

# Namespace Lib9c

## Classes

[Currencies](#)

[SerializeKeys](#)

# Class Currencies

Namespace: [Lib9c](#)

Assembly: Lib9c.dll

```
public static class Currencies
```

## Inheritance

[object](#) ← Currencies

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Crystal

```
public static readonly Currency Crystal
```

Field Value

Currency

### DailyRewardRune

```
public static readonly Currency DailyRewardRune
```

Field Value

Currency

### FreyaBlessingRune

```
public static readonly Currency FreyaBlessingRune
```

Field Value

Currency

## FreyaLiberationRune

```
public static readonly Currency FreyaLiberationRune
```

Field Value

Currency

## Garage

```
public static readonly Currency Garage
```

Field Value

Currency

## GuildGold

```
public static readonly Currency GuildGold
```

Field Value

Currency

## Mead

```
public static readonly Currency Mead
```

Field Value

Currency

## OdinWeaknessRune

```
public static readonly Currency OdinWeaknessRune
```

Field Value

Currency

## OdinWisdomRune

```
public static readonly Currency OdinWisdomRune
```

Field Value

Currency

## StakeRune

```
public static readonly Currency StakeRune
```

Field Value

Currency

## Methods

GetItemCurrency(int, bool)

```
public static Currency GetItemCurrency(int itemId, bool tradable)
```

## Parameters

itemId [int](#)

tradable [bool](#)

## Returns

Currency

## GetMinterlessCurrency(string?)

Covers the reward.CurrencyTicker is following cases: - Currencies.Crystal.Ticker - Currencies.Garage.Ticker - lower case is starting with "rune\_" or "runestone\_" - lower case is starting with "soulstone\_"

```
public static Currency GetMinterlessCurrency(string? ticker)
```

## Parameters

ticker [string](#)

## Returns

Currency

## Exceptions

[ArgumentNullException](#)

[ArgumentException](#)

## GetRune(string?)

```
public static Currency GetRune(string? ticker)
```

Parameters

`ticker string`

Returns

`Currency`

## GetRunes(RuneSheet?)

```
public static IOrderedEnumerable<Currency> GetRunes(RuneSheet? sheet)
```

Parameters

`sheet RuneSheet`

Returns

`IOrderedEnumerable<Currency>`

## GetRunes(params string?[])

```
public static IOrderedEnumerable<Currency> GetRunes(params string?[] tickers)
```

Parameters

`tickers string[]`

Returns

`IOrderedEnumerable<Currency>`

## GetSoulStone(string?)

```
public static Currency GetSoulStone(string? ticker)
```

Parameters

`ticker` [string](#)

Returns

`Currency`

## GetSoulStones(PetSheet?)

```
public static IOrderedEnumerable<Currency> GetSoulStones(PetSheet? sheet)
```

Parameters

`sheet` [PetSheet](#)

Returns

[IOrderedEnumerable](#)<Currency>

## GetSoulStones(params string?[])

```
public static IOrderedEnumerable<Currency> GetSoulStones(params string?[] tickers)
```

Parameters

`tickers` [string](#)[]

Returns

[IOrderedEnumerable](#)<Currency>

## GetUnwrappedCurrency(Currency)

```
public static Currency GetUnwrappedCurrency(Currency currency)
```

Parameters

**currency** Currency

Returns

Currency

## GetWrappedCurrency(Currency)

```
public static Currency GetWrappedCurrency(Currency currency)
```

Parameters

**currency** Currency

Returns

Currency

## IsRuneTicker(string)

```
public static bool IsRuneTicker(string ticker)
```

Parameters

**ticker** [string](#)

Returns

[bool](#)

## IsSoulstoneTicker(string)

```
public static bool IsSoulstoneTicker(string ticker)
```

Parameters

`ticker` [string](#)

Returns

[bool](#)

## IsWrappedCurrency(Currency)

```
public static bool IsWrappedCurrency(Currency currency)
```

Parameters

`currency` [Currency](#)

Returns

[bool](#)

## ParseItemCurrency(Currency)

```
public static (bool tradable, int itemId) ParseItemCurrency(Currency currency)
```

Parameters

`currency` [Currency](#)

Returns

[\(bool](#) [tradable](#), [int](#) [itemId](#))

## PickAddress(Currency, Address, Address)

pick address by currency ticker

```
public static Address PickAddress(Currency currency, Address agentAddress,  
Address avatarAddress)
```

## Parameters

**currency** Currency

Libplanet.Types.Assets.Currency

**agentAddress** Address

Libplanet.Crypto.Address

**avatarAddress** Address

Libplanet.Crypto.Address

## Returns

Address

agentAddress when ticker is NCG or Crystal or Garage or Mead else avatarAddress

# Class SerializeKeys

Namespace: [Lib9c](#)

Assembly: Lib9c.dll

```
public static class SerializeKeys
```

## Inheritance

[object](#) ← SerializeKeys

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### AchievementsKey

```
public const string AchievementsKey = "ach"
```

Field Value

[string](#)

### ActionPointKey

```
public const string ActionPointKey = "ap"
```

Field Value

[string](#)

### AdditionalStatValueKey

```
public const string AdditionalStatValueKey = "av"
```

Field Value

[string](#)

## AddressKey

```
public const string AddressKey = "a"
```

Field Value

[string](#)

## AgentAddressKey

```
public const string AgentAddressKey = "aga"
```

Field Value

[string](#)

## AmountKey

```
public const string AmountKey = "am"
```

Field Value

[string](#)

## AvatarAddressKey

```
public const string AvatarAddressKey = "aa"
```

Field Value

[string](#)

## BlockIndexKey

```
public const string BlockIndexKey = "bi"
```

Field Value

[string](#)

## BuffSkillsKey

```
public const string BuffSkillsKey = "bs"
```

Field Value

[string](#)

## BuyerAgentAddressKey

```
public const string BuyerAgentAddressKey = "bga"
```

Field Value

[string](#)

## BuyerAvatarAddressKey

```
public const string BuyerAvatarAddressKey = "ba"
```

Field Value

[string](#)

## ByCustomCraftKey

```
public const string ByCustomCraftKey = "bcc"
```

Field Value

[string](#)

## CancellableBlockIndexKey

```
public const string CancellableBlockIndexKey = "cbi"
```

Field Value

[string](#)

## ChampionshipIdKey

```
public const string ChampionshipIdKey = "chi"
```

Field Value

[string](#)

## ChanceKey

```
public const string ChanceKey = "ch"
```

Field Value

[string](#)

## CharacterIdKey

```
public const string CharacterIdKey = "ci"
```

Field Value

[string](#)

## ClaimDataKey

```
public const string ClaimDataKey = "cd"
```

Field Value

[string](#)

## CombatPointKey

```
public const string CombatPointKey = "cp"
```

Field Value

[string](#)

## CombinationSlotAddressesKey

```
public const string CombinationSlotAddressesKey = "csa"
```

Field Value

[string](#)

## CompletedQuestIdsKeyDeprecated

```
public const string CompletedQuestIdsKeyDeprecated = "completedQuestIds"
```

Field Value

[string](#)

## CostumeKey

```
public const string CostumeKey = "c"
```

Field Value

[string](#)

## CostumesKey

```
public const string CostumesKey = "cs"
```

Field Value

[string](#)

## CouponWalletKey

```
public const string CouponWalletKey = "coupon_wallet"
```

Field Value

[string](#)

## CraftWithRandomKey

```
public const string CraftWithRandomKey = "cwr"
```

Field Value

[string](#)

## DailyRewardReceivedIndexKey

```
public const string DailyRewardReceivedIndexKey = "dri"
```

Field Value

[string](#)

## EarKey

```
public const string EarKey = "e"
```

Field Value

[string](#)

## ElementalTypeKey

```
public const string ElementalTypeKey = "et"
```

Field Value

[string](#)

## EndKey

```
public const string EndKey = "ed"
```

Field Value

[string](#)

## EnemyAvatarAddressKey

```
public const string EnemyAvatarAddressKey = "eaa"
```

Field Value

[string](#)

## EquipmentExpKey

```
public const string EquipmentExpKey = "eq_exp"
```

Field Value

[string](#)

## EquipmentIconIdKey

```
public const string EquipmentIconIdKey = "icon_id"
```

Field Value

[string](#)

## EquipmentsKey

```
public const string EquipmentsKey = "es"
```

Field Value

[string](#)

## ErrorCodeKey

```
public const string ErrorCodeKey = "ec"
```

Field Value

[string](#)

## EventMapKey

```
public const string EventMapKey = "em"
```

Field Value

[string](#)

## ExpKey

```
public const string ExpKey = "exp"
```

Field Value

[string](#)

## ExpiredBlockIndexKey

```
public const string ExpiredBlockIndexKey = "ebi"
```

Field Value

[string](#)

## GradeKey

```
public const string GradeKey = "g"
```

Field Value

[string](#)

## GrandFinaleIdKey

```
public const string GrandFinaleIdKey = "gfi"
```

Field Value

[string](#)

## HairKey

```
public const string HairKey = "h"
```

Field Value

[string](#) ↗

## HasRandomOnlyIconKey

```
public const string HasRandomOnlyIconKey = "hroi"
```

Field Value

[string](#) ↗

## IdKey

```
public const string IdKey = "id"
```

Field Value

[string](#) ↗

## ItemCountKey

```
public const string ItemCountKey = "item_count"
```

Field Value

[string](#) ↗

## ItemIdKey

```
public const string ItemIdKey = "ii"
```

Field Value

[string](#)

## ItemMapKey

```
public const string ItemMapKey = "im"
```

Field Value

[string](#)

## ItemSubTypeKey

```
public const string ItemSubTypeKey = "ist"
```

Field Value

[string](#)

## ItemTypeKey

```
public const string ItemTypeKey = "it"
```

Field Value

[string](#)

## LegacyActionPointKey

```
public const string LegacyActionPointKey = "actionPoint"
```

Field Value

[string](#)

## LegacyAdditionalStatValueKey

```
public const string LegacyAdditionalStatValueKey = "additionalValue"
```

Field Value

[string](#)

## LegacyAddressKey

```
public const string LegacyAddressKey = "address"
```

Field Value

[string](#)

## LegacyAgentAddressKey

```
public const string LegacyAgentAddressKey = "agentAddress"
```

Field Value

[string](#)

## LegacyBlockIndexKey

```
public const string LegacyBlockIndexKey = "blockIndex"
```

Field Value

[string](#)

## LegacyBuffSkillsKey

```
public const string LegacyBuffSkillsKey = "buffSkills"
```

Field Value

[string](#)

## LegacyChanceKey

```
public const string LegacyChanceKey = "chance"
```

Field Value

[string](#)

## LegacyCharacterIdKey

```
public const string LegacyCharacterIdKey = "characterId"
```

Field Value

[string](#)

## LegacyCombinationSlotAddressesKey

```
public const string LegacyCombinationSlotAddressesKey = "combinationSlotAddresses"
```

Field Value

[string](#)

## LegacyCostumeItemIdKey

```
public const string LegacyCostumeItemIdKey = "item_id"
```

Field Value

[string](#)

## LegacyCostumeKey

```
public const string LegacyCostumeKey = "costume"
```

Field Value

[string](#)

## LegacyDailyRewardReceivedIndexKey

```
public const string LegacyDailyRewardReceivedIndexKey = "dailyRewardReceivedIndex"
```

Field Value

[string](#)

## LegacyEarKey

```
public const string LegacyEarKey = "ear"
```

Field Value

[string](#)

## LegacyElementTypeKey

```
public const string LegacyElementTypeKey = "elemental_type"
```

Field Value

[string](#)

## LegacyEquippedKey

```
public const string LegacyEquippedKey = "equipped"
```

Field Value

[string](#)

## LegacyEventMapKey

```
public const string LegacyEventMapKey = "eventMap"
```

Field Value

[string](#)

## LegacyGradeKey

```
public const string LegacyGradeKey = "grade"
```

Field Value

[string](#)

## LegacyHairKey

```
public const string LegacyHairKey = "hair"
```

Field Value

[string](#)

## LegacyInventoryKey

```
public const string LegacyInventoryKey = "inventory"
```

Field Value

[string](#)

## LegacyItemIdKey

```
public const string LegacyItemIdKey = "itemId"
```

Field Value

[string](#)

## LegacyItemMapKey

```
public const string LegacyItemMapKey = "itemMap"
```

Field Value

[string](#)

## LegacyItemSubTypeKey

```
public const string LegacyItemSubTypeKey = "item_sub_type"
```

Field Value

[string](#)

## LegacyItemTypeKey

```
public const string LegacyItemTypeKey = "item_type"
```

Field Value

[string](#)

## LegacyItemUsableKey

```
public const string LegacyItemUsableKey = "itemUsable"
```

Field Value

[string](#)

## LegacyLevelKey

```
public const string LegacyLevelKey = "level"
```

Field Value

[string](#)

## LegacyMailBoxKey

```
public const string LegacyMailBoxKey = "mailBox"
```

Field Value

[string](#)

## LegacyMonsterMapKey

```
public const string LegacyMonsterMapKey = "monsterMap"
```

Field Value

[string](#)

## LegacyNameKey

```
public const string LegacyNameKey = "name"
```

Field Value

[string](#)

## LegacyNonceKey

```
public const string LegacyNonceKey = "nonce"
```

Field Value

[string](#)

## LegacyPowerKey

```
public const string LegacyPowerKey = "power"
```

Field Value

[string](#)

## LegacyPriceKey

```
public const string LegacyPriceKey = "price"
```

Field Value

[string](#)

## LegacyProductIdKey

```
public const string LegacyProductIdKey = "productId"
```

Field Value

[string](#)

## LegacyProductsKey

```
public const string LegacyProductsKey = "products"
```

Field Value

[string](#)

## LegacyQuestListKey

```
public const string LegacyQuestListKey = "questList"
```

Field Value

[string](#)

## LegacyRankingMapAddressKey

```
public const string LegacyRankingMapAddressKey = "ranking_map_address"
```

Field Value

[string](#)

## LegacyRequiredBlockIndexKey

```
public const string LegacyRequiredBlockIndexKey = "requiredBlockIndex"
```

Field Value

[string](#)

## LegacySellerAgentAddressKey

```
public const string LegacySellerAgentAddressKey = "sellerAgentAddress"
```

Field Value

[string](#)

## LegacySellerAvatarAddressKey

```
public const string LegacySellerAvatarAddressKey = "sellerAvatarAddress"
```

Field Value

[string](#)

## LegacySetIdKey

```
public const string LegacySetIdKey = "set_id"
```

Field Value

[string](#)

## LegacySkillRowKey

```
public const string LegacySkillRowKey = "skillRow"
```

Field Value

[string](#)

## LegacySkillsKey

```
public const string LegacySkillsKey = "skills"
```

Field Value

[string](#)

## LegacySpineResourcePathKey

```
public const string LegacySpineResourcePathKey = "spine_resource_path"
```

Field Value

[string](#)

## LegacyStageMapKey

```
public const string LegacyStageMapKey = "stageMap"
```

Field Value

[string](#)

## LegacyStatKey

```
public const string LegacyStatKey = "stat"
```

Field Value

[string](#)

## LegacyStatTypeKey

```
public const string LegacyStatTypeKey = "statType"
```

Field Value

[string](#)

## LegacyStatValueKey

```
public const string LegacyStatValueKey = "value"
```

Field Value

[string](#)

## LegacyStatsMapKey

```
public const string LegacyStatsMapKey = "statsMap"
```

Field Value

[string](#)

## LegacyTailKey

```
public const string LegacyTailKey = "tail"
```

Field Value

[string](#)

## LegacyUpdatedAtKey

```
public const string LegacyUpdatedAtKey = "updatedAt"
```

Field Value

[string](#)

## LegacyWorldInformationKey

```
public const string LegacyWorldInformationKey = "worldInformation"
```

Field Value

[string](#)

## LensKey

```
public const string LensKey = "lens"
```

Field Value

[string](#)

## LevelKey

```
public const string LevelKey = "l"
```

Field Value

[string](#)

## MadeWithMimisbrunnrRecipeKey

```
public const string MadeWithMimisbrunnrRecipeKey = "mwmr"
```

Field Value

[string](#)

## MailBoxKey

```
public const string MailBoxKey = "mb"
```

Field Value

[string](#)

## MemoKey

```
public const string MemoKey = "m"
```

Field Value

[string](#)

## MonsterCollectionResultKey

```
public const string MonsterCollectionResultKey = "mcr"
```

Field Value

[string](#)

## MonsterCollectionRoundKey

```
public const string MonsterCollectionRoundKey = "s"
```

Field Value

[string](#)

## MonsterMapKey

```
public const string MonsterMapKey = "mm"
```

Field Value

[string](#)

## MyAvatarAddressKey

```
public const string MyAvatarAddressKey = "maa"
```

Field Value

[string](#)

## NameKey

```
public const string NameKey = "n"
```

Field Value

[string](#)

## OptionCountFromCombinationKey

```
public const string OptionCountFromCombinationKey = "oc"
```

Field Value

[string](#)

## OrderDigestListKey

```
public const string OrderDigestListKey = "odl"
```

Field Value

[string](#)

## OrderIdKey

```
public const string OrderIdKey = "oi"
```

Field Value

[string](#)

## OrderReceiptListKey

```
public const string OrderReceiptListKey = "orl"
```

Field Value

[string](#)

## OrderTypeKey

```
public const string OrderTypeKey = "ot"
```

Field Value

[string](#)

## PowerKey

```
public const string PowerKey = "pw"
```

Field Value

[string](#)

## PriceKey

```
public const string PriceKey = "p"
```

Field Value

[string](#)

## ProductIdKey

```
public const string ProductIdKey = "pi"
```

Field Value

[string](#)

## ProductsKey

```
public const string ProductsKey = "pr"
```

Field Value

[string](#)

## PurchaseInfosKey

```
public const string PurchaseInfosKey = "pis"
```

Field Value

[string](#)

## PurchaseResultsKey

```
public const string PurchaseResultsKey = "prs"
```

Field Value

[string](#)

## QuantityKey

```
public const string QuantityKey = "q"
```

Field Value

[string](#)

## QuestsKeyDeprecated

```
public const string QuestsKeyDeprecated = "quests"
```

Field Value

[string](#)

## RankingMapAddressKey

```
public const string RankingMapAddressKey = "rma"
```

Field Value

[string](#)

## RateKey

```
public const string RateKey = "r"
```

Field Value

[string](#)

## ReceivedBlockIndexKey

```
public const string ReceivedBlockIndexKey = "rbi2"
```

Field Value

[string](#)

## RequiredBlockIndexKey

```
public const string RequiredBlockIndexKey = "rbi"
```

Field Value

[string](#)

## RequiredCharacterLevelKey

```
public const string RequiredCharacterLevelKey = "rc"
```

Field Value

[string](#)

## RewardLevelKey

```
public const string RewardLevelKey = "rl"
```

Field Value

[string](#)

## RewardLevelMapKey

```
public const string RewardLevelMapKey = "rlm"
```

Field Value

[string](#)

## RewardMapKey

```
public const string RewardMapKey = "rm"
```

Field Value

[string](#)

## RoundKey

```
public const string RoundKey = "rd"
```

Field Value

[string](#)

## RuneInfos

```
public const string RuneInfos = "ri"
```

Field Value

[string](#)

## SellerAgentAddressKey

```
public const string SellerAgentAddressKey = "sga"
```

Field Value

[string](#)

## SellerAvatarAddressKey

```
public const string SellerAvatarAddressKey = "sva"
```

Field Value

[string](#)

## SellerResultsKey

```
public const string SellerResultsKey = "srs"
```

Field Value

[string](#)

## SetIdKey

```
public const string SetIdKey = "si"
```

Field Value

[string](#)

## SkillsKey

```
public const string SkillsKey = "sk"
```

Field Value

[string](#)

## SpineResourcePathKey

```
public const string SpineResourcePathKey = "srp"
```

Field Value

[string](#)

## StageMapKey

```
public const string StageMapKey = "sm"
```

Field Value

[string](#)

## StartedBlockIndexKey

```
public const string StartedBlockIndexKey = "sbi"
```

Field Value

[string](#)

## StatKey

```
public const string StatKey = "st"
```

Field Value

[string](#)

## StatTypeKey

```
public const string StatTypeKey = "stt"
```

Field Value

[string](#) ↗

## StatValueKey

```
public const string StatValueKey = "v"
```

Field Value

[string](#) ↗

## TailKey

```
public const string TailKey = "t"
```

Field Value

[string](#) ↗

## TicketKey

```
public const string TicketKey = "tk"
```

Field Value

[string](#) ↗

## TradableFungibleItemCountKey

```
public const string TradableFungibleItemCountKey = "tradable_fungible_item_count"
```

Field Value

[string](#)

## TradableFungibleItemKey

```
public const string TradableFungibleItemKey = "tradable_fungible_item"
```

Field Value

[string](#)

## TradableIdKey

```
public const string TradableIdKey = "ti"
```

Field Value

[string](#)

## UpdateSellInfoKey

```
public const string UpdateSellInfoKey = "usi"
```

Field Value

[string](#)

## UpdatedAtKey

```
public const string UpdatedAtKey = "ua"
```

Field Value

[string](#) ↗

## updateSellOrderIdKey

```
public const string updateSellOrderIdKey = "roi"
```

Field Value

[string](#) ↗

# Namespace Lib9c.Exceptions

## Classes

[InvalidStateTypeException](#)

# Class InvalidStateTypeException

Namespace: [Lib9c.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidStateTypeException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidStateTypeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidStateTypeException(string)

```
public InvalidStateTypeException(string message)
```

## Parameters

message [string](#)

# Namespace Lib9c.Model.Order

## Classes

[FungibleOrder](#)

[NonFungibleOrder](#)

[Order](#)

[OrderBase](#)

[OrderDigest](#)

[OrderDigestListState](#)

[OrderFactory](#)

[OrderReceipt](#)

## Enums

[Order.OrderType](#)

# Class FungibleOrder

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FungibleOrder : Order
```

## Inheritance

[object](#) ← [OrderBase](#) ← [Order](#) ← FungibleOrder

## Inherited Members

[Order.ExpirationInterval](#) , [Order.DeriveAddress\(Guid\)](#) , [Order.SellerAgentAddress](#) ,  
[Order.SellerAvatarAddress](#) , [Order.Price](#) , [Order.ItemSubType](#) , [Order.GetTax\(\)](#) ,  
[Order.Cancel\(AvatarState, long\)](#) , [Order.Equals\(Order\)](#) , [OrderBase.OrderId](#) ,  
[OrderBase.TradableId](#) , [OrderBase.StartedBlockIndex](#) , [OrderBase.ExpiredBlockIndex](#) ,  
[OrderBase.Equals\(OrderBase\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### FungibleOrder(Dictionary)

```
public FungibleOrder(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### FungibleOrder(Address, Address, Guid, FungibleAssetValue, Guid, long, ItemSubType, int)

```
public FungibleOrder(Address sellerAgentAddress, Address sellerAvatarAddress, Guid
orderId, FungibleAssetValue price, Guid tradableId, long startedBlockIndex,
ItemSubType itemSubType, int itemCount)
```

## Parameters

`sellerAgentAddress` Address

`sellerAvatarAddress` Address

`orderId` [Guid](#)

`price` FungibleAssetValue

`tradableId` [Guid](#)

`startedBlockIndex` [long](#)

`itemSubType` [ItemSubType](#)

`ItemCount` [int](#)

## Fields

### ItemCount

`public readonly int ItemCount`

Field Value

[int](#)

## Properties

### Type

`public override Order.OrderType Type { get; }`

Property Value

[Order.OrderType](#)

# Methods

## Cancel2(AvatarState, long)

```
[Obsolete("Use Cancel")]
public override ITradableItem Cancel2(AvatarState avatarState, long blockIndex)
```

### Parameters

avatarState [AvatarState](#)

blockIndex [long](#)

### Returns

[ITradableItem](#)

## Digest(AvatarState, CostumeStatSheet)

```
public override OrderDigest Digest(AvatarState avatarState,
CostumeStatSheet costumeStatSheet)
```

### Parameters

avatarState [AvatarState](#)

costumeStatSheet [CostumeStatSheet](#)

### Returns

[OrderDigest](#)

## Digest2(AvatarState, CostumeStatSheet)

```
[Obsolete("Use Digest")]
public override OrderDigest Digest2(AvatarState avatarState,
CostumeStatSheet costumeStatSheet)
```

Parameters

avatarState [AvatarState](#)

costumeStatSheet [CostumeStatSheet](#)

Returns

[OrderDigest](#)

## Equals(FungibleOrder)

```
protected bool Equals(FungibleOrder other)
```

Parameters

other [FungibleOrder](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Sell(AvatarState)

```
public override ITradableItem Sell(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell2(AvatarState)

```
[Obsolete("Use Sell")]
public override ITradableItem Sell2(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell3(AvatarState)

```
[Obsolete("Use Sell")]
public override ITradableItem Sell3(AvatarState avatarState)
```

### Parameters

avatarState [AvatarState](#)

### Returns

[ITradableItem](#)

## Sell4(AvatarState)

```
[Obsolete("Use Sell")]
public override ITradableItem Sell4(AvatarState avatarState)
```

### Parameters

avatarState [AvatarState](#)

### Returns

[ITradableItem](#)

## Serialize()

```
public override IValue Serialize()
```

### Returns

IValue

## Transfer(AvatarState, AvatarState, long)

```
public override OrderReceipt Transfer(AvatarState seller, AvatarState buyer,  
long blockIndex)
```

## Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

## Returns

[OrderReceipt](#)

## Transfer2(AvatarState, AvatarState, long)

```
[Obsolete("Use Transfer")]  
public override OrderReceipt Transfer2(AvatarState seller, AvatarState buyer,  
long blockIndex)
```

## Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

## Returns

[OrderReceipt](#)

## Transfer3(AvatarState, AvatarState, long)

```
[Obsolete("Use Transfer")]  
public override OrderReceipt Transfer3(AvatarState seller, AvatarState buyer,  
long blockIndex)
```

Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

Returns

[OrderReceipt](#)

## Validate(AvatarState, int)

```
public override void Validate(AvatarState avatarState, int count)
```

Parameters

avatarState [AvatarState](#)

count [int](#)

## ValidateCancelOrder(AvatarState, Guid)

```
public override void ValidateCancelOrder(AvatarState avatarState, Guid tradableId)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

## ValidateCancelOrder2(AvatarState, Guid)

```
[Obsolete("Use ValidateCancelOrder")]
```

```
public override void ValidateCancelOrder2(AvatarState avatarState, Guid tradableId)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

## ValidateTransfer(AvatarState, Guid, FungibleAssetValue, long)

```
public override int ValidateTransfer(AvatarState avatarState, Guid tradableId,  
FungibleAssetValue price, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

price FungibleAssetValue

blockIndex [long](#)

Returns

[int](#)

## ValidateTransfer2(AvatarState, Guid, FungibleAssetValue, long)

```
[Obsolete("Use ValidateTransfer")]  
public override int ValidateTransfer2(AvatarState avatarState, Guid tradableId,  
FungibleAssetValue price, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

**price** FungibleAssetValue

**blockIndex** [long](#)

Returns

[int](#)

# Class NonFungibleOrder

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NonFungibleOrder : Order
```

## Inheritance

[object](#) ← [OrderBase](#) ← [Order](#) ← NonFungibleOrder

## Inherited Members

[Order.ExpirationInterval](#) , [Order.DeriveAddress\(Guid\)](#) , [Order.SellerAgentAddress](#) ,  
[Order.SellerAvatarAddress](#) , [Order.Price](#) , [Order.ItemSubType](#) , [Order.GetTax\(\)](#) ,  
[Order.Cancel\(AvatarState, long\)](#) , [Order.Serialize\(\)](#) , [Order.Equals\(Order\)](#) ,  
[Order.Equals\(object\)](#) , [Order.GetHashCode\(\)](#) , [OrderBase.OrderId](#) , [OrderBase.TradableId](#) ,  
[OrderBase.StartedBlockIndex](#) , [OrderBase.ExpiredBlockIndex](#) ,  
[OrderBase.Equals\(OrderBase\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### NonFungibleOrder(Dictionary)

```
public NonFungibleOrder(Dictionary serialized)
```

#### Parameters

`serialized` Dictionary

### NonFungibleOrder(Address, Address, Guid, FungibleAssetValue, Guid, long, ItemSubType)

```
public NonFungibleOrder(Address sellerAgentAddress, Address sellerAvatarAddress,
Guid orderId, FungibleAssetValue price, Guid tradableId, long startedBlockIndex,
```

```
ItemSubType itemSubType)
```

## Parameters

sellerAgentAddress Address

sellerAvatarAddress Address

orderId [Guid](#)

price FungibleAssetValue

tradableId [Guid](#)

startedBlockIndex [long](#)

itemSubType [ItemSubType](#)

## Properties

### Type

```
public override Order.OrderType Type { get; }
```

## Property Value

[Order.OrderType](#)

## Methods

### Cancel2(AvatarState, long)

```
[Obsolete("Use Cancel")]
public override ITradableItem Cancel2(AvatarState avatarState, long blockIndex)
```

## Parameters

avatarState [AvatarState](#)

blockIndex [long](#)

Returns

[ITractableItem](#)

## Digest(AvatarState, CostumeStatSheet)

```
public override OrderDigest Digest(AvatarState avatarState,  
CostumeStatSheet costumeStatSheet)
```

Parameters

avatarState [AvatarState](#)

costumeStatSheet [CostumeStatSheet](#)

Returns

[OrderDigest](#)

## Digest2(AvatarState, CostumeStatSheet)

```
[Obsolete("Use Digest")]  
public override OrderDigest Digest2(AvatarState avatarState,  
CostumeStatSheet costumeStatSheet)
```

Parameters

avatarState [AvatarState](#)

costumeStatSheet [CostumeStatSheet](#)

Returns

[OrderDigest](#)

## Sell(AvatarState)

```
public override ITradableItem Sell(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell2(AvatarState)

```
[Obsolete("Use Sell")]
public override ITradableItem Sell2(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell3(AvatarState)

```
[Obsolete("Use Sell")]
public override ITradableItem Sell3(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell4(AvatarState)

```
[Obsolete("Use Sell")]
public override ITradableItem Sell4(AvatarState avatarState)
```

### Parameters

avatarState [AvatarState](#)

### Returns

[ITradableItem](#)

## Transfer(AvatarState, AvatarState, long)

```
public override OrderReceipt Transfer(AvatarState seller, AvatarState buyer,
long blockIndex)
```

### Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

### Returns

[OrderReceipt](#)

## Transfer2(AvatarState, AvatarState, long)

```
[Obsolete("Use Transfer")]
public override OrderReceipt Transfer2(AvatarState seller, AvatarState buyer,
long blockIndex)
```

Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

Returns

[OrderReceipt](#)

## Transfer3(AvatarState, AvatarState, long)

```
[Obsolete("Use Transfer")]
public override OrderReceipt Transfer3(AvatarState seller, AvatarState buyer,
long blockIndex)
```

Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

Returns

[OrderReceipt](#)

## Validate(AvatarState, int)

```
public override void Validate(AvatarState avatarState, int count)
```

Parameters

avatarState [AvatarState](#)

count [int](#)

## ValidateCancelOrder(AvatarState, Guid)

```
public override void ValidateCancelOrder(AvatarState avatarState, Guid tradableId)
```

### Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

## ValidateCancelOrder2(AvatarState, Guid)

```
[Obsolete("Use ValidateCancelOrder")]
```

```
public override void ValidateCancelOrder2(AvatarState avatarState, Guid tradableId)
```

### Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

## ValidateTransfer(AvatarState, Guid, FungibleAssetValue, long)

```
public override int ValidateTransfer(AvatarState avatarState, Guid tradableId,  
FungibleAssetValue price, long blockIndex)
```

### Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

price FungibleAssetValue

blockIndex [long](#)

Returns

[int](#)

## ValidateTransfer2(AvatarState, Guid, FungibleAssetValue, long)

```
[Obsolete("Use ValidateTransfer")]
public override int ValidateTransfer2(AvatarState avatarState, Guid tradableId,
FungibleAssetValue price, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

price FungibleAssetValue

blockIndex [long](#)

Returns

[int](#)

# Class Order

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Order : OrderBase
```

## Inheritance

[object](#) ← [OrderBase](#) ← Order

## Derived

[FungibleOrder](#), [NonFungibleOrder](#)

## Inherited Members

[OrderBase.OrderId](#), [OrderBase.TradableId](#), [OrderBase.StartedBlockIndex](#),  
[OrderBase.ExpiredBlockIndex](#), [OrderBase.Equals\(OrderBase\)](#),  
[object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

### Order(Dictionary)

```
protected Order(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Order(Address, Address, Guid, FungibleAssetValue, Guid, long, ItemSubType)

```
protected Order(Address sellerAgentAddress, Address sellerAvatarAddress, Guid
orderId, FungibleAssetValue price, Guid tradableId, long startedBlockIndex,
ItemSubType itemSubType)
```

## Parameters

`sellerAgentAddress` Address

`sellerAvatarAddress` Address

`orderId` [Guid](#)

`price` FungibleAssetValue

`tradableId` [Guid](#)

`startedBlockIndex` [long](#)

`itemSubType` [ItemSubType](#)

## Fields

### ExpirationInterval

```
public const long ExpirationInterval = 36000
```

Field Value

[long](#)

## Properties

### ItemSubType

```
public ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

## Price

```
public FungibleAssetValue Price { get; }
```

Property Value

FungibleAssetValue

## SellerAgentAddress

```
public Address SellerAgentAddress { get; }
```

Property Value

Address

## SellerAvatarAddress

```
public Address SellerAvatarAddress { get; }
```

Property Value

Address

## Type

```
public abstract Order.OrderType Type { get; }
```

Property Value

[Order.OrderType](#)

## Methods

Cancel(AvatarState, long)

```
public ITradableItem Cancel(AvatarState avatarState, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

blockIndex [long](#)

Returns

[ITradableItem](#)

## Cancel2(AvatarState, long)

```
[Obsolete("Use Cancel")]
```

```
public abstract ITradableItem Cancel2(AvatarState avatarState, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

blockIndex [long](#)

Returns

[ITradableItem](#)

## DeriveAddress(Guid)

```
public static Address DeriveAddress(Guid orderId)
```

Parameters

orderId [Guid](#)

Returns

Address

## Digest(AvatarState, CostumeStatSheet)

```
public abstract OrderDigest Digest(AvatarState avatarState,  
CostumeStatSheet costumeStatSheet)
```

Parameters

avatarState [AvatarState](#)

costumeStatSheet [CostumeStatSheet](#)

Returns

[OrderDigest](#)

## Digest2(AvatarState, CostumeStatSheet)

```
[Obsolete("Use Digest")]  
public abstract OrderDigest Digest2(AvatarState avatarState,  
CostumeStatSheet costumeStatSheet)
```

Parameters

avatarState [AvatarState](#)

costumeStatSheet [CostumeStatSheet](#)

Returns

[OrderDigest](#)

## Equals(Order)

```
protected bool Equals(Order other)
```

Parameters

[other Order](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

[obj object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetTax()

```
public FungibleAssetValue GetTax()
```

Returns

FungibleAssetValue

## Sell(AvatarState)

```
public abstract ITradableItem Sell(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell2(AvatarState)

```
[Obsolete("Use Sell")]
public abstract ITradableItem Sell2(AvatarState avatarState)
```

Parameters

avatarState [AvatarState](#)

Returns

[ITradableItem](#)

## Sell3(AvatarState)

```
[Obsolete("Use Sell")]
public abstract ITradableItem Sell3(AvatarState avatarState)
```

## Parameters

avatarState [AvatarState](#)

## Returns

[ITradableItem](#)

## Sell4(AvatarState)

```
[Obsolete("Use Sell")]
public abstract ITradableItem Sell4(AvatarState avatarState)
```

## Parameters

avatarState [AvatarState](#)

## Returns

[ITradableItem](#)

## Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

## Transfer(AvatarState, AvatarState, long)

```
public abstract OrderReceipt Transfer(AvatarState seller, AvatarState buyer,  
long blockIndex)
```

## Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#) ↗

## Returns

[OrderReceipt](#)

## Transfer2(AvatarState, AvatarState, long)

```
[Obsolete("Use Transfer")]  
public abstract OrderReceipt Transfer2(AvatarState seller, AvatarState buyer,  
long blockIndex)
```

## Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#) ↗

## Returns

[OrderReceipt](#)

## Transfer3(AvatarState, AvatarState, long)

```
[Obsolete("Use Transfer")]  
public abstract OrderReceipt Transfer3(AvatarState seller, AvatarState buyer,  
long blockIndex)
```

Parameters

seller [AvatarState](#)

buyer [AvatarState](#)

blockIndex [long](#)

Returns

[OrderReceipt](#)

## Validate(AvatarState, int)

```
public virtual void Validate(AvatarState avatarState, int count)
```

Parameters

avatarState [AvatarState](#)

count [int](#)

## ValidateCancelOrder(AvatarState, Guid)

```
public virtual void ValidateCancelOrder(AvatarState avatarState, Guid tradableId)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

## ValidateCancelOrder2(AvatarState, Guid)

```
[Obsolete("Use ValidateCancelOrder")]
```

```
public virtual void ValidateCancelOrder2(AvatarState avatarState, Guid tradableId)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

## ValidateTransfer(AvatarState, Guid, FungibleAssetValue, long)

```
public virtual int ValidateTransfer(AvatarState avatarState, Guid tradableId,  
FungibleAssetValue price, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

price FungibleAssetValue

blockIndex [long](#)

Returns

[int](#)

## ValidateTransfer2(AvatarState, Guid, FungibleAssetValue, long)

```
[Obsolete("Use ValidateTransfer")]  
public virtual int ValidateTransfer2(AvatarState avatarState, Guid tradableId,  
FungibleAssetValue price, long blockIndex)
```

Parameters

avatarState [AvatarState](#)

tradableId [Guid](#)

**price** FungibleAssetValue

**blockIndex** [long](#)

Returns

[int](#)

# Enum Order.OrderType

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
public enum Order.OrderType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Fungible = 0

NonFungible = 1

# Class OrderBase

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderBase
```

## Inheritance

[object](#) ← OrderBase

## Derived

[Order](#), [OrderDigest](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### OrderBase(Dictionary)

```
public OrderBase(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### OrderBase(Guid, Guid, long, long)

```
public OrderBase(Guid orderId, Guid tradableId, long startedBlockIndex,
long expiredBlockIndex)
```

#### Parameters

orderId [Guid](#)

tradableId [Guid ↗](#)

startedBlockIndex [long ↗](#)

expiredBlockIndex [long ↗](#)

## Fields

### OrderId

`public readonly Guid OrderId`

### Field Value

[Guid ↗](#)

### TradableId

`public readonly Guid TradableId`

### Field Value

[Guid ↗](#)

## Properties

### ExpiredBlockIndex

`public long ExpiredBlockIndex { get; }`

### Property Value

[long ↗](#)

# StartedBlockIndex

```
public long StartedBlockIndex { get; }
```

Property Value

[long](#) ↗

## Methods

### Equals(OrderBase)

```
protected bool Equals(OrderBase other)
```

Parameters

other [OrderBase](#)

Returns

[bool](#) ↗

### Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#) ↗

The object to compare with the current object.

Returns

[bool](#) ↗

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public virtual IValue Serialize()
```

Returns

IValue

# Class OrderDigest

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderDigest : OrderBase
```

## Inheritance

[object](#) ← [OrderBase](#) ← OrderDigest

## Inherited Members

[OrderBase.OrderId](#) , [OrderBase.TradableId](#) , [OrderBase.StartedBlockIndex](#) ,  
[OrderBase.ExpiredBlockIndex](#) , [OrderBase.Equals\(OrderBase\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### OrderDigest(Dictionary)

```
public OrderDigest(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### OrderDigest(Address, long, long, Guid, Guid, FungibleAssetValue, int, int, int)

```
public OrderDigest(Address sellerAgentAddress, long startedBlockIndex, long
expiredBlockIndex, Guid orderId, Guid tradableId, FungibleAssetValue price, int
combatPoint, int level, int itemId, int itemCount)
```

#### Parameters

`sellerAgentAddress` Address

`startedBlockIndex` [long](#)

`expiredBlockIndex` [long](#)

`orderId` [Guid](#)

`tradableId` [Guid](#)

`price` FungibleAssetValue

`combatPoint` [int](#)

`level` [int](#)

`itemId` [int](#)

`itemCount` [int](#)

## Fields

### CombatPoint

`public readonly int CombatPoint`

### Field Value

[int](#)

### ItemCount

`public readonly int ItemCount`

### Field Value

[int](#)

## ItemId

```
public readonly int ItemId
```

### Field Value

[int](#)

## Level

```
public readonly int Level
```

### Field Value

[int](#)

## Price

```
public readonly FungibleAssetValue Price
```

### Field Value

FungibleAssetValue

## SellerAgentAddress

```
public readonly Address SellerAgentAddress
```

### Field Value

Address

## Methods

## Equals(OrderDigest)

```
protected bool Equals(OrderDigest other)
```

Parameters

other [OrderDigest](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int ↗](#)

A hash code for the current object.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class OrderDigestListState

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderDigestListState
```

## Inheritance

[object](#) ← OrderDigestListState

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### OrderDigestListState(Dictionary)

```
public OrderDigestListState(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### OrderDigestListState(Address)

```
public OrderDigestListState(Address address)
```

#### Parameters

address Address

## Fields

# Address

```
public readonly Address Address
```

## Field Value

Address

# Properties

## OrderDigestList

```
public IReadOnlyList<OrderDigest> OrderDigestList { get; }
```

## Property Value

[IReadOnlyList](#)<[OrderDigest](#)>

# Methods

## Add(OrderDigest)

```
public void Add(OrderDigest orderDigest)
```

## Parameters

orderDigest [OrderDigest](#)

## DeriveAddress(Address)

```
public static Address DeriveAddress(Address avatarAddress)
```

## Parameters

`avatarAddress` Address

Returns

Address

## Equals(OrderDigestListState)

`protected bool Equals(OrderDigestListState other)`

Parameters

`other` [OrderDigestListState](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

`public override bool Equals(object obj)`

Parameters

`obj` [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Remove(Guid)

```
public void Remove(Guid orderId)
```

Parameters

orderId [Guid](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class OrderFactory

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
public static class OrderFactory
```

## Inheritance

[object](#) ← OrderFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

Create(Address, Address, Guid, FungibleAssetValue, Guid, long, ItemSubType, int)

```
public static Order Create(Address agentAddress, Address avatarAddress, Guid  
orderId, FungibleAssetValue price, Guid tradableId, long startedIndex, ItemSubType  
itemSubType, int count)
```

## Parameters

agentAddress Address

avatarAddress Address

orderId [Guid](#)

price FungibleAssetValue

tradableId [Guid](#)

startedIndex [long](#)

itemSubType [ItemSubType](#)

count [int](#)

Returns

[Order](#)

CreateFungibleOrder(Address, Address, Guid, FungibleAssetValue, Guid, long, int, ItemSubType)

```
public static FungibleOrder CreateFungibleOrder(Address sellerAgentAddress, Address sellerAvatarAddress, Guid orderId, FungibleAssetValue price, Guid itemId, long startedBlockIndex, int count, ItemSubType itemSubType)
```

Parameters

sellerAgentAddress Address

sellerAvatarAddress Address

orderId [Guid](#)

price FungibleAssetValue

itemId [Guid](#)

startedBlockIndex [long](#)

count [int](#)

itemSubType [ItemSubType](#)

Returns

[FungibleOrder](#)

CreateNonFungibleOrder(Address, Address, Guid, FungibleAssetValue, Guid, long, ItemSubType)

```
public static NonFungibleOrder CreateNonFungibleOrder(Address sellerAgentAddress,  
Address sellerAvatarAddress, Guid orderId, FungibleAssetValue price, Guid itemId,  
long startedBlockIndex, ItemSubType itemSubType)
```

## Parameters

**sellerAgentAddress** Address

**sellerAvatarAddress** Address

**orderId** [Guid](#)

**price** FungibleAssetValue

**itemId** [Guid](#)

**startedBlockIndex** [long](#)

**itemSubType** [ItemSubType](#)

## Returns

[NonFungibleOrder](#)

## Deserialize(Dictionary)

```
public static Order Deserialize(Dictionary dictionary)
```

## Parameters

**dictionary** Dictionary

## Returns

[Order](#)

# Class OrderReceipt

Namespace: [Lib9c.Model.Order](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderReceipt
```

## Inheritance

[object](#) ← OrderReceipt

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### OrderReceipt(Dictionary)

```
public OrderReceipt(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### OrderReceipt(Guid, Address, Address, long)

```
public OrderReceipt(Guid orderId, Address buyerAgentAddress, Address
buyerAvatarAddress, long transferredBlockIndex)
```

#### Parameters

orderId [Guid](#)

buyerAgentAddress Address

`buyerAvatarAddress` Address

`transferredBlockIndex` [long](#)

## Fields

### BuyerAgentAddress

`public readonly Address BuyerAgentAddress`

Field Value

Address

### BuyerAvatarAddress

`public readonly Address BuyerAvatarAddress`

Field Value

Address

### OrderId

`public readonly Guid OrderId`

Field Value

[Guid](#)

### TransferredBlockIndex

`public readonly long TransferredBlockIndex`

## Field Value

[long](#) ↴

## Methods

### DeriveAddress(Guid)

```
public static Address DeriveAddress(Guid orderId)
```

#### Parameters

orderId [Guid](#) ↴

#### Returns

Address

### Serialize()

```
public IValue Serialize()
```

#### Returns

IValue

# Namespace Nekoyume

## Classes

[ActionObsoleteAttributeExtension](#)

[ActionObsoleteConfig](#)

[Addresses](#)

[CurrencyExtensions](#)

[GameConfig](#)

[GameConfig.MaxEquipmentSlotCount](#)

[GameConfig.RequireCharacterLevel](#)

[GameConfig.RequireClearedStageLevel](#)

[MeadConfig](#)

[PlanetExtension](#)

[StoreUtils](#)

## Structs

[AppProtocolVersionExtra](#)

## Enums

[Planet](#)

# Class ActionObsoleteAttributeExtension

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class ActionObsoleteAttributeExtension
```

## Inheritance

[object](#) ← ActionObsoleteAttributeExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

**IsObsolete(IEnumerable<ActionObsoleteAttribute>, Planet, long)**

```
public static bool IsObsolete(this IEnumerable<ActionObsoleteAttribute> attrs,  
    Planet planet, long blockIndex)
```

### Parameters

attrs [IEnumerable](#)<[ActionObsoleteAttribute](#)>

planet [Planet](#)

blockIndex [long](#)

### Returns

[bool](#)

# Class ActionObsoleteConfig

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class ActionObsoleteConfig
```

## Inheritance

[object](#) ← ActionObsoleteConfig

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### V100080ObsoleteIndex

```
public const long V100080ObsoleteIndex = 2448000
```

Field Value

[long](#)

### V100081ObsoleteIndex

```
public const long V100081ObsoleteIndex = 2550000
```

Field Value

[long](#)

### V100083ObsoleteIndex

```
public const long V1000830bssoleteIndex = 2680000
```

Field Value

[long](#) ↗

## V100086ObsoleteIndex

```
public const long V1000860bssoleteIndex = 2800001
```

Field Value

[long](#) ↗

## V100089ObsoleteIndex

```
public const long V1000890bssoleteIndex = 2908000
```

Field Value

[long](#) ↗

## V100093ObsoleteIndex

```
public const long V1000930bssoleteIndex = 3050000
```

Field Value

[long](#) ↗

## V100095ObsoleteIndex

```
public const long V1000950bssoleteIndex = 3317632
```

Field Value

[long](#) ↗

## V100096ObsoleteIndex

```
public const long V1000960bssoleteIndex = 3317632
```

Field Value

[long](#) ↗

## V100170ObsoleteIndex

```
public const long V1001700bssoleteIndex = 3810000
```

Field Value

[long](#) ↗

## V100190ObsoleteIndex

```
public const long V1001900bssoleteIndex = 4204863
```

Field Value

[long](#) ↗

## V100193ObsoleteIndex

```
public const long V1001930ObsoleteIndex = 3975929
```

Field Value

[long](#) ↗

## V100200ObsoleteIndex

```
public const long V100200ObsoleteIndex = 4246225
```

Field Value

[long](#) ↗

## V100210ObsoleteIndex

```
public const long V100210ObsoleteIndex = 4366622
```

Field Value

[long](#) ↗

## V100220ObsoleteIndex

```
public const long V100220ObsoleteIndex = 4390272
```

Field Value

[long](#) ↗

## V100230ObsoleteIndex

```
public const long V1002300ObsoleteIndex = 4558559
```

Field Value

[long](#) ↗

## V100240ObsoleteIndex

```
public const long V1002400ObsoleteIndex = 4578480
```

Field Value

[long](#) ↗

## V100260ObsoleteIndex

```
public const long V1002600ObsoleteIndex = 4596180
```

Field Value

[long](#) ↗

## V100270ObsoleteIndex

```
public const long V1002700ObsoleteIndex = 4841774
```

Field Value

[long](#) ↗

## V100282ObsoleteIndex

```
public const long V1002820bsoleteIndex = 4835445
```

Field Value

[long](#) ↗

## V100290ObsoleteIndex

```
public const long V1002900bsoleteIndex = 4913153
```

Field Value

[long](#) ↗

## V100300ObsoleteIndex

```
public const long V1003000bsoleteIndex = 5150000
```

Field Value

[long](#) ↗

## V100301ExecutedBlockIndex

```
public const long V100301ExecutedBlockIndex = 5048399
```

Field Value

[long](#) ↗

## V100310ExecutedBlockIndex

```
public const long V100310ExecutedBlockIndex = 5217577
```

Field Value

[long](#) ↗

## V100310ObsoleteIndex

```
public const long V100310ObsoleteIndex = 5300000
```

Field Value

[long](#) ↗

## V100320ObsoleteIndex

```
public const long V100320ObsoleteIndex = 5398001
```

Field Value

[long](#) ↗

## V100340ObsoleteIndex

```
public const long V100340ObsoleteIndex = 5800000
```

Field Value

[long](#) ↗

## V100351ObsoleteIndex

```
public const long V1003510ObsoleteIndex = 5828750
```

Field Value

[long](#) ↗

## V100360ObsoleteIndex

```
public const long V1003600ObsoleteIndex = 6020000
```

Field Value

[long](#) ↗

## V200010ObsoleteIndex

```
public const long V2000100ObsoleteIndex = 6642114
```

Field Value

[long](#) ↗

## V200020AccidentObsoleteIndex

```
public const long V200020AccidentObsoleteIndex = 7000000
```

Field Value

[long](#) ↗

## V200020ObsoleteIndex

```
public const long V2000200bssoleteIndex = 6856587
```

Field Value

[long](#) ↗

## V200030ObsoleteIndex

```
public const long V2000300bssoleteIndex = 7200000
```

Field Value

[long](#) ↗

## V200031ObsoleteIndex

```
public const long V2000310bssoleteIndex = 7206000
```

Field Value

[long](#) ↗

## V200040ObsoleteIndex

```
public const long V2000400bssoleteIndex = 7330000
```

Field Value

[long](#) ↗

## V200060ObsoleteIndex

```
public const long V2000600bssoleteIndex = 7649999
```

Field Value

[long](#) ↗

## V200061ObsoleteIndex

```
public const long V2000610bssoleteIndex = 7660000
```

Field Value

[long](#) ↗

## V200063ObsoleteIndex

```
public const long V2000630bssoleteIndex = 7700000
```

Field Value

[long](#) ↗

## V200070ObsoleteIndex

```
public const long V2000700bssoleteIndex = 7716400
```

Field Value

[long](#) ↗

## V200071ObsoleteIndex

```
public const long V2000710ObsoleteIndex = 7718878
```

Field Value

[long](#) ↗

## V200080ObsoleteIndex

```
public const long V2000800ObsoleteIndex = 7983895
```

Field Value

[long](#) ↗

## V200090ObsoleteIndex

```
public const long V2000900ObsoleteIndex = 8070865
```

Field Value

[long](#) ↗

## V200092ObsoleteIndex

```
public const long V2000920ObsoleteIndex = 8324909
```

Field Value

[long](#) ↗

# Class Addresses

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class Addresses
```

## Inheritance

[object](#) ← Addresses

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### AbstainHistory

An address of an account having [AbstainHistory](#).

```
public static readonly Address AbstainHistory
```

Field Value

Address

### ActionPoint

```
public static readonly Address ActionPoint
```

Field Value

Address

## ActivatedAccount

```
public static readonly Address ActivatedAccount
```

Field Value

Address

## Admin

```
public static readonly Address Admin
```

Field Value

Address

## AdventureBoss

```
public static readonly Address AdventureBoss
```

Field Value

Address

## Agent

```
public static readonly Address Agent
```

Field Value

Address

## Arena

```
public static readonly Address Arena
```

Field Value

Address

## AssetMinters

```
public static readonly Address AssetMinters
```

Field Value

Address

## AuthorizedMiners

```
public static readonly Address AuthorizedMiners
```

Field Value

Address

## Avatar

```
public static readonly Address Avatar
```

Field Value

Address

## Blacksmith

```
public static readonly Address Blacksmith
```

Field Value

Address

## BountyBoard

```
public static readonly Address BountyBoard
```

Field Value

Address

## Collection

```
public static readonly Address Collection
```

Field Value

Address

## CombinationSlot

```
public static readonly Address CombinationSlot
```

Field Value

Address

## CommunityPool

An address for community fund.

```
public static readonly Address CommunityPool
```

Field Value

Address

## Credits

```
public static readonly Address Credits
```

Field Value

Address

## DailyReward

```
public static readonly Address DailyReward
```

Field Value

Address

## EmptyAccountAddress

```
public static readonly Address EmptyAccountAddress
```

Field Value

Address

## EventDungeon

```
public static readonly Address EventDungeon
```

Field Value

Address

## ExploreBoard

```
public static readonly Address ExploreBoard
```

Field Value

Address

## ExplorerList

```
public static readonly Address ExplorerList
```

Field Value

Address

## GameConfig

```
public static readonly Address GameConfig
```

Field Value

Address

## GarageWallet

```
public static readonly Address GarageWallet
```

Field Value

Address

## GasPool

```
public static readonly Address GasPool
```

Field Value

Address

## GoldCurrency

```
public static readonly Address GoldCurrency
```

Field Value

Address

## GoldDistribution

```
public static readonly Address GoldDistribution
```

Field Value

Address

## Guild

An address of an account having [Guild](#).

```
public static readonly Address Guild
```

Field Value

Address

## GuildApplication

An address of an account having `Nekoyume.Model.Guild.GuildApplication`.

```
public static readonly Address GuildApplication
```

Field Value

Address

## GuildBond

An address of an account having [Bond](#).

```
public static readonly Address GuildBond
```

Field Value

Address

## GuildDelegateeMetadata

An address of an account having [DelegateeMetadata](#).

```
public static readonly Address GuildDelegateeMetadata
```

Field Value

Address

## GuildDelegatorMetadata

An address of an account having [DelegatorMetadata](#).

```
public static readonly Address GuildDelegatorMetadata
```

Field Value

Address

## GuildLumpSumRewardsRecord

An address of an account having [LumpSumRewardsRecord](#).

```
public static readonly Address GuildLumpSumRewardsRecord
```

Field Value

Address

## GuildMemberCounter

An address of an account having Bencodex.Types.Integer which means the number of the guild.

```
public static readonly Address GuildMemberCounter
```

Field Value

Address

## GuildParticipant

An address of an account having [GuildParticipant](#)

```
public static readonly Address GuildParticipant
```

Field Value

Address

## GuildRebondGrace

An address of an account having [RebondGrace](#).

```
public static readonly Address GuildRebondGrace
```

Field Value

Address

## GuildRejoinCooldown

An address of an account having [GuildRejoinCooldown](#)

```
public static readonly Address GuildRejoinCooldown
```

Field Value

Address

## GuildUnbondLockIn

An address of an account having [UnbondLockIn](#).

```
public static readonly Address GuildUnbondLockIn
```

Field Value

Address

## GuildUnbondingSet

An address of an account having [UnbondingSet](#).

```
public static readonly Address GuildUnbondingSet
```

Field Value

Address

## Inventory

```
public static readonly Address Inventory
```

Field Value

Address

## Market

```
public static readonly Address Market
```

Field Value

Address

## MaterialCost

```
public static readonly Address MaterialCost
```

Field Value

Address

## Migration

An account address for migration.

```
public static readonly Address Migration
```

Field Value

Address

## MortgagePool

```
public static readonly Address MortgagePool
```

Field Value

Address

## NonValidatorDelegatee

An address for non validator.

```
public static readonly Address NonValidatorDelegatee
```

Field Value

Address

## PendingActivation

```
public static readonly Address PendingActivation
```

Field Value

Address

## QuestList

```
public static readonly Address QuestList
```

### Field Value

Address

## Raid

```
public static readonly Address Raid
```

### Field Value

Address

## Ranking

```
public static readonly Address Ranking
```

### Field Value

Address

## RedeemCode

```
public static readonly Address RedeemCode
```

### Field Value

Address

## Relationship

```
public static readonly Address Relationship
```

Field Value

Address

## RewardPool

An address for reward pool of validators.

```
public static readonly Address RewardPool
```

Field Value

Address

## Rune

```
public static readonly Address Rune
```

Field Value

Address

## RuneState

```
public static readonly Address RuneState
```

Field Value

Address

## Shop

```
public static readonly Address Shop
```

Field Value

Address

## StageRandomBuff

```
public static readonly Address StageRandomBuff
```

Field Value

Address

## SuperCraft

```
public static readonly Address SuperCraft
```

Field Value

Address

## TableSheet

```
public static readonly Address TableSheet
```

Field Value

Address

## UnlockEquipmentRecipe

```
public static readonly Address UnlockEquipmentRecipe
```

Field Value

Address

## UnlockWorld

```
public static readonly Address UnlockWorld
```

Field Value

Address

## ValidatorBond

An address of an account having [Bond](#).

```
public static readonly Address ValidatorBond
```

Field Value

Address

## ValidatorDelegatee

An address of an account having [ValidatorDelegatee](#).

```
public static readonly Address ValidatorDelegatee
```

Field Value

Address

## ValidatorDelegateeMetadata

An address of an account having [DelegateeMetadata](#).

```
public static readonly Address ValidatorDelegateeMetadata
```

Field Value

Address

## ValidatorDelegator

An address of an account having [ValidatorDelegator](#).

```
public static readonly Address ValidatorDelegator
```

Field Value

Address

## ValidatorDelegatorMetadata

An address of an account having [DelegatorMetadata](#).

```
public static readonly Address ValidatorDelegatorMetadata
```

Field Value

Address

## ValidatorList

An address of an account having [ValidatorList](#).

```
public static readonly Address ValidatorList
```

Field Value

Address

## ValidatorLumpSumRewardsRecord

An address of an account having [LumpSumRewardsRecord](#).

```
public static readonly Address ValidatorLumpSumRewardsRecord
```

Field Value

Address

## ValidatorRebondGrace

An address of an account having [RebondGrace](#).

```
public static readonly Address ValidatorRebondGrace
```

Field Value

Address

## ValidatorUnbondLockIn

An address of an account having [UnbondLockIn](#).

```
public static readonly Address ValidatorUnbondLockIn
```

Field Value

Address

## ValidatorUnbondingSet

An address of an account having [UnbondingSet](#).

```
public static readonly Address ValidatorUnbondingSet
```

Field Value

Address

## WeeklyArena

```
public static readonly Address WeeklyArena
```

Field Value

Address

## WorldInformation

```
public static readonly Address WorldInformation
```

Field Value

Address

## Methods

### AdventureSeasonAddress(long)

```
public static Address AdventureSeasonAddress(long season)
```

Parameters

season [long](#)

Returns

Address

## CheckAgentHasPermissionOnBalanceAddr(Address, Address)

```
public static bool CheckAgentHasPermissionOnBalanceAddr(Address agentAddr,  
Address balanceAddr)
```

Parameters

agentAddr Address

balanceAddr Address

Returns

[bool](#)

## CheckAvatarAddrIsContainedInAgent(Address, Address)

```
public static bool CheckAvatarAddrIsContainedInAgent(Address agentAddr,  
Address avatarAddr)
```

Parameters

agentAddr Address

avatarAddr Address

Returns

[bool](#)

## CheckInventoryAddrIsContainedInAgent(Address, Address)

```
public static bool CheckInventoryAddrIsContainedInAgent(Address agentAddr,  
Address inventoryAddr)
```

### Parameters

agentAddr Address

inventoryAddr Address

### Returns

[bool](#)

## GetArenaParticipantAccountAddress(int, int)

Get the account address of an arena participant.

```
public static Address GetArenaParticipantAccountAddress(int championshipId,  
int round)
```

### Parameters

championshipId [int](#)

round [int](#)

### Returns

Address

The account address of an arena participant.

"01000000000000000000000000000000" ~  
"01999999999999999999999999999999".

## GetAvatarAddress(Address, int)

```
public static Address GetAvatarAddress(Address agentAddr, int index)
```

Parameters

agentAddr Address

index [int](#)

Returns

Address

## GetCombinationSlotAddress(Address, int)

```
public static Address GetCombinationSlotAddress(Address avatarAddr, int index)
```

Parameters

avatarAddr Address

index [int](#)

Returns

Address

## GetDailyCrystalCostAddress(int)

```
public static Address GetDailyCrystalCostAddress(int index)
```

Parameters

index [int](#)

Returns

Address

## GetGarageAddress(Address, HashDigest<SHA256>)

```
public static Address GetGarageAddress(Address agentAddr,  
HashDigest<SHA256> fungibleId)
```

### Parameters

agentAddr Address

fungibleId HashDigest<[SHA256](#)>

### Returns

Address

## GetGarageBalanceAddress(Address)

```
public static Address GetGarageBalanceAddress(Address agentAddr)
```

### Parameters

agentAddr Address

### Returns

Address

## GetGuildBanAccountAddress(Address)

Build an Libplanet.Crypto.Address of an Libplanet.Action.State.Account, represented as agentAddress ↛ Bencodex.Types.Boolean, indicates whether the agentAddress is banned.

```
public static Address GetGuildBanAccountAddress(Address guildAddress)
```

### Parameters

**guildAddress** Address

The guild address.

Returns

Address

An account address.

## GetHammerPointStateAddress(Address, int)

```
public static Address GetHammerPointStateAddress(Address avatarAddress,  
int recipeId)
```

Parameters

**avatarAddress** Address

**recipeId** [int](#)

Returns

Address

## GetInventoryAddress(Address, int)

```
public static Address GetInventoryAddress(Address agentAddr, int avatarIndex)
```

Parameters

**agentAddr** Address

**avatarIndex** [int](#)

Returns

Address

## GetItemAddress(Guid)

```
public static Address GetItemAddress(Guid itemId)
```

Parameters

itemId [Guid](#)

Returns

Address

## GetRaiderAddress(Address, int)

```
public static Address GetRaiderAddress(Address avatarAddress, int raidId)
```

Parameters

avatarAddress Address

raidId [int](#)

Returns

Address

## GetRaiderListAddress(int)

```
public static Address GetRaiderListAddress(int raidId)
```

Parameters

raidId [int](#)

Returns

Address

## GetSheetAddress(string)

```
public static Address GetSheetAddress(string sheetName)
```

Parameters

sheetName [string](#)

Returns

Address

## GetSheetAddress<T>()

```
public static Address GetSheetAddress<T>() where T : ISheet
```

Returns

Address

Type Parameters

T

## GetSkillStateAddressFromAvatarAddress(Address)

```
public static Address GetSkillStateAddressFromAvatarAddress(Address avatarAddress)
```

Parameters

avatarAddress Address

Returns

Address

## GetWeeklyCrystalCostAddress(int)

```
public static Address GetWeeklyCrystalCostAddress(int index)
```

Parameters

index [int](#)

Returns

Address

## GetWorldBossAddress(int)

```
public static Address GetWorldBossAddress(int raidId)
```

Parameters

raidId [int](#)

Returns

Address

## GetWorldBossKillRewardRecordAddress(Address, int)

```
public static Address GetWorldBossKillRewardRecordAddress(Address avatarAddress,  
int raidId)
```

Parameters

avatarAddress Address

raidId [int](#)

Returns

Address

## IsArenaParticipantAccountAddress(Address)

Check if the given address is within the range of arena participant account addresses.

"01000000000000000000000000000000" ~

"01999999999999999999999999999999".

```
public static bool IsArenaParticipantAccountAddress(Address address)
```

Parameters

**address** Address

Returns

bool ↗

# Struct AppProtocolVersionExtra

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public readonly struct AppProtocolVersionExtra
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### AppProtocolVersionExtra(Dictionary)

```
public AppProtocolVersionExtra(Dictionary dictionary)
```

#### Parameters

**dictionary** Dictionary

### AppProtocolVersionExtra(string, string, DateTimeOffset)

```
public AppProtocolVersionExtra(string macosBinaryUrl, string windowsBinaryUrl,  
DateTimeOffset timestamp)
```

#### Parameters

**macosBinaryUrl** [string](#)

**windowsBinaryUrl** [string](#)

**timestamp** [DateTimeOffset](#)

## Fields

### MacOSBinaryUrl

```
public readonly string MacOSBinaryUrl
```

#### Field Value

[string](#) ↗

### Timestamp

```
public readonly DateTimeOffset Timestamp
```

#### Field Value

[DateTimeOffset](#) ↗

### WindowsBinaryUrl

```
public readonly string WindowsBinaryUrl
```

#### Field Value

[string](#) ↗

## Methods

### Serialize()

```
public IValue Serialize()
```

#### Returns



# Class CurrencyExtensions

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class CurrencyExtensions
```

## Inheritance

[object](#) ← CurrencyExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Deserialize(Dictionary)

```
public static Currency Deserialize(Dictionary serialized)
```

#### Parameters

serialized Dictionary

#### Returns

Currency

### Serialize(Currency)

```
public static Dictionary Serialize(this Currency currency)
```

#### Parameters

**currency** Currency

Returns

Dictionary

# Class GameConfig

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class GameConfig
```

## Inheritance

[object](#) ← GameConfig

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### ArenaChallengeCountMax

```
public const int ArenaChallengeCountMax = 5
```

Field Value

[int](#)

### ArenaScoreDefault

```
public const int ArenaScoreDefault = 1000
```

Field Value

[int](#)

### AvatarNickNamePattern

```
public const string AvatarNickNamePattern = "^[0-9a-zA-Z]{2,20}$"
```

Field Value

[string](#)

## DefaultAttackId

```
public const int DefaultAttackId = 1000000
```

Field Value

[int](#)

## DefaultAvatarArmorId

```
public const int DefaultAvatarArmorId = 102000000
```

Field Value

[int](#)

## DefaultAvatarCharacterId

```
public const int DefaultAvatarCharacterId = 100010
```

Field Value

[int](#)

## DefaultAvatarWeaponId

```
public const int DefaultAvatarWeaponId = 10100000
```

Field Value

[int](#)

## EnhanceEquipmentCostAP

```
public const int EnhanceEquipmentCostAP = 0
```

Field Value

[int](#)

## IsEditor

```
public static readonly bool IsEditor
```

Field Value

[bool](#)

## MaximumProbability

```
public const int MaximumProbability = 10000
```

Field Value

[int](#)

## MimisbrunnrStartStageId

```
public const int MimirunnrStartStageId = 10000001
```

Field Value

[int](#)

## MimirunnrWorldId

```
public const int MimirunnrWorldId = 10001
```

Field Value

[int](#)

## RequiredAppraiseBlock

```
[Obsolete("Use GameConfigState.RequiredAppraiseBlock")]
public const int RequiredAppraiseBlock = 50
```

Field Value

[int](#)

## SlotCount

```
public const int SlotCount = 3
```

Field Value

[int](#)

# Class GameConfig.MaxEquipmentSlotCount

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class GameConfig.MaxEquipmentSlotCount
```

## Inheritance

[object](#) ← GameConfig.MaxEquipmentSlotCount

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Armor

```
public const int Armor = 1
```

#### Field Value

[int](#)

### Aura

```
public const int Aura = 1
```

#### Field Value

[int](#)

## Belt

```
public const int Belt = 1
```

Field Value

[int ↗](#)

## Grimoire

```
public const int Grimoire = 1
```

Field Value

[int ↗](#)

## Necklace

```
public const int Necklace = 1
```

Field Value

[int ↗](#)

## Ring

```
public const int Ring = 2
```

Field Value

[int ↗](#)

## Weapon

```
public const int Weapon = 1
```

Field Value

[int](#) ↗

# Class GameConfig.RequireCharacterLevel

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class GameConfig.RequireCharacterLevel
```

## Inheritance

[object](#) ← GameConfig.RequireCharacterLevel

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### CharacterConsumableSlot1

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_ConsumableSlot1")]
public const int CharacterConsumableSlot1 = 1
```

Field Value

[int](#)

### CharacterConsumableSlot2

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_ConsumableSlot2")]
public const int CharacterConsumableSlot2 = 35
```

Field Value

[int](#)

## CharacterConsumableSlot3

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_ConsumableSlot3")]
public const int CharacterConsumableSlot3 = 100
```

Field Value

[int](#)

## CharacterConsumableSlot4

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_ConsumableSlot4")]
public const int CharacterConsumableSlot4 = 200
```

Field Value

[int](#)

## CharacterConsumableSlot5

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_ConsumableSlot5")]
public const int CharacterConsumableSlot5 = 350
```

Field Value

[int](#)

## CharacterEarCostumeSlot

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EarCostumeSlot")]
public const int CharacterEarCostumeSlot = 2
```

Field Value

[int](#)

## CharacterEquipmentSlotArmor

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotArmor")]
public const int CharacterEquipmentSlotArmor = 3
```

Field Value

[int](#)

## CharacterEquipmentSlotAura

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotAura")]
public const int CharacterEquipmentSlotAura = 1
```

Field Value

[int](#)

## CharacterEquipmentSlotBelt

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotBelt")]
public const int CharacterEquipmentSlotBelt = 5
```

Field Value

[int](#)

## CharacterEquipmentSlotNecklace

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotNecklace")]
public const int CharacterEquipmentSlotNecklace = 8
```

Field Value

[int](#)

## CharacterEquipmentSlotRing1

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotRing1")]
public const int CharacterEquipmentSlotRing1 = 13
```

Field Value

[int](#)

## CharacterEquipmentSlotRing2

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotRing2")]
public const int CharacterEquipmentSlotRing2 = 46
```

Field Value

[int](#)

## CharacterEquipmentSlotWeapon

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EquipmentSlotWeapon")]
public const int CharacterEquipmentSlotWeapon = 1
```

Field Value

[int](#)

## CharacterEyeCostumeSlot

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_EyeCostumeSlot")]
public const int CharacterEyeCostumeSlot = 2
```

Field Value

[int](#)

## CharacterFullCostumeSlot

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_FullCostumeSlot")]
public const int CharacterFullCostumeSlot = 2
```

Field Value

[int](#)

## CharacterHairCostumeSlot

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_HairCostumeSlot")]
public const int CharacterHairCostumeSlot = 2
```

Field Value

[int](#)

## CharacterTailCostumeSlot

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_TailCostumeSlot")]
public const int CharacterTailCostumeSlot = 2
```

Field Value

[int](#)

## CharacterTitleSlot

```
[Obsolete("Use GameConfigState.RequireCharacterLevel_TitleSlot")]

```

```
public const int CharacterTitleSlot = 1
```

Field Value

[int](#) ↗

# Class GameConfig.RequireClearedStageLevel

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class GameConfig.RequireClearedStageLevel
```

## Inheritance

[object](#) ← GameConfig.RequireClearedStageLevel

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### ActionsInMimisbrunnr

```
[Obsolete("Not used anymore since v200092")]
public const int ActionsInMimisbrunnr = 100
```

Field Value

[int](#)

### ActionsInRaid

```
[Obsolete("Not used anymore since v200092")]
public const int ActionsInRaid = 50
```

Field Value

[int](#)

## ActionsInRankingBoard

```
[Obsolete("Not used anymore since v200092")]
public const int ActionsInRankingBoard = 25
```

Field Value

[int](#)

## ActionsInShop

```
[Obsolete("Not used anymore since v200092")]
public const int ActionsInShop = 17
```

Field Value

[int](#)

## CombinationConsumableAction

```
[Obsolete("Not used anymore since v200092")]
public const int CombinationConsumableAction = 6
```

Field Value

[int](#)

## CombinationEquipmentAction

```
[Obsolete("Not used anymore since v200092")]
public const int CombinationEquipmentAction = 3
```

Field Value

[int](#)

## ItemEnhancementAction

```
[Obsolete("Not used anymore since v200092")]
public const int ItemEnhancementAction = 9
```

Field Value

[int](#)

# Class MeadConfig

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class MeadConfig
```

## Inheritance

[object](#) ← MeadConfig

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### PatronAddress

```
public static readonly Address PatronAddress
```

#### Field Value

Address

# Enum Planet

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public enum Planet : byte
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Heimdall = 1

HeimdallInternal = 4

Idun = 2

Odin = 0

OdinInternal = 3

# Class PlanetExtension

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class PlanetExtension
```

## Inheritance

[object](#) ← PlanetExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### DeterminePlanet(ITransaction)

```
public static Planet? DeterminePlanet(this ITransaction tx)
```

#### Parameters

[tx](#) ITransaction

#### Returns

[Planet?](#)

# Class StoreUtils

Namespace: [Nekoyume](#)

Assembly: Lib9c.dll

```
public static class StoreUtils
```

## Inheritance

[object](#) ← StoreUtils

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### BackupNameFormat

```
public const string BackupNameFormat = "{0}_{1:yyyyMMdd_HHmmss}"
```

Field Value

[string](#)

### BackupNamePattern

```
public const string BackupNamePattern = "{0}_*"
```

Field Value

[string](#)

## Methods

## ClearBackupStores(string)

```
public static void ClearBackupStores(string storePath)
```

### Parameters

storePath [string](#)

## ResetStore(string)

```
public static void ResetStore(string storePath)
```

### Parameters

storePath [string](#)

# Namespace Nekoyume.Action

## Classes

[ActionBase](#)

[ActionContextExtensions](#)

[ActionObsoleteAttribute](#)

An attribute on an Libplanet.Action.IAction to indicate that such Libplanet.Action.IAction is obsoleted after a certain index.

Due to a bug introduced earlier and to keep backward compatibility, an Libplanet.Action.IAction with this attribute is obsoleted, i.e. cannot be included, starting from [Obsolete Index + 2](#).

[ActionObsoletedException](#)

[ActionPointExceededException](#)

[ActionUnavailableException](#)

[ActivateCollection](#)

[ActivatedAccountsDoesNotExistsException](#)

[ActivationException](#)

[AddRedeemCode](#)

[AddressExtension](#)

[AdminPermissionException](#)

[AgentStateNotContainsAvatarAddressException](#)

[AlreadyActivatedException](#)

[AlreadyContractedException](#)

[AlreadyReceivedException](#)

[AlreadyRecipeUnlockedException](#)

[AlreadyWorldUnlockedException](#)

[AppraiseBlockNotReachedException](#)

[ApprovePledge](#)

[ArenaNotEndedException](#)

[AssetInfo](#)

[AttachmentActionResult](#)

[AuraSummon](#)

[AvatarIndexAlreadyUsedException](#)

[AvatarIndexOutOfRangeException](#)

[BalanceDoesNotExistException](#)

[BattleArena](#)

Introduce at <https://github.com/planetarium/lib9c/pull/2229> Changed at <https://github.com/planetarium/lib9c/pull/2242>

[BurnAsset](#)

[Buy](#)

Updated at <https://github.com/planetarium/lib9c/pull/1164>

[Buy7](#)

[Buy7.BuyerMultipleResult](#)

[Buy7.BuyerResult](#)

[Buy7.PurchaseResult](#)

[Buy7.SellerMultipleResult](#)

[Buy7.SellerResult](#)

[BuyMultiple](#)

Introduced at <https://github.com/planetarium/lib9c/pull/331> Updated at many pull requests Obsoleted at <https://github.com/planetarium/lib9c/pull/487> Updated at many pull requests Updated at <https://github.com/planetarium/lib9c/pull/957>

[BuyMultiple.BuyerResult](#)

[BuyMultiple.PurchaseInfo](#)

[BuyMultiple.PurchaseResult](#)

[BuyMultiple.SellerResult](#)

[BuyProduct](#)

[CancelProductRegistration](#)

[ChargeActionPoint](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/430> Updated at <https://github.com/planetarium/lib9c/pull/474> Updated at <https://github.com/planetarium/lib9c/pull/602> Updated at <https://github.com/planetarium/lib9c/pull/861> Updated at <https://github.com/planetarium/lib9c/pull/957>

[ClaimItems](#)

[ClaimRaidReward](#)

[ClaimStakeReward](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2097>

[ClaimStakeReward2](#)

[ClaimStakeReward8](#)

[ClaimWordBossKillReward](#)

[CombinationConsumable](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

[CombinationConsumable5](#)

[CombinationConsumable5.ResultModel](#)

[CombinationEquipment](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

[CombinationEquipment5](#)

[CombinationSlotResultNullException](#)

[CombinationSlotUnlockException](#)

[ConsumableSlotOutOfRangeException](#)

[ConsumableSlotUnlockException](#)

[CostumeSlotUnlockException](#)

## [CreateAvatar](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

## [CreatePendingActivation](#)

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at <https://github.com/planetarium/lib9c/pull/957>

## [CreatePendingActivations](#)

Updated at <https://github.com/planetarium/lib9c/pull/746> Updated at <https://github.com/planetarium/lib9c/pull/957>

## [CreatePledge](#)

### [DailyReward](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/1828> Updated at <https://github.com/planetarium/lib9c/pull/1828>

### [DailyReward2](#)

### [DailyReward2.DailyRewardResult](#)

### [DuplicateCostumeException](#)

### [DuplicateEquipmentException](#)

### [DuplicateMaterialException](#)

### [DuplicateOrderIdException](#)

### [EndPledge](#)

### [EquipmentLevelExceededException](#)

### [EquipmentSlotUnlockException](#)

### [EventConsumableItemCrafts](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

### [EventDungeonBattle](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

### [EventMaterialItemCrafts](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

### [FailedLoadSheetException](#)

[FailedLoadStateException](#)

[FavProductInfo](#)

[GameAction](#)

[GoldDistribution](#)

[Grinding](#)

[HackAndSlash](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

[HackAndSlashRandomBuff](#)

Created at <https://github.com/planetarium/lib9c/pull/1031>

[HackAndSlashSweep](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

[InitializeStates](#)

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at NCG distribution(7e4515b6e14cc5d6eb716d5ebb587ab04b4246f9) Updated at

<https://github.com/planetarium/lib9c/pull/19> Updated at

<https://github.com/planetarium/lib9c/pull/36> Updated at

<https://github.com/planetarium/lib9c/pull/42> Updated at

<https://github.com/planetarium/lib9c/pull/55> Updated at

<https://github.com/planetarium/lib9c/pull/57> Updated at

<https://github.com/planetarium/lib9c/pull/60> Updated at

<https://github.com/planetarium/lib9c/pull/102> Updated at

<https://github.com/planetarium/lib9c/pull/128> Updated at

<https://github.com/planetarium/lib9c/pull/167> Updated at

<https://github.com/planetarium/lib9c/pull/422> Updated at

<https://github.com/planetarium/lib9c/pull/747> Updated at

<https://github.com/planetarium/lib9c/pull/798> Updated at

<https://github.com/planetarium/lib9c/pull/957>

[InvalidAddressException](#)

[InvalidClaimException](#)

[InvalidCurrencyException](#)

[InvalidElementalException](#)

[InvalidEquipmentException](#)

[InvalidItemCountException](#)

[InvalidItemTypeException](#)

[InvalidLevelException](#)

[InvalidMaterialException](#)

[InvalidMinterException](#)

[InvalidMonsterCollectionRoundException](#)

[InvalidNamePatternException](#)

[InvalidPriceException](#)

[InvalidProductTypeException](#)

[InvalidRecipieIdException](#)

[InvalidRepeatPlayException](#)

[InvalidShopItemException](#)

[InvalidSignatureException](#)

[InvalidSlotIndexException](#)

[InvalidStageException](#)

[InvalidTradableIdException](#)

[InvalidTradableItemException](#)

[InvalidTransferCurrencyException](#)

[InvalidTransferMinterException](#)

[InvalidTransferRecipientException](#)

[InvalidTransferSignerException](#)

[InvalidTransferUnactivatedRecipientException](#)

[InvalidWorldException](#)

[IssueToken](#)

[IssueTokensFromGarage](#)

[ItemDoesNotExistException](#)

[ItemEnhancement](#)

Updated at <https://github.com/planetarium/lib9c/pull/2195>

[ItemEnhancement10](#)

Updated at <https://github.com/planetarium/lib9c/pull/1176>

[ItemEnhancement10.ResultModel](#)

[ItemEnhancement11](#)

Updated at <https://github.com/planetarium/lib9c/pull/1164>

[ItemEnhancement11.ResultModel](#)

[ItemEnhancement12](#)

Updated at <https://github.com/planetarium/lib9c/pull/2068>

[ItemEnhancement12.ResultModel](#)

[ItemEnhancement13](#)

Updated at <https://github.com/planetarium/lib9c/pull/2068>

[ItemEnhancement13.ResultModel](#)

[ItemEnhancement7](#)

[ItemEnhancement7.ResultModel](#)

[ItemEnhancement9](#)

[ItemEnhancement9.ResultModel](#)

[ItemProductInfo](#)

[JoinArena](#)

Introduced at <https://github.com/planetarium/lib9c/pull/2195>

[JoinArena1](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1495>

[JoinArena3](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1663>

[MemoLengthOverflowException](#)

[MigrateAgentAvatar](#)

[MigrateFee](#)

[MigrateMonsterCollection](#)

An action to claim remained monster collection rewards and to migrate [MonsterCollectionState](#) into [LegacyStakeState](#) without cancellation, to keep its staked period.

[MigrationActivatedAccountsState](#)

[MintAssets](#)

[MonsterCollectionExistingClaimableException](#)

[MonsterCollectionExistingException](#)

An exception thrown when there is unexpected monster collection.

[MonsterCollectionExpiredException](#)

[MonsterCollectionLevelException](#)

[NotEnoughActionPointException](#)

[NotEnoughAvatarLevelException](#)

[NotEnoughClearedStageLevelException](#)

[NotEnoughCombatPointException](#)

[NotEnoughFungibleAssetValueException](#)

[NotEnoughHammerPointException](#)

[NotEnoughMaterialException](#)

[NotEnoughMedalException](#)

[NotEnoughRankException](#)

[NotEnoughStarException](#)

[NotEnoughWeeklyArenaChallengeCountException](#)

[OrderIdDoesNotExistException](#)

[PatchTableSheet](#)

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at  
<https://github.com/planetarium/lib9c/pull/42> Updated at  
<https://github.com/planetarium/lib9c/pull/101> Updated at  
<https://github.com/planetarium/lib9c/pull/287> Updated at  
<https://github.com/planetarium/lib9c/pull/315> Updated at  
<https://github.com/planetarium/lib9c/pull/957> Updated at  
<https://github.com/planetarium/lib9c/pull/1560>

[PendingActivationDoesNotExistsException](#)

[PermissionDeniedException](#)

[PetEnhancement](#)

[PetIsLockedException](#)

[PlayCountIsZeroException](#)

[PolicyExpiredException](#)

[PrepareRewardAssets](#)

[ProductNotFoundException](#)

[PurchaseInfo](#)

[PurchaseInfo0](#)

[Raid](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

[RankingBattle](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/941> Updated at  
<https://github.com/planetarium/lib9c/pull/1135>

[RankingBattle11](#)

Updated at <https://github.com/planetarium/lib9c/pull/1176>

[RankingExceededException](#)

[RapidCombination](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

[RapidCombination0](#)

[RapidCombination0.ResultModel](#)

[RapidCombination5](#)

[RapidCombination5.ResultModel](#)

[ReRegisterProduct](#)

[RedeemCode](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/602> Updated at  
<https://github.com/planetarium/lib9c/pull/861> Updated at  
<https://github.com/planetarium/lib9c/pull/957>

[RedeemCode2](#)

[RegisterInfo](#)

[RegisterProduct](#)

[RegisterProduct0](#)

[RenewAdminState](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/805> Updated at  
<https://github.com/planetarium/lib9c/pull/815> Updated at  
<https://github.com/planetarium/lib9c/pull/957>

[RequestPledge](#)

[RequiredBlockIndexException](#)

[RequiredBlockIntervalException](#)

[RetrieveAvatarAssets](#)

[RewardGold](#)

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at  
many pull requests Updated at <https://github.com/planetarium/lib9c/pull/1135>

[RuneEnhancement](#)

[RuneSlotInfo](#)

[RuneSummon](#)

[SecureMiningReward](#)

[Sell](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/1640>

[Sell6](#)

[SellCancellation](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/602> Updated at  
<https://github.com/planetarium/lib9c/pull/609> Updated at  
<https://github.com/planetarium/lib9c/pull/620> Updated at  
<https://github.com/planetarium/lib9c/pull/861> Updated at  
<https://github.com/planetarium/lib9c/pull/957>

[SellCancellation.Result](#)

[ShopItemExpiredException](#)

[SlotAlreadyUnlockedException](#)

[StageNotClearedException](#)

[Stake](#)

[Stake2](#)

[StakeExistingClaimableException](#)

[TotalSupplyDoesNotExistException](#)

[TransferAsset](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/2143> Updated at  
<https://github.com/planetarium/lib9c/pull/2143>

[TransferAsset3](#)

[TransferAssets](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/636> Updated at  
<https://github.com/planetarium/lib9c/pull/957>

[UnlockCombinationSlot](#)

[UnlockEquipmentRecipe](#)

[UnlockRuneSlot](#)

[UnlockWorld](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/1309>

[UpdateSell](#)

Hard forked at <https://github.com/planetarium/lib9c/pull/1640>

[UpdateSellInfo](#)

[UsageLimitExceedException](#)

[ValidatorSetOperate](#)

[ValidatorSetOperatorTypeExtensions](#)

[WeeklyArenaStateAlreadyEndedException](#)

[WeeklyArenaStateNotContainsAvatarAddressException](#)

## Structs

[IssueTokensFromGarage.Spec](#)

[MintAssets.MintSpec](#)

[RuneEnhancement.LevelUpResult](#)

## Interfaces

[IBuy0](#)

Common interface used before [IBuy5](#).

[IBuy5](#)

Common interface used after [IBuy5](#).

[IClaimItems](#)

[IClaimStakeReward](#)

[IHackAndSlashV10](#)

[IProductInfo](#)

[IPurchaseInfo](#)

[IRegisterInfo](#)

[ITransferAsset](#)

[ITransferAssets](#)

## Enums

[ItemEnhancement10.EnhancementResult](#)

[ItemEnhancement11.EnhancementResult](#)

[ItemEnhancement12.EnhancementResult](#)

[ItemEnhancement13.EnhancementResult](#)

[ItemEnhancement9.EnhancementResult](#)

[ShopErrorType](#)

[ValidatorSetOperatorType](#)

# Class ActionBase

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ActionBase : IAction
```

## Inheritance

[object](#) ← ActionBase

## Implements

IAction

## Derived

[ApprovePledge](#), [BurnAsset](#), [CreatePendingActivation](#), [CreatePendingActivations](#),  
[CreatePledge](#), [CustomEquipmentCraft](#), [EndPledge](#), [GameAction](#), [BanGuildMember](#),  
[ClaimGuildReward](#), [ClaimReward](#), [JoinGuild](#), [MakeGuild](#), [MigrateDelegation](#),  
[MigrateDelegationHeight](#), [MigratePlanetariumGuild](#), [MoveGuild](#), [QuitGuild](#), [RemoveGuild](#),  
[UnbanGuildMember](#), [IssueToken](#), [IssueTokensFromGarage](#), [MigrateAgentAvatar](#), [MigrateFee](#),  
[MigrateMonsterCollection](#), [MintAssets](#), [PrepareRewardAssets](#), [RequestPledge](#),  
[RetrieveAvatarAssets](#), [RewardGold](#), [SecureMiningReward](#), [TransferAsset](#), [TransferAssets](#),  
[AllocateGuildReward](#), [AllocateReward](#), [ClaimValidatorRewardSelf](#), [DelegateValidator](#),  
[Mortgage](#), [PromoteValidator](#), [RecordProposer](#), [Refund](#), [ReleaseValidatorUnbondings](#),  
[Reward](#), [SetValidatorCommission](#), [SlashValidator](#), [UndelegateValidator](#), [UnjailValidator](#),  
[UpdateValidators](#), [ValidatorSetOperate](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public abstract IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### CheckObsolete(long, IActionContext)

```
protected void CheckObsolete(long obsoleteIndex, IActionContext ctx)
```

#### Parameters

obsoleteIndex [long](#)

ctx IActionContext

### CheckPermission(IActionContext)

```
protected void CheckPermission(IActionContext ctx)
```

#### Parameters

ctx IActionContext

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public abstract IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetSignerAndOtherAddressesHex(IActionContext, params Address[])

returns "[Signer Address, AvatarState Address, ...]"

```
protected string GetSignerAndOtherAddressesHex(IActionContext ctx, params Address[] addresses)
```

### Parameters

`ctx` IActionContext

`addresses` Address[]

### Returns

[string](#)

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public abstract void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

## TryGetAdminState(IActionContext, out AdminState)

```
protected bool TryGetAdminState(IActionContext ctx, out AdminState state)
```

## Parameters

**ctx** IActionContext

**state** [AdminState](#)

## Returns

[bool](#) ↗

# Class ActionContextExtensions

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public static class ActionContextExtensions
```

## Inheritance

[object](#) ← ActionContextExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### IsMainNet(IActionContext)

```
public static bool IsMainNet(this IActionContext context)
```

#### Parameters

context IActionContext

#### Returns

[bool](#)

### Since(IActionContext, long)

```
public static bool Since(this IActionContext context, long blockIndex)
```

#### Parameters

context IActionContext

blockIndex [long](#)

Returns

[bool](#)

# Class ActionObsoleteAttribute

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

An attribute on an Libplanet.Action.IAction to indicate that such Libplanet.Action.IAction is obsoleted after a certain index.

Due to a bug introduced earlier and to keep backward compatibility, an Libplanet.Action.IAction with this attribute is obsoleted, i.e. cannot be included, starting from [ObsoleteIndex + 2](#).

```
[AttributeUsage(AttributeTargets.Class, AllowMultiple = true)]
public class ActionObsoleteAttribute : Attribute
```

## Inheritance

[object](#) ← [Attribute](#) ← ActionObsoleteAttribute

## Inherited Members

[Attribute.Equals\(object\)](#) , [Attribute.GetCustomAttribute\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Assembly, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type\)](#) ,  
[Attribute.GetCustomAttribute\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttribute\(ParameterInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Assembly, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, bool\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(MemberInfo, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module\)](#) , [Attribute.GetCustomAttributes\(Module, bool\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type\)](#) ,  
[Attribute.GetCustomAttributes\(Module, Type, bool\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, bool\)](#) ,

[Attribute.GetCustomAttributes\(ParameterInfo, Type\)](#) ,  
[Attribute.GetCustomAttributes\(ParameterInfo, Type, bool\)](#) , [Attribute.GetHashCode\(\)](#) ,  
[Attribute.IsDefaultAttribute\(\)](#) , [Attribute.IsDefined\(Assembly, Type\)](#) ,  
[Attribute.IsDefined\(Assembly, Type, bool\)](#) , [Attribute.IsDefined\(MemberInfo, Type\)](#) ,  
[Attribute.IsDefined\(MemberInfo, Type, bool\)](#) , [Attribute.IsDefined\(Module, Type\)](#) ,  
[Attribute.IsDefined\(Module, Type, bool\)](#) , [Attribute.IsDefined\(ParameterInfo, Type\)](#) ,  
[Attribute.IsDefined\(ParameterInfo, Type, bool\)](#) , [Attribute.Match\(object\)](#) ,  
[Attribute.TypeId](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ActionObsoleteAttribute(Planet, long)

```
public ActionObsoleteAttribute(Planet planet, long obsoleteIndex)
```

#### Parameters

planet [Planet](#)

obsoleteIndex [long](#)

### ActionObsoleteAttribute(long)

```
public ActionObsoleteAttribute(long obsoleteIndex)
```

#### Parameters

obsoleteIndex [long](#)

## Fields

### AffectedPlanet

```
public readonly Planet? AffectedPlanet
```

Field Value

[Planet?](#)

## ObsoleteIndex

`public readonly long ObsoleteIndex`

Field Value

[long](#) ↗

# Class ActionObsoleteException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActionObsoleteException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← ActionObsoleteException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ActionObsoleteException()

```
public ActionObsoleteException()
```

### ActionObsoleteException(SerializationInfo, StreamingContext)

```
protected ActionObsoleteException(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ActionObsoleteException(string)

public ActionObsoleteException([string](#) s)

Parameters

s [string](#)

# Class ActionPointExceededException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActionPointExceededException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← ActionPointExceededException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ActionPointExceededException()

```
public ActionPointExceededException()
```

### ActionPointExceededException(SerializationInfo, StreamingContext)

```
protected ActionPointExceededException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ActionPointExceededException(string)

public ActionPointExceededException(string s)

Parameters

s [string](#)

# Class ActionUnavailableException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActionUnavailableException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← ActionUnavailableException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ActionUnavailableException()

```
public ActionUnavailableException()
```

### ActionUnavailableException(SerializationInfo, StreamingContext)

```
protected ActionUnavailableException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ActionUnavailableException(string)

public ActionUnavailableException(string s)

Parameters

s [string](#)

# Class ActivateCollection

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("activate_collection")]
public class ActivateCollection : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ActivateCollection

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### CollectionData

```
public List<(int collectionId, List<ICollectionMaterial> materials)> CollectionData
```

## Field Value

[List](#)<(int [collectionId](#), [List](#)<ICollectionMaterial> [materials](#))>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the [context](#) object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

[context](#) IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class ActivatedAccountsDoesNotExistsException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActivatedAccountsDoesNotExistsException : ActivationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [ActivationException](#) ← ActivatedAccountsDoesNotExistsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ActivatedAccountsDoesNotExistsException()

```
public ActivatedAccountsDoesNotExistsException()
```

### ActivatedAccountsDoesNotExistsException(SerializationInfo, StreamingContext)

```
public ActivatedAccountsDoesNotExistsException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

# Class ActivationException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public abstract class ActivationException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ActivationException

## Implements

[ISerializable](#)

## Derived

[ActivatedAccountsDoesNotExistsException](#), [InvalidSignatureException](#),  
[PendingActivationDoesNotExistsException](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ActivationException()

```
protected ActivationException()
```

### ActivationException(SerializationInfo, StreamingContext)

```
protected ActivationException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

# Class AddRedeemCode

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("add_redeem_code")]
public class AddRedeemCode : GameAction, IAction, IAddRedeemCodeV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← AddRedeemCode

## Implements

IAction, IAddRedeemCodeV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### redeemCsv

```
public string redeemCsv
```

## Field Value

[string](#)

## Properties

# PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class AddressExtension

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public static class AddressExtension
```

## Inheritance

[object](#) ← AddressExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Derive(Address, byte[])

```
public static Address Derive(this Address address, byte[] key)
```

#### Parameters

address Address

key [byte](#)[]

#### Returns

Address

### Derive(Address, string)

```
public static Address Derive(this Address address, string key)
```

Parameters

address Address

key [string](#)

Returns

Address

## GetPledgeAddress(Address)

```
public static Address GetPledgeAddress(this Address address)
```

Parameters

address Address

Returns

Address

# Class AdminPermissionException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class AdminPermissionException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AdminPermissionException

## Implements

[ISerializable](#)

## Derived

[PermissionDeniedException](#), [PolicyExpiredException](#)

## Inherited Members

[Exception.GetBaseException\(\)](#), [Exception.GetType\(\)](#), [Exception.ToString\(\)](#),  
[Exception.Data](#), [Exception.HelpLink](#), [Exception.HResult](#), [Exception.InnerException](#),  
[Exception.Message](#), [Exception.Source](#), [Exception.StackTrace](#), [Exception.TargetSite](#),  
[Exception.SerializeObjectState](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### AdminPermissionException(AdminState)

```
public AdminPermissionException(AdminState policy)
```

## Parameters

policy [AdminState](#)

# AdminPermissionException(SerializationInfo, StreamingContext)

```
protected AdminPermissionException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Policy

```
public AdminState Policy { get; }
```

## Property Value

[AdminState](#)

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

`context StreamingContext`

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

[ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class AgentStateNotContainsAvatarAddressException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AgentStateNotContainsAvatarAddressException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AgentStateNotContainsAvatarAddressException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AgentStateNotContainsAvatarAddressException(SerializationInfo, StreamingContext)

```
public AgentStateNotContainsAvatarAddressException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AgentStateNotContainsAvatarAddressException(string)

```
public AgentStateNotContainsAvatarAddressException(string message)
```

### Parameters

message [string](#)

# Class AlreadyActivatedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyActivatedException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
AlreadyActivatedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyActivatedException(SerializationInfo, StreamingContext)

```
protected AlreadyActivatedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyActivatedException(string)

```
public AlreadyActivatedException(string s)
```

### Parameters

s [string](#)

# Class AlreadyContractedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyContractedException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
AlreadyContractedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyContractedException(SerializationInfo, StreamingContext)

```
protected AlreadyContractedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyContractedException(string)

```
public AlreadyContractedException(string s)
```

### Parameters

s [string](#)

# Class AlreadyReceivedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyReceivedException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
AlreadyReceivedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyReceivedException(SerializationInfo, StreamingContext)

```
protected AlreadyReceivedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyReceivedException(string)

```
public AlreadyReceivedException(string s)
```

### Parameters

s [string](#)

# Class AlreadyRecipeUnlockedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyRecipeUnlockedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AlreadyRecipeUnlockedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyRecipeUnlockedException()

```
public AlreadyRecipeUnlockedException()
```

### AlreadyRecipeUnlockedException(SerializationInfo, StreamingContext)

```
protected AlreadyRecipeUnlockedException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyRecipeUnlockedException(string)

public AlreadyRecipeUnlockedException(string msg)

Parameters

msg [string](#)

# Class AlreadyWorldUnlockedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyWorldUnlockedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AlreadyWorldUnlockedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyWorldUnlockedException()

```
public AlreadyWorldUnlockedException()
```

### AlreadyWorldUnlockedException(SerializationInfo, StreamingContext)

```
protected AlreadyWorldUnlockedException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyWorldUnlockedException(string)

public AlreadyWorldUnlockedException(string msg)

Parameters

msg [string](#)

# Class AppraiseBlockNotReachedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AppraiseBlockNotReachedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AppraiseBlockNotReachedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AppraiseBlockNotReachedException(SerializationInfo, StreamingContext)

```
protected AppraiseBlockNotReachedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AppraiseBlockNotReachedException(string)

```
public AppraiseBlockNotReachedException(string message)
```

### Parameters

message [string](#)

# Class ApprovePledge

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("approve_pledge")]
public class ApprovePledge : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← ApprovePledge

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ApprovePledge()

```
public ApprovePledge()
```

## Fields

### PatronAddress

```
public Address PatronAddress
```

## Field Value

Address

## TypeIdentifier

```
public const string TypeIdentifier = "approve_pledge"
```

Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### `context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

[IActionContext](#)

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` `IValue`

Data (made by `Libplanet.Action.IAction.PlainValue` property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class ArenaNotEndedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaNotEndedException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
ArenaNotEndedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

ArenaNotEndedException(SerializationInfo,  
StreamingContext)

```
protected ArenaNotEndedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaNotEndedException(string)

```
public ArenaNotEndedException(string s)
```

### Parameters

s [string](#)

# Class AssetInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class AssetInfo : IRegisterInfo
```

## Inheritance

[object](#) ← AssetInfo

## Implements

[IRegisterInfo](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### AssetInfo()

```
public AssetInfo()
```

### AssetInfo(List)

```
public AssetInfo(List serialized)
```

## Parameters

**serialized** List

## Properties

## Asset

```
public FungibleAssetValue Asset { get; set; }
```

Property Value

FungibleAssetValue

## AvatarAddress

```
public Address AvatarAddress { get; set; }
```

Property Value

Address

## Price

```
public FungibleAssetValue Price { get; set; }
```

Property Value

FungibleAssetValue

## Type

```
public ProductType Type { get; set; }
```

Property Value

[ProductType](#)

## Methods

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Validate()

```
public void Validate()
```

## ValidateAddress(Address)

```
public void ValidateAddress(Address avatarAddress)
```

Parameters

avatarAddress Address

## ValidatePrice(Currency)

```
public void ValidatePrice(Currency ncg)
```

Parameters

ncg Currency

# Class AttachmentActionResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class AttachmentActionResult : IState
```

## Inheritance

[object](#) ← AttachmentActionResult

## Implements

[IState](#)

## Derived

[Buy7.BuyerResult](#), [Buy7.SellerResult](#), [CombinationConsumable5.ResultModel](#),  
[DailyReward2.DailyRewardResult](#), [ItemEnhancement10.ResultModel](#),  
[ItemEnhancement11.ResultModel](#), [ItemEnhancement12.ResultModel](#),  
[ItemEnhancement13.ResultModel](#), [ItemEnhancement7.ResultModel](#),  
[ItemEnhancement9.ResultModel](#), [RapidCombination5.ResultModel](#), [SellCancellation.Result](#),  
[MonsterCollectionResult](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### AttachmentActionResult()

```
protected AttachmentActionResult()
```

### AttachmentActionResult(Dictionary)

```
protected AttachmentActionResult(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Fields

### costume

```
public Costume costume
```

#### Field Value

[Costume](#)

### itemUsable

```
public ItemUsable itemUsable
```

#### Field Value

[ItemUsable](#)

### tradableFungibleItem

```
public ITradableFungibleItem tradableFungibleItem
```

#### Field Value

[ITradableFungibleItem](#)

### tradableFungibleItemCount

```
public int tradableFungibleItemCount
```

Field Value

[int](#)

## Properties

### TypeId

```
protected abstract string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Deserialize(Dictionary)

```
public static AttachmentActionResult Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[AttachmentActionResult](#)

### Serialize()

```
public virtual IValue Serialize()
```

Returns

IValue

## SerializeBackup1()

```
public virtual IValue SerializeBackup1()
```

Returns

IValue

# Class AuraSummon

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("aura_summon")]
public class AuraSummon : GameAction, IAction, IAuraSummonV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [AuraSummon](#)

## Implements

IAction, IAuraSummonV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AuraSummon()

```
public AuraSummon()
```

### AuraSummon(Address, int, int)

```
public AuraSummon(Address avatarAddress, int groupId, int summonCount)
```

## Parameters

avatarAddress Address

groupId [int](#)

summonCount [int](#)

## Fields

### AvatarAddress

[public](#) Address AvatarAddress

#### Field Value

Address

### AvatarAddressKey

[public const string](#) AvatarAddressKey = "aa"

#### Field Value

[string](#)

### GroupId

[public int](#) GroupId

#### Field Value

[int](#)

### GroupIdKey

```
public const string GroupIdKey = "gid"
```

Field Value

[string](#)

## SummonCount

```
public int SummonCount
```

Field Value

[int](#)

## SummonCountKey

```
public const string SummonCountKey = "sc"
```

Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

## AddAndUnlockOption(AgentState, Equipment, IRandom, Row, EquipmentItemOptionSheet, SkillSheet)

```
public static void AddAndUnlockOption(AgentState agentState, Equipment equipment,
IRandom random, EquipmentItemSubRecipeSheetV2.Row subRecipe,
EquipmentItemOptionSheet optionSheet, SkillSheet skillSheet)
```

### Parameters

agentState [AgentState](#)

equipment [Equipment](#)

random IRandom

subRecipe [EquipmentItemSubRecipeSheetV2.Row](#)

optionSheet [EquipmentItemOptionSheet](#)

skillSheet [SkillSheet](#)

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

## SimulateSummon(string, AgentState, EquipmentItemRecipeSheet, EquipmentItemSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet, SkillSheet, Row, int, IRandom, long)

```
public static IEnumerable<(int, Equipment)> SimulateSummon(string addressesHex, AgentState agentState, EquipmentItemRecipeSheet recipeSheet, EquipmentItemSheet equipmentItemSheet, EquipmentItemSubRecipeSheetV2 equipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet optionSheet, SkillSheet skillSheet, SummonSheet.Row summonRow, int summonCount, IRandom random, long blockIndex)
```

### Parameters

addressesHex [string](#)

agentState [AgentState](#)

recipeSheet [EquipmentItemRecipeSheet](#)

equipmentItemSheet [EquipmentItemSheet](#)

equipmentItemSubRecipeSheetV2 [EquipmentItemSubRecipeSheetV2](#)

optionSheet [EquipmentItemOptionSheet](#)

skillSheet [SkillSheet](#)

summonRow [SummonSheet.Row](#)

summonCount [int](#)

random [IRandom](#)

blockIndex [long](#)

Returns

[IEnumerable](#)<([int](#), [Equipment](#))>

# Class AvatarIndexAlreadyUsedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AvatarIndexAlreadyUsedException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← AvatarIndexAlreadyUsedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AvatarIndexAlreadyUsedException(SerializationInfo, StreamingContext)

```
protected AvatarIndexAlreadyUsedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AvatarIndexAlreadyUsedException(string)

```
public AvatarIndexAlreadyUsedException(string s)
```

### Parameters

s [string](#)

# Class AvatarIndexOutOfRangeException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class AvatarIndexOutOfRangeException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AvatarIndexOutOfRangeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AvatarIndexOutOfRangeException(SerializationInfo, StreamingContext)

```
protected AvatarIndexOutOfRangeException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AvatarIndexOutOfRangeException(string)

```
public AvatarIndexOutOfRangeException(string message)
```

### Parameters

message [string](#)

# Class BalanceDoesNotExistsException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BalanceDoesNotExistsException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← BalanceDoesNotExistsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### BalanceDoesNotExistsException(Address, Currency)

```
public BalanceDoesNotExistsException(Address address, Currency currency)
```

## Parameters

address Address

currency Currency

### BalanceDoesNotExistsException(SerializationInfo, StreamingContext)

```
protected BalanceDoesNotExistException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Address

```
public Address Address { get; }
```

#### Property Value

Address

### Currency

```
public Currency Currency { get; }
```

#### Property Value

Currency

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

### info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

### context [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

### [ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class BattleArena

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduce at <https://github.com/planetarium/lib9c/pull/2229> Changed at <https://github.com/planetarium/lib9c/pull/2242>

```
[Serializable]
[ActionType("battle_arena15")]
public class BattleArena : GameAction, IAction, IBattleArenaV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← BattleArena

## Implements

IAction, IBattleArenaV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### HpIncreasingModifier

```
public const int HpIncreasingModifier = 5
```

#### Field Value

[int](#)

## PurchasedCountKey

```
public const string PurchasedCountKey = "purchased_count_during_interval"
```

### Field Value

[string](#)

## championshipId

```
public int championshipId
```

### Field Value

[int](#)

## costumes

```
public List<Guid> costumes
```

### Field Value

[List](#)<[Guid](#)>

## enemyAvatarAddress

```
public Address enemyAvatarAddress
```

### Field Value

Address

## equipments

```
public List<Guid> equipments
```

Field Value

[List](#)<[Guid](#)>

## myAvatarAddress

```
public Address myAvatarAddress
```

Field Value

Address

## round

```
public int round
```

Field Value

[int](#)

## runeInfos

```
public List<RuneSlotInfo> runeInfos
```

Field Value

[List](#)<[RuneSlotInfo](#)>

## ticket

```
public int ticket
```

## Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class BurnAsset

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("burn_asset")]
public class BurnAsset : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← BurnAsset

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### BurnAsset()

```
public BurnAsset()
```

### BurnAsset(Address, FungibleAssetValue, string)

```
public BurnAsset(Address owner, FungibleAssetValue amount, string memo)
```

## Parameters

**owner** Address

amount FungibleAssetValue

memo [string](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "burn_asset"
```

Field Value

[string](#)

## Properties

### Amount

```
public FungibleAssetValue Amount { get; }
```

Property Value

FungibleAssetValue

### Memo

```
public string Memo { get; }
```

Property Value

[string](#)

### Owner

```
public Address Owner { get; }
```

Property Value

Address

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class Buy

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/1164>

```
[Serializable]
[ActionObsolete(8324909)]
[ActionTypes("buy12")]
public class Buy : GameAction, IBuy5, IAction, IBuyV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Buy

## Implements

[IBuy5](#), IAction, IBuyV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ErrorCodeDuplicateSell

```
public const int ErrorCodeDuplicateSell = 10
```

## Field Value

[int](#)

## ErrorCodeFailedLoadingState

```
public const int ErrorCodeFailedLoadingState = 1
```

Field Value

[int ↗](#)

## ErrorCodeInsufficientBalance

```
public const int ErrorCodeInsufficientBalance = 4
```

Field Value

[int ↗](#)

## ErrorCodeInvalidAddress

```
public const int ErrorCodeInvalidAddress = 5
```

Field Value

[int ↗](#)

## ErrorCodeInvalidItemType

```
public const int ErrorCodeInvalidItemType = 9
```

Field Value

[int ↗](#)

## ErrorCodeInvalidOrderId

```
public const int ErrorCodeInvalidOrderId = 7
```

Field Value

[int](#)

## ErrorCodeInvalidPrice

```
public const int ErrorCodeInvalidPrice = 6
```

Field Value

[int](#)

## ErrorCodeInvalidTradableId

```
public const int ErrorCodeInvalidTradableId = 8
```

Field Value

[int](#)

## ErrorCodeItemDoesNotExist

```
public const int ErrorCodeItemDoesNotExist = 2
```

Field Value

[int](#)

## ErrorCodeShopItemExpired

```
public const int ErrorCodeShopItemExpired = 3
```

Field Value

[int](#)

## TaxRate

```
public const int TaxRate = 8
```

Field Value

[int](#)

## errors

```
public List<(Guid orderId, int errorCode)> errors
```

Field Value

[List](#)<[Guid](#) [orderId](#), [int](#) [errorCode](#)>

## purchaselInfos

```
public IEnumerable<PurchaseInfo> purchaseInfos
```

Field Value

[IEnumerable](#)<[PurchaseInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), IValue>

## buyerAvatarAddress

```
public Address buyerAvatarAddress { get; set; }
```

Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class Buy7

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("buy7")]
public class Buy7 : GameAction, IBuy5, IAction, IBuyV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Buy7

## Implements

[IBuy5](#), IAction, IBuyV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ErrorCodeFailedLoadingState

```
public const int ErrorCodeFailedLoadingState = 1
```

#### Field Value

[int](#)

### ErrorCodeInsufficientBalance

```
public const int ErrorCodeInsufficientBalance = 4
```

Field Value

[int](#)

## ErrorCodeInvalidAddress

```
public const int ErrorCodeInvalidAddress = 5
```

Field Value

[int](#)

## ErrorCodeInvalidPrice

```
public const int ErrorCodeInvalidPrice = 6
```

Field Value

[int](#)

## ErrorCodeItemDoesNotExist

```
public const int ErrorCodeItemDoesNotExist = 2
```

Field Value

[int](#)

## ErrorCodeShopItemExpired

```
public const int ErrorCodeShopItemExpired = 3
```

Field Value

[int](#)

## TaxRate

```
public const int TaxRate = 8
```

Field Value

[int](#)

## buyerMultipleResult

```
public Buy7.BuyerMultipleResult buyerMultipleResult
```

Field Value

[Buy7.BuyerMultipleResult](#)

## purchaselInfos

```
public IEnumerable<PurchaseInfo0> purchaseInfos
```

Field Value

[IEnumerable](#)<[PurchaseInfo0](#)>

## sellerMultipleResult

```
public Buy7.SellerMultipleResult sellerMultipleResult
```

Field Value

[Buy7.SellerMultipleResult](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

### buyerAvatarAddress

```
public Address buyerAvatarAddress { get; set; }
```

Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class Buy7.BuyerMultipleResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Buy7.BuyerMultipleResult
```

## Inheritance

[object](#) ← Buy7.BuyerMultipleResult

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### BuyerMultipleResult()

```
public BuyerMultipleResult()
```

### BuyerMultipleResult(Dictionary)

```
public BuyerMultipleResult(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Fields

### purchaseResults

```
public IEnumerable<Buy7.PurchaseResult> purchaseResults
```

Field Value

[IEnumerable](#) <[Buy7.PurchaseResult](#)>

Methods

Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class Buy7.BuyerResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuyerResult : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← BuyerResult

## Implements

[IState](#)

## Derived

[Buy7.PurchaseResult](#), [BuyMultiple.PurchaseResult](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### BuyerResult()

```
public BuyerResult()
```

### BuyerResult(Dictionary)

```
public BuyerResult(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Fields

### id

```
public Guid id
```

### Field Value

[Guid](#)

### shopItem

```
public ShopItem shopItem
```

### Field Value

[ShopItem](#)

## Properties

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

[string](#)

## Methods

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class Buy7.PurchaseResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Buy7.PurchaseResult : Buy7.BuyerResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← [Buy7.BuyerResult](#) ← [Buy7.PurchaseResult](#)

## Implements

[IState](#)

## Inherited Members

[Buy7.BuyerResult.shopItem](#) , [Buy7.BuyerResult.id](#) , [Buy7.BuyerResult.TypeId](#) ,  
[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### PurchaseResult(Dictionary)

```
public PurchaseResult(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### PurchaseResult(Guid)

```
public PurchaseResult(Guid shopProductId)
```

## Parameters

shopProductId [Guid](#)

## Fields

### errorCode

```
public int errorCode
```

#### Field Value

[int](#)

### productId

```
public readonly Guid productId
```

#### Field Value

[Guid](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

#### Returns

IValue

# Class Buy7.SellerMultipleResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Buy7.SellerMultipleResult
```

## Inheritance

[object](#) ← Buy7.SellerMultipleResult

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### SellerMultipleResult()

```
public SellerMultipleResult()
```

### SellerMultipleResult(Dictionary)

```
public SellerMultipleResult(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### sellerResults

```
public IEnumerable<Buy7.SellerResult> sellerResults
```

Field Value

[IEnumerable](#) <[Buy7.SellerResult](#)>

Methods

Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class Buy7.SellerResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Buy7.SellerResult : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← Buy7.SellerResult

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SellerResult()

```
public SellerResult()
```

### SellerResult(Dictionary)

```
public SellerResult(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### gold

```
public FungibleAssetValue gold
```

#### Field Value

FungibleAssetValue

### id

```
public Guid id
```

#### Field Value

[Guid](#)

### shopItem

```
public ShopItem shopItem
```

#### Field Value

[ShopItem](#)

## Properties

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class BuyMultiple

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/331> Updated at many pull requests  
Obsolete at <https://github.com/planetarium/lib9c/pull/487> Updated at many pull requests  
Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionObsolete(7000000)]
[ActionTypes("buy_multiple")]
public class BuyMultiple : GameAction, IAction, IBuyMultipleV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← BuyMultiple

## Implements

IAction, IBuyMultipleV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ERROR\_CODE\_FAILED\_LOADING\_STATE

```
public const int ERROR_CODE_FAILED_LOADING_STATE = 1
```

## Field Value

[int](#)

## ERROR\_CODE\_INSUFFICIENT\_BALANCE

```
public const int ERROR_CODE_INSUFFICIENT_BALANCE = 4
```

Field Value

[int](#)

## ERROR\_CODE\_ITEM\_DOES\_NOT\_EXIST

```
public const int ERROR_CODE_ITEM_DOES_NOT_EXIST = 2
```

Field Value

[int](#)

## ERROR\_CODE\_SHOPITEM\_EXPIRED

```
public const int ERROR_CODE_SHOPITEM_EXPIRED = 3
```

Field Value

[int](#)

## buyerAvatarAddress

```
public Address buyerAvatarAddress
```

Field Value

Address

## buyerResult

```
public BuyMultiple.BuyerResult buyerResult
```

### Field Value

[BuyMultiple.BuyerResult](#)

## purchaselInfos

```
public IEnumerable<BuyMultiple.PurchaseInfo> purchaseInfos
```

### Field Value

[IEnumerable<BuyMultiple.PurchaseInfo>](#)

## sellerResult

```
public BuyMultiple.SellerResult sellerResult
```

### Field Value

[BuyMultiple.SellerResult](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary<string, IValue>](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class BuyMultiple.BuyerResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuyMultiple.BuyerResult
```

## Inheritance

[object](#) ← BuyMultiple.BuyerResult

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### BuyerResult()

```
public BuyerResult()
```

### BuyerResult(Dictionary)

```
public BuyerResult(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### purchaseResults

```
public IEnumerable<BuyMultiple.PurchaseResult> purchaseResults
```

Field Value

[IEnumerable](#) <[BuyMultiple.PurchaseResult](#)>

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class BuyMultiple.PurchaseInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuyMultiple.PurchaseInfo : IComparable<BuyMultiple.PurchaseInfo>,
IComparable
```

## Inheritance

[object](#) ← BuyMultiple.PurchaseInfo

## Implements

[IComparable](#)<[BuyMultiple.PurchaseInfo](#)>, [IComparable](#)

## Inherited Members

[object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

### PurchaseInfo(Dictionary)

```
public PurchaseInfo(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### PurchaseInfo(Guid, Address, Address)

```
public PurchaseInfo(Guid productId, Address sellerAgentAddress,
Address sellerAvatarAddress)
```

#### Parameters

`productId` [Guid ↗](#)

`sellerAgentAddress` `Address`

`sellerAvatarAddress` `Address`

## Fields

### `productId`

`public Guid productId`

Field Value

[Guid ↗](#)

### `sellerAgentAddress`

`public Address sellerAgentAddress`

Field Value

`Address`

### `sellerAvatarAddress`

`public Address sellerAvatarAddress`

Field Value

`Address`

## Methods

## CompareTo(PurchaseInfo)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(BuyMultiple.PurchaseInfo other)
```

### Parameters

**other** [BuyMultiple.PurchaseInfo](#)

An object to compare with this instance.

### Returns

[int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <b>other</b> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <b>other</b> .
<b>Greater than zero</b>	This instance follows <b>other</b> in the sort order.

## CompareTo(object)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(object obj)
```

## Parameters

### `obj` [object](#)

An object to compare with this instance.

## Returns

### [int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <code>obj</code> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <code>obj</code> .
<b>Greater than zero</b>	This instance follows <code>obj</code> in the sort order.

## Exceptions

### [ArgumentException](#)

`obj` is not the same type as this instance.

## Equals(PurchaseInfo)

```
protected bool Equals(BuyMultiple.PurchaseInfo other)
```

## Parameters

### `other` [BuyMultiple.PurchaseInfo](#)

## Returns

### [bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Operators

## operator ==(PurchaseInfo, PurchaseInfo)

```
public static bool operator ==(BuyMultiple.PurchaseInfo left,  
BuyMultiple.PurchaseInfo right)
```

### Parameters

left [BuyMultiple.PurchaseInfo](#)

right [BuyMultiple.PurchaseInfo](#)

### Returns

[bool](#)

## operator >(PurchaseInfo, PurchaseInfo)

```
public static bool operator >(BuyMultiple.PurchaseInfo left,  
BuyMultiple.PurchaseInfo right)
```

### Parameters

left [BuyMultiple.PurchaseInfo](#)

right [BuyMultiple.PurchaseInfo](#)

### Returns

[bool](#)

## operator >=(PurchaseInfo, PurchaseInfo)

```
public static bool operator >=(BuyMultiple.PurchaseInfo left,  
BuyMultiple.PurchaseInfo right)
```

Parameters

`left` [BuyMultiple.PurchaseInfo](#)

`right` [BuyMultiple.PurchaseInfo](#)

Returns

[bool](#)

## operator !=(PurchaseInfo, PurchaseInfo)

```
public static bool operator !=(BuyMultiple.PurchaseInfo left,  
BuyMultiple.PurchaseInfo right)
```

Parameters

`left` [BuyMultiple.PurchaseInfo](#)

`right` [BuyMultiple.PurchaseInfo](#)

Returns

[bool](#)

## operator <(PurchaseInfo, PurchaseInfo)

```
public static bool operator <(BuyMultiple.PurchaseInfo left,  
BuyMultiple.PurchaseInfo right)
```

Parameters

`left` [BuyMultiple.PurchaseInfo](#)

`right` [BuyMultiple.PurchaseInfo](#)

Returns

[bool](#)

## operator <=(PurchaseInfo, PurchaseInfo)

```
public static bool operator <=(BuyMultiple.PurchaseInfo left,  
BuyMultiple.PurchaseInfo right)
```

### Parameters

left [BuyMultiple.PurchaseInfo](#)

right [BuyMultiple.PurchaseInfo](#)

### Returns

[bool](#)

# Class BuyMultiple.PurchaseResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuyMultiple.PurchaseResult : Buy7.BuyerResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← [Buy7.BuyerResult](#) ← [BuyMultiple.PurchaseResult](#)

## Implements

[IState](#)

## Inherited Members

[Buy7.BuyerResult.shopItem](#) , [Buy7.BuyerResult.id](#) , [Buy7.BuyerResult.TypeId](#) ,  
[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### PurchaseResult(Dictionary)

```
public PurchaseResult(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### PurchaseResult(Guid)

```
public PurchaseResult(Guid shopProductId)
```

## Parameters

shopProductId [Guid](#)

## Fields

errorCode

```
public int errorCode
```

## Field Value

[int](#)

productId

```
public Guid productId
```

## Field Value

[Guid](#)

## Methods

Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

# Class BuyMultiple.SellerResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuyMultiple.SellerResult
```

## Inheritance

[object](#) ← BuyMultiple.SellerResult

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### SellerResult()

```
public SellerResult()
```

### SellerResult(Dictionary)

```
public SellerResult(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### sellerResults

```
public IEnumerable<Buy7.SellerResult> sellerResults
```

Field Value

[IEnumerable](#) <[Buy7.SellerResult](#)>

Methods

Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class BuyProduct

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("buy_product3")]
public class BuyProduct : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← BuyProduct

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

### Capacity

```
public const int Capacity = 20
```

## Field Value

[int](#)

## ProductInfos

```
public IEnumerable<IProductInfo> ProductInfos
```

## Field Value

[IEnumerable](#)<[IProductInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class CancelProductRegistration

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("cancel_product_registration2")]
public class CancelProductRegistration : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← CancelProductRegistration

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

### Capacity

```
public const int Capacity = 100
```

Field Value

[int ↗](#)

## ChargeAp

`public bool ChargeAp`

Field Value

[bool ↗](#)

## CostAp

`public const int CostAp = 5`

Field Value

[int ↗](#)

## ProductInfos

`public List<IProductInfo> ProductInfos`

Field Value

[List ↗ <IProductInfo>](#)

## Properties

### PlainValueInternal

`protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }`

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Cancel(ProductsState, IProductInfo, IWorld, AvatarState, IActionContext)

```
public static IWorld Cancel(ProductsState productsState, IProductInfo productInfo,
IWorld states, AvatarState avatarState, IActionContext context)
```

#### Parameters

productsState [ProductsState](#)

productInfo [IProductInfo](#)

states IWorld

avatarState [AvatarState](#)

context IActionContext

#### Returns

IWorld

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class ChargeActionPoint

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/430> Updated at <https://github.com/planetarium/lib9c/pull/474> Updated at <https://github.com/planetarium/lib9c/pull/602> Updated at <https://github.com/planetarium/lib9c/pull/861> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("charge_action_point3")]
public class ChargeActionPoint : GameAction, IAction, IChargeActionPointV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ChargeActionPoint

## Implements

IAction, IChargeActionPointV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

### Field Value

Address

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

[LoadPlainValueInternal\(IImmutableDictionary<string, IValue>\)](#)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class ClaimItems

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("claim_items")]
public class ClaimItems : GameAction, IAction, IClaimItems
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ClaimItems

## Implements

IAction, [IClaimItems](#)

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ClaimItems()

```
public ClaimItems()
```

### ClaimItems(IReadOnlyList<(Address, IReadOnlyList<FungibleAssetValue>>, string)

```
public ClaimItems(IReadOnlyList<(Address, IReadOnlyList<FungibleAssetValue>>
claimData, string memo = null)
```

## Parameters

```
claimData IReadOnlyList<(Address address, IReadOnlyList<FungibleAssetValue>  
fungibleAssetValues)>
```

memo [string](#)

## Fields

### Memo

```
public string Memo
```

### Field Value

[string](#)

## Properties

### ClaimData

```
public IReadOnlyList<(Address address, IReadOnlyList<FungibleAssetValue>  
fungibleAssetValues)> ClaimData { get; }
```

### Property Value

[IReadOnlyList](#)<(Address [address](#), [IReadOnlyList](#)<FungibleAssetValue> [fungibleAssetValues](#))>

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

`plainValue` [IImmutableDictionary](#)<[string](#), `IValue`>

# Class ClaimRaidReward

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("claim_raid_reward")]
public class ClaimRaidReward : GameAction, IAction, IClaimRaidRewardV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ClaimRaidReward

## Implements

IAction, IClaimRaidRewardV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ClaimRaidReward()

```
public ClaimRaidReward()
```

### ClaimRaidReward(Address)

```
public ClaimRaidReward(Address avatarAddress)
```

## Parameters

avatarAddress Address

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

#### Property Value

[IImmutableDictionary](#)<[string](#), IValue>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class ClaimStakeReward

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2097>

```
[ActionType("claim_stake_reward9")]
public class ClaimStakeReward : GameAction, IClaimStakeReward,
IAction, IClaimStakeRewardV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ClaimStakeReward

## Implements

[IClaimStakeReward](#), IAction, IClaimStakeRewardV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ClaimStakeReward()

```
public ClaimStakeReward()
```

### ClaimStakeReward(Address)

```
public ClaimStakeReward(Address avatarAddress)
```

## Parameters

avatarAddress Address

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class ClaimStakeReward2

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public static class ClaimStakeReward2
```

## Inheritance

[object](#) ← ClaimStakeReward2

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### ObsoleteIndex

```
public const long ObsoleteIndex = 5549200
```

#### Field Value

[long](#)

# Class ClaimStakeReward8

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class ClaimStakeReward8
```

## Inheritance

[object](#) ← ClaimStakeReward8

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### ObsoleteBlockIndex

```
public const long ObsoleteBlockIndex = 7983895
```

#### Field Value

[long](#)

# Class ClaimWordBossKillReward

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("claim_world_boss_kill_reward")]
public class ClaimWordBossKillReward : GameAction,
IAction, IClaimWordBossKillRewardV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ClaimWordBossKillReward

## Implements

IAction, IClaimWordBossKillRewardV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

## Properties

# PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class CombinationConsumable

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("combination_consumable9")]
public class CombinationConsumable : GameAction, IAction, ICombinationConsumableV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← CombinationConsumable

## Implements

IAction, ICombinationConsumableV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddressKey

```
public const string AvatarAddressKey = "a"
```

Field Value

[string](#)

## RecipIdKey

```
public const string RecipeIdKey = "r"
```

Field Value

[string](#)

## SlotIndexKey

```
public const string SlotIndexKey = "s"
```

Field Value

[string](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## recipId

```
public int recipeId
```

Field Value

[int](#)

## slotIndex

```
public int slotIndex
```

## Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class CombinationConsumable5

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("combination_consumable5")]
public class CombinationConsumable5 : GameAction, IAction, ICombinationConsumableV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← CombinationConsumable5

## Implements

IAction, ICombinationConsumableV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

### recipeld

```
public int recipeId
```

Field Value

[int](#)

## slotIndex

```
public int slotIndex
```

Field Value

[int](#)

# Properties

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class CombinationConsumable5.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationConsumable5.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← CombinationConsumable5.ResultModel

## Implements

[IState](#)

## Derived

[RapidCombination0.ResultModel](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Fields

**actionPoint**

```
public int actionPoint
```

Field Value

[int](#)

**gold**

```
public BigInteger gold
```

Field Value

[BigInteger](#)

**id**

```
public Guid id
```

Field Value

[Guid](#)

**itemType**

```
public ItemType itemType
```

Field Value

[ItemType](#)

## materials

```
public Dictionary<Material, int> materials
```

Field Value

[Dictionary](#)<[Material](#), [int](#)>

## recipId

```
public int recipeId
```

Field Value

[int](#)

## subRecipId

```
public int? subRecipeId
```

Field Value

[int](#)?

## Properties

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class CombinationEquipment

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("combination_equipment17")]
public class CombinationEquipment : GameAction, IAction, ICombinationEquipmentV4
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← CombinationEquipment

## Implements

IAction, ICombinationEquipmentV4

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddressKey

```
public const string AvatarAddressKey = "a"
```

Field Value

[string](#)

## BasicSubRecipeHammerPoint

```
public const int BasicSubRecipeHammerPoint = 1
```

Field Value

[int](#)

## PayByCrystalKey

```
public const string PayByCrystalKey = "p"
```

Field Value

[string](#)

## PetIdKey

```
public const string PetIdKey = "pid"
```

Field Value

[string](#)

## RecipeIdKey

```
public const string RecipeIdKey = "r"
```

Field Value

[string](#)

## SlotIndexKey

```
public const string SlotIndexKey = "s"
```

Field Value

[string](#)

## SpecialSubRecipeHammerPoint

```
public const int SpecialSubRecipeHammerPoint = 2
```

Field Value

[int](#)

## SubRecipeIdKey

```
public const string SubRecipeIdKey = "i"
```

Field Value

[string](#)

## UseHammerPointKey

```
public const string UseHammerPointKey = "h"
```

Field Value

[string](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## payByCrystal

```
public bool payByCrystal
```

Field Value

[bool](#)

## petId

```
public int? petId
```

Field Value

[int](#)?

## recipieId

```
public int recipeId
```

Field Value

[int](#)

## slotIndex

```
public int slotIndex
```

Field Value

[int](#)

## subRecipeId

```
public int? subRecipeId
```

Field Value

[int](#)?

## useHammerPoint

```
public bool useHammerPoint
```

Field Value

[bool](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

## AddAndUnlockOption(AgentState, PetState, Equipment, IRandom, Row, EquipmentItemOptionSheet, PetOptionSheet, SkillSheet)

```
public static void AddAndUnlockOption(AgentState agentState, PetState petState,
Equipment equipment, IRandom random, EquipmentItemSubRecipeSheetV2.Row
subRecipe, EquipmentItemOptionSheet optionSheet, PetOptionSheet petOptionSheet,
SkillSheet skillSheet)
```

### Parameters

agentState [AgentState](#)

petState [PetState](#)

equipment [Equipment](#)

random IRandom

subRecipe [EquipmentItemSubRecipeSheetV2.Row](#)

optionSheet [EquipmentItemOptionSheet](#)

petOptionSheet [PetOptionSheet](#)

skillSheet [SkillSheet](#)

## AddSkillOption(AgentState, Equipment, IRandom, Row, EquipmentItemOptionSheet, SkillSheet)

```
public static void AddSkillOption(AgentState agentState, Equipment equipment,
IRandom random, EquipmentItemSubRecipeSheetV2.Row subRecipe,
EquipmentItemOptionSheet optionSheet, SkillSheet skillSheet)
```

### Parameters

agentState [AgentState](#)

equipment [Equipment](#)

`random` `IRandom`

`subRecipe` [EquipmentItemSubRecipeSheetV2.Row](#)

`optionSheet` [EquipmentItemOptionSheet](#)

`skillSheet` [SkillSheet](#)

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override Iworld Execute(IActionContext context)
```

### Parameters

`context` `IActionContext`

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

`IWorld`

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetSkill(Row, SkillSheet, IRandom)

```
public static Skill GetSkill(EquipmentItemOptionSheet.Row row, SkillSheet  
skillSheet, IRandom random)
```

### Parameters

row [EquipmentItemOptionSheet.Row](#)

skillSheet [SkillSheet](#)

random IRandom

Returns

[Skill](#)

LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class CombinationEquipment5

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public static class CombinationEquipment5
```

## Inheritance

[object](#) ← CombinationEquipment5

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### BlacksmithAddress

```
public static readonly Address BlacksmithAddress
```

## Field Value

Address

## Methods

### GetSkill(Row, SkillSheet, IRandom)

```
public static Skill GetSkill(EquipmentItemOptionSheet.Row row, SkillSheet  
skillSheet, IRandom random)
```

## Parameters

row [EquipmentItemOptionSheet.Row](#)

skillSheet [SkillSheet](#)

random IRandom

Returns

[Skill](#)

## GetStat(Row, IRandom)

```
public static DecimalStat GetStat(EquipmentItemOptionSheet.Row row, IRandom random)
```

Parameters

row [EquipmentItemOptionSheet.Row](#)

random IRandom

Returns

[DecimalStat](#)

## SelectOption(EquipmentItemOptionSheet, SkillSheet, Row, IRandom, Equipment)

```
public static HashSet<int> SelectOption(EquipmentItemOptionSheet optionSheet,  
SkillSheet skillSheet, EquipmentItemSubRecipeSheet.Row subRecipe, IRandom random,  
Equipment equipment)
```

Parameters

optionSheet [EquipmentItemOptionSheet](#)

skillSheet [SkillSheet](#)

subRecipe [EquipmentItemSubRecipeSheet.Row](#)

random IRandom

equipment [Equipment](#)

Returns

[HashSet](#)<[int](#)>

# Class CombinationSlotResultNullException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationSlotResultNullException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← CombinationSlotResultNullException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### CombinationSlotResultNullException(SerializationInfo, StreamingContext)

```
public CombinationSlotResultNullException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## CombinationSlotResultNullException(string)

```
public CombinationSlotResultNullException(string message)
```

### Parameters

message [string](#)

# Class CombinationSlotUnlockException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationSlotUnlockException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← CombinationSlotUnlockException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### CombinationSlotUnlockException(SerializationInfo, StreamingContext)

```
protected CombinationSlotUnlockException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## CombinationSlotUnlockException(string)

```
public CombinationSlotUnlockException(string s)
```

### Parameters

s [string](#)

# Class ConsumableSlotOutOfRangeException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ConsumableSlotOutOfRangeException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ConsumableSlotOutOfRangeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ConsumableSlotOutOfRangeException()

```
public ConsumableSlotOutOfRangeException()
```

### ConsumableSlotOutOfRangeException(SerializationInfo, StreamingContext)

```
protected ConsumableSlotOutOfRangeException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#) ↗

context [StreamingContext](#) ↗

## ConsumableSlotOutOfRangeException(string)

public ConsumableSlotOutOfRangeException(string message)

Parameters

message [string](#) ↗

# Class ConsumableSlotUnlockException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ConsumableSlotUnlockException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← ConsumableSlotUnlockException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ConsumableSlotUnlockException(SerializationInfo, StreamingContext)

```
protected ConsumableSlotUnlockException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ConsumableSlotUnlockException(string)

```
public ConsumableSlotUnlockException(string s)
```

### Parameters

s [string](#)

# Class CostumeSlotUnlockException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CostumeSlotUnlockException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← CostumeSlotUnlockException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### CostumeSlotUnlockException(SerializationInfo, StreamingContext)

```
protected CostumeSlotUnlockException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## CostumeSlotUnlockException(string)

```
public CostumeSlotUnlockException(string s)
```

### Parameters

s [string](#)

# Class CreateAvatar

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("create_avatar11")]
public class CreateAvatar : GameAction, IAction, ICreateAvatarV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← CreateAvatar

## Implements

IAction, ICreateAvatarV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### DeriveFormat

```
public const string DeriveFormat = "avatar-state-{0}"
```

### Field Value

[string](#)

**ear**

```
public int ear
```

Field Value

[int](#)

**hair**

```
public int hair
```

Field Value

[int](#)

**index**

```
public int index
```

Field Value

[int](#)

**lens**

```
public int lens
```

Field Value

[int](#)

**name**

```
public string name
```

Field Value

[string](#)

tail

```
public int tail
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### AddItem(ItemSheet, CreateAvatarItemSheet, AvatarState, IRandom)

```
public static void AddItem(ItemSheet itemSheet, CreateAvatarItemSheet  
createAvatarItemSheet, AvatarState avatarState, IRandom random)
```

Parameters

itemSheet [ItemSheet](#)

createAvatarItemSheet [CreateAvatarItemSheet](#)

avatarState [AvatarState](#)

random IRandom

## CreateAvatarState(string, Address, IActionContext, MaterialItemSheet, Address)

```
public static AvatarState CreateAvatarState(string name, Address avatarAddress,  
IActionContext ctx, MaterialItemSheet materialItemSheet, Address rankingMapAddress)
```

### Parameters

name [string](#)

avatarAddress Address

ctx IActionContext

materialItemSheet [MaterialItemSheet](#)

rankingMapAddress Address

### Returns

[AvatarState](#)

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

## MintAsset(CreateAvatarFavSheet, AvatarState, IWorld, IActionContext)

```
public static IWorld MintAsset(CreateAvatarFavSheet favSheet, AvatarState avatarState, IWorld states, IActionContext context)
```

### Parameters

favSheet [CreateAvatarFavSheet](#)

avatarState [AvatarState](#)

states IWorld

context IActionContext

### Returns



# Class CreatePendingActivation

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("create_pending_activation")]
[ActionObsolete(7200000)]
public class CreatePendingActivation : ActionBase,
IAction, ICreatePendingActivationV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← CreatePendingActivation

## Implements

IAction, ICreatePendingActivationV1

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CreatePendingActivation()

```
public CreatePendingActivation()
```

### CreatePendingActivation(PendingActivationState)

```
public CreatePendingActivation(PendingActivationState activationKey)
```

## Parameters

activationKey [PendingActivationState](#)

## Properties

### PendingActivation

```
public PendingActivationState PendingActivation { get; }
```

#### Property Value

[PendingActivationState](#)

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

#### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

[IACTIONCONTEXT](#)

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` `IValue`

Data (made by `Libplanet.Action.IAction.PlainValue` property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class CreatePendingActivations

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/746> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("create_pending_activations")]
[ActionObsolete(7200000)]
public class CreatePendingActivations : ActionBase,
IAction, ICreatePendingActivationsV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← CreatePendingActivations

## Implements

IAction, ICreatePendingActivationsV1

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CreatePendingActivations()

```
public CreatePendingActivations()
```

### CreatePendingActivations(IEnumerable<PendingActivationState>)

```
public CreatePendingActivations(IEnumerable<PendingActivationState> states)
```

## Parameters

states [IEnumerable](#)<[PendingActivationState](#)>

## Properties

### PendingActivations

```
public IList<(byte[] Address, byte[] Nonce, byte[] PublicKey)> PendingActivations {  
    get; }
```

## Property Value

[IList](#)<([byte](#)[] [Address](#), [byte](#)[] [Nonce](#), [byte](#)[] [PublicKey](#))>

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class CreatePledge

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("create_pledge")]
public class CreatePledge : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← CreatePledge

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AgentAddresses

```
public IEnumerable<(Address, Address)> AgentAddresses
```

#### Field Value

[IEnumerable](#)<(Address, Address)>

### Mead

```
public int Mead
```

Field Value

[int](#)

## PatronAddress

```
public Address PatronAddress
```

Field Value

Address

## TypeIdentifier

```
public const string TypeIdentifier = "create_pledge"
```

Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and [Libplanet.Action.IActionContexts](#), the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class DailyReward

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/1828> Updated at <https://github.com/planetarium/lib9c/pull/1828>

```
[Serializable]
[ActionType("daily_reward7")]
public class DailyReward : GameAction, IAction, IDailyRewardV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← DailyReward

## Implements

IAction, IDailyRewardV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ActionPointMax

```
public const int ActionPointMax = 120
```

#### Field Value

[int](#)

## AvatarAddressKey

```
public const string AvatarAddressKey = "a"
```

Field Value

[string](#)

## DailyRewardInterval

```
public const long DailyRewardInterval = 2550
```

Field Value

[long](#)

## DailyRuneRewardAmount

```
public const int DailyRuneRewardAmount = 1
```

Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## Properties

# PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class DailyReward2

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("daily_reward2")]
public class DailyReward2 : GameAction, IAction, IDailyRewardV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← DailyReward2

## Implements

IAction, IDailyRewardV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

### dailyRewardResult

```
public DailyReward2.DailyRewardResult dailyRewardResult
```

## Field Value

[DailyReward2.DailyRewardResult](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class DailyReward2.DailyRewardResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DailyReward2.DailyRewardResult : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← DailyReward2.DailyRewardResult

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### DailyRewardResult()

```
public DailyRewardResult()
```

### DailyRewardResult(Dictionary)

```
public DailyRewardResult(Dictionary serialized)
```

## Parameters

serialized Dictionary

# Fields

## id

```
public Guid id
```

### Field Value

[Guid](#)

## materials

```
public Dictionary<Material, int> materials
```

### Field Value

[Dictionary](#)<[Material](#), [int](#)>

# Properties

## TypeId

```
protected override string TypeId { get; }
```

### Property Value

[string](#)

# Methods

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class DuplicateCostumeException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicateCostumeException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
DuplicateCostumeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

DuplicateCostumeException(SerializationInfo,  
StreamingContext)

```
protected DuplicateCostumeException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicateCostumeException(string)

```
public DuplicateCostumeException(string s)
```

### Parameters

s [string](#)

# Class DuplicateEquipmentException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicateEquipmentException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
DuplicateEquipmentException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### DuplicateEquipmentException(SerializationInfo, StreamingContext)

```
protected DuplicateEquipmentException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicateEquipmentException(string)

```
public DuplicateEquipmentException(string s)
```

### Parameters

s [string](#)

# Class DuplicateMaterialException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicateMaterialException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
DuplicateMaterialException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### DuplicateMaterialException(SerializationInfo, StreamingContext)

```
protected DuplicateMaterialException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicateMaterialException(string)

```
public DuplicateMaterialException(string s)
```

### Parameters

s [string](#)

# Class DuplicateOrderIdException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicateOrderIdException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
DuplicateOrderIdException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

DuplicateOrderIdException(SerializationInfo,  
StreamingContext)

```
protected DuplicateOrderIdException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicateOrderIdException(string)

```
public DuplicateOrderIdException(string s)
```

### Parameters

s [string](#)

# Class EndPledge

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("end_pledge")]
public class EndPledge : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← EndPledge

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### EndPledge()

```
public EndPledge()
```

## Fields

### AgentAddress

```
public Address AgentAddress
```

## Field Value

Address

## TypeIdentifier

```
public const string TypeIdentifier = "end_pledge"
```

Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### `context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

[IActionContext](#)

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` `IValue`

Data (made by `Libplanet.Action.IAction.PlainValue` property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class EquipmentLevelExceededException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EquipmentLevelExceededException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← EquipmentLevelExceededException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### EquipmentLevelExceededException(SerializationInfo, StreamingContext)

```
protected EquipmentLevelExceededException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## EquipmentLevelExceededException(string)

```
public EquipmentLevelExceededException(string s)
```

### Parameters

s [string](#)

# Class EquipmentSlotUnlockException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EquipmentSlotUnlockException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← EquipmentSlotUnlockException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### EquipmentSlotUnlockException()

```
public EquipmentSlotUnlockException()
```

### EquipmentSlotUnlockException(SerializationInfo, StreamingContext)

```
protected EquipmentSlotUnlockException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## EquipmentSlotUnlockException(string)

public [EquipmentSlotUnlockException](#)([string](#) msg)

Parameters

msg [string](#)

# Class EventConsumableItemCrafts

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("event_consumable_item_crafts2")]
public class EventConsumableItemCrafts : GameAction,
IAction, IEventConsumableItemCraftsV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← EventConsumableItemCrafts

## Implements

IAction, IEventConsumableItemCraftsV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

## EventConsumableItemRecipId

```
public int EventConsumableItemRecipeId
```

Field Value

[int](#)

## EventScheduleId

```
public int EventScheduleId
```

Field Value

[int](#)

## SlotIndex

```
public int SlotIndex
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class EventDungeonBattle

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("event_dungeon_battle6")]
public class EventDungeonBattle : GameAction, IAction, IEventDungeonBattleV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← EventDungeonBattle

## Implements

IAction, IEventDungeonBattleV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

## BuyTicketIfNeeded

```
public bool BuyTicketIfNeeded
```

Field Value

[bool](#)

## Costumes

```
public List<Guid> Costumes
```

Field Value

[List](#)<[Guid](#)>

## Equipments

```
public List<Guid> Equipments
```

Field Value

[List](#)<[Guid](#)>

## EventDungeonId

```
public int EventDungeonId
```

Field Value

[int](#)

## EventDungeonStageId

```
public int EventDungeonStageId
```

Field Value

[int](#)

## EventScheduleId

```
public int EventScheduleId
```

Field Value

[int](#)

## Foods

```
public List<Guid> Foods
```

Field Value

[List](#)<[Guid](#)>

## PlayCount

```
public const int PlayCount = 1
```

Field Value

[int](#)

## RunelInfos

```
public List<RuneSlotInfo> RuneInfos
```

## Field Value

[List](#)<[RuneSlotInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class EventMaterialItemCrafts

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("event_material_item_crafts2")]
public class EventMaterialItemCrafts : GameAction,
IAction, IEventMaterialItemCraftsV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← EventMaterialItemCrafts

## Implements

IAction, IEventMaterialItemCraftsV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

## EventMaterialItemRecipId

```
public int EventMaterialItemRecipeId
```

Field Value

[int](#)

## EventScheduleId

```
public int EventScheduleId
```

Field Value

[int](#)

## MaterialsToUse

```
public Dictionary<int, int> MaterialsToUse
```

Field Value

[Dictionary](#)<[int](#), [int](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class FailedLoadSheetException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FailedLoadSheetException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← FailedLoadSheetException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### FailedLoadSheetException(SerializationInfo, StreamingContext)

```
protected FailedLoadSheetException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## FailedLoadSheetException(string)

```
public FailedLoadSheetException(string message)
```

### Parameters

message [string](#)

## FailedLoadSheetException(Type)

```
public FailedLoadSheetException(Type sheetType)
```

### Parameters

sheetType [Type](#)

# Class FailedLoadStateException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FailedLoadStateException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← FailedLoadStateException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### FailedLoadStateException(Address, Type, Exception)

```
public FailedLoadStateException(Address address, Type stateType, Exception
innerException = null)
```

## Parameters

address Address

stateType [Type](#)

innerException [Exception](#)

## FailedLoadStateException(SerializationInfo, StreamingContext)

```
protected FailedLoadStateException(SerializationInfo info, StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## FailedLoadStateException(string)

```
public FailedLoadStateException(string message)
```

### Parameters

message [string](#)

## FailedLoadStateException(string, Exception)

```
public FailedLoadStateException(string message, Exception innerException = null)
```

### Parameters

message [string](#)

innerException [Exception](#)

## FailedLoadStateException(string, string, string, Exception)

```
public FailedLoadStateException(string actionType, string addressesHex, string message, Exception innerException = null)
```

## Parameters

actionType [string](#)

addressesHex [string](#)

message [string](#)

innerException [Exception](#)

## FailedLoadStateException(string, string, Type, Address)

```
public FailedLoadStateException(string actionType, string addressesHex, Type  
stateType, Address address)
```

## Parameters

actionType [string](#)

addressesHex [string](#)

stateType [Type](#)

address Address

# Class FavProductInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class FavProductInfo : IProductInfo
```

## Inheritance

[object](#) ← FavProductInfo

## Implements

[IProductInfo](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### FavProductInfo()

```
public FavProductInfo()
```

### FavProductInfo(List)

```
public FavProductInfo(List serialized)
```

## Parameters

serialized List

## Properties

## AgentAddress

```
public Address AgentAddress { get; set; }
```

Property Value

Address

## AvatarAddress

```
public Address AvatarAddress { get; set; }
```

Property Value

Address

## Price

```
public FungibleAssetValue Price { get; set; }
```

Property Value

FungibleAssetValue

## ProductId

```
public Guid ProductId { get; set; }
```

Property Value

[Guid](#) ↗

## Type

```
public ProductType Type { get; set; }
```

Property Value

[ProductType](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

### ValidateType()

```
public void ValidateType()
```

# Class GameAction

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class GameAction : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← GameAction

## Implements

IAction

## Derived

[ActivateCollection](#), [AddRedeemCode](#), [ClaimAdventureBossReward](#), [ExploreAdventureBoss](#),  
[SweepAdventureBoss](#), [UnlockFloor](#), [Wanted](#), [AuraSummon](#), [BattleArena](#), [Buy](#), [Buy7](#),  
[BuyMultiple](#), [BuyProduct](#), [CancelProductRegistration](#), [ChargeActionPoint](#), [ClaimItems](#),  
[ClaimRaidReward](#), [ClaimStakeReward](#), [ClaimWordBossKillReward](#), [CombinationConsumable](#),  
[CombinationConsumable5](#), [CombinationEquipment](#), [IssueCoupons](#), [RedeemCoupon](#),  
[TransferCoupons](#), [CreateAvatar](#), [DailyReward](#), [DailyReward2](#), [EventConsumableItemCrafts](#),  
[EventDungeonBattle](#), [EventMaterialItemCrafts](#), [BulkUnloadFromGarages](#),  
[DeliverToOthersGarages](#), [LoadIntoMyGarages](#), [UnloadFromMyGarages](#), [Grinding](#),  
[HackAndSlash](#), [HackAndSlashRandomBuff](#), [HackAndSlashSweep](#), [InitializeStates](#),  
[ItemEnhancement](#), [ItemEnhancement10](#), [ItemEnhancement11](#), [ItemEnhancement12](#),  
[ItemEnhancement13](#), [ItemEnhancement7](#), [ItemEnhancement9](#), [JoinArena](#), [JoinArena1](#),  
[JoinArena3](#), [MigrationActivatedAccountsState](#), [PatchTableSheet](#), [PetEnhancement](#), [Raid](#),  
[RankingBattle](#), [RankingBattle11](#), [RapidCombination](#), [RapidCombination0](#),  
[RapidCombination5](#), [ReRegisterProduct](#), [RedeemCode](#), [RedeemCode2](#), [RegisterProduct](#),  
[RegisterProduct0](#), [RenewAdminState](#), [RuneEnhancement](#), [RuneSummon](#), [Sell](#), [Sell6](#),  
[SellCancellation](#), [Stake](#), [Stake2](#), [UnlockCombinationSlot](#), [UnlockEquipmentRecipe](#),  
[UnlockRuneSlot](#), [UnlockWorld](#), [UpdateSell](#)

## Inherited Members

[ActionBase.Execute\(IActionContext\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GameAction()

```
protected GameAction()
```

## Properties

### Id

```
public Guid Id { get; }
```

### Property Value

[Guid](#)

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override sealed IValue PlainValue { get; }
```

### Property Value

#### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

LoadPlainValue(IValue)

## PlainValueInternal

```
protected abstract IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), IValue>

## Methods

### LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override sealed void LoadPlainValue(IValue plainValue)
```

Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

#### See Also

PlainValue

### LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected abstract void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class GoldDistribution

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GoldDistribution : IEquatable<GoldDistribution>
```

## Inheritance

[object](#) ← GoldDistribution

## Implements

[IEquatable](#)<[GoldDistribution](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GoldDistribution()

```
public GoldDistribution()
```

### GoldDistribution(Dictionary)

```
public GoldDistribution(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### GoldDistribution(IValue)

```
public GoldDistribution(IValue serialized)
```

## Parameters

serialized IValue

## Fields

### Address

[Ignore]

```
public Address Address
```

### Field Value

Address

## Properties

### AddressString

[Index(0, -1)]

```
public string AddressString { get; set; }
```

### Property Value

[string](#)

### AmountPerBlock

[Index(1, -1)]

```
public BigInteger AmountPerBlock { get; set; }
```

### Property Value

[BigInteger](#)

## EndBlock

```
[Index(3, -1)]  
public long EndBlock { get; set; }
```

Property Value

[long](#)

## StartBlock

```
[Index(2, -1)]  
public long StartBlock { get; set; }
```

Property Value

[long](#)

## Methods

### Equals(GoldDistribution)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(GoldDistribution other)
```

Parameters

**other** [GoldDistribution](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

### Parameters

[obj](#) [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetAmount(long)

```
public BigInteger GetAmount(long blockIndex)
```

### Parameters

[blockIndex](#) [long](#)

### Returns

[BigInteger](#)

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## LoadInDescendingEndBlockOrder(string)

```
public static GoldDistribution[] LoadInDescendingEndBlockOrder(string csvPath)
```

Parameters

[csvPath](#) [string](#)

Returns

[GoldDistribution\[\]](#)

## Serialize()

```
public Dictionary Serialize()
```

Returns

[Dictionary](#)

# Class Grinding

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("grinding2")]
public class Grinding : GameAction, IAction, IGrindingV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Grinding

## Implements

IAction, IGrindingV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

### ChargeAp

```
public bool ChargeAp
```

Field Value

[bool](#)

CostAp

```
public const int CostAp = 5
```

Field Value

[int](#)

EquipmentIds

```
public List<Guid> EquipmentIds
```

Field Value

[List](#)<[Guid](#)>

Limit

```
public const int Limit = 50
```

Field Value

[int](#)

Properties

PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### CalculateMaterialReward(IEnumerable<Equipment>, CrystalEquipmentGrindingSheet, MaterialItemSheet)

```
public static Dictionary<Material, int>
CalculateMaterialReward(IEnumerable<Equipment> equipmentList,
CrystalEquipmentGrindingSheet crystalEquipmentGrindingSheet,
MaterialItemSheet materialItemSheet)
```

## Parameters

equipmentList [IEnumerable](#)<[Equipment](#)>

crystalEquipmentGrindingSheet [CrystalEquipmentGrindingSheet](#)

materialItemSheet [MaterialItemSheet](#)

## Returns

[Dictionary](#)<[Material](#), [int](#)>

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

# Parameters

## context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

# Returns

## IWorld

A map of changed states (so-called "dirty").

# Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class HackAndSlash

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("hack_and_slash22")]
public class HackAndSlash : GameAction, IAction, IHackAndSlashV10
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← HackAndSlash

## Implements

IAction, [IHackAndSlashV10](#)

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ApStoneCount

```
public int ApStoneCount
```

#### Field Value

[int](#)

## AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

## Costumes

```
public List<Guid> Costumes
```

### Field Value

[List](#)<[Guid](#)>

## Equipments

```
public List<Guid> Equipments
```

### Field Value

[List](#)<[Guid](#)>

## Foods

```
public List<Guid> Foods
```

### Field Value

[List](#)<[Guid](#)>

## RunelInfos

```
public List<RuneSlotInfo> RuneInfos
```

Field Value

[List](#)<[RuneSlotInfo](#)>

## StageBuffId

```
public int? StageBuffId
```

Field Value

[int](#)?

## StageId

```
public int StageId
```

Field Value

[int](#)

## TotalPlayCount

```
public int TotalPlayCount
```

Field Value

[int](#)

## UsableApStoneCount

```
public const int UsableApStoneCount = 10
```

Field Value

[int](#)

## WorldId

```
public int WorldId
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## Execute(IWorld, Address, long, IRandom)

```
public IWorld Execute(IWorld states, Address signer, long blockIndex,  
IRandom random)
```

### Parameters

**states** IWorld

**signer** Address

**blockIndex** [long](#)

**random** IRandom

### Returns

IWorld

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class HackAndSlashRandomBuff

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Created at <https://github.com/planetarium/lib9c/pull/1031>

```
[Serializable]
[ActionType("hack_and_slash_random_buff")]
public class HackAndSlashRandomBuff : GameAction, IAction, IHackAndSlashRandomBuffV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← HackAndSlashRandomBuff

## Implements

IAction, IHackAndSlashRandomBuffV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AdvancedGacha

```
public bool AdvancedGacha
```

#### Field Value

[bool](#)

## AdvancedGachaCount

```
public const int AdvancedGachaCount = 10
```

Field Value

[int](#)

## AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

## NormalGachaCount

```
public const int NormalGachaCount = 5
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class HackAndSlashSweep

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("hack_and_slash_sweep10")]
public class HackAndSlashSweep : GameAction, IAction, IHackAndSlashSweepV3
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← HackAndSlashSweep

## Implements

IAction, IHackAndSlashSweepV3

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### UsableApStoneCount

```
public const int UsableApStoneCount = 20
```

#### Field Value

[int](#)

## actionPoint

```
public int actionPoint
```

Field Value

[int](#)

## apStoneCount

```
public int apStoneCount
```

Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## costumes

```
public List<Guid> costumes
```

Field Value

[List](#)<[Guid](#)>

## equipments

```
public List<Guid> equipments
```

Field Value

[List](#)<[Guid](#)>

## runeInfos

```
public List<RuneSlotInfo> runeInfos
```

Field Value

[List](#)<[RuneSlotInfo](#)>

## stageId

```
public int stageId
```

Field Value

[int](#)

## worldId

```
public int worldId
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), IValue>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetRewardItems(IRandom, int, Row, MaterialItemSheet)

```
public static List<ItemBase> GetRewardItems(IRandom random, int playCount,  
StageSheet.Row stageRow, MaterialItemSheet materialItemSheet)
```

## Parameters

random IRandom

playCount [int](#)

stageRow [StageSheet.Row](#)

materialItemSheet [MaterialItemSheet](#)

Returns

[List](#)<[ItemBase](#)>

**LoadPlainValueInternal(IIImmutableDictionary<string, IValue>)**

```
protected override void LoadPlainValueInternal(IIImmutableDictionary<string, IValue> plainValue)
```

Parameters

plainValue [IIImmutableDictionary](#)<[string](#), IValue>

# Interface IBuy0

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Common interface used before [IBuy5](#).

```
public interface IBuy0 : IAction
```

## Inherited Members

IAction.PlainValue , IAction.LoadPlainValue(IValue) , IAction.Execute(IActionContext)

## Properties

### buyerAvatarAddress

```
Address buyerAvatarAddress { get; }
```

Property Value

Address

### productId

```
Guid productId { get; }
```

Property Value

[Guid](#)

### sellerAgentAddress

```
Address sellerAgentAddress { get; }
```

Property Value

Address

## **sellerAvatarAddress**

```
Address sellerAvatarAddress { get; }
```

Property Value

Address

## **See Also**

[IBuy5](#)

# Interface IBuy5

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Common interface used after [IBuy5](#).

```
public interface IBuy5 : IAction
```

## Inherited Members

IAction.PlainValue , IAction.LoadPlainValue(IValue) , IAction.Execute(IActionContext)

## Properties

### buyerAvatarAddress

```
Address buyerAvatarAddress { get; }
```

Property Value

Address

### purchaseInfos

```
IEnumerable<IPurchaseInfo> purchaseInfos { get; }
```

Property Value

[IEnumerable](#)<[IPurchaseInfo](#)>

## See Also

[IBuy5](#)

# Interface IClaimItems

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface IClaimItems
```

## Properties

### ClaimData

```
IReadOnlyList<(Address address, IReadOnlyList<FungibleAssetValue>  
fungibleAssetValues)> ClaimData { get; }
```

### Property Value

[IReadOnlyList](#)<(Address [address](#), [IReadOnlyList](#)<FungibleAssetValue> [fungibleAssetValues](#))>

# Interface IClaimStakeReward

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface IClaimStakeReward : IAction
```

## Inherited Members

IAction.PlainValue , IAction.LoadPlainValue(IValue) , IAction.Execute(IActionContext)

# Interface IHackAndSlashV10

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface IHackAndSlashV10
```

## Properties

### ApStoneCount

```
int ApStoneCount { get; }
```

#### Property Value

[int](#)

### AvatarAddress

```
Address AvatarAddress { get; }
```

#### Property Value

Address

### Costumes

```
IEnumerable<Guid> Costumes { get; }
```

#### Property Value

[IEnumerable](#)<[Guid](#)>

## Equipments

IEnumerable<Guid> Equipments { [get](#); }

Property Value

[IEnumerable](#)<[Guid](#)>

## Foods

IEnumerable<Guid> Foods { [get](#); }

Property Value

[IEnumerable](#)<[Guid](#)>

## RuneSlotInfos

IEnumerable<IValue> RuneSlotInfos { [get](#); }

Property Value

[IEnumerable](#)<[IValue](#)>

## StageBuffId

[int](#)? StageBuffId { [get](#); }

Property Value

[int](#)?

## StageId

```
int StageId { get; }
```

Property Value

[int](#)

## TotalPlayCount

```
int TotalPlayCount { get; }
```

Property Value

[int](#)

## WorldId

```
int WorldId { get; }
```

Property Value

[int](#)

# Interface IProductInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface IProductInfo
```

## Properties

### AgentAddress

```
Address AgentAddress { get; set; }
```

#### Property Value

Address

### AvatarAddress

```
Address AvatarAddress { get; set; }
```

#### Property Value

Address

### Price

```
FungibleAssetValue Price { get; set; }
```

#### Property Value

FungibleAssetValue

## ProductId

```
Guid ProductId { get; set; }
```

Property Value

[Guid](#)

## Type

```
ProductType Type { get; set; }
```

Property Value

[ProductType](#)

## Methods

### Serialize()

```
IValue Serialize()
```

Returns

IValue

### ValidateType()

```
void ValidateType()
```

# Interface IPurchaseInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface IPurchaseInfo
```

## Properties

### ItemSubType

```
ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

### OrderId

```
Guid? OrderId { get; }
```

Property Value

[Guid](#)?

### Price

```
FungibleAssetValue Price { get; }
```

Property Value

FungibleAssetValue

## SellerAgentAddress

```
Address SellerAgentAddress { get; }
```

Property Value

Address

## SellerAvatarAddress

```
Address SellerAvatarAddress { get; }
```

Property Value

Address

# Interface IRegisterInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface IRegisterInfo
```

## Properties

### AvatarAddress

```
Address AvatarAddress { get; set; }
```

#### Property Value

Address

### Price

```
FungibleAssetValue Price { get; set; }
```

#### Property Value

FungibleAssetValue

### Type

```
ProductType Type { get; set; }
```

#### Property Value

[ProductType](#)

# Methods

## Serialize()

```
IValue Serialize()
```

Returns

IValue

## Validate()

```
void Validate()
```

## ValidateAddress(Address)

```
void ValidateAddress(Address avatarAddress)
```

Parameters

avatarAddress Address

## ValidatePrice(Currency)

```
void ValidatePrice(Currency ncg)
```

Parameters

ncg Currency

# Interface ITransferAsset

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface ITransferAsset
```

## Properties

### Amount

FungibleAssetValue Amount { [get](#); }

### Property Value

FungibleAssetValue

### Memo

[string?](#) Memo { [get](#); }

### Property Value

[string](#) ↗

### Recipient

Address Recipient { [get](#); }

### Property Value

Address

# Sender

```
Address Sender { get; }
```

Property Value

Address

# Interface ITransferAssets

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public interface ITransferAssets
```

## Properties

### Memo

```
string? Memo { get; }
```

Property Value

[string](#)

### Recipients

```
List<(Address recipient, FungibleAssetValue amount)> Recipients { get; }
```

Property Value

[List](#)<(Address [recipient](#), FungibleAssetValue [amount](#))>

### Sender

```
Address Sender { get; }
```

Property Value

Address

# Class InitializeStates

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at NCG distribution(7e4515b6e14cc5d6eb716d5ebb587ab04b4246f9) Updated at <https://github.com/planetarium/lib9c/pull/19> Updated at <https://github.com/planetarium/lib9c/pull/36> Updated at <https://github.com/planetarium/lib9c/pull/42> Updated at <https://github.com/planetarium/lib9c/pull/55> Updated at <https://github.com/planetarium/lib9c/pull/57> Updated at <https://github.com/planetarium/lib9c/pull/60> Updated at <https://github.com/planetarium/lib9c/pull/102> Updated at <https://github.com/planetarium/lib9c/pull/128> Updated at <https://github.com/planetarium/lib9c/pull/167> Updated at <https://github.com/planetarium/lib9c/pull/422> Updated at <https://github.com/planetarium/lib9c/pull/747> Updated at <https://github.com/planetarium/lib9c/pull/798> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("initialize_states")]
public class InitializeStates : GameAction, IAction, IInitializeStatesV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← InitializeStates

## Implements

IAction, IInitializeStatesV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## InitializeStates()

```
public InitializeStates()
```

InitializeStates(ValidatorSet, RankingState0, ShopState,  
Dictionary<string, string>, GameConfigState,  
RedeemCodeState, ActivatedAccountsState,  
GoldCurrencyState, GoldDistribution[],  
PendingActivationState[], AdminState,  
AuthorizedMinersState, CreditsState, ISet<Address>)

```
public InitializeStates(ValidatorSet validatorSet, RankingState0 rankingState,  
ShopState shopState, Dictionary<string, string> tableSheets, GameConfigState  
gameConfigState, RedeemCodeState redeemCodeState, ActivatedAccountsState  
activatedAccountsState, GoldCurrencyState goldCurrencyState, GoldDistribution[]  
goldDistributions, PendingActivationState[] pendingActivationStates, AdminState  
adminAddressState = null, AuthorizedMinersState authorizedMinersState = null,  
CreditsState creditsState = null, ISet<Address> assetMinters = null)
```

## Parameters

validatorSet ValidatorSet

rankingState [RankingState0](#)

shopState [ShopState](#)

tableSheets [Dictionary<string, string>](#)

gameConfigState [GameConfigState](#)

redeemCodeState [RedeemCodeState](#)

activatedAccountsState [ActivatedAccountsState](#)

goldCurrencyState [GoldCurrencyState](#)

goldDistributions [GoldDistribution\[\]](#)

pendingActivationStates [PendingActivationState\[\]](#)

adminAddressState [AdminState](#)

authorizedMinersState [AuthorizedMinersState](#)

creditsState [CreditsState](#)

assetMinters [ISet<Address>](#)

## Properties

### ActivatedAccounts

```
public Dictionary ActivatedAccounts { get; set; }
```

Property Value

Dictionary

### AdminAddressState

```
public Dictionary AdminAddressState { get; set; }
```

Property Value

Dictionary

### AssetMinters

```
public ISet<Address> AssetMinters { get; set; }
```

Property Value

[ISet<Address>](#)

## AuthorizedMiners

```
public Dictionary AuthorizedMiners { get; set; }
```

Property Value

Dictionary

## Credits

```
public Dictionary Credits { get; set; }
```

Property Value

Dictionary

## GameConfig

```
public Dictionary GameConfig { get; set; }
```

Property Value

Dictionary

## GoldCurrency

```
public Dictionary GoldCurrency { get; set; }
```

Property Value

Dictionary

## GoldDistributions

```
public List GoldDistributions { get; set; }
```

Property Value

List

## PendingActivations

```
public List PendingActivations { get; set; }
```

Property Value

List

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Ranking

```
public Dictionary Ranking { get; set; }
```

Property Value

Dictionary

## RedeemCode

```
public Dictionary RedeemCode { get; set; }
```

Property Value

Dictionary

## Shop

```
public Dictionary Shop { get; set; }
```

Property Value

Dictionary

## TableSheets

```
public Dictionary<string, string> TableSheets { get; set; }
```

Property Value

[Dictionary](#)<[string](#), [string](#)>

## ValidatorSet

```
public IValue ValidatorSet { get; set; }
```

Property Value

IValue

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but

equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class InvalidAddressException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidAddressException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidAddressException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidAddressException()

```
public InvalidAddressException()
```

### InvalidAddressException(SerializationInfo, StreamingContext)

```
protected InvalidAddressException(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidAddressException(string)

public InvalidAddressException([string](#) msg)

Parameters

msg [string](#)

# Class InvalidClaimException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidClaimException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidClaimException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidClaimException()

```
public InvalidClaimException()
```

### InvalidClaimException(SerializationInfo, StreamingContext)

```
protected InvalidClaimException(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidOperationException(string)

public [InvalidOperationException](#)(string msg)

Parameters

msg [string](#)

# Class InvalidCurrencyException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidCurrencyException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← InvalidCurrencyException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidCurrencyException(SerializationInfo, StreamingContext)

```
protected InvalidCurrencyException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidCurrencyException(string)

```
public InvalidCurrencyException(string msg)
```

### Parameters

msg [string](#)

# Class InvalidElementException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidElementException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidElementException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidElementException()

```
public InvalidElementException()
```

### InvalidElementException(SerializationInfo, StreamingContext)

```
protected InvalidElementException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidElementalException(string)

public InvalidElementalException(string message)

Parameters

message [string](#)

# Class InvalidEquipmentException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidEquipmentException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidEquipmentException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidEquipmentException()

```
public InvalidEquipmentException()
```

### InvalidEquipmentException(SerializationInfo, StreamingContext)

```
protected InvalidEquipmentException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidOperationException(string)

public [InvalidOperationException](#)(string msg)

Parameters

msg [string](#)

# Class InvalidItemCountException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidItemCountException : ArgumentOutOfRangeException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [ArgumentException](#) ←  
[ArgumentOutOfRangeException](#) ← [InvalidItemCountException](#)

## Implements

[ISerializable](#)

## Inherited Members

[ArgumentOutOfRangeException.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ArgumentOutOfRangeException.ActualValue](#) , [ArgumentOutOfRangeException.Message](#) ,  
[ArgumentException.ParamName](#) , [Exception.GetBaseException\(\)](#) ,  
[Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) ,  
[Exception.HResult](#) , [Exception.InnerException](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidItemCountException()

```
public InvalidItemCountException()
```

### InvalidItemCountException(SerializationInfo, StreamingContext)

```
protected InvalidItemCountException(SerializationInfo info,
```

```
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidItemCountException(string)

```
public InvalidItemCountException(string msg)
```

## Parameters

msg [string](#)

# Class InvalidItemTypeException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidItemTypeException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidItemTypeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

InvalidItemTypeException(SerializationInfo,  
StreamingContext)

```
protected InvalidItemTypeException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidItemTypeException(string)

```
public InvalidItemTypeException(string msg)
```

### Parameters

msg [string](#)

# Class InvalidLevelException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidLevelException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← InvalidLevelException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

InvalidLevelException(SerializationInfo, StreamingContext)

```
protected InvalidLevelException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidOperationException(string)

```
public InvalidOperationException(string s)
```

### Parameters

s [string](#)

# Class InvalidMaterialException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidMaterialException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← InvalidMaterialException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

InvalidMaterialException(SerializationInfo, StreamingContext)

```
protected InvalidMaterialException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidMaterialException(string)

```
public InvalidMaterialException(string s)
```

### Parameters

s [string](#)

# Class InvalidMinterException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidMinterException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidMinterException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidMinterException()

```
public InvalidMinterException()
```

### InvalidMinterException(Address)

```
public InvalidMinterException(Address signer)
```

## Parameters

**signer** Address

# InvalidMinterException(SerializationInfo, StreamingContext)

```
protected InvalidMinterException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

context [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

[ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class InvalidMonsterCollectionRoundException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidMonsterCollectionRoundException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidMonsterCollectionRoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidMonsterCollectionRoundException(SerializationInfo, StreamingContext)

```
protected InvalidMonsterCollectionRoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidMonsterCollectionRoundException(string)

public [InvalidMonsterCollectionRoundException](#)([string](#) msg)

### Parameters

msg [string](#)

# Class InvalidNamePatternException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidNamePatternException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidNamePatternException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

InvalidNamePatternException(SerializationInfo,  
StreamingContext)

```
protected InvalidNamePatternException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidNamePatternException(string)

```
public InvalidNamePatternException(string s)
```

### Parameters

s [string](#)

# Class InvalidPriceException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidPriceException : ArgumentOutOfRangeException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [ArgumentException](#) ←  
[ArgumentOutOfRangeException](#) ← InvalidPriceException

## Implements

[ISerializable](#)

## Inherited Members

[ArgumentOutOfRangeException.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ArgumentOutOfRangeException.ActualValue](#) , [ArgumentOutOfRangeException.Message](#) ,  
[ArgumentException.ParamName](#) , [Exception.GetBaseException\(\)](#) ,  
[Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) ,  
[Exception.HResult](#) , [Exception.InnerException](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidPriceException(SerializationInfo, StreamingContext)

```
protected InvalidPriceException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidPriceException(string)

```
public InvalidPriceException(string msg)
```

### Parameters

msg [string](#)

# Class InvalidProductTypeException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidProductTypeException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidProductTypeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidProductTypeException(SerializationInfo, StreamingContext)

```
protected InvalidProductTypeException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidProductTypeException(string)

```
public InvalidProductTypeException(string msg)
```

### Parameters

msg [string](#)

# Class InvalidRecipIdException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidRecipeIdException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidRecipIdException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidRecipIdException()

```
public InvalidRecipeIdException()
```

### InvalidRecipIdException(SerializationInfo, StreamingContext)

```
protected InvalidRecipeIdException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidRecipeIdException(string)

public [InvalidRecipeIdException](#)([string](#) msg)

### Parameters

msg [string](#)

# Class InvalidRepeatPlayException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidRepeatPlayException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidRepeatPlayException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidRepeatPlayException()

```
public InvalidRepeatPlayException()
```

### InvalidRepeatPlayException(SerializationInfo, StreamingContext)

```
protected InvalidRepeatPlayException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidRepeatPlayException(string)

public [InvalidRepeatPlayException](#)(string msg)

Parameters

msg [string](#)

# Class InvalidShopItemException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidShopItemException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidShopItemException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

InvalidShopItemException(SerializationInfo,  
StreamingContext)

```
protected InvalidShopItemException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidShopItemException(string)

```
public InvalidShopItemException(string s)
```

### Parameters

s [string](#)

# Class InvalidSignatureException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidSignatureException : ActivationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [ActivationException](#) ← InvalidSignatureException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidSignatureException(PendingActivationState, byte[])

```
public InvalidSignatureException(PendingActivationState pending, byte[] signature)
```

## Parameters

pending [PendingActivationState](#)

signature [byte](#)[]

# InvalidOperationException(SerializationInfo, StreamingContext)

```
public InvalidOperationException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Pending

```
public PendingActivationState Pending { get; }
```

## Property Value

[PendingActivationState](#)

## Signature

```
public byte[] Signature { get; }
```

## Property Value

[byte](#)[]

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

`info` [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

`context` [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

[ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class InvalidSlotIndexException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidSlotIndexException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidSlotIndexException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

InvalidSlotIndexException(SerializationInfo,  
StreamingContext)

```
protected InvalidSlotIndexException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidOperationException(string)

```
public InvalidOperationException(string s)
```

### Parameters

s [string](#)

# Class InvalidStageException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidStageException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidStageException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidStageException()

```
public InvalidStageException()
```

### InvalidStageException(SerializationInfo, StreamingContext)

```
protected InvalidStageException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidStageException(string)

public InvalidStageException(string msg)

### Parameters

msg [string](#)

# Class InvalidTradableIdException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidTradableIdException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidTradableIdException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTradableIdException(SerializationInfo, StreamingContext)

```
protected InvalidTradableIdException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidTradableIdException(string)

```
public InvalidTradableIdException(string message)
```

### Parameters

message [string](#)

# Class InvalidTradableItemException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class InvalidTradableItemException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidTradableItemException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTradableItemException(SerializationInfo, StreamingContext)

```
protected InvalidTradableItemException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidTradableItemException(string)

```
public InvalidTradableItemException(string message)
```

### Parameters

message [string](#)

# Class InvalidTransferCurrencyException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidTransferCurrencyException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidTransferCurrencyException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTransferCurrencyException(SerializationInfo, StreamingContext)

```
protected InvalidTransferCurrencyException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidTransferCurrencyException(string)

```
public InvalidTransferCurrencyException(string message)
```

### Parameters

message [string](#)

# Class InvalidTransferMinterException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidTransferMinterException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidTransferMinterException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTransferMinterException(IEnumerable<Address>, Address, Address)

```
public InvalidTransferMinterException(IEnumerable<Address> minters, Address sender,
Address recipient)
```

## Parameters

minters [IEnumerable](#)<Address>

sender Address

recipient Address

## InvalidTransferMinterException(SerializationInfo, StreamingContext)

```
protected InvalidTransferMinterException(SerializationInfo info,  
StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Minters

```
public HashSet<Address> Minters { get; }
```

### Property Value

[HashSet](#)<Address>

### Recipient

```
public Address Recipient { get; }
```

### Property Value

Address

### Sender

```
public Address Sender { get; }
```

Property Value

Address

# Class InvalidTransferRecipientException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidTransferRecipientException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidTransferRecipientException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTransferRecipientException(Address, Address)

```
public InvalidTransferRecipientException(Address sender, Address recipient)
```

## Parameters

**sender** Address

**recipient** Address

### InvalidTransferRecipientException(SerializationInfo, StreamingContext)

```
public InvalidTransferRecipientException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Recipient

```
public Address Recipient { get; }
```

#### Property Value

Address

### Sender

```
public Address Sender { get; }
```

#### Property Value

Address

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

### `info` [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

### `context` [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

### [ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class InvalidTransferSignerException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidTransferSignerException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidTransferSignerException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTransferSignerException(Address, Address, Address)

```
public InvalidTransferSignerException(Address txSigner, Address sender,
Address recipient)
```

## Parameters

**txSigner** Address

**sender** Address

**recipient** Address

# InvalidTransferSignerException(SerializationInfo, StreamingContext)

```
public InvalidTransferSignerException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Recipient

```
public Address Recipient { get; }
```

#### Property Value

Address

### Sender

```
public Address Sender { get; }
```

#### Property Value

Address

### TxSigner

```
public Address TxSigner { get; }
```

Property Value

Address

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

`info` [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

`context` [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

[ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class InvalidTransferUnactivatedRecipientException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidTransferUnactivatedRecipientException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidTransferUnactivatedRecipientException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidTransferUnactivatedRecipientException(Address, Address)

```
public InvalidTransferUnactivatedRecipientException(Address sender,
Address recipient)
```

## Parameters

sender Address

`recipient` Address

## InvalidTransferUnactivatedRecipientException(SerializationInfo, StreamingContext)

```
public InvalidTransferUnactivatedRecipientException(SerializationInfo info,  
StreamingContext context)
```

### Parameters

`info` [SerializationInfo](#)

`context` [StreamingContext](#)

## Properties

### Recipient

```
public Address Recipient { get; }
```

### Property Value

Address

### Sender

```
public Address Sender { get; }
```

### Property Value

Address

## Methods

# GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

**info** [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

**context** [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

[ArgumentNullException](#)

The **info** parameter is a null reference (**Nothing** in Visual Basic).

# Class InvalidWorldException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidWorldException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidWorldException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidWorldException()

```
public InvalidWorldException()
```

### InvalidWorldException(SerializationInfo, StreamingContext)

```
protected InvalidWorldException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidWorldException(string)

public InvalidWorldException(string msg)

### Parameters

msg [string](#)

# Class IssueToken

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("issue_token")]
public class IssueToken : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← IssueToken

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### IssueToken()

```
public IssueToken()
```

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

## Field Value

Address

## FungibleAssetValues

```
public List<FungibleAssetValue> FungibleAssetValues
```

Field Value

[List](#)<FungibleAssetValue>

## Items

```
public List<(int id, int count, bool tradable)> Items
```

Field Value

[List](#)<(int id, int count, bool tradable)>

## TypeIdentifier

```
public const string TypeIdentifier = "issue_token"
```

Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other "bound" information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class IssueTokensFromGarage

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("issue_tokens_from_garage")]
public class IssueTokensFromGarage : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← IssueTokensFromGarage

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### IssueTokensFromGarage()

```
public IssueTokensFromGarage()
```

### IssueTokensFromGarage(IEnumerable<Spec>)

```
public IssueTokensFromGarage(IEnumerable<IssueTokensFromGarage.Spec> specs)
```

## Parameters

specs [IEnumerable](#)<[IssueTokensFromGarage.Spec](#)>

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "issue_tokens_from_garage"
```

## Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

# Specs

```
public IEnumerable<IssueTokensFromGarage.Spec> Specs { get; }
```

## Property Value

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called "dirty").

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a

Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Struct IssueTokensFromGarage.Spec

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public readonly struct IssueTokensFromGarage.Spec
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### Spec(List)

```
public Spec(List bencoded)
```

## Parameters

bencoded List

## Properties

### Assets

```
public FungibleAssetValue? Assets { get; }
```

## Property Value

FungibleAssetValue?

## Items

```
public FungibleItemValue? Items { get; }
```

Property Value

[FungibleItemValue?](#)

## Methods

### FromFungibleAssetValue(FungibleAssetValue)

```
public static IssueTokensFromGarage.Spec FromFungibleAssetValue(FungibleAssetValue assets)
```

Parameters

assets FungibleAssetValue

Returns

[IssueTokensFromGarage.Spec](#)

### FromFungibleItemValue(FungibleItemValue)

```
public static IssueTokensFromGarage.Spec FromFungibleItemValue(FungibleItemValue items)
```

Parameters

items [FungibleItemValue](#)

Returns

[IssueTokensFromGarage.Spec](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class ItemDoesNotExistException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemDoesNotExistException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← ItemDoesNotExistException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ItemDoesNotExistException(SerializationInfo, StreamingContext)

```
public ItemDoesNotExistException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ItemDoesNotExistException(string)

```
public ItemDoesNotExistException(string msg)
```

### Parameters

msg [string](#)

# Class ItemEnhancement

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("item_enhancement14")]
public class ItemEnhancement : GameAction, IAction, IItemEnhancementV5
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement

## Implements

IAction, IItemEnhancementV5

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### HammerBannedTypes

```
public static readonly IReadOnlyCollection<ItemSubType> HammerBannedTypes
```

#### Field Value

[IReadOnlyCollection](#)<[ItemSubType](#)>

## HammerIds

```
public static readonly IReadOnlyCollection<int> HammerIds
```

### Field Value

[IReadOnlyCollection](#)<[int](#)>

## MaterialCountLimit

```
public const int MaterialCountLimit = 50
```

### Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

### Field Value

Address

## hammers

```
public Dictionary<int, int> hammers
```

### Field Value

[Dictionary](#)<[int](#), [int](#)>

## itemId

```
public Guid itemId
```

Field Value

[Guid](#)

## materialIds

```
public List<Guid> materialIds
```

Field Value

[List](#)<[Guid](#)>

## slotIndex

```
public int slotIndex
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetEquipmentMaxLevel(Equipment, EnhancementCostSheetV3)

```
public static int GetEquipmentMaxLevel(Equipment equipment,  
EnhancementCostSheetV3 sheet)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

### Returns

[int](#)

## GetRequiredBlockCount(Equipment, Equipment, EnhancementCostSheetV3)

```
public static int GetRequiredBlockCount(Equipment preEquipment, Equipment targetEquipment, EnhancementCostSheetV3 sheet)
```

### Parameters

preEquipment [Equipment](#)

targetEquipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

### Returns

[int](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

## TryGetRow(Equipment, EnhancementCostSheetV3, out Row)

```
public static bool TryGetRow(Equipment equipment, EnhancementCostSheetV3 sheet, out EnhancementCostSheetV3.Row row)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

row [EnhancementCostSheetV3.Row](#)

Returns

[bool](#) ↗

# Class ItemEnhancement10

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/1176>

```
[Serializable]
[ActionType("item_enhancement10")]
[ActionObsolete(7200000)]
public class ItemEnhancement10 : GameAction, IAction, IItemEnhancementV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement10

## Implements

IAction, IItemEnhancementV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

### Field Value

Address

## itemId

```
public Guid itemId
```

### Field Value

[Guid](#)

## materialId

```
public Guid materialId
```

### Field Value

[Guid](#)

## slotIndex

```
public int slotIndex
```

### Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetEnhancementResult(Row, IRandom)

```
public static ItemEnhancement10.EnhancementResult  
GetEnhancementResult(EnhancementCostSheetV2.Row row, IRandom random)
```

### Parameters

row [EnhancementCostSheetV2.Row](#)

random IRandom

### Returns

[ItemEnhancement10.EnhancementResult](#)

## GetEquipmentMaxLevel(Equipment, EnhancementCostSheetV2)

```
public static int GetEquipmentMaxLevel(Equipment equipment,  
EnhancementCostSheetV2 sheet)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV2](#)

### Returns

[int](#)

## GetFeeStoreAddress()

```
public static Address GetFeeStoreAddress()
```

### Returns

Address

## GetRequiredAp()

```
public static int GetRequiredAp()
```

### Returns

[int](#)

## GetRequiredBlockCount(Row, EnhancementResult)

```
public static int GetRequiredBlockCount(EnhancementCostSheetV2.Row row,  
ItemEnhancement10.EnhancementResult result)
```

## Parameters

row [EnhancementCostSheetV2.Row](#)

result [ItemEnhancement10.EnhancementResult](#)

## Returns

[int](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

## TryGetRow(Equipment, EnhancementCostSheetV2, out Row)

```
public static bool TryGetRow(Equipment equipment, EnhancementCostSheetV2 sheet, out  
EnhancementCostSheetV2.Row row)
```

## Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV2](#)

row [EnhancementCostSheetV2.Row](#)

## Returns

[bool](#) ↗

# Enum ItemEnhancement10.EnhancementResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ItemEnhancement10.EnhancementResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Fail = 2

GreatSuccess = 0

Success = 1

# Class ItemEnhancement10.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancement10.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← ItemEnhancement10.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### actionPoint

```
public int actionPoint
```

#### Field Value

[int](#)

### enhancementResult

```
public ItemEnhancement10.EnhancementResult enhancementResult
```

#### Field Value

[ItemEnhancement10.EnhancementResult](#)

### gold

```
public BigInteger gold
```

#### Field Value

[BigInteger](#)

### id

```
public Guid id
```

#### Field Value

[Guid](#)

## materialItemIdList

```
public IEnumerable<Guid> materialItemIdList
```

Field Value

[IEnumerable](#)<[Guid](#)>

## preItemUsable

```
public ItemUsable preItemUsable
```

Field Value

[ItemUsable](#)

## Properties

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ItemEnhancement11

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/1164>

```
[Serializable]
[ActionType("item_enhancement11")]
public class ItemEnhancement11 : GameAction, IAction, IItemEnhancementV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement11

## Implements

IAction, IItemEnhancementV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

## itemId

```
public Guid itemId
```

### Field Value

[Guid](#)

## materialId

```
public Guid materialId
```

### Field Value

[Guid](#)

## slotIndex

```
public int slotIndex
```

### Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetEnhancementResult(Row, IRandom)

```
public static ItemEnhancement11.EnhancementResult  
GetEnhancementResult(EnhancementCostSheetV2.Row row, IRandom random)
```

### Parameters

row [EnhancementCostSheetV2.Row](#)

random IRandom

### Returns

[ItemEnhancement11.EnhancementResult](#)

## GetEquipmentMaxLevel(Equipment, EnhancementCostSheetV2)

```
public static int GetEquipmentMaxLevel(Equipment equipment,  
EnhancementCostSheetV2 sheet)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV2](#)

### Returns

[int](#)

## GetRequiredAp()

```
public static int GetRequiredAp()
```

### Returns

[int](#)

## GetRequiredBlockCount(Row, EnhancementResult)

```
public static int GetRequiredBlockCount(EnhancementCostSheetV2.Row row,  
ItemEnhancement11.EnhancementResult result)
```

### Parameters

row [EnhancementCostSheetV2.Row](#)

result [ItemEnhancement11.EnhancementResult](#)

### Returns

[int](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

## TryGetRow(Equipment, EnhancementCostSheetV2, out Row)

```
public static bool TryGetRow(Equipment equipment, EnhancementCostSheetV2 sheet, out EnhancementCostSheetV2.Row row)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV2](#)

row [EnhancementCostSheetV2.Row](#)

### Returns

[bool](#)

# Enum ItemEnhancement11.EnhancementResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ItemEnhancement11.EnhancementResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Fail = 2

GreatSuccess = 0

Success = 1

# Class ItemEnhancement11.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancement11.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← ItemEnhancement11.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### CRYSTAL

```
public FungibleAssetValue CRYSTAL
```

#### Field Value

FungibleAssetValue

### actionPoint

```
public int actionPoint
```

#### Field Value

[int](#)

### enhancementResult

```
public ItemEnhancement11.EnhancementResult enhancementResult
```

#### Field Value

[ItemEnhancement11.EnhancementResult](#)

### gold

```
public BigInteger gold
```

#### Field Value

[BigInteger](#)

## id

`public Guid id`

### Field Value

[Guid](#)

## materialItemIdList

`public IEnumerable<Guid> materialItemIdList`

### Field Value

[IEnumerable](#)<[Guid](#)>

## preItemUsable

`public ItemUsable preItemUsable`

### Field Value

[ItemUsable](#)

## Properties

### TypeId

`protected override string TypeId { get; }`

### Property Value

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ItemEnhancement12

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/2068>

```
[Serializable]
[ActionObsolete(7718878)]
[ActionTypes("item_enhancement12")]
public class ItemEnhancement12 : GameAction, IAction, IItemEnhancementV4
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement12

## Implements

IAction, IItemEnhancementV4

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### MaterialCountLimit

```
public const int MaterialCountLimit = 50
```

#### Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

### Field Value

Address

## itemId

```
public Guid itemId
```

### Field Value

[Guid](#)

## materialIds

```
public List<Guid> materialIds
```

### Field Value

[List](#)<[Guid](#)>

## slotIndex

```
public int slotIndex
```

### Field Value

[int](#)

## Properties

# PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetEquipmentMaxLevel(Equipment, EnhancementCostSheetV3)

```
public static int GetEquipmentMaxLevel(Equipment equipment,  
EnhancementCostSheetV3 sheet)
```

Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

Returns

[int](#)

## GetRequiredAp()

```
public static int GetRequiredAp()
```

Returns

[int](#)

## GetRequiredBlockCount(Equipment, Equipment, EnhancementCostSheetV3)

```
public static int GetRequiredBlockCount(Equipment preEquipment, Equipment targetEquipment, EnhancementCostSheetV3 sheet)
```

Parameters

preEquipment [Equipment](#)

targetEquipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

Returns

[int](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

## TryGetRow(Equipment, EnhancementCostSheetV3, out Row)

```
public static bool TryGetRow(Equipment equipment, EnhancementCostSheetV3 sheet, out EnhancementCostSheetV3.Row row)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

row [EnhancementCostSheetV3.Row](#)

### Returns

[bool](#)

# Enum ItemEnhancement12.EnhancementResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ItemEnhancement12.EnhancementResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Success = 1

# Class ItemEnhancement12.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancement12.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← ItemEnhancement12.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### CRYSTAL

```
public FungibleAssetValue CRYSTAL
```

#### Field Value

FungibleAssetValue

### actionPoint

```
public int actionPoint
```

#### Field Value

[int](#)

### enhancementResult

```
public ItemEnhancement12.EnhancementResult enhancementResult
```

#### Field Value

[ItemEnhancement12.EnhancementResult](#)

### gold

```
public BigInteger gold
```

#### Field Value

[BigInteger](#)

## id

```
public Guid id
```

### Field Value

[Guid](#)

## materialItemIdList

```
public IEnumerable<Guid> materialItemIdList
```

### Field Value

[IEnumerable](#)<[Guid](#)>

## preItemUsable

```
public ItemUsable preItemUsable
```

### Field Value

[ItemUsable](#)

## Properties

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ItemEnhancement13

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/2068>

```
[Serializable]
[ActionObsolete(8324909)]
[ActionTypes("item_enhancement13")]
public class ItemEnhancement13 : GameAction, IAction, IItemEnhancementV4
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement13

## Implements

IAction, IItemEnhancementV4

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### MaterialCountLimit

```
public const int MaterialCountLimit = 50
```

#### Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

### Field Value

Address

## itemId

```
public Guid itemId
```

### Field Value

[Guid](#)

## materialIds

```
public List<Guid> materialIds
```

### Field Value

[List](#)<[Guid](#)>

## slotIndex

```
public int slotIndex
```

### Field Value

[int](#)

## Properties

# PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetEquipmentMaxLevel(Equipment, EnhancementCostSheetV3)

```
public static int GetEquipmentMaxLevel(Equipment equipment,  
EnhancementCostSheetV3 sheet)
```

Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

Returns

[int](#)

## GetRequiredAp()

```
public static int GetRequiredAp()
```

Returns

[int](#)

## GetRequiredBlockCount(Equipment, Equipment, EnhancementCostSheetV3)

```
public static int GetRequiredBlockCount(Equipment preEquipment, Equipment targetEquipment, EnhancementCostSheetV3 sheet)
```

Parameters

preEquipment [Equipment](#)

targetEquipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

Returns

[int](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

## TryGetRow(Equipment, EnhancementCostSheetV3, out Row)

```
public static bool TryGetRow(Equipment equipment, EnhancementCostSheetV3 sheet, out EnhancementCostSheetV3.Row row)
```

### Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV3](#)

row [EnhancementCostSheetV3.Row](#)

### Returns

[bool](#)

# Enum ItemEnhancement13.EnhancementResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ItemEnhancement13.EnhancementResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Success = 1

# Class ItemEnhancement13.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancement13.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← ItemEnhancement13.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### CRYSTAL

```
public FungibleAssetValue CRYSTAL
```

#### Field Value

FungibleAssetValue

### actionPoint

```
public int actionPoint
```

#### Field Value

[int](#)

### enhancementResult

```
public ItemEnhancement13.EnhancementResult enhancementResult
```

#### Field Value

[ItemEnhancement13.EnhancementResult](#)

### gold

```
public BigInteger gold
```

#### Field Value

[BigInteger](#)

## id

`public Guid id`

### Field Value

[Guid](#)

## materialItemIdList

`public IEnumerable<Guid> materialItemIdList`

### Field Value

[IEnumerable](#)<[Guid](#)>

## preItemUsable

`public ItemUsable preItemUsable`

### Field Value

[ItemUsable](#)

## Properties

### TypeId

`protected override string TypeId { get; }`

### Property Value

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ItemEnhancement7

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("item_enhancement7")]
public class ItemEnhancement7 : GameAction, IAction, IItemEnhancementV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement7

## Implements

IAction, IItemEnhancementV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BlacksmithAddress

```
public static readonly Address BlacksmithAddress
```

#### Field Value

Address

### RequiredBlockCount

```
public const int RequiredBlockCount = 1
```

Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## itemId

```
public Guid itemId
```

Field Value

[Guid](#)

## materialId

```
public Guid materialId
```

Field Value

[Guid](#)

## slotIndex

```
public int slotIndex
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## GetRequiredAp()

```
public static int GetRequiredAp()
```

Returns

[int](#)

## GetRequiredNCG(EnhancementCostSheet, int, int)

```
public static BigInteger GetRequiredNCG(EnhancementCostSheet costSheet, int grade,  
int level)
```

Parameters

costSheet [EnhancementCostSheet](#)

grade [int](#)

level [int](#)

Returns

[BigInteger](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# UpgradeEquipment(Equipment)

```
public static Equipment UpgradeEquipment(Equipment equipment)
```

Parameters

equipment [Equipment](#)

Returns

[Equipment](#)

# Class ItemEnhancement7.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancement7.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← ItemEnhancement7.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### actionPoint

```
public int actionPoint
```

#### Field Value

[int](#)

### gold

```
public BigInteger gold
```

#### Field Value

[BigInteger](#)

### id

```
public Guid id
```

#### Field Value

[Guid](#)

### materialItemIdList

```
public IEnumerable<Guid> materialItemIdList
```

#### Field Value

[IEnumerable](#) <[Guid](#)>

## Properties

### TypeId

```
protected override string TypeId { get; }
```

#### Property Value

[string](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

#### Returns

IValue

# Class ItemEnhancement9

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("item_enhancement9")]
public class ItemEnhancement9 : GameAction, IAction, IItemEnhancementV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ItemEnhancement9

## Implements

IAction, IItemEnhancementV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BlacksmithAddress

```
public static readonly Address BlacksmithAddress
```

#### Field Value

Address

### avatarAddress

```
public Address avatarAddress
```

## Field Value

Address

## itemId

```
public Guid itemId
```

## Field Value

[Guid](#)

## materialId

```
public Guid materialId
```

## Field Value

[Guid](#)

## slotIndex

```
public int slotIndex
```

## Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetEnhancementResult(Row, IRandom)

```
public static ItemEnhancement9.EnhancementResult  
GetEnhancementResult(EnhancementCostSheetV2.Row row, IRandom random)
```

### Parameters

row [EnhancementCostSheetV2.Row](#)

`random` IRandom

Returns

[ItemEnhancement9.EnhancementResult](#)

## GetEquipmentMaxLevel(Equipment, EnhancementCostSheetV2)

```
public static int GetEquipmentMaxLevel(Equipment equipment,  
EnhancementCostSheetV2 sheet)
```

Parameters

`equipment` [Equipment](#)

`sheet` [EnhancementCostSheetV2](#)

Returns

[int](#)

## GetRequiredAp()

```
public static int GetRequiredAp()
```

Returns

[int](#)

## GetRequiredBlockCount(Row, EnhancementResult)

```
public static int GetRequiredBlockCount(EnhancementCostSheetV2.Row row,  
ItemEnhancement9.EnhancementResult result)
```

Parameters

row [EnhancementCostSheetV2.Row](#)

result [ItemEnhancement9EnhancementResult](#)

Returns

[int](#)

**LoadPlainValueInternal(IImmutableDictionary<string, IValue>)**

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

**TryGetRow(Equipment, EnhancementCostSheetV2, out Row)**

```
public static bool TryGetRow(Equipment equipment, EnhancementCostSheetV2 sheet, out EnhancementCostSheetV2.Row row)
```

Parameters

equipment [Equipment](#)

sheet [EnhancementCostSheetV2](#)

row [EnhancementCostSheetV2.Row](#)

Returns

[bool](#)

# Enum ItemEnhancement9.EnhancementResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ItemEnhancement9.EnhancementResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Fail = 2

GreatSuccess = 0

Success = 1

# Class ItemEnhancement9.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancement9.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← ItemEnhancement9.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel()

```
public ResultModel()
```

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

serialized Dictionary

## Fields

### actionPoint

```
public int actionPoint
```

#### Field Value

[int](#)

### enhancementResult

```
public ItemEnhancement9.EnhancementResult enhancementResult
```

#### Field Value

[ItemEnhancement9.EnhancementResult](#)

### gold

```
public BigInteger gold
```

#### Field Value

[BigInteger](#)

### id

```
public Guid id
```

#### Field Value

[Guid](#)

## materialItemIdList

```
public IEnumerable<Guid> materialItemIdList
```

Field Value

[IEnumerable](#)<[Guid](#)>

## preItemUsable

```
public ItemUsable preItemUsable
```

Field Value

[ItemUsable](#)

## Properties

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ItemProductInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class ItemProductInfo : IProductInfo
```

## Inheritance

[object](#) ← ItemProductInfo

## Implements

[IProductInfo](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemProductInfo()

```
public ItemProductInfo()
```

### ItemProductInfo(List)

```
public ItemProductInfo(List serialized)
```

## Parameters

serialized List

## Properties

## AgentAddress

```
public Address AgentAddress { get; set; }
```

Property Value

Address

## AvatarAddress

```
public Address AvatarAddress { get; set; }
```

Property Value

Address

## ItemSubType

```
public ItemSubType ItemSubType { get; set; }
```

Property Value

[ItemSubType](#)

## Legacy

```
public bool Legacy { get; set; }
```

Property Value

[bool](#) ↗

## Price

```
public FungibleAssetValue Price { get; set; }
```

Property Value

FungibleAssetValue

## ProductId

```
public Guid ProductId { get; set; }
```

Property Value

[Guid](#)

## TradableId

```
public Guid TradableId { get; set; }
```

Property Value

[Guid](#)

## Type

```
public ProductType Type { get; set; }
```

Property Value

[ProductType](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

## ValidateType()

```
public void ValidateType()
```

# Class JoinArena

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("join_arena4")]
public class JoinArena : GameAction, IAction, IJoinArenaV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← JoinArena

## Implements

IAction, IJoinArenaV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

## championshipId

```
public int championshipId
```

### Field Value

[int](#)

## costumes

```
public List<Guid> costumes
```

### Field Value

[List](#)<[Guid](#)>

## equipments

```
public List<Guid> equipments
```

### Field Value

[List](#)<[Guid](#)>

## round

```
public int round
```

### Field Value

[int](#)

## runeInfos

```
public List<RuneSlotInfo> runeInfos
```

## Field Value

[List](#)<[RuneSlotInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class JoinArena1

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1495>

```
[Serializable]
[ActionObsolete(7000000)]
[ActionTypes("join_arena")]
public class JoinArena1 : GameAction, IAction, IJoinArenaV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [JoinArena1](#)

## Implements

IAction, IJoinArenaV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

## Field Value

Address

## championshipId

```
public int championshipId
```

### Field Value

[int](#)

## costumes

```
public List<Guid> costumes
```

### Field Value

[List](#)<[Guid](#)>

## equipments

```
public List<Guid> equipments
```

### Field Value

[List](#)<[Guid](#)>

## round

```
public int round
```

### Field Value

[int](#)

## Properties

# PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called “dirty”).

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class JoinArena3

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1663>

```
[Serializable]
[ActionObsolete(8324909)]
[ActionTypes("join_arena3")]
public class JoinArena3 : GameAction, IAction, IJoinArenaV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← JoinArena3

## Implements

IAction, IJoinArenaV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

## Field Value

Address

## championshipId

```
public int championshipId
```

### Field Value

[int](#)

## costumes

```
public List<Guid> costumes
```

### Field Value

[List](#)<[Guid](#)>

## equipments

```
public List<Guid> equipments
```

### Field Value

[List](#)<[Guid](#)>

## round

```
public int round
```

### Field Value

[int](#)

## runeInfos

```
public List<RuneSlotInfo> runeInfos
```

## Field Value

[List](#)<[RuneSlotInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class MemoLengthOverflowException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MemoLengthOverflowException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
MemoLengthOverflowException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### MemoLengthOverflowException(SerializationInfo, StreamingContext)

```
protected MemoLengthOverflowException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MemoLengthOverflowException(string)

```
public MemoLengthOverflowException(string message)
```

### Parameters

message [string](#)

# Class MigrateAgentAvatar

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("migrate_agent_avatar")]
public class MigrateAgentAvatar : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MigrateAgentAvatar

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MigrateAgentAvatar()

```
public MigrateAgentAvatar()
```

## Fields

### AgentAddresses

```
public List<Address> AgentAddresses
```

## Field Value

## TypeIdentifier

```
public const string TypeIdentifier = "migrate_agent_avatar"
```

### Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

[IActionContext](#)

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` `IValue`

Data (made by `Libplanet.Action.IAction.PlainValue` property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class MigrateFee

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("migrate_fee")]
public class MigrateFee : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MigrateFee

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MigrateFee()

```
public MigrateFee()
```

## Fields

### Memo

```
public string Memo
```

### Field Value

[string](#)

## TransferData

```
public List<(Address sender, Address recipient, BigInteger amount)> TransferData
```

### Field Value

[List](#)<(Address [sender](#), Address [recipient](#), [BigInteger](#) [amount](#))>

## TypeIdentifier

```
public const string TypeIdentifier = "migrate_fee"
```

### Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

#### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class MigrateMonsterCollection

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

An action to claim remained monster collection rewards and to migrate [MonsterCollectionState](#) into [LegacyStakeState](#) without cancellation, to keep its staked period.

```
[ActionType("migrate_monster_collection")]
public class MigrateMonsterCollection : ActionBase,
IAction, IMigrateMonsterCollectionV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← MigrateMonsterCollection

## Implements

IAction, IMigrateMonsterCollectionV1

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MigrateMonsterCollection()

```
public MigrateMonsterCollection()
```

### MigrateMonsterCollection(Address)

```
public MigrateMonsterCollection(Address avatarAddress)
```

## Parameters

avatarAddress Address

## Properties

### AvatarAddress

```
public Address AvatarAddress { get; }
```

Property Value

Address

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but

equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class MigrationActivatedAccountsState

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("migration_activated_accounts_state")]
public class MigrationActivatedAccountsState : GameAction, IAction,
IMigrationActivatedAccountsStateV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← MigrationActivatedAccountsState

## Implements

IAction, IMigrationActivatedAccountsStateV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class MintAssets

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("mint_assets")]
public class MintAssets : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MintAssets

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MintAssets()

```
public MintAssets()
```

### MintAssets(IEnumerable<MintSpec>, string?)

```
public MintAssets(IEnumerable<MintAssets.MintSpec> specs, string? memo)
```

## Parameters

specs [IEnumerable](#)<[MintAssets.MintSpec](#)>

`memo` [string](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "mint_assets"
```

#### Field Value

[string](#)

## Properties

### Memo

```
public string? Memo { get; }
```

#### Property Value

[string](#)

### MintSpecs

```
public List<MintAssets.MintSpec>? MintSpecs { get; }
```

#### Property Value

[List](#)<[MintAssets.MintSpec](#)>

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Struct MintAssets.MintSpec

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public readonly struct MintAssets.MintSpec
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### MintSpec(List)

```
public MintSpec(List bencoded)
```

#### Parameters

bencoded List

### MintSpec(Address, FungibleAssetValue?, FungibleItemValue?)

```
public MintSpec(Address recipient, FungibleAssetValue? assets,  
FungibleItemValue? items)
```

#### Parameters

recipient Address

assets FungibleAssetValue?

items [FungibleItemValue?](#)

# Properties

## Assets

```
public FungibleAssetValue? Assets { get; }
```

Property Value

FungibleAssetValue?

## Items

```
public FungibleItemValue? Items { get; }
```

Property Value

[FungibleItemValue?](#)

## Recipient

```
public Address Recipient { get; }
```

Property Value

Address

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns



# Class MonsterCollectionExistingClaimableException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionExistingClaimableException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← MonsterCollectionExistingClaimableException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### MonsterCollectionExistingClaimableException()

```
public MonsterCollectionExistingClaimableException()
```

### MonsterCollectionExistingClaimableException(SerializationInfo, StreamingContext)

```
protected MonsterCollectionExistingClaimableException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MonsterCollectionExistingClaimableException(string)

```
public MonsterCollectionExistingClaimableException(string message)
```

## Parameters

message [string](#)

## MonsterCollectionExistingClaimableException(string, Exception)

```
public MonsterCollectionExistingClaimableException(string message,  
Exception innerException)
```

## Parameters

message [string](#)

innerException [Exception](#)

# Class MonsterCollectionExistingException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

An exception thrown when there is unexpected monster collection.

```
[Serializable]
public class MonsterCollectionExistingException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← MonsterCollectionExistingException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### MonsterCollectionExistingException()

```
public MonsterCollectionExistingException()
```

### MonsterCollectionExistingException(SerializationInfo, StreamingContext)

```
protected MonsterCollectionExistingException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MonsterCollectionExistingException(string)

```
public MonsterCollectionExistingException(string message)
```

Parameters

message [string](#)

## MonsterCollectionExistingException(string, Exception)

```
public MonsterCollectionExistingException(string message, Exception innerException)
```

Parameters

message [string](#)

innerException [Exception](#)

# Class MonsterCollectionExpiredException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionExpiredException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
MonsterCollectionExpiredException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

MonsterCollectionExpiredException(SerializationInfo,  
StreamingContext)

```
protected MonsterCollectionExpiredException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MonsterCollectionExpiredException(string)

```
public MonsterCollectionExpiredException(string msg)
```

### Parameters

msg [string](#)

# Class MonsterCollectionLevelException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionLevelException : InvalidOperationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
MonsterCollectionLevelException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### MonsterCollectionLevelException()

```
public MonsterCollectionLevelException()
```

### MonsterCollectionLevelException(SerializationInfo, StreamingContext)

```
protected MonsterCollectionLevelException(SerializationInfo info,
```

```
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MonsterCollectionLevelException(string)

```
public MonsterCollectionLevelException(string msg)
```

## Parameters

msg [string](#)

# Class NotEnoughActionPointException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughActionPointException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughActionPointException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughActionPointException(SerializationInfo, StreamingContext)

```
public NotEnoughActionPointException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughActionPointException(string)

```
public NotEnoughActionPointException(string msg)
```

### Parameters

msg [string](#)

# Class NotEnoughAvatarLevelException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughAvatarLevelException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughAvatarLevelException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughAvatarLevelException(int, bool, int, int)

```
public NotEnoughAvatarLevelException(int itemId, bool isMadeWithMimisbrunnrRecipe,
int require, int current)
```

## Parameters

itemId [int](#)

isMadeWithMimisbrunnrRecipe [bool](#)

require [int](#)

current [int](#)

## NotEnoughAvatarLevelException(int, int)

```
public NotEnoughAvatarLevelException(int require, int current)
```

### Parameters

require [int](#)

current [int](#)

## NotEnoughAvatarLevelException(SerializationInfo, StreamingContext)

```
public NotEnoughAvatarLevelException(SerializationInfo info,  
StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughAvatarLevelException(string)

```
public NotEnoughAvatarLevelException(string message)
```

### Parameters

message [string](#)

# Class NotEnoughClearedStageLevelException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughClearedStageLevelException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughClearedStageLevelException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughClearedStageLevelException(SerializationInfo , StreamingContext)

```
public NotEnoughClearedStageLevelException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughClearedStageLevelException(string)

```
public NotEnoughClearedStageLevelException(string message)
```

### Parameters

message [string](#)

## NotEnoughClearedStageLevelException(string, int, int)

```
public NotEnoughClearedStageLevelException(string addressesHex, int require,  
int current)
```

### Parameters

addressesHex [string](#)

require [int](#)

current [int](#)

## NotEnoughClearedStageLevelException(string, string, int, int)

```
public NotEnoughClearedStageLevelException(string actionType, string addressesHex,  
int require, int current)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

require [int](#)

current [int](#)



# Class NotEnoughCombatPointException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class NotEnoughCombatPointException : InvalidOperationException,  
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
NotEnoughCombatPointException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughCombatPointException(SerializationInfo, StreamingContext)

```
protected NotEnoughCombatPointException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughCombatPointException(string)

```
public NotEnoughCombatPointException(string s)
```

### Parameters

s [string](#)

# Class NotEnoughFungibleAssetValueException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughFungibleAssetValueException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughFungibleAssetValueException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughFungibleAssetValueException(SerializationInfo, StreamingContext)

```
public NotEnoughFungibleAssetValueException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughFungibleAssetValueException(string, Exception)

```
public NotEnoughFungibleAssetValueException(string message, Exception innerException = null)
```

### Parameters

message [string](#)

innerException [Exception](#)

## NotEnoughFungibleAssetValueException(string, BigInteger, FungibleAssetValue)

```
public NotEnoughFungibleAssetValueException(string addressesHex, BigInteger require, FungibleAssetValue current)
```

### Parameters

addressesHex [string](#)

require [BigInteger](#)

current FungibleAssetValue

## NotEnoughFungibleAssetValueException(string, string, FungibleAssetValue, FungibleAssetValue)

```
public NotEnoughFungibleAssetValueException(string actionType, string addressesHex, FungibleAssetValue require, FungibleAssetValue current)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

require FungibleAssetValue

current FungibleAssetValue

## NotEnoughFungibleAssetValueException(string, string, FungibleAssetValue, BigInteger)

```
public NotEnoughFungibleAssetValueException(string actionType, string addressesHex,  
FungibleAssetValue require, BigInteger current)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

require FungibleAssetValue

current [BigInteger](#)

## NotEnoughFungibleAssetValueException(string, string, BigInteger, FungibleAssetValue)

```
public NotEnoughFungibleAssetValueException(string actionType, string addressesHex,  
BigInteger require, FungibleAssetValue current)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

require [BigInteger](#)

current FungibleAssetValue

## NotEnoughFungibleAssetValueException(string, string, BigInteger, BigInteger)

```
public NotEnoughFungibleAssetValueException(string actionType, string addressesHex,  
BigInteger require, BigInteger current)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

require [BigInteger](#)

current [BigInteger](#)

## NotEnoughFungibleAssetValueException(string, string, string, Exception)

```
public NotEnoughFungibleAssetValueException(string addressesHex, string require,  
string current, Exception innerException = null)
```

### Parameters

addressesHex [string](#)

require [string](#)

current [string](#)

innerException [Exception](#)

## NotEnoughFungibleAssetValueException(string, string, string, string, Exception)

```
public NotEnoughFungibleAssetValueException(string actionType, string addressesHex,  
string require, string current, Exception innerException = null)
```

## Parameters

actionType [string](#)

addressesHex [string](#)

require [string](#)

current [string](#)

innerException [Exception](#)

# Class NotEnoughHammerPointException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class NotEnoughHammerPointException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughHammerPointException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughHammerPointException(SerializationInfo, StreamingContext)

```
protected NotEnoughHammerPointException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughHammerPointException(string)

```
public NotEnoughHammerPointException(string s)
```

### Parameters

s [string](#)

# Class NotEnoughMaterialException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughMaterialException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
NotEnoughMaterialException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughMaterialException(SerializationInfo, StreamingContext)

```
protected NotEnoughMaterialException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughMaterialException(string)

```
public NotEnoughMaterialException(string s)
```

### Parameters

s [string](#)

# Class NotEnoughMedalException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughMedalException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughMedalException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughMedalException(SerializationInfo, StreamingContext)

```
public NotEnoughMedalException(SerializationInfo info, StreamingContext context)
```

#### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughMedalException(string)

```
public NotEnoughMedalException(string msg)
```

### Parameters

msg [string](#)

# Class NotEnoughRankException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughRankException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
NotEnoughRankException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughRankException()

```
public NotEnoughRankException()
```

### NotEnoughRankException(SerializationInfo, StreamingContext)

```
public NotEnoughRankException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

# Class NotEnoughStarException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class NotEnoughStarException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughStarException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughStarException(SerializationInfo, StreamingContext)

```
protected NotEnoughStarException(SerializationInfo info, StreamingContext context)
```

#### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

### NotEnoughStarException(string)

```
public NotEnoughStarException(string s)
```

## Parameters

s [string](#)

# Class NotEnoughWeeklyArenaChallengeCountException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughWeeklyArenaChallengeCountException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughWeeklyArenaChallengeCountException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughWeeklyArenaChallengeCountException()

```
public NotEnoughWeeklyArenaChallengeCountException()
```

### NotEnoughWeeklyArenaChallengeCountException(SerializationInfo, StreamingContext)

```
public NotEnoughWeeklyArenaChallengeCountException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughWeeklyArenaChallengeCountException(string )

```
public NotEnoughWeeklyArenaChallengeCountException(string message)
```

## Parameters

message [string](#)

## Fields

### BaseMessage

```
public const string BaseMessage = "Aborted as the arena state reached the
daily limit."
```

## Field Value

[string](#)

# Class OrderIdDoesNotExistException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderIdDoesNotExistException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← OrderIdDoesNotExistException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### OrderIdDoesNotExistException(SerializationInfo, StreamingContext)

```
protected OrderIdDoesNotExistException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## OrderIdDoesNotExistException(string)

```
public OrderIdDoesNotExistException(string msg)
```

### Parameters

msg [string](#)

# Class PatchTableSheet

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at  
<https://github.com/planetarium/lib9c/pull/42> Updated at  
<https://github.com/planetarium/lib9c/pull/101> Updated at  
<https://github.com/planetarium/lib9c/pull/287> Updated at  
<https://github.com/planetarium/lib9c/pull/315> Updated at  
<https://github.com/planetarium/lib9c/pull/957> Updated at  
<https://github.com/planetarium/lib9c/pull/1560>

```
[Serializable]
[ActionType("patch_table_sheet")]
public class PatchTableSheet : GameAction, IAction, IPatchTableSheetV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← PatchTableSheet

## Implements

IAction, IPatchTableSheetV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### TableCsv

```
public string TableCsv
```

## Field Value

[string](#)

## TableName

```
public string TableName
```

## Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class PendingActivationDoesNotExistsException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PendingActivationDoesNotExistsException : ActivationException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [ActivationException](#) ← PendingActivationDoesNotExistsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### PendingActivationDoesNotExistsException(Address)

```
public PendingActivationDoesNotExistsException(Address pendingAddress)
```

## Parameters

pendingAddress Address

## PendingActivationDoesNotExistException(SerializationInfo, StreamingContext)

```
public PendingActivationDoesNotExistException(SerializationInfo info,  
StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### PendingAddress

```
public Address PendingAddress { get; }
```

### Property Value

Address

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

`context StreamingContext`

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

[ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class PermissionDeniedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PermissionDeniedException : AdminPermissionException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [AdminPermissionException](#) ← [PermissionDeniedException](#)

## Implements

[ISerializable](#)

## Inherited Members

[AdminPermissionException.Policy](#) , [Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PermissionDeniedException(AdminState, Address)

```
public PermissionDeniedException(AdminState policy, Address signer)
```

#### Parameters

policy [AdminState](#)

signer Address

### PermissionDeniedException(SerializationInfo, StreamingContext)

```
public PermissionDeniedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Signer

```
public Address Signer { get; }
```

## Property Value

Address

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

context [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

### [ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class PetEnhancement

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("pet_enhancement")]
public class PetEnhancement : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← PetEnhancement

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ActionTypeIdentifier

```
public const string ActionTypeIdentifier = "pet_enhancement"
```

#### Field Value

[string](#)

### AvatarAddress

```
public Address AvatarAddress
```

## Field Value

Address

## PetId

```
public int PetId
```

## Field Value

[int](#)

## TargetLevel

```
public int TargetLevel
```

## Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class PetIsLockedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetIsLockedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← PetIsLockedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PetIsLockedException(SerializationInfo, StreamingContext)

```
public PetIsLockedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## PetIsLockedException(string)

```
public PetIsLockedException(string msg)
```

### Parameters

msg [string](#)

# Class PlayCountIsZeroException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PlayCountIsZeroException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← PlayCountIsZeroException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PlayCountIsZeroException(SerializationInfo, StreamingContext)

```
public PlayCountIsZeroException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## PlayCountIsZeroException(string)

```
public PlayCountIsZeroException(string msg)
```

### Parameters

msg [string](#)

# Class PolicyExpiredException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PolicyExpiredException : AdminPermissionException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [AdminPermissionException](#) ← [PolicyExpiredException](#)

## Implements

[ISerializable](#)

## Inherited Members

[AdminPermissionException.Policy](#) , [Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PolicyExpiredException(AdminState, long)

```
public PolicyExpiredException(AdminState policy, long blockIndex)
```

## Parameters

policy [AdminState](#)

blockIndex [long](#)

### PolicyExpiredException(SerializationInfo, StreamingContext)

```
public PolicyExpiredException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### BlockIndex

```
public long BlockIndex { get; }
```

## Property Value

[long](#)

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

context [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

### [ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class PrepareRewardAssets

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("prepare_reward_assets")]
public class PrepareRewardAssets : ActionBase, IAction, IPrepareRewardAssetsV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← PrepareRewardAssets

## Implements

IAction, IPrepareRewardAssetsV1

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### PrepareRewardAssets()

```
public PrepareRewardAssets()
```

### PrepareRewardAssets(Address, List<FungibleAssetValue>)

```
public PrepareRewardAssets(Address rewardPoolAddress, List<FungibleAssetValue>
assets)
```

## Parameters

`rewardPoolAddress` Address

`assets` [List](#)<FungibleAssetValue>

## Fields

### Assets

`public List<FungibleAssetValue> Assets`

#### Field Value

[List](#)<FungibleAssetValue>

## RewardPoolAddress

`public Address RewardPoolAddress`

#### Field Value

Address

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

`public override IValue PlainValue { get; }`

#### Property Value

## IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

## **plainValue** **IValue**

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### **See Also**

[PlainValue](#)

# Class ProductNotFoundException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ProductNotFoundException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
ProductNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ProductNotFoundException(SerializationInfo, StreamingContext)

```
public ProductNotFoundException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ProductNotFoundException(string)

```
public ProductNotFoundException(string msg)
```

### Parameters

msg [string](#)

# Class PurchasInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PurchaseInfo : IComparable<PurchaseInfo>, IComparable, IPurchaseInfo
```

## Inheritance

[object](#) ← PurchasInfo

## Implements

[IComparable](#)<[PurchaseInfo](#)>, [IComparable](#), [IPurchaseInfo](#)

## Inherited Members

[object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

### PurchasInfo(Dictionary)

```
public PurchaseInfo(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### PurchasInfo(Guid, Guid, Address, Address, ItemSubType, FungibleAssetValue)

```
public PurchaseInfo(Guid orderId, Guid tradableId, Address agentAddress, Address
avatarAddress, ItemSubType type, FungibleAssetValue itemPrice)
```

#### Parameters

`orderId` [Guid ↗](#)

`tradableId` [Guid ↗](#)

`agentAddress` Address

`avatarAddress` Address

`type` [ItemSubType](#)

`itemPrice` FungibleAssetValue

## Fields

### OrderId

`public readonly Guid OrderId`

Field Value

[Guid ↗](#)

### TradableId

`public readonly Guid TradableId`

Field Value

[Guid ↗](#)

## Properties

### ItemSubType

`public ItemSubType ItemSubType { get; }`

Property Value

[ItemSubType](#)

Price

```
public FungibleAssetValue Price { get; }
```

Property Value

FungibleAssetValue

SellerAgentAddress

```
public Address SellerAgentAddress { get; }
```

Property Value

Address

SellerAvatarAddress

```
public Address SellerAvatarAddress { get; }
```

Property Value

Address

Methods

**CompareTo(PurchaseInfo)**

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(PurchaseInfo other)
```

## Parameters

### other [PurchaseInfo](#)

An object to compare with this instance.

## Returns

### [int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <a href="#">other</a> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <a href="#">other</a> .
<b>Greater than zero</b>	This instance follows <a href="#">other</a> in the sort order.

## CompareTo(object)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(object obj)
```

## Parameters

### obj [object](#)

An object to compare with this instance.

Returns

[int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <code>obj</code> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <code>obj</code> .
<b>Greater than zero</b>	This instance follows <code>obj</code> in the sort order.

Exceptions

[ArgumentException](#)

`obj` is not the same type as this instance.

## Equals(PurchaseInfo)

`protected bool Equals(PurchaseInfo other)`

Parameters

`other` [PurchaseInfo](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

`public override bool Equals(object obj)`

## Parameters

**obj** [object](#)

The object to compare with the current object.

## Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

## Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

## Returns

IValue

## Operators

### operator ==(PurchaseInfo, PurchaseInfo)

```
public static bool operator ==(PurchaseInfo left, PurchaseInfo right)
```

Parameters

`left` [PurchaseInfo](#)

`right` [PurchaseInfo](#)

Returns

[bool](#)

## operator >(PurchaseInfo, PurchaseInfo)

```
public static bool operator >(PurchaseInfo left, PurchaseInfo right)
```

Parameters

`left` [PurchaseInfo](#)

`right` [PurchaseInfo](#)

Returns

[bool](#)

## operator >=(PurchaseInfo, PurchaseInfo)

```
public static bool operator >=(PurchaseInfo left, PurchaseInfo right)
```

Parameters

`left` [PurchaseInfo](#)

`right` [PurchaseInfo](#)

Returns

[bool](#)

## operator !=(PurchaseInfo, PurchaseInfo)

```
public static bool operator !=(PurchaseInfo left, PurchaseInfo right)
```

Parameters

left [PurchaseInfo](#)

right [PurchaseInfo](#)

Returns

[bool](#)

## operator <(PurchaseInfo, PurchaseInfo)

```
public static bool operator <(PurchaseInfo left, PurchaseInfo right)
```

Parameters

left [PurchaseInfo](#)

right [PurchaseInfo](#)

Returns

[bool](#)

## operator <=(PurchaseInfo, PurchaseInfo)

```
public static bool operator <=(PurchaseInfo left, PurchaseInfo right)
```

Parameters

left [PurchaseInfo](#)

right [PurchaseInfo](#)

## Returns

[bool](#) ↗

# Class PurchaseInfo0

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PurchaseInfo0 : IComparable<PurchaseInfo0>, IComparable, IPurchaseInfo
```

## Inheritance

[object](#) ← PurchaseInfo0

## Implements

[IComparable](#)<PurchaseInfo0>, [IComparable](#), [IPurchaseInfo](#)

## Inherited Members

[object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

### PurchaseInfo0(Dictionary)

```
public PurchaseInfo0(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### PurchaseInfo0(Guid, Address, Address, ItemSubType, FungibleAssetValue)

```
public PurchaseInfo0(Guid id, Address agentAddress, Address avatarAddress,  
ItemSubType type, FungibleAssetValue itemPrice = default)
```

#### Parameters

**id** [Guid](#)

**agentAddress** Address

**avatarAddress** Address

**type** [ItemSubType](#)

**itemPrice** FungibleAssetValue

## Fields

### itemSubType

```
public readonly ItemSubType itemSubType
```

Field Value

[ItemSubType](#)

### price

```
public readonly FungibleAssetValue price
```

Field Value

FungibleAssetValue

### productId

```
public readonly Guid productId
```

Field Value

[Guid](#)

## **sellerAgentAddress**

```
public readonly Address sellerAgentAddress
```

Field Value

Address

## **sellerAvatarAddress**

```
public readonly Address sellerAvatarAddress
```

Field Value

Address

## **Methods**

### **CompareTo(PurchaseInfo0)**

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(PurchaseInfo0 other)
```

Parameters

**other** [PurchaseInfo0](#)

An object to compare with this instance.

Returns

[int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <i>other</i> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <i>other</i> .
<b>Greater than zero</b>	This instance follows <i>other</i> in the sort order.

## CompareTo(object)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(object obj)
```

### Parameters

*obj* [object](#)

An object to compare with this instance.

### Returns

[int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <i>obj</i> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <i>obj</i> .

Value	Meaning
<b>Greater than zero</b>	This instance follows <code>obj</code> in the sort order.

## Exceptions

### [ArgumentException](#)

`obj` is not the same type as this instance.

## Equals(PurchaseInfo0)

`protected bool Equals(PurchaseInfo0 other)`

### Parameters

`other` [PurchaseInfo0](#)

### Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

`public override bool Equals(object obj)`

### Parameters

`obj` [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Operators

### operator ==(PurchaseInfo0, PurchaseInfo0)

```
public static bool operator ==(PurchaseInfo0 left, PurchaseInfo0 right)
```

Parameters

left [PurchaseInfo0](#)

right [PurchaseInfo0](#)

Returns

[bool](#)

## operator >(PurchaseInfo0, PurchaseInfo0)

```
public static bool operator >(PurchaseInfo0 left, PurchaseInfo0 right)
```

Parameters

[left](#) [PurchaseInfo0](#)

[right](#) [PurchaseInfo0](#)

Returns

[bool](#)

## operator >=(PurchaseInfo0, PurchaseInfo0)

```
public static bool operator >=(PurchaseInfo0 left, PurchaseInfo0 right)
```

Parameters

[left](#) [PurchaseInfo0](#)

[right](#) [PurchaseInfo0](#)

Returns

[bool](#)

## operator !=(PurchaseInfo0, PurchaseInfo0)

```
public static bool operator !=(PurchaseInfo0 left, PurchaseInfo0 right)
```

Parameters

`left` [PurchaseInfo0](#)

`right` [PurchaseInfo0](#)

Returns

[bool](#) ↗

## operator <(PurchaseInfo0, PurchaseInfo0)

```
public static bool operator <(PurchaseInfo0 left, PurchaseInfo0 right)
```

Parameters

`left` [PurchaseInfo0](#)

`right` [PurchaseInfo0](#)

Returns

[bool](#) ↗

## operator <=(PurchaseInfo0, PurchaseInfo0)

```
public static bool operator <=(PurchaseInfo0 left, PurchaseInfo0 right)
```

Parameters

`left` [PurchaseInfo0](#)

`right` [PurchaseInfo0](#)

Returns

[bool](#) ↗

# Class Raid

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("raid7")]
public class Raid : GameAction, IAction, IRaidV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Raid

## Implements

IAction, IRaidV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

## CostumeIds

```
public List<Guid> CostumeIds
```

### Field Value

[List](#)<[Guid](#)>

## EquipmentIds

```
public List<Guid> EquipmentIds
```

### Field Value

[List](#)<[Guid](#)>

## FoodIds

```
public List<Guid> FoodIds
```

### Field Value

[List](#)<[Guid](#)>

## PayNcg

```
public bool PayNcg
```

### Field Value

[bool](#)

## RunelInfos

```
public List<RuneSlotInfo> RuneInfos
```

## Field Value

[List](#)<[RuneSlotInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class RankingBattle

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/941> Updated at <https://github.com/planetarium/lib9c/pull/1135>

```
[Serializable]
[ActionObsolete(8324909)]
[ActionType("ranking_battle12")]
public class RankingBattle : GameAction, IAction, IRankingBattleV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [RankingBattle](#)

## Implements

IAction, IRankingBattleV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### PreviousArenaInfo

```
public ArenaInfo PreviousArenaInfo
```

#### Field Value

[ArenaInfo](#)

## PreviousEnemyArenaInfo

```
public ArenaInfo PreviousEnemyArenaInfo
```

Field Value

[ArenaInfo](#)

## PreviousEnemyPlayerDigest

```
public EnemyPlayerDigest PreviousEnemyPlayerDigest
```

Field Value

[EnemyPlayerDigest](#)

## StageId

```
public const int StageId = 999999
```

Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## costumeIds

```
public List<Guid> costumeIds
```

Field Value

[List](#)<[Guid](#)>

## enemyAddress

```
public Address enemyAddress
```

Field Value

Address

## equipmentIds

```
public List<Guid> equipmentIds
```

Field Value

[List](#)<[Guid](#)>

## weeklyArenaAddress

```
public Address weeklyArenaAddress
```

Field Value

Address

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` [IActionContext](#)

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

[IWorld](#)

A map of changed states (so-called “dirty”).

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and [Libplanet.Action.IActionContexts](#), the same result should be returned. Side effects should

be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class RankingBattle11

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Updated at <https://github.com/planetarium/lib9c/pull/1176>

```
[Serializable]
[ActionType("ranking_battle11")]
[ActionObsolete(7200000)]
public class RankingBattle11 : GameAction, IAction, IRankingBattleV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RankingBattle11

## Implements

IAction, IRankingBattleV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ArenaInfo

```
public ArenaInfo ArenaInfo
```

#### Field Value

[ArenaInfo](#)

## EnemyArenaInfo

```
public ArenaInfo EnemyArenaInfo
```

Field Value

[ArenaInfo](#)

## EnemyPlayerDigest

```
public EnemyPlayerDigest EnemyPlayerDigest
```

Field Value

[EnemyPlayerDigest](#)

## EntranceFee

```
public static readonly BigInteger EntranceFee
```

Field Value

[BigInteger](#)

## StageId

```
public const int StageId = 999999
```

Field Value

[int](#)

## UpdateTargetBlockIndex

```
public const long UpdateTargetBlockIndex = 3808000
```

Field Value

[long](#)

## UpdateTargetWeeklyArenaIndex

```
public const int UpdateTargetWeeklyArenaIndex = 68
```

Field Value

[int](#)

## avatarAddress

```
public Address avatarAddress
```

Field Value

Address

## costumeIds

```
public List<Guid> costumeIds
```

Field Value

[List](#)<[Guid](#)>

## enemyAddress

```
public Address enemyAddress
```

## Field Value

Address

## equipmentIds

```
public List<Guid> equipmentIds
```

## Field Value

[List](#)<[Guid](#)>

## weeklyArenaAddress

```
public Address weeklyArenaAddress
```

## Field Value

Address

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class RankingExceededException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RankingExceededException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
RankingExceededException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RankingExceededException()

```
public RankingExceededException()
```

### RankingExceededException(SerializationInfo, StreamingContext)

```
protected RankingExceededException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#) ↗

context [StreamingContext](#) ↗

# Class RapidCombination

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2195>

```
[Serializable]
[ActionType("rapid_combination10")]
public class RapidCombination : GameAction, IAction, IRapidCombinationV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [RapidCombination](#)

## Implements

IAction, IRapidCombinationV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

# slotIndexList

```
public List<int> slotIndexList
```

## Field Value

[List](#)<[int](#)>

# Properties

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

### `IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class RapidCombination0

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("rapid_combination")]
public class RapidCombination0 : GameAction, IAction, IRapidCombinationV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RapidCombination0

## Implements

IAction, IRapidCombinationV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

### slotIndex

```
public int slotIndex
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### CalculateHourglassCount(GameConfigState, long)

```
public static int CalculateHourglassCount(GameConfigState state, long diff)
```

Parameters

state [GameConfigState](#)

diff [long](#)

Returns

[int](#)

### CalculateHourglassCount(decimal, long)

```
public static int CalculateHourglassCount(decimal hourglassPerBlock, long diff)
```

## Parameters

hourglassPerBlock [decimal](#)

diff [long](#)

## Returns

[int](#)

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

context [IActionContext](#)

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called “dirty”).

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

`LoadPlainValueInternal(IImmutableDictionary<string, IValue>)`

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class RapidCombination0.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RapidCombination0.ResultModel : CombinationConsumable5.ResultModel,
IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← [CombinationConsumable5.ResultModel](#) ←  
RapidCombination0.ResultModel

## Implements

[IState](#)

## Inherited Members

[CombinationConsumable5.ResultModel.materials](#) ,  
[CombinationConsumable5.ResultModel.id](#) , [CombinationConsumable5.ResultModel.gold](#) ,  
[CombinationConsumable5.ResultModel.actionPoint](#) ,  
[CombinationConsumable5.ResultModel.recipeId](#) ,  
[CombinationConsumable5.ResultModel.subRecipeId](#) ,  
[CombinationConsumable5.ResultModel.itemType](#) , [AttachmentActionResult.itemUsable](#) ,  
[AttachmentActionResult.costume](#) , [AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Fields

### cost

```
public Dictionary<Material, int> cost
```

### Field Value

[Dictionary](#)<[Material](#), [int](#)>

## Properties

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

[string](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

### Returns

IValue

# Class RapidCombination5

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("rapid_combination5")]
public class RapidCombination5 : GameAction, IAction, IRapidCombinationV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RapidCombination5

## Implements

IAction, IRapidCombinationV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

### slotIndex

```
public int slotIndex
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class RapidCombination5.ResultModel

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RapidCombination5.ResultModel : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← RapidCombination5.ResultModel

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ResultModel(Dictionary)

```
public ResultModel(Dictionary serialized)
```

#### Parameters

serialized Dictionary

## Fields

cost

```
public Dictionary<Material, int> cost
```

Field Value

[Dictionary](#) <[Material](#), [int](#)>

id

```
public Guid id
```

Field Value

[Guid](#)

## Properties

TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ReRegisterProduct

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("re_register_product2")]
public class ReRegisterProduct : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ReRegisterProduct

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### Capacity

```
public const int Capacity = 100
```

Field Value

[int](#)

## ChargeAp

`public bool ChargeAp`

Field Value

[bool](#)

## CostAp

`public const int CostAp = 5`

Field Value

[int](#)

## ReRegisterInfos

`public List<(IProductInfo, IRegisterInfo)> ReRegisterInfos`

Field Value

[List](#)<[IProductInfo](#), [IRegisterInfo](#)>

## Properties

### PlainValueInternal

`protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }`

## Property Value

[IImmutableDictionary<string, IValue>](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called "dirty").

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class RedeemCode

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/602> Updated at <https://github.com/planetarium/lib9c/pull/861> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("redeem_code3")]
public class RedeemCode : GameAction, IAction, IRedeemCodeV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [RedeemCode](#)

## Implements

IAction, IRedeemCodeV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RedeemCode()

```
public RedeemCode()
```

### RedeemCode(string, Address)

```
public RedeemCode(string code, Address avatarAddress)
```

## Parameters

code [string](#)

avatarAddress Address

## Properties

### AvatarAddress

```
public Address AvatarAddress { get; }
```

#### Property Value

Address

### Code

```
public string Code { get; }
```

#### Property Value

[string](#)

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

#### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class RedeemCode2

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("redeem_code2")]
public class RedeemCode2 : GameAction, IAction, IRedeemCodeV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [RedeemCode2](#)

## Implements

IAction, IRedeemCodeV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RedeemCode2()

```
public RedeemCode2()
```

### RedeemCode2(string, Address)

```
public RedeemCode2(string code, Address avatarAddress)
```

## Parameters

code [string](#)

avatarAddress Address

## Properties

### AvatarAddress

```
public Address AvatarAddress { get; }
```

Property Value

Address

## Code

```
public string Code { get; }
```

Property Value

[string](#)

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### `context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

[IActionContext](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class RegisterInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public class RegisterInfo : IRegisterInfo
```

## Inheritance

[object](#) ← RegisterInfo

## Implements

[IRegisterInfo](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RegisterInfo()

```
public RegisterInfo()
```

### RegisterInfo(List)

```
public RegisterInfo(List serialized)
```

## Parameters

serialized List

## Properties

## AvatarAddress

```
public Address AvatarAddress { get; set; }
```

Property Value

Address

## ItemCount

```
public int ItemCount { get; set; }
```

Property Value

[int](#)

## Price

```
public FungibleAssetValue Price { get; set; }
```

Property Value

FungibleAssetValue

## TradableId

```
public Guid TradableId { get; set; }
```

Property Value

[Guid](#)

## Type

```
public ProductType Type { get; set; }
```

Property Value

[ProductType](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

### Validate()

```
public void Validate()
```

### ValidateAddress(Address)

```
public void ValidateAddress(Address avatarAddress)
```

Parameters

avatarAddress Address

### ValidatePrice(Currency)

```
public void ValidatePrice(Currency ncg)
```

## Parameters

**ncg** Currency

# Class RegisterProduct

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("register_product3")]
public class RegisterProduct : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RegisterProduct

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

### Capacity

```
public const int Capacity = 100
```

Field Value

[int](#)

## ChargeAp

`public bool ChargeAp`

Field Value

[bool](#)

## CostAp

`public const int CostAp = 5`

Field Value

[int](#)

## NonTradableTickerCurrencies

`public static readonly IReadOnlyCollection<Currency> NonTradableTickerCurrencies`

Field Value

[IReadOnlyCollection](#) <Currency>

## RegisterInfos

`public IEnumerable<IRegisterInfo> RegisterInfos`

Field Value

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

`LoadPlainValueInternal(IImmutableDictionary<string, IValue>)`

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

## Register(IActionContext, IRegisterInfo, AvatarState, ProductsState, IWorld, IRandom)

```
public static IWorld Register(IActionContext context, IRegisterInfo info,  
AvatarState avatarState, ProductsState productsState, IWorld states, IRandom random)
```

## Parameters

context IActionContext

info [IRegisterInfo](#)

avatarState [AvatarState](#)

productsState [ProductsState](#)

states IWorld

random IRandom

## Returns

IWorld

# Class RegisterProduct0

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("register_product")]
[ActionObsolete(8324909)]
public class RegisterProduct0 : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RegisterProduct0

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### Capacity

```
public const int Capacity = 100
```

Field Value

[int](#)

## ChargeAp

```
public bool ChargeAp
```

Field Value

[bool](#)

## CostAp

```
public const int CostAp = 5
```

Field Value

[int](#)

## RegisterInfos

```
public IEnumerable<IRegisterInfo> RegisterInfos
```

Field Value

[IEnumerable](#) <[IRegisterInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), IValue>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

## Register(IActionContext, IRegisterInfo, AvatarState, ProductsState, IWorld, IRandom)

```
public static IWorld Register(IActionContext context, IRegisterInfo info,
AvatarState avatarState, ProductsState productsState, IWorld states, IRandom random)
```

### Parameters

context [IActionContext](#)

info [IRegisterInfo](#)

avatarState [AvatarState](#)

productsState [ProductsState](#)

states [IWorld](#)

random [IRandom](#)

### Returns

[IWorld](#)

# Class RenewAdminState

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/805> Updated at <https://github.com/planetarium/lib9c/pull/815> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("renew_admin_state")]
public class RenewAdminState : GameAction, IAction, IRenewAdminStateV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RenewAdminState

## Implements

IAction, IRenewAdminStateV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RenewAdminState()

```
public RenewAdminState()
```

### RenewAdminState(long)

```
public RenewAdminState(long newValidUntil)
```

## Parameters

`newValidUntil long`

## Properties

### NewValidUntil

```
public long NewValidUntil { get; }
```

## Property Value

`long`

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

`IImmutableDictionary<string>, IValue>`

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

**context** IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class RequestPledge

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("request_pledge")]
public class RequestPledge : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← RequestPledge

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RequestPledge()

```
public RequestPledge()
```

## Fields

### AgentAddress

```
public Address AgentAddress
```

## Field Value

Address

## DefaultRefillMead

```
public const int DefaultRefillMead = 4
```

Field Value

[int](#)

## RefillMead

```
public int RefillMead
```

Field Value

[int](#)

## TypeIdentifier

```
public const string TypeIdentifier = "request_pledge"
```

Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other "bound" information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class RequiredBlockIndexException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RequiredBlockIndexException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RequiredBlockIndexException

## Implements

[ISerializable](#)

## Derived

[RequiredBlockIntervalException](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RequiredBlockIndexException()

```
public RequiredBlockIndexException()
```

### RequiredBlockIndexException(SerializationInfo, StreamingContext)

```
protected RequiredBlockIndexException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RequiredBlockIndexException(string)

```
public RequiredBlockIndexException(string msg)
```

## Parameters

msg [string](#)

## RequiredBlockIndexException(string, string, long)

```
public RequiredBlockIndexException(string actionType, string addressesHex,  
long currentBlockIndex)
```

## Parameters

actionType [string](#)

addressesHex [string](#)

currentBlockIndex [long](#)

## RequiredBlockIndexException(string, string, long, long)

```
public RequiredBlockIndexException(string actionType, string addressesHex, long  
requiredBlockIndex, long currentBlockIndex)
```

## Parameters

actionType [string](#)

addressesHex [string](#)

requiredBlockIndex [long](#)

currentBlockIndex [long](#)

# Class RequiredBlockIntervalException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RequiredBlockIntervalException : RequiredBlockIndexException,
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [RequiredBlockIndexException](#) ← RequiredBlockIntervalException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RequiredBlockIntervalException()

```
public RequiredBlockIntervalException()
```

### RequiredBlockIntervalException(SerializationInfo, StreamingContext)

```
protected RequiredBlockIntervalException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RequiredBlockIntervalException(string)

```
public RequiredBlockIntervalException(string msg)
```

Parameters

msg [string](#)

## RequiredBlockIntervalException(string, string, long)

```
public RequiredBlockIntervalException(string actionType, string addressesHex,  
long currentBlockIndex)
```

Parameters

actionType [string](#)

addressesHex [string](#)

currentBlockIndex [long](#)

## RequiredBlockIntervalException(string, string, long, long)

```
public RequiredBlockIntervalException(string actionType, string addressesHex, long  
requiredBlockIndex, long currentBlockIndex)
```

Parameters

actionType [string](#)

addressesHex [string](#)

requiredBlockIndex [long](#)

currentBlockIndex [long](#)

# Class RetrieveAvatarAssets

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("retrieve_avatar_assets")]
public class RetrieveAvatarAssets : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← RetrieveAvatarAssets

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RetrieveAvatarAssets()

```
public RetrieveAvatarAssets()
```

### RetrieveAvatarAssets(Address)

```
public RetrieveAvatarAssets(Address avatarAddress)
```

## Parameters

avatarAddress Address

# Fields

## AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

## TypeIdentifier

```
public const string TypeIdentifier = "retrieve_avatar_assets"
```

### Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

#### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class RewardGold

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Introduced at Initial commit(2e645be18a4e2caea031c347f00777fbad5dbcc6) Updated at many pull requests Updated at <https://github.com/planetarium/lib9c/pull/1135>

```
[Serializable]  
public class RewardGold : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← RewardGold

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### FilterInactiveArenaInfoBlockIndex

```
public const long FilterInactiveArenaInfoBlockIndex = 3976000
```

## Field Value

[long](#)

## Properties

# PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GenesisGoldDistribution(IActionContext, IWorld)

```
public IWorld GenesisGoldDistribution(IActionContext ctx, IWorld states)
```

Parameters

`ctx` IActionContext

`states` IWorld

Returns

IWorld

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

Parameters

`plainValue` IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

PlainValue

## MinerReward(IActionContext, IWorld)

```
public IWorld MinerReward(IActionContext ctx, IWorld states)
```

Parameters

`ctx` IActionContext

`states` IWorld

Returns

IWorld

## PrepareNextArena(IActionContext, IWorld)

```
public IWorld PrepareNextArena(IActionContext ctx, IWorld states)
```

Parameters

`ctx` IActionContext

`states` IWorld

Returns

IWorld

## ResetChallengeCount(IActionContext, IWorld)

```
public IWorld ResetChallengeCount(IActionContext ctx, IWorld states)
```

Parameters

`ctx` IActionContext

`states` IWorld

Returns

IWorld

## TransferMead(IActionContext, IWorld)

```
public static IWorld TransferMead(IActionContext context, IWorld states)
```

### Parameters

**context** IActionContext

**states** IWorld

### Returns

IWorld

## WeeklyArenaRankingBoard(IActionContext, IWorld)

```
[Obsolete("Use WeeklyArenaRankingBoard2 for performance.")]
```

```
public IWorld WeeklyArenaRankingBoard(IActionContext ctx, IWorld states)
```

### Parameters

**ctx** IActionContext

**states** IWorld

### Returns

IWorld

## WeeklyArenaRankingBoard2(IActionContext, IWorld)

```
public IWorld WeeklyArenaRankingBoard2(IActionContext ctx, IWorld states)
```

### Parameters

**ctx** IActionContext

states IWorld

Returns

IWorld

# Class RuneEnhancement

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("runeEnhancement2")]
public class RuneEnhancement : GameAction, IAction, IRuneEnhancementV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RuneEnhancement

## Implements

IAction, IRuneEnhancementV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

### Runeld

```
public int RuneId
```

Field Value

[int](#)

## TryCount

```
public int TryCount
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

**context** IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Struct RuneEnhancement.LevelUpResult

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public struct RuneEnhancement.LevelUpResult
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Properties

### CrystalCost

```
public int CrystalCost { readonly get; set; }
```

Property Value

[int](#)

### LevelUpCount

```
public int LevelUpCount { readonly get; set; }
```

Property Value

[int](#)

### NcgCost

```
public int NcgCost { readonly get; set; }
```

Property Value

[int](#)

## RuneCost

```
public int RuneCost { readonly get; set; }
```

Property Value

[int](#)

## Methods

### ToString()

Returns the fully qualified type name of this instance.

```
public override string ToString()
```

Returns

[string](#)

The fully qualified type name.

# Class RuneSlotInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneSlotInfo
```

## Inheritance

[object](#) ← RuneSlotInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RuneSlotInfo(List)

```
public RuneSlotInfo(List serialized)
```

#### Parameters

serialized List

### RuneSlotInfo(int, int)

```
public RuneSlotInfo(int slotIndex, int runeId)
```

#### Parameters

slotIndex [int](#)

runeId [int](#)

# Properties

## RunelId

```
public int RuneId { get; }
```

### Property Value

[int](#)

## SlotIndex

```
public int SlotIndex { get; }
```

### Property Value

[int](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Class RuneSummon

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("rune_summon")]
public class RuneSummon : GameAction, IAction, IRuneSummonV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← RuneSummon

## Implements

IAction, IRuneSummonV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

### AvatarAddressKey

```
public const string AvatarAddressKey = "aa"
```

Field Value

[string](#) ↗

## GroupId

```
public int GroupId
```

Field Value

[int](#) ↗

## GroupIdKey

```
public const string GroupIdKey = "gid"
```

Field Value

[string](#) ↗

## RuneQuantity

```
public const int RuneQuantity = 10
```

Field Value

[int](#) ↗

## SummonCount

```
public int SummonCount
```

Field Value

[int](#)

## SummonCountKey

```
public const string SummonCountKey = "sc"
```

### Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the [context](#) object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

# Parameters

## context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

# Returns

## IWorld

A map of changed states (so-called "dirty").

# Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# SimulateSummon(RuneSheet, Row, int, IRandom)

```
public static Dictionary<Currency, int> SimulateSummon(RuneSheet runeSheet, SummonSheet.Row summonRow, int summonCount, IRandom random)
```

## Parameters

runeSheet [RuneSheet](#)

summonRow [SummonSheet.Row](#)

summonCount [int](#)

random IRandom

## Returns

[Dictionary<Currency, int>](#)

# Summon(IActionContext, Address, RuneSheet, Row, int, IRandom, IWorld)

```
public static IWorld Summon(IActionContext context, Address avatarAddress,  
RuneSheet runeSheet, SummonSheet.Row summonRow, int summonCount, IRandom random,  
IWorld states)
```

## Parameters

context IActionContext

avatarAddress Address

runeSheet [RuneSheet](#)

summonRow [SummonSheet.Row](#)

summonCount [int](#)

random IRandom

states IWorld

## Returns

IWorld

# Class SecureMiningReward

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("secure_mining_reward")]
public class SecureMiningReward : ActionBase, IAction, ISecureMiningReward
```

## Inheritance

[object](#) ← [ActionBase](#) ← SecureMiningReward

## Implements

IAction, ISecureMiningReward

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SecureMiningReward()

```
public SecureMiningReward()
```

### SecureMiningReward(Address)

```
public SecureMiningReward(Address recipient)
```

## Parameters

recipient Address

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

`LoadPlainValue(IValue)`

## Recipient

```
public Address Recipient { get; }
```

## Property Value

### Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class Sell

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/1640>

```
[Serializable]
[ActionType("sell12")]
[ActionObsolete(7200000)]
public class Sell : GameAction, IAction, ISellV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Sell

## Implements

IAction, ISellV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### count

```
public int count
```

### Field Value

[int](#)

## itemSubType

```
public ItemSubType itemSubType
```

### Field Value

[ItemSubType](#)

## orderId

```
public Guid orderId
```

### Field Value

[Guid](#)

## price

```
public FungibleAssetValue price
```

### Field Value

FungibleAssetValue

## sellerAvatarAddress

```
public Address sellerAvatarAddress
```

### Field Value

Address

## tradableId

```
public Guid tradableId
```

## Field Value

[Guid](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class Sell6

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionObsolete(7000000)]
[ActionType("sell6")]
public class Sell6 : GameAction, IAction, ISellV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Sell6

## Implements

IAction, ISellV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ExpiredBlockIndex

```
public const long ExpiredBlockIndex = 16000
```

#### Field Value

[long](#)

#### count

```
public int count
```

Field Value

[int](#)

## itemSubType

```
public ItemSubType itemSubType
```

Field Value

[ItemSubType](#)

## price

```
public FungibleAssetValue price
```

Field Value

FungibleAssetValue

## sellerAvatarAddress

```
public Address sellerAvatarAddress
```

Field Value

Address

## tradableId

```
public Guid tradableId
```

## Field Value

[Guid](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class SellCancellation

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/602> Updated at <https://github.com/planetarium/lib9c/pull/609> Updated at <https://github.com/planetarium/lib9c/pull/620> Updated at <https://github.com/planetarium/lib9c/pull/861> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionObsolete(8324909)]
[ActionType("sell_cancellation9")]
public class SellCancellation : GameAction, IAction, ISellCancellationV3
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← SellCancellation

## Implements

IAction, ISellCancellationV3

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### itemSubType

```
public ItemSubType itemSubType
```

### Field Value

## [ItemSubType](#)

### orderId

```
public Guid orderId
```

#### Field Value

[Guid](#)

### sellerAvatarAddress

```
public Address sellerAvatarAddress
```

#### Field Value

Address

### tradableId

```
public Guid tradableId
```

#### Field Value

[Guid](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

#### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Cancel(IActionContext, IWorld, AvatarState, string, Order)

```
public static IWorld Cancel(IActionContext context, IWorld states, AvatarState
avatarState, string addressesHex, Order order)
```

#### Parameters

**context** [IActionContext](#)

**states** [IWorld](#)

**avatarState** [AvatarState](#)

**addressesHex** [string](#)

**order** [Order](#)

#### Returns

[IWorld](#)

### CancelV2(IActionContext, IWorld, AvatarState, string, Order, Guid, ItemSubType)

```
public static IWorld CancelV2(IActionContext context, IWorld states,
AvatarState avatarState, string addressesHex, Order order, Guid tradableId,
ItemSubType itemSubType)
```

#### Parameters

**context** [IActionContext](#)

**states** [IWorld](#)

avatarState [AvatarState](#)

addressesHex [string](#)

order [Order](#)

tradableId [Guid](#)

itemSubType [ItemSubType](#)

Returns

IWorld

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

Returns

IWorld

A map of changed states (so-called “dirty”).

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

`LoadPlainValueInternal(IImmutableDictionary<string, IValue>)`

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class SellCancellation.Result

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SellCancellation.Result : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← SellCancellation.Result

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Result()

```
public Result()
```

### Result(Dictionary)

```
public Result(Dictionary serialized)
```

## Parameters

serialized Dictionary

# Fields

## id

```
public Guid id
```

### Field Value

[Guid](#)

## shopItem

```
public ShopItem shopItem
```

### Field Value

[ShopItem](#)

# Properties

## TypeId

```
protected override string TypeId { get; }
```

### Property Value

[string](#)

# Methods

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Enum ShopErrorType

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ShopErrorType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

ERROR\_CODE\_DUPLICATE\_SELL = 10

ERROR\_CODE\_FAILED\_LOADING\_STATE = 1

ERROR\_CODE\_INSUFFICIENT\_BALANCE = 4

ERROR\_CODE\_INVALID\_ADDRESS = 5

ERROR\_CODE\_INVALID\_ITEMTYPE = 9

ERROR\_CODE\_INVALID\_ORDERID = 7

ERROR\_CODE\_INVALID\_PRICE = 6

ERROR\_CODE\_INVALID\_TRADEABLEID = 8

ERROR\_CODE\_ITEM\_DOES\_NOT\_EXIST = 2

ERROR\_CODE\_SHOPITEM\_EXPIRED = 3

# Class ShopItemExpiredException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ShopItemExpiredException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ShopItemExpiredException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ShopItemExpiredException()

```
public ShopItemExpiredException()
```

### ShopItemExpiredException(SerializationInfo, StreamingContext)

```
protected ShopItemExpiredException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ShopItemExpiredException(string)

public [ShopItemExpiredException](#)([string](#) msg)

### Parameters

msg [string](#)

# Class SlotAlreadyUnlockedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SlotAlreadyUnlockedException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← SlotAlreadyUnlockedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SlotAlreadyUnlockedException(SerializationInfo, StreamingContext)

```
protected SlotAlreadyUnlockedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SlotAlreadyUnlockedException(string)

```
public SlotAlreadyUnlockedException(string s)
```

### Parameters

s [string](#)

# Class StageNotClearedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageNotClearedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← StageNotClearedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### StageNotClearedException()

```
public StageNotClearedException()
```

### StageNotClearedException(SerializationInfo, StreamingContext)

```
protected StageNotClearedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## StageNotClearedException(string)

```
public StageNotClearedException(string message)
```

Parameters

message [string](#)

## StageNotClearedException(string, Exception)

```
public StageNotClearedException(string message, Exception innerException)
```

Parameters

message [string](#)

innerException [Exception](#)

## StageNotClearedException(string, string, int, int)

```
public StageNotClearedException(string actionType, string addressesHex, int requiredToClearedStage, int currentClearedStage)
```

Parameters

actionType [string](#)

addressesHex [string](#)

requiredToClearedStage [int](#)

currentClearedStage [int](#)

# Class Stake

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("stake3")]
public class Stake : GameAction, IAction, IStakeV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Stake

## Implements

IAction, IStakeV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Stake()

```
public Stake()
```

### Stake(BigInteger)

```
public Stake(BigInteger amount)
```

## Parameters

amount [BigInteger](#)

# Properties

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

`LoadPlainValueInternal(IImmutableDictionary<string, IValue>)`

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string,  
IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class Stake2

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("stake2")]
[ActionObsolete(7983895)]
public class Stake2 : GameAction, IAction, IStakeV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Stake2

## Implements

IAction, IStakeV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Stake2()

```
public Stake2()
```

### Stake2(BigInteger)

```
public Stake2(BigInteger amount)
```

## Parameters

amount [BigInteger](#)

## Fields

### ObsoleteBlockIndex

```
public const long ObsoleteBlockIndex = 7983895
```

Field Value

[long](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class StakeExistingClaimableException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeExistingClaimableException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← StakeExistingClaimableException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### StakeExistingClaimableException()

```
public StakeExistingClaimableException()
```

### StakeExistingClaimableException(SerializationInfo, StreamingContext)

```
protected StakeExistingClaimableException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## StakeExistingClaimableException(string)

```
public StakeExistingClaimableException(string message)
```

Parameters

message [string](#)

## StakeExistingClaimableException(string, Exception)

```
public StakeExistingClaimableException(string message, Exception innerException)
```

Parameters

message [string](#)

innerException [Exception](#)

# Class TotalSupplyDoesNotExistException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TotalSupplyDoesNotExistException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← TotalSupplyDoesNotExistException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Message](#) , [Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### TotalSupplyDoesNotExistException(Currency)

```
public TotalSupplyDoesNotExistException(Currency currency)
```

## Parameters

**currency** Currency

### TotalSupplyDoesNotExistException(SerializationInfo, StreamingContext)

```
protected TotalSupplyDoesNotExistException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Currency

```
public Currency Currency { get; }
```

## Property Value

Currency

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

When overridden in a derived class, sets the [SerializationInfo](#) with information about the exception.

```
public override void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) that holds the serialized object data about the exception being thrown.

context [StreamingContext](#)

The [StreamingContext](#) that contains contextual information about the source or destination.

## Exceptions

### [ArgumentNullException](#)

The `info` parameter is a null reference (`Nothing` in Visual Basic).

# Class TransferAsset

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/2143> Updated at <https://github.com/planetarium/lib9c/pull/2143>

```
[Serializable]
[ActionType("transfer_asset5")]
public class TransferAsset : ActionBase, IAction, ISerializable,
ITransferAsset, ITransferAssetV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← TransferAsset

## Implements

IAction, [ISerializable](#), [ITransferAsset](#), ITransferAssetV1

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### TransferAsset()

```
public TransferAsset()
```

### TransferAsset(Address, Address, FungibleAssetValue, string)

```
public TransferAsset(Address sender, Address recipient, FungibleAssetValue amount,  
string memo = null)
```

## Parameters

sender Address

recipient Address

amount FungibleAssetValue

memo [string](#)

## TransferAsset(SerializationInfo, StreamingContext)

```
protected TransferAsset(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "transfer_asset5"
```

## Field Value

[string](#)

## Properties

### Amount

```
public FungibleAssetValue Amount { get; }
```

## Property Value

FungibleAssetValue

## Memo

```
public string Memo { get; }
```

## Property Value

[string](#)

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

## Recipient

```
public Address Recipient { get; }
```

Property Value

Address

## Sender

```
public Address Sender { get; }
```

Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

# GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

**info** [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

**context** [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

# LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by `Libplanet.Action.IAction.PlainValue` property) to be deserialized and assigned to this action's properties (or fields).

## See Also

`PlainValue`

## ThrowIfStakeState(IWorld, Address)

```
public static void ThrowIfStakeState(IWorld state, Address recipient)
```

### Parameters

**state** IWorld

**recipient** Address

# Class TransferAsset3

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public static class TransferAsset3
```

## Inheritance

[object](#) ← TransferAsset3

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### AllowedCrystalTransfers

```
public static readonly IReadOnlyList<Address> AllowedCrystalTransfers
```

#### Field Value

[IReadOnlyList](#)<Address>

### CrystalTransferringRestrictionStartIndex

```
public const long CrystalTransferringRestrictionStartIndex = 6220000
```

#### Field Value

[long](#)

## Methods

## CheckCrystalSender(Currency, long, Address)

```
public static void CheckCrystalSender(Currency currency, long blockIndex,  
Address sender)
```

### Parameters

**currency** Currency

**blockIndex** [long](#)

**sender** Address

# Class TransferAssets

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/636> Updated at <https://github.com/planetarium/lib9c/pull/957>

```
[Serializable]
[ActionType("transfer_assets3")]
public class TransferAssets : ActionBase, IAction, ISerializable,
ITransferAssets, ITransferAssetsV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← TransferAssets

## Implements

IAction, [ISerializable](#), [ITransferAssets](#), ITransferAssetsV1

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### TransferAssets()

```
public TransferAssets()
```

### TransferAssets(Address, List<(Address, FungibleAssetValue)>, string)

```
public TransferAssets(Address sender, List<(Address, FungibleAssetValue)>
recipients, string memo = null)
```

## Parameters

sender Address

recipients [List](#)<(Address [recipient](#), FungibleAssetValue [amount](#))>

memo [string](#)

## TransferAssets(SerializationInfo, StreamingContext)

```
protected TransferAssets(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

### RecipientsCapacity

```
public const int RecipientsCapacity = 100
```

## Field Value

[int](#)

### TypeIdentifier

```
public const string TypeIdentifier = "transfer_assets3"
```

## Field Value

[string](#) ↗

## Properties

### Memo

```
public string Memo { get; }
```

## Property Value

[string](#) ↗

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Recipients

```
public List<(Address recipient, FungibleAssetValue amount)> Recipients { get; }
```

## Property Value

[List](#)<(Address [recipient](#), FungibleAssetValue [amount](#))>

## Sender

`public Address Sender { get; }`

## Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

`public override IWorld Execute(IActionContext context)`

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

**info** [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

**context** [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class UnlockCombinationSlot

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("unlock_combination_slot")]
public class UnlockCombinationSlot : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UnlockCombinationSlot](#)

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### GoldenDustId

```
public const int GoldenDustId = 600201
```

Field Value

[int](#)

## RubyDustId

```
public const int RubyDustId = 600202
```

Field Value

[int](#)

## SlotIndex

```
public int SlotIndex
```

Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class UnlockEquipmentRecipe

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[ActionType("unlock_equipment_recipe2")]
public class UnlockEquipmentRecipe : GameAction, IAction, IUnlockEquipmentRecipeV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UnlockEquipmentRecipe](#)

## Implements

IAction, IUnlockEquipmentRecipeV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

### Recipelds

```
public List<int> RecipeIds
```

## Field Value

[List](#)<[int](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

## UnlockedIds(IWorld, Address, EquipmentItemRecipeSheet, WorldInformation, List<int>)

```
public static List<int> UnlockedIds(IWorld states, Address unlockedRecipeIdsAddress, EquipmentItemRecipeSheet equipmentRecipeSheet, WorldInformation worldInformation, List<int> recipeIds)
```

### Parameters

states IWorld

unlockedRecipeIdsAddress Address

equipmentRecipeSheet [EquipmentItemRecipeSheet](#)

worldInformation [WorldInformation](#)

recipeIds [List<int>](#)

### Returns

[List<int>](#)

# Class UnlockRuneSlot

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("unlock_rune_slot")]
public class UnlockRuneSlot : GameAction, IAction, IUnlockRuneSlotV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UnlockRuneSlot](#)

## Implements

IAction, IUnlockRuneSlotV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

### Field Value

Address

### SlotIndex

```
public int SlotIndex
```

## Field Value

[int](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#) <[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class UnlockWorld

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/1309>

```
[ActionType("unlock_world2")]
public class UnlockWorld : GameAction, IAction, IUnlockWorldV1
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UnlockWorld](#)

## Implements

IAction, IUnlockWorldV1

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

#### Field Value

Address

### WorldIds

```
public List<int> WorldIds
```

## Field Value

[List](#)<[int](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class UpdateSell

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

Hard forked at <https://github.com/planetarium/lib9c/pull/1640>

```
[Serializable]
[ActionType("update_sell")]
[ActionObsolete(7200000)]
public class UpdateSell : GameAction, IAction, IUpdateSellV2
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UpdateSell](#)

## Implements

IAction, IUpdateSellV2

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### sellerAvatarAddress

```
public Address sellerAvatarAddress
```

#### Field Value

Address

# updateSellInfos

```
public IEnumerable<UpdateSellInfo> updateSellInfos
```

## Field Value

[IEnumerable](#)<[UpdateSellInfo](#)>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Cancel(IWorld, UpdateSellInfo, string, AvatarState, OrderDigestListState, IActionContext, Address)

```
public static IWorld Cancel(IWorld states, UpdateSellInfo updateSellInfo, string  
addressesHex, AvatarState avatarState, OrderDigestListState digestList,  
IActionContext context, Address sellerAvatarAddress)
```

## Parameters

states [IWorld](#)

updateSellInfo [UpdateSellInfo](#)

addressesHex [string](#)

avatarState [AvatarState](#)

`digestList` [OrderDigestListState](#)

`context` IActionContext

`sellerAvatarAddress` Address

Returns

IWorld

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

Returns

IWorld

A map of changed states (so-called “dirty”).

Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), [IValue](#)>

# Class UpdateSellInfo

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class UpdateSellInfo
```

## Inheritance

[object](#) ← UpdateSellInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### UpdateSellInfo(List)

```
public UpdateSellInfo(List serialized)
```

#### Parameters

serialized List

### UpdateSellInfo(Guid, Guid, Guid, ItemSubType, FungibleAssetValue, int)

```
public UpdateSellInfo(Guid orderId, Guid updateSellOrderId, Guid tradableId,
ItemSubType itemSubType, FungibleAssetValue price, int count)
```

#### Parameters

orderId [Guid](#)

updateSellOrderId [Guid](#)

tradableId [Guid](#)

itemSubType [ItemSubType](#)

price FungibleAssetValue

count [int](#)

## Fields

### count

`public int count`

### Field Value

[int](#)

### itemSubType

`public ItemSubType itemSubType`

### Field Value

[ItemSubType](#)

### orderId

`public Guid orderId`

### Field Value

[Guid](#)

## price

```
public FungibleAssetValue price
```

### Field Value

FungibleAssetValue

## tradableId

```
public Guid tradableId
```

### Field Value

[Guid](#)

## updateSellOrderId

```
public Guid updateSellOrderId
```

### Field Value

[Guid](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Class UsageLimitExceedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class UsageLimitExceedException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ← UsageLimitExceedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### UsageLimitExceedException()

```
public UsageLimitExceedException()
```

### UsageLimitExceedException(SerializationInfo, StreamingContext)

```
protected UsageLimitExceedException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## UsageLimitExceedException(string)

public UsageLimitExceedException(string message)

Parameters

message [string](#)

# Class ValidatorSetOperate

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("op_validator_set")]
public sealed class ValidatorSetOperate : ActionBase, IAction,
IEquatable<ValidatorSetOperate>
```

## Inheritance

[object](#) ← [ActionBase](#) ← ValidatorSetOperate

## Implements

IAction, [IEquatable](#)<[ValidatorSetOperate](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ValidatorSetOperate()

```
public ValidatorSetOperate()
```

### ValidatorSetOperate(ValidatorSetOperatorType, Validator)

```
public ValidatorSetOperate(ValidatorSetOperatorType @operator, Validator operand)
```

## Parameters

operator [ValidatorSetOperatorType](#)

operand Validator

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "op_validator_set"
```

### Field Value

[string](#) ↗

## ValidatorSetOperateKey

```
public static readonly string ValidatorSetOperateKey
```

### Field Value

[string](#) ↗

# Properties

## Error

```
public string Error { get; }
```

### Property Value

[string](#) ↗

## Operand

```
public Validator Operand { get; }
```

Property Value

Validator

## Operator

```
public ValidatorSetOperatorType Operator { get; }
```

Property Value

[ValidatorSetOperatorType](#)

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Append(Validator)

```
public static ValidatorSetOperate Append(Validator operand)
```

Parameters

**operand** Validator

Returns

[ValidatorSetOperate](#)

## Equals(ValidatorSetOperate)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(ValidatorSetOperate other)
```

Parameters

**other** [ValidatorSetOperate](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

**obj** [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the [context](#) object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

[context](#) IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

## Remove(Validator)

```
public static ValidatorSetOperate Remove(Validator operand)
```

## Parameters

**operand** Validator

## Returns

[ValidatorSetOperate](#)

## Update(Validator)

```
public static ValidatorSetOperate Update(Validator operand)
```

## Parameters

**operand** Validator

## Returns

## ValidatorSetOperate

# Enum ValidatorSetOperatorType

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public enum ValidatorSetOperatorType
```

## Extension Methods

[ValidatorSetOperatorTypeExtensions.ToFunc\(ValidatorSetOperatorType\)](#) ,  
[StateExtensions.Serialize\(Enum\)](#)

## Fields

**Append = 0**

Append the validator. no-op if validator public key already exists.

**Remove = 1**

Set validator's power to 0.

**Update = 2**

Update validator. if not exists, append it.

# Class ValidatorSetOperatorTypeExtensions

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
public static class ValidatorSetOperatorTypeExtensions
```

## Inheritance

[object](#) ← ValidatorSetOperatorTypeExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ToFunc(ValidatorSetOperatorType)

```
public static Func<ValidatorSet, Validator, Validator> ToFunc(this  
ValidatorSetOperatorType @operator)
```

#### Parameters

operator [ValidatorSetOperatorType](#)

#### Returns

[Func](#)<ValidatorSet, Validator, Validator>

# Class WeeklyArenaStateAlreadyEndedException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WeeklyArenaStateAlreadyEndedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← WeeklyArenaStateAlreadyEndedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### WeeklyArenaStateAlreadyEndedException()

```
public WeeklyArenaStateAlreadyEndedException()
```

### WeeklyArenaStateAlreadyEndedException(SerializationInfo, StreamingContext)

```
public WeeklyArenaStateAlreadyEndedException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## WeeklyArenaStateAlreadyEndedException(string)

```
public WeeklyArenaStateAlreadyEndedException(string message)
```

Parameters

message [string](#)

## Fields

### BaseMessage

```
public const string BaseMessage = "Aborted as the weekly arena state already ended."
```

Field Value

[string](#)

# Class WeeklyArenaStateNotContainsAvatarAddressException

Namespace: [Nekoyume.Action](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WeeklyArenaStateNotContainsAvatarAddressException :  
Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← WeeklyArenaStateNotContainsAvatarAddressException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### WeeklyArenaStateNotContainsAvatarAddressException( SerializationInfo, StreamingContext)

```
public WeeklyArenaStateNotContainsAvatarAddressException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## WeeklyArenaStateNotContainsAvatarAddressException( string)

```
public WeeklyArenaStateNotContainsAvatarAddressException(string message)
```

### Parameters

message [string](#)

## WeeklyArenaStateNotContainsAvatarAddressException( string, Address)

```
public WeeklyArenaStateNotContainsAvatarAddressException(string addressesHex,  
Address avatarAddress)
```

### Parameters

addressesHex [string](#)

avatarAddress [Address](#)

# Namespace Nekoyume.Action.Adventure Boss

## Classes

[ClaimAdventureBossReward](#)

[ExploreAdventureBoss](#)

[SweepAdventureBoss](#)

[UnlockFloor](#)

[Wanted](#)

# Class ClaimAdventureBossReward

Namespace: [Nekoyume.Action.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("claim_adventure_boss_reward")]
public class ClaimAdventureBossReward : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ClaimAdventureBossReward

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### TypeIdentifier

```
public const string TypeIdentifier = "claim_adventure_boss_reward"
```

## Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class ExploreAdventureBoss

Namespace: [Nekoyume.Action.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("explore_adventure_boss")]
public class ExploreAdventureBoss : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← ExploreAdventureBoss

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### Costumes

```
public List<Guid> Costumes
```

Field Value

[List](#) <[Guid](#)>

## Equipments

```
public List<Guid> Equipments
```

Field Value

[List](#) <[Guid](#)>

## Foods

```
public List<Guid> Foods
```

Field Value

[List](#) <[Guid](#)>

## RunelInfos

```
public List<RuneSlotInfo> RuneInfos
```

Field Value

[List](#) <[RuneSlotInfo](#)>

## Season

```
public int Season
```

Field Value

[int](#)

## StageBuffId

```
public int? StageBuffId
```

Field Value

[int](#)?

## TypeIdentifier

```
public const string TypeIdentifier = "explore_adventure_boss"
```

Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class SweepAdventureBoss

Namespace: [Nekoyume.Action.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("sweep_adventure_boss")]
public class SweepAdventureBoss : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← SweepAdventureBoss

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

## Costumes

```
public List<Guid> Costumes
```

Field Value

[List](#)<[Guid](#)>

## Equipments

```
public List<Guid> Equipments
```

Field Value

[List](#)<[Guid](#)>

## RuneInfos

```
public List<RuneSlotInfo> RuneInfos
```

Field Value

[List](#)<[RuneSlotInfo](#)>

## Season

```
public int Season
```

Field Value

[int](#)

## TypeIdentifier

```
public const string TypeIdentifier = "sweep_adventure_boss"
```

## Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers.

See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class UnlockFloor

Namespace: [Nekoyume.Action.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("unlock_floor")]
public class UnlockFloor : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UnlockFloor](#)

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### GoldenDustId

```
public const int GoldenDustId = 600201
```

Field Value

[int](#)

## OpeningFloor

```
public const int OpeningFloor = 5
```

Field Value

[int](#)

## Season

```
public int Season
```

Field Value

[int](#)

## TotalFloor

```
public const int TotalFloor = 20
```

Field Value

[int](#)

## TypeIdentifier

```
public const string TypeIdentifier = "unlock_floor"
```

Field Value

[string](#)

## UseNcg

```
public bool UseNcg
```

Field Value

[bool](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

**context** IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class Wanted

Namespace: [Nekoyume.Action.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("wanted")]
public class Wanted : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← Wanted

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### Bounty

```
public FungibleAssetValue Bounty
```

## Field Value

FungibleAssetValue

## Season

```
public int Season
```

## Field Value

[int](#)

## TypeIdentifier

```
public const string TypeIdentifier = "wanted"
```

## Field Value

[string](#)

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

## Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

# Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Namespace Nekoyume.Action.Coupons

## Classes

[IssueCoupons](#)

[RedeemCoupon](#)

[TransferCoupons](#)

# Class IssueCoupons

Namespace: [Nekoyume.Action.Coupons](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("issue_coupons")]
public sealed class IssueCoupons : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← IssueCoupons

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### IssueCoupons()

```
public IssueCoupons()
```

### IssueCoupons(IImmutableDictionary<RewardSet, uint>, Address)

```
public IssueCoupons(IImmutableDictionary<RewardSet, uint> rewards,  
Address recipient)
```

## Parameters

rewards [IImmutableDictionary](#)<[RewardSet](#), [uint](#)>

`recipient` Address

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Recipient

```
public Address Recipient { get; }
```

Property Value

Address

## Rewards

```
public IImmutableDictionary<RewardSet, uint> Rewards { get; }
```

Property Value

[IImmutableDictionary](#)<[RewardSet](#), [uint](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### `context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

[IActionContext](#)

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class RedeemCoupon

Namespace: [Nekoyume.Action.Coupons](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("redeem_coupon")]
public sealed class RedeemCoupon : GameAction, IAction
```

## Inheritance

[object](#) ↳ [ActionBase](#) ↳ [GameAction](#) ↳ [RedeemCoupon](#)

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RedeemCoupon()

```
public RedeemCoupon()
```

### RedeemCoupon(Guid, Address)

```
public RedeemCoupon(Guid couponId, Address avatarAddress)
```

## Parameters

couponId [Guid](#)

avatarAddress Address

# Properties

## AvatarAddress

```
public Address AvatarAddress { get; }
```

### Property Value

Address

## CouponId

```
public Guid CouponId { get; }
```

### Property Value

[Guid](#)

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class TransferCoupons

Namespace: [Nekoyume.Action.Coupons](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("transfer_coupons")]
public sealed class TransferCoupons : GameAction, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← TransferCoupons

## Implements

IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### TransferCoupons()

```
public TransferCoupons()
```

### TransferCoupons(ImmutableDictionary<Address, ImmutableSet<Guid>>)

```
public TransferCoupons(ImmutableDictionary<Address, ImmutableSet<Guid>>  
couponsPerRecipient)
```

## Parameters

couponsPerRecipient [ImmutableDictionary](#)<Address, [ImmutableSet](#)<[Guid](#)>>

# Properties

## CouponsPerRecipient

```
public IImmutableDictionary<Address, IImmutableSet<Guid>> CouponsPerRecipient {  
    get; }
```

Property Value

[IImmutableDictionary](#)<Address, [IImmutableSet](#)<Guid>>>

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<string, IValue>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

### `IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Namespace Nekoyume.Action.Custom EquipmentCraft

## Classes

[CustomEquipmentCraft](#)

## Structs

[CustomCraftData](#)

# Struct CustomCraftData

Namespace: [Nekoyume.Action.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public struct CustomCraftData
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Fields

### IconId

```
public int IconId
```

#### Field Value

[int](#)

### RecipeId

```
public int RecipeId
```

#### Field Value

[int](#)

### SlotIndex

```
public int SlotIndex
```

Field Value

[int](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class CustomEquipmentCraft

Namespace: [Nekoyume.Action.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
[ActionType("custom_equipment_craft")]
public class CustomEquipmentCraft : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← CustomEquipmentCraft

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

### CraftList

```
public List<CustomCraftData> CraftList
```

## Field Value

[List](#) <CustomCraftData>

## RandomIconId

```
public const int RandomIconId = 0
```

## Field Value

[int](#)

## TypeIdentifier

```
public const string TypeIdentifier = "custom_equipment_craft"
```

## Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

IWorld

A map of changed states (so-called “dirty”).

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Namespace Nekoyume.Action.Exceptions

## Classes

[DuplicatedCraftSlotIndexException](#)

[EmptyRewardException](#)

[InvalidSummonCountException](#)

# Class DuplicatedCraftSlotIndexException

Namespace: [Nekoyume.Action.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicatedCraftSlotIndexException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← DuplicatedCraftSlotIndexException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### DuplicatedCraftSlotIndexException(SerializationInfo, StreamingContext)

```
protected DuplicatedCraftSlotIndexException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicatedCraftSlotIndexException(string)

```
public DuplicatedCraftSlotIndexException(string message)
```

### Parameters

message [string](#)

# Class EmptyRewardException

Namespace: [Nekoyume.Action.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EmptyRewardException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← EmptyRewardException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### EmptyRewardException()

```
public EmptyRewardException()
```

### EmptyRewardException(SerializationInfo, StreamingContext)

```
protected EmptyRewardException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## EmptyRewardException(string)

public EmptyRewardException(string msg)

### Parameters

msg [string](#)

# Class InvalidSummonCountException

Namespace: [Nekoyume.Action.Exceptions](#)

Assembly: Lib9c.dll

```
public class InvalidSummonCountException : ArgumentException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [ArgumentException](#) ←  
InvalidSummonCountException

## Implements

[ISerializable](#)

## Inherited Members

[ArgumentException.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ArgumentException.Message](#) , [ArgumentException.ParamName](#) ,  
[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidSummonCountException()

```
public InvalidSummonCountException()
```

### InvalidSummonCountException(SerializationInfo, StreamingContext)

```
protected InvalidSummonCountException(SerializationInfo info,  
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidSummonCountException(string)

public InvalidSummonCountException([string](#) msg)

Parameters

msg [string](#)

# Namespace Nekoyume.Action.Exceptions. AdventureBoss

## Classes

[AlreadyClaimedException](#)

[ClaimExpiredException](#)

[InsufficientStakingException](#)

[InvalidAdventureBossSeasonException](#)

[InvalidBountyException](#)

[MaxInvestmentCountExceededException](#)

[PreviousBountyException](#)

[SeasonInProgressException](#)

# Class AlreadyClaimedException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyClaimedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AlreadyClaimedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyClaimedException()

```
public AlreadyClaimedException()
```

### AlreadyClaimedException(SerializationInfo, StreamingContext)

```
protected AlreadyClaimedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyClaimedException(string)

public AlreadyClaimedException(string msg)

### Parameters

msg [string](#)

# Class ClaimExpiredException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ClaimExpiredException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ClaimExpiredException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ClaimExpiredException()

```
public ClaimExpiredException()
```

### ClaimExpiredException(SerializationInfo, StreamingContext)

```
protected ClaimExpiredException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ClaimExpiredException(string)

public **ClaimExpiredException**(string msg)

### Parameters

msg [string](#)

# Class InsufficientStakingException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InsufficientStakingException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InsufficientStakingException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InsufficientStakingException()

```
public InsufficientStakingException()
```

### InsufficientStakingException(SerializationInfo, StreamingContext)

```
protected InsufficientStakingException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InsufficientStakingException(string)

public [InsufficientStakingException](#)([string](#) msg)

Parameters

msg [string](#)

# Class InvalidAdventureBossSeasonException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidAdventureBossSeasonException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidAdventureBossSeasonException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidAdventureBossSeasonException()

```
public InvalidAdventureBossSeasonException()
```

### InvalidAdventureBossSeasonException(SerializationInfo, StreamingContext)

```
protected InvalidAdventureBossSeasonException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidOperationException(string)

public **InvalidOperationException**(**string** message)

Parameters

message [string](#)

# Class InvalidBountyException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidBountyException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidBountyException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidBountyException()

```
public InvalidBountyException()
```

### InvalidBountyException(SerializationInfo, StreamingContext)

```
protected InvalidBountyException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidBountyException(string)

public [InvalidBountyException](#)([string](#) msg)

### Parameters

msg [string](#)

# Class MaxInvestmentCountExceededException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MaxInvestmentCountExceededException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← MaxInvestmentCountExceededException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### MaxInvestmentCountExceededException()

```
public MaxInvestmentCountExceededException()
```

### MaxInvestmentCountExceededException(SerializationInfo, StreamingContext)

```
protected MaxInvestmentCountExceededException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MaxInvestmentCountExceededException(string)

public MaxInvestmentCountExceededException(string message)

Parameters

message [string](#)

# Class PreviousBountyException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PreviousBountyException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← PreviousBountyException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PreviousBountyException()

```
public PreviousBountyException()
```

### PreviousBountyException(SerializationInfo, StreamingContext)

```
protected PreviousBountyException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## PreviousBountyException(string)

public PreviousBountyException(string msg)

### Parameters

msg [string](#)

# Class SeasonInProgressException

Namespace: [Nekoyume.Action.Exceptions.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SeasonInProgressException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SeasonInProgressException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SeasonInProgressException()

```
public SeasonInProgressException()
```

### SeasonInProgressException(SerializationInfo, StreamingContext)

```
protected SeasonInProgressException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SeasonInProgressException(string)

public SeasonInProgressException(string msg)

Parameters

msg [string](#)

# Namespace Nekoyume.Action.Exceptions.Arena

## Classes

[AlreadyJoinedArenaException](#)

# Class AlreadyJoinedArenaException

Namespace: [Nekoyume.Action.Exceptions.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyJoinedArenaException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AlreadyJoinedArenaException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyJoinedArenaException(int, int, Address)

```
public AlreadyJoinedArenaException(int championshipId, int round,
Address avatarAddress)
```

## Parameters

championshipId [int](#)

round [int](#)

avatarAddress Address

## AlreadyJoinedArenaException(SerializationInfo, StreamingContext)

```
protected AlreadyJoinedArenaException(SerializationInfo info,  
StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AlreadyJoinedArenaException(string)

```
public AlreadyJoinedArenaException(string msg)
```

### Parameters

msg [string](#)

# Namespace Nekoyume.Action.Exceptions. CustomEquipmentCraft

## Classes

[NotEnoughRelationshipException](#)

[RandomOnlyIconException](#)

# Class NotEnoughRelationshipException

Namespace: [Nekoyume.Action.Exceptions.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughRelationshipException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughRelationshipException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughRelationshipException(SerializationInfo, StreamingContext)

```
protected NotEnoughRelationshipException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughRelationshipException(string)

```
public NotEnoughRelationshipException(string s)
```

### Parameters

s [string](#)

# Class RandomOnlyIconException

Namespace: [Nekoyume.Action.Exceptions.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
public class RandomOnlyIconException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RandomOnlyIconException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RandomOnlyIconException(int)

```
public RandomOnlyIconException(int iconId)
```

#### Parameters

iconId [int](#)

### RandomOnlyIconException(SerializationInfo, StreamingContext)

```
protected RandomOnlyIconException(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RandomOnlyIconException(string)

```
public RandomOnlyIconException(string s)
```

Parameters

s [string](#)

# Namespace Nekoyume.Action.Factory

## Classes

[ClaimStakeRewardFactory](#)

# Class ClaimStakeRewardFactory

Namespace: [Nekoyume.Action.Factory](#)

Assembly: Lib9c.dll

```
public static class ClaimStakeRewardFactory
```

## Inheritance

[object](#) ← ClaimStakeRewardFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### CreateByBlockIndex(long, Address)

```
public static IClaimStakeReward CreateByBlockIndex(long blockIndex,  
Address avatarAddress)
```

#### Parameters

blockIndex [long](#)

avatarAddress Address

#### Returns

[IClaimStakeReward](#)

### CreateByVersion(int, Address)

```
public static IClaimStakeReward CreateByVersion(int version, Address avatarAddress)
```

## Parameters

`version int`

`avatarAddress Address`

## Returns

[`IClaimStakeReward`](#)

# Namespace Nekoyume.Action.Garages

## Classes

[BulkUnloadFromGarages](#)

[DeliverToOthersGarages](#)

[GarageUtils](#)

[LoadIntoMyGarages](#)

[UnloadFromMyGarages](#)

# Class BulkUnloadFromGarages

Namespace: [Nekoyume.Action.Garages](#)

Assembly: Lib9c.dll

```
[ActionType("bulk_unload_from_garages")]
public class BulkUnloadFromGarages : GameAction, IBulkUnloadFromGaragesV1, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← BulkUnloadFromGarages

## Implements

IBulkUnloadFromGaragesV1, IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### BulkUnloadFromGarages()

```
public BulkUnloadFromGarages()
```

```
BulkUnloadFromGarages(IReadOnlyList<(Address  
recipientAvatarAddress, IEnumerable<(Address  
balanceAddress, FungibleAssetValue value)>?  
fungibleAssetValues,  
IEnumerable<(HashDigest<SHA256> fungibleId, int  
count)>? fungibleIdAndCounts, string? memo)>)
```

```
public BulkUnloadFromGarages(IReadOnlyList<(Address recipientAvatarAddress,
IEnumerable<(Address balanceAddress, FungibleAssetValue value)>?
fungibleAssetValues, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>?
fungibleIdAndCounts, string? memo)> unloadData)
```

## Parameters

unloadData [IReadOnlyList](#)<(Address [recipientAvatarAddress](#), [IEnumerable](#)<(Address [balanceAddress](#), FungibleAssetValue [value](#))> [fungibleAssetValues](#), [IEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))> [fungibleIdAndCounts](#), [string](#) [memo](#))>

## Properties

### PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

### UnloadData

```
public IReadOnlyList<(Address recipientAvatarAddress, IOrderedEnumerable<(Address
balanceAddress, FungibleAssetValue value)>? fungibleAssetValues,
IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count)>? fungibleIdAndCounts,
string? memo)> UnloadData { get; }
```

### Property Value

[IReadOnlyList](#)<(Address [recipientAvatarAddress](#), [IOrderedEnumerable](#)<(Address [balanceAddress](#), FungibleAssetValue [value](#))> [fungibleAssetValues](#), [IOrderedEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))> [fungibleIdAndCounts](#), [string](#) [memo](#))>

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary<string, IValue>](#)

# Class DeliverToOthersGarages

Namespace: [Nekoyume.Action.Garages](#)

Assembly: Lib9c.dll

```
[ActionType("deliver_to_others_garages")]
public class DeliverToOthersGarages : GameAction, IDeliverToOthersGaragesV1, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← DeliverToOthersGarages

## Implements

IDeliverToOthersGaragesV1, IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### DeliverToOthersGarages()

```
public DeliverToOthersGarages()
```

DeliverToOthersGarages(Address,  
IEnumerable<FungibleAssetValue>?,  
IEnumerable<(HashDigest<SHA256> fungibleId, int  
count)>?, string?)

```
public DeliverToOthersGarages(Address recipientAgentAddr,  
IEnumerable<FungibleAssetValue>? fungibleAssetValues,  
IEnumerable<(HashDigest<SHA256> fungibleId, int count)>? fungibleIdAndCounts,  
string? memo)
```

## Parameters

recipientAgentAddr Address

fungibleAssetValues [IEnumerable](#)<FungibleAssetValue>

fungibleIdAndCounts [IEnumerable](#)<(HashDigest<[SHA256](#)> fungibleId, int count)>

memo [string](#)

## Properties

### FungibleAssetValues

```
public IOorderedEnumerable<FungibleAssetValue>? FungibleAssetValues { get; }
```

#### Property Value

[IOorderedEnumerable](#)<FungibleAssetValue>

### FungibleIdAndCounts

```
public IOorderedEnumerable<(HashDigest<SHA256> fungibleId, int count)>?  
FungibleIdAndCounts { get; }
```

#### Property Value

[IOorderedEnumerable](#)<(HashDigest<[SHA256](#)> fungibleId, int count)>

### Memo

```
public string? Memo { get; }
```

Property Value

[string](#)

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), IValue>

## RecipientAgentAddr

```
public Address RecipientAgentAddr { get; }
```

Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language

and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

## Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Class GarageUtils

Namespace: [Nekoyume.Action.Garages](#)

Assembly: Lib9c.dll

```
public static class GarageUtils
```

## Inheritance

[object](#) ← GarageUtils

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

ConvertToFungibleItemGarage(IValue?, Address,  
HashDigest<SHA256>)

```
public static FungibleItemGarage ConvertToFungibleItemGarage(IValue? garageState,  
Address garageAddr, HashDigest<SHA256> fungibleId)
```

### Parameters

garageState IValue

garageAddr Address

fungibleId HashDigest<[SHA256](#)>

### Returns

[FungibleItemGarage](#)

## MergeAndSort(IEnumerable<(HashDigest<SHA256> fungibleId, int count)>?)

```
public static IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count)>? MergeAndSort(IEnumerable<(HashDigest<SHA256> fungibleId, int count)>? fungibleIdAndCounts)
```

### Parameters

fungibleIdAndCounts [IEnumerable](#)<(HashDigest<[SHA256](#)> fungibleId, [int](#) count)>

### Returns

[IOrderedEnumerable](#)<(HashDigest<[SHA256](#)> fungibleId, [int](#) count)>

## MergeAndSort(IEnumerable<(Address balanceAddr, FungibleAssetValue value)>?)

```
public static IOrderedEnumerable<(Address balanceAddr, FungibleAssetValue value)>? MergeAndSort(IEnumerable<(Address balanceAddr, FungibleAssetValue value)>? fungibleAssetValues)
```

### Parameters

fungibleAssetValues [IEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

### Returns

[IOrderedEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

## WithGarageStateTuples(Address, IWorldState, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>)

```
public static IOorderedEnumerable<(HashDigest<SHA256> fungibleId, int count,
Address garageAddr, IValue? garageState)> WithGarageStateTuples(Address
agentAddr, IWorldState states, IOenumerable<(HashDigest<SHA256> fungibleId, int
count)> fungibleIdAndCounts)
```

## Parameters

agentAddr Address

states IWorldState

fungibleIdAndCounts [IOenumerable](#)<(HashDigest<[SHA256](#)> fungibleId, int count)>

## Returns

[IOorderedEnumerable](#)<(HashDigest<[SHA256](#)> fungibleId, int count, Address garage
Addr, IValue garageState)>

## WithGarageStateTuples(Address, Address, IWorldState, IOenumerable<(HashDigest<SHA256> fungibleId, int count)>)

```
public static IOorderedEnumerable<(HashDigest<SHA256> fungibleId, int count, Address
senderGarageAddr, FungibleItemGarage senderGarage, Address recipientGarageAddr,
IValue? recipientGarageState)> WithGarageStateTuples(Address senderAgentAddr,
Address recipientAgentAddr, IWorldState states, IOenumerable<(HashDigest<SHA256>
fungibleId, int count)> fungibleIdAndCounts)
```

## Parameters

senderAgentAddr Address

recipientAgentAddr Address

states IWorldState

fungibleIdAndCounts [IOenumerable](#)<(HashDigest<[SHA256](#)> fungibleId, int count)>

## Returns

`IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count, Address senderGarageAddr, FungibleItemGarage senderGarage, Address recipientGarageAddr, IValue recipientGarageState)>`

`WithGarageTuples(Address, IWorldState, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>)`

```
public static IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count, Address garageAddr, FungibleItemGarage garage)> WithGarageTuples(Address agentAddr, IWorldState states, IEnumerable<(HashDigest<SHA256> fungibleId, int count)> fungibleIdAndCounts)
```

## Parameters

`agentAddr` Address

`states` IWorldState

`fungibleIdAndCounts` `IEnumerable<(HashDigest<SHA256> fungibleId, int count)>`

## Returns

`IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count, Address garageAddr, FungibleItemGarage garage)>`

# Class LoadIntoMyGarages

Namespace: [Nekoyume.Action.Garages](#)

Assembly: Lib9c.dll

```
[ActionType("load_into_my_garages")]
public class LoadIntoMyGarages : GameAction, ILoadIntoMyGaragesV1, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← LoadIntoMyGarages

## Implements

ILoadIntoMyGaragesV1, IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### LoadIntoMyGarages()

```
public LoadIntoMyGarages()
```

LoadIntoMyGarages(IEnumerable<(Address  
balanceAddr, FungibleAssetValue value)>?, Address?,  
IEnumerable<(HashDigest<SHA256> fungibleId, int  
count)>?, string?)

```
public LoadIntoMyGarages(IEnumerable<(Address balanceAddr, FungibleAssetValue value)>? fungibleAssetValues, Address? avatarAddr, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>? fungibleIdAndCounts, string? memo)
```

## Parameters

fungibleAssetValues [IEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

avatarAddr Address?

fungibleIdAndCounts [IEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))>

memo [string](#)

## Properties

### AvatarAddr

This address should belong to one of the signer's avatars. If the avatar state is v1, there is no separate inventory, so it should be execute another action first to migrate the avatar state to v2. And then, the inventory address will be set.

```
public Address? AvatarAddr { get; }
```

### Property Value

Address?

### FungibleAssetValues

```
public IOrderedEnumerable<(Address balanceAddr, FungibleAssetValue value)>? FungibleAssetValues { get; }
```

### Property Value

[IOrderedEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

## FungibleIdAndCounts

```
public IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count)?>
FungibleIdAndCounts { get; }
```

### Property Value

[IOrderedEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))>

## Memo

```
public string? Memo { get; }
```

### Property Value

[string](#)

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

### Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the [context](#) object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IIImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IIImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IIImmutableDictionary](#)<[string](#), IValue>

# Class UnloadFromMyGarages

Namespace: [Nekoyume.Action.Garages](#)

Assembly: Lib9c.dll

```
[ActionType("unload_from_my_garages")]
public class UnloadFromMyGarages : GameAction, IUnloadFromMyGaragesV1, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← [GameAction](#) ← [UnloadFromMyGarages](#)

## Implements

IUnloadFromMyGaragesV1, IAction

## Inherited Members

[GameAction.Id](#) , [GameAction.PlainValue](#) , [GameAction.LoadPlainValue\(IValue\)](#) ,  
[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnloadFromMyGarages()

```
public UnloadFromMyGarages()
```

UnloadFromMyGarages(Address, IEnumerable<(Address  
balanceAddr, FungibleAssetValue value)>?,  
IEnumerable<(HashDigest<SHA256> fungibleId, int  
count)>?, string?)

```
public UnloadFromMyGarages(Address recipientAvatarAddr, IEnumerable<(Address balanceAddr, FungibleAssetValue value)>? fungibleAssetValues, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>? fungibleIdAndCounts, string? memo)
```

## Parameters

recipientAvatarAddr Address

fungibleAssetValues [IEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

fungibleIdAndCounts [IEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))>

memo [string](#)

## Properties

### FungibleAssetValues

```
public IOrderedEnumerable<(Address balanceAddr, FungibleAssetValue value)>? FungibleAssetValues { get; }
```

#### Property Value

[IOrderedEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

### FungibleIdAndCounts

```
public IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count)>? FungibleIdAndCounts { get; }
```

#### Property Value

[IOrderedEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))>

## Memo

```
public string? Memo { get; }
```

Property Value

[string](#)

## PlainValueInternal

```
protected override IImmutableDictionary<string, IValue> PlainValueInternal { get; }
```

Property Value

[IImmutableDictionary](#)<[string](#), [IValue](#)>

## RecipientAvatarAddr

If the avatar state is v1, there is no separate inventory, so it should be execute another action first to migrate the avatar state to v2. And then, the inventory address will be set.

```
public Address RecipientAvatarAddr { get; }
```

Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValueInternal(IImmutableDictionary<string, IValue>)

```
protected override void LoadPlainValueInternal(IImmutableDictionary<string, IValue> plainValue)
```

### Parameters

plainValue [IImmutableDictionary](#)<[string](#), IValue>

# Namespace Nekoyume.Action.Guild

## Classes

[BanGuildMember](#)

[ClaimGuildReward](#)

[ClaimReward](#)

[GuildConfig](#)

[JoinGuild](#)

[MakeGuild](#)

[MoveGuild](#)

[QuitGuild](#)

[RemoveGuild](#)

An action to remove the guild.

[UnbanGuildMember](#)

# Class BanGuildMember

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
public class BanGuildMember : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← BanGuildMember

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### BanGuildMember()

```
public BanGuildMember()
```

### BanGuildMember(AgentAddress)

```
public BanGuildMember(AgentAddress target)
```

## Parameters

target [AgentAddress](#)

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "ban_guild_member"
```

### Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

#### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

#### See Also

[LoadPlainValue\(IValue\)](#)

## Target

```
public AgentAddress Target { get; }
```

### Property Value

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called “dirty”).

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a

Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## **See Also**

[PlainValue](#)

# Class ClaimGuildReward

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
[ActionType("claim_guild_reward")]
public sealed class ClaimGuildReward : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← ClaimGuildReward

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ClaimGuildReward()

```
public ClaimGuildReward()
```

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "claim_guild_reward"
```

### Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

`LoadPlainValue(IValue)`

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

### `IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue IValue`

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class ClaimReward

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
[ActionType("claim_reward")]
public sealed class ClaimReward : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← ClaimReward

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ClaimReward()

```
public ClaimReward()
```

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "claim_reward"
```

## Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

`LoadPlainValue(IValue)`

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

### `IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue IValue`

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class GuildConfig

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
public static class GuildConfig
```

## Inheritance

[object](#) ← GuildConfig

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### PlanetariumGuildOwner

```
public static readonly AgentAddress PlanetariumGuildOwner
```

#### Field Value

[AgentAddress](#)

# Class JoinGuild

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
public class JoinGuild : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← JoinGuild

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### JoinGuild()

```
public JoinGuild()
```

### JoinGuild(GuildAddress)

```
public JoinGuild(GuildAddress guildAddress)
```

## Parameters

guildAddress [GuildAddress](#)

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "join_guild"
```

### Field Value

[string](#)

# Properties

## GuildAddress

```
public GuildAddress GuildAddress { get; }
```

### Property Value

[GuildAddress](#)

# PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called “dirty”).

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a

Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class MakeGuild

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
[ActionType("make_guild")]
public class MakeGuild : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MakeGuild

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MakeGuild()

```
public MakeGuild()
```

### MakeGuild(Address)

```
public MakeGuild(Address validatorAddress)
```

## Parameters

validatorAddress Address

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "make_guild"
```

### Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

#### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

#### See Also

[LoadPlainValue\(IValue\)](#)

## ValidatorAddress

```
public Address ValidatorAddress { get; }
```

### Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called "dirty").

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a

Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## ExecutePublic(IActionContext)

```
public IWorld ExecutePublic(IActionContext context)
```

### Parameters

context IActionContext

### Returns

IWorld

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

### Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

PlainValue

# Class MoveGuild

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
public class MoveGuild : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MoveGuild

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MoveGuild()

```
public MoveGuild()
```

### MoveGuild(GuildAddress)

```
public MoveGuild(GuildAddress guildAddress)
```

## Parameters

guildAddress [GuildAddress](#)

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "move_guild"
```

### Field Value

[string](#)

# Properties

## GuildAddress

```
public GuildAddress GuildAddress { get; }
```

### Property Value

[GuildAddress](#)

# PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called "dirty").

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a

Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## **See Also**

[PlainValue](#)

# Class QuitGuild

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
public class QuitGuild : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← QuitGuild

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "quit_guild"
```

#### Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

### Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

PlainValue

# Class RemoveGuild

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

An action to remove the guild.

```
public class RemoveGuild : ActionBase, IAction
```

## Inheritance

[object](#) ↴ ← [ActionBase](#) ← RemoveGuild

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "remove_guild"
```

### Field Value

[string](#) ↴

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

### Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

PlainValue

# Class UnbanGuildMember

Namespace: [Nekoyume.Action.Guild](#)

Assembly: Lib9c.dll

```
public class UnbanGuildMember : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← UnbanGuildMember

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnbanGuildMember()

```
public UnbanGuildMember()
```

### UnbanGuildMember(Address)

```
public UnbanGuildMember(Address target)
```

## Parameters

**target** Address

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "unban_guild_member"
```

## Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Target

```
public Address Target { get; }
```

## Property Value

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called "dirty").

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a

Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Namespace Nekoyume.Action.Guild. Migration

## Classes

[GuildMigrationFailedException](#)

[MigrateDelegation](#)

An action to migrate guild delegation.

[MigrateDelegationHeight](#)

An action to migrate the delegation height.

[MigratePlanetariumGuild](#)

An action to migrate the planetarium guild.

# Class GuildMigrationFailedException

Namespace: [Nekoyume.Action.Guild.Migration](#)

Assembly: Lib9c.dll

```
public class GuildMigrationFailedException : InvalidOperationException,  
ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
GuildMigrationFailedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### GuildMigrationFailedException(string)

```
public GuildMigrationFailedException(string message)
```

## Parameters

message [string](#)

# Class MigrateDelegation

Namespace: [Nekoyume.Action.Guild.Migration](#)

Assembly: Lib9c.dll

An action to migrate guild delegation.

```
[ActionType("migrate_delegation")]
public class MigrateDelegation : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MigrateDelegation

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MigrateDelegation()

```
[Obsolete("Don't call in code.", false)]
public MigrateDelegation()
```

### MigrateDelegation(AgentAddress)

```
public MigrateDelegation(AgentAddress target)
```

## Parameters

target [AgentAddress](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "migrate_delegation"
```

#### Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

#### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

`LoadPlainValue(IValue)`

## Target

```
public AgentAddress Target { get; }
```

## Property Value

[AgentAddress](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

#### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

#### Returns

IWorld

A map of changed states (so-called "dirty").

#### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IAction Context) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class MigrateDelegationHeight

Namespace: [Nekoyume.Action.Guild.Migration](#)

Assembly: Lib9c.dll

An action to migrate the delegation height.

```
[ActionType("migrate_delegation_height")]
public class MigrateDelegationHeight : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MigrateDelegationHeight

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MigrateDelegationHeight()

```
public MigrateDelegationHeight()
```

### MigrateDelegationHeight(long)

```
public MigrateDelegationHeight(long height)
```

## Parameters

height [long](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "migrate_delegation_height"
```

#### Field Value

[string](#)

## Properties

### Height

```
public long Height { get; }
```

#### Property Value

[long](#)

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

#### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and [Libplanet.Action.IActionContexts](#), the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

[PlainValue](#)

# Class MigratePlanetariumGuild

Namespace: [Nekoyume.Action.Guild.Migration](#)

Assembly: Lib9c.dll

An action to migrate the planetarium guild.

```
[ActionType("migrate_planetarium_guild")]
public class MigratePlanetariumGuild : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← MigratePlanetariumGuild

## Implements

IAction

## Inherited Members

[ActionBase.GetSignerAndOtherAddressesHex\(IActionContext, params Address\[\]\)](#) ,  
[ActionBase.TryGetAdminState\(IActionContext, out AdminState\)](#) ,  
[ActionBase.CheckPermission\(IActionContext\)](#) ,  
[ActionBase.CheckObsolete\(long, IActionContext\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MigratePlanetariumGuild()

```
public MigratePlanetariumGuild()
```

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "migrate_planetarium_guild"
```

## Field Value

[string](#) ↗

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

### Property Value

#### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

#### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

# Parameters

## context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

# Returns

## IWorld

A map of changed states (so-called "dirty").

# Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

# LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Namespace Nekoyume.Action.Guild. Migration.LegacyModels

## Classes

[LegacyGuild](#)

[LegacyGuildParticipant](#)

[MigrationModule](#)

# Class LegacyGuild

Namespace: [Nekoyume.Action.Guild.Migration.LegacyModels](#)

Assembly: Lib9c.dll

```
public class LegacyGuild : IEquatable<LegacyGuild>, IBencodable
```

## Inheritance

[object](#) ← LegacyGuild

## Implements

[IEquatable](#)<[LegacyGuild](#)>, [IBencodable](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### LegacyGuild(List)

```
public LegacyGuild(List list)
```

#### Parameters

list List

### LegacyGuild(AgentAddress)

```
public LegacyGuild(AgentAddress guildMasterAddress)
```

#### Parameters

guildMasterAddress [AgentAddress](#)

# Fields

## GuildMasterAddress

```
public readonly AgentAddress GuildMasterAddress
```

### Field Value

[AgentAddress](#)

# Properties

## Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

# Methods

## Equals(LegacyGuild)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(LegacyGuild other)
```

### Parameters

other [LegacyGuild](#)

An object to compare with this object.

### Returns

[bool](#) ↗

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

### Parameters

[obj](#) [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

### Returns

[int](#)

A hash code for the current object.

# Class LegacyGuildParticipant

Namespace: [Nekoyume.Action.Guild.Migration.LegacyModels](#)

Assembly: Lib9c.dll

```
public class LegacyGuildParticipant : IBencodable,  
IEquatable<LegacyGuildParticipant>
```

## Inheritance

[object](#) ← LegacyGuildParticipant

## Implements

IBencodable, [IEquatable](#)<[LegacyGuildParticipant](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### LegacyGuildParticipant(List)

```
public LegacyGuildParticipant(List list)
```

#### Parameters

list List

### LegacyGuildParticipant(GuildAddress)

```
public LegacyGuildParticipant(GuildAddress guildAddress)
```

#### Parameters

guildAddress [GuildAddress](#)

# Fields

## GuildAddress

```
public readonly GuildAddress GuildAddress
```

### Field Value

[GuildAddress](#)

# Properties

## Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

# Methods

## Equals(LegacyGuildParticipant)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(LegacyGuildParticipant other)
```

### Parameters

other [LegacyGuildParticipant](#)

An object to compare with this object.

### Returns

[bool](#) ↗

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

### Parameters

[obj](#) [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

### Returns

[int](#)

A hash code for the current object.

# Class MigrationModule

Namespace: [Nekoyume.Action.Guild.Migration.LegacyModels](#)

Assembly: Lib9c.dll

```
public static class MigrationModule
```

## Inheritance

[object](#) ← MigrationModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### DelegationMigrationHeight

An address for delegation height migration.

```
public static readonly Address DelegationMigrationHeight
```

## Field Value

Address

## Methods

### GetDelegationMigrationHeight(IWorldState)

```
public static long? GetDelegationMigrationHeight(this IWorldState worldState)
```

## Parameters

**worldState** IWorldState

Returns

[long](#)?

## SetDelegationMigrationHeight(IWorld, long)

```
public static IWorld SetDelegationMigrationHeight(this IWorld world, long height)
```

Parameters

**world** IWorld

**height** [long](#)

Returns

IWorld

# Namespace Nekoyume.Action.Loader

## Classes

### [NCActionLoader](#)

An Libplanet.Action.Loader.IActionLoader implementation for Nine Chronicles's BlockChain. This is a simple wrapper around a Libplanet.Action.Loader.TypedActionLoader for loading all Libplanet.Action.IAction classes within the same assembly as the [Action Base](#) class, inheriting from the [ActionBase](#) class, and has an Libplanet.Action.ActionType Attribute from Bencodex.Types.IValues

# Class NCActionLoader

Namespace: [Nekoyume.Action.Loader](#)

Assembly: Lib9c.dll

An Libplanet.Action.Loader.IActionLoader implementation for Nine Chronicles's BlockChain. This is a simple wrapper around a Libplanet.Action.Loader.TypedActionLoader for loading all Libplanet.Action.IAction classes within the same assembly as the [ActionBase](#) class, inheriting from the [ActionBase](#) class, and has an Libplanet.Action.ActionTypeAttribute from Bencodex.Types.IValues

```
public class NCActionLoader : IActionLoader
```

## Inheritance

[object](#) ← NCActionLoader

## Implements

IActionLoader

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### NCActionLoader()

```
public NCActionLoader()
```

## Methods

### LoadAction(long, IValue)

Loads a concrete Libplanet.Action.IAction from [value](#) given [index](#).

```
public IAction LoadAction(long index, IValue value)
```

## Parameters

**index** [long](#) ↗

The index of a block to use as context for deserializing **value**.

**value** [IValue](#)

The Bencodex.Types.IValue to deserialize to an Libplanet.Action.IAction.

## Returns

[IAction](#)

An Libplanet.Action.IAction instantiated with **value**.

## Exceptions

[InvalidActionException](#)

Thrown when an Libplanet.Action.IAction cannot be instantiated with given **index** and **value**.

## See Also

[TypedActionLoader](#)

# Namespace Nekoyume.Action.Validator Delegation

## Classes

[AllocateGuildReward](#)

[AllocateReward](#)

[ClaimValidatorRewardSelf](#)

[DelegateValidator](#)

[Mortgage](#)

An action for mortgage gas fee for a transaction. Should be executed at the beginning of the tx.

[PromoteValidator](#)

[RecordProposer](#)

An action for recording proposer of the block to use in next block's reward distribution.

[Refund](#)

An action for refund gas fee for a transaction. Should be executed at the beginning of the tx.

[ReleaseValidatorUnbondings](#)

[Reward](#)

An action for reward for a transaction. Should be executed at the beginning of the tx.

[SetValidatorCommission](#)

[SlashValidator](#)

[UndelegateValidator](#)

[UnjailValidator](#)

[UpdateValidators](#)

[ValidatorConfig](#)

# Class AllocateGuildReward

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class AllocateGuildReward : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← AllocateGuildReward

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AllocateGuildReward()

```
public AllocateGuildReward()
```

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

## IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

## plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

[PlainValue](#)

# Class AllocateReward

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class AllocateReward : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← AllocateReward

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AllocateReward()

```
public AllocateReward()
```

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

## IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

## **plainValue** **IValue**

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### **See Also**

[PlainValue](#)

# Class ClaimValidatorRewardSelf

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
[ActionType("claim_validator_reward_self")]
public sealed class ClaimValidatorRewardSelf : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← ClaimValidatorRewardSelf

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ClaimValidatorRewardSelf()

```
public ClaimValidatorRewardSelf()
```

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "claim_validator_reward_self"
```

### Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

`LoadPlainValue(IValue)`

## ValidatorDelegatee

```
public Address ValidatorDelegatee { get; }
```

## Property Value

### Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

**context** IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [Current Culture](#) on .NET; if you format numbers, dates and times, currencies, or other such things

into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class DelegateValidator

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
[ActionType("delegate_validator")]
public sealed class DelegateValidator : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← DelegateValidator

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### DelegateValidator()

```
public DelegateValidator()
```

### DelegateValidator(FungibleAssetValue)

```
public DelegateValidator(FungibleAssetValue fav)
```

## Parameters

**fav** FungibleAssetValue

## Fields

# TypeIdentifier

```
public const string TypeIdentifier = "delegate_validator"
```

Field Value

[string](#)

## Properties

FAV

```
public FungibleAssetValue FAV { get; }
```

Property Value

FungibleAssetValue

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

`LoadPlainValue(IValue)`

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class Mortgage

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

An action for mortgage gas fee for a transaction. Should be executed at the beginning of the tx.

```
public sealed class Mortgage : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← Mortgage

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Mortgage()

Creates a new instance of [Mortgage](#).

```
public Mortgage()
```

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other "bound" information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class PromoteValidator

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
[ActionType("promote_validator")]
public sealed class PromoteValidator : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← PromoteValidator

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### PromoteValidator()

```
public PromoteValidator()
```

### PromoteValidator(PublicKey, FungibleAssetValue)

```
public PromoteValidator(PublicKey publicKey, FungibleAssetValue fav)
```

## Parameters

**publicKey** PublicKey

**fav** FungibleAssetValue

# PromoteValidator(PublicKey, FungibleAssetValue, BigInteger)

```
public PromoteValidator(PublicKey publicKey, FungibleAssetValue fav,  
BigInteger commissionPercentage)
```

## Parameters

publicKey PublicKey

fav FungibleAssetValue

commissionPercentage [BigInteger](#)

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "promote_validator"
```

## Field Value

[string](#)

## Properties

### CommissionPercentage

```
public BigInteger CommissionPercentage { get; }
```

## Property Value

[BigInteger](#)

## FAV

```
public FungibleAssetValue FAV { get; }
```

Property Value

FungibleAssetValue

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## PublicKey

```
public PublicKey PublicKey { get; }
```

Property Value

PublicKey

## Methods

Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` `IActionContext`

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

`IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but

equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## ExecutePublic(IActionContext)

```
public IWorld ExecutePublic(IActionContext context)
```

### Parameters

context IActionContext

### Returns

IWorld

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

### `plainValue IValue`

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

[PlainValue](#)

# Class RecordProposer

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

An action for recording proposer of the block to use in next block's reward distribution.

```
public sealed class RecordProposer : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← RecordProposer

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RecordProposer()

Creates a new instance of [RecordProposer](#).

```
public RecordProposer()
```

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other "bound" information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class Refund

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

An action for refund gas fee for a transaction. Should be executed at the beginning of the tx.

```
public sealed class Refund : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← Refund

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## Refund()

Creates a new instance of [Refund](#).

```
public Refund()
```

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other "bound" information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class ReleaseValidatorUnbondings

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class ReleaseValidatorUnbondings : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← ReleaseValidatorUnbondings

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ReleaseValidatorUnbondings()

```
public ReleaseValidatorUnbondings()
```

### ReleaseValidatorUnbondings(Address)

```
public ReleaseValidatorUnbondings(Address validatorDelegatee)
```

## Parameters

validatorDelegatee Address

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

LoadPlainValue(IValue)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

### IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

### Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

PlainValue

# Class Reward

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

An action for reward for a transaction. Should be executed at the beginning of the tx.

```
public sealed class Reward : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← Reward

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Reward()

Creates a new instance of [Refund](#).

```
public Reward()
```

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other "bound" information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See [Libplanet.Action.IActionContext](#) for details.

## Returns

### IWorld

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class SetValidatorCommission

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
[ActionType("set_validator_commission")]
public sealed class SetValidatorCommission : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← SetValidatorCommission

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SetValidatorCommission()

```
public SetValidatorCommission()
```

### SetValidatorCommission(Address, BigInteger)

```
public SetValidatorCommission(Address validatorDelegatee,
BigInteger commissionPercentage)
```

## Parameters

validatorDelegatee Address

commissionPercentage [BigInteger](#)

# Fields

## TypeIdentifier

```
public const string TypeIdentifier = "set_validator_commission"
```

### Field Value

[string](#) ↗

# Properties

## CommissionPercentage

```
public BigInteger CommissionPercentage { get; }
```

### Property Value

[BigInteger](#) ↗

# PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

### Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

LoadPlainValue(IValue)

## ValidatorDelegatee

```
public Address ValidatorDelegatee { get; }
```

Property Value

Address

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

Returns

IWorld

A map of changed states (so-called “dirty”).

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class SlashValidator

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class SlashValidator : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← SlashValidator

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SlashValidator()

```
public SlashValidator()
```

## Properties

### AbstainJailTime

```
public static long AbstainJailTime { get; }
```

Property Value

[long](#)

### DuplicateVoteSlashFactor

```
public static BigInteger DuplicateVoteSlashFactor { get; }
```

Property Value

[BigInteger](#)

## LivenessSlashFactor

```
public static BigInteger LivenessSlashFactor { get; }
```

Property Value

[BigInteger](#)

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

Property Value

IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

[Execute\(IActionContext\)](#)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` `IActionContext`

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

## Returns

`IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but

equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

**plainValue** IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class UndelegateValidator

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
[ActionType("undelegate_validator")]
public sealed class UndelegateValidator : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← UndelegateValidator

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UndelegateValidator()

```
public UndelegateValidator()
```

### UndelegateValidator(BigInteger)

```
public UndelegateValidator(BigInteger share)
```

## Parameters

share [BigInteger](#)

## Fields

# TypeIdentifier

```
public const string TypeIdentifier = "undelegate_validator"
```

## Field Value

[string](#)

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue) method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Share

```
public BigInteger Share { get; }
```

## Property Value

[BigInteger](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

#### context IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

#### IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should be avoided, because an action's Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext) method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the Blockchain.Renderers.IRenderer interface separately and attach it to a Blockchain.BlockChain instance.

For randomness, *never* use [Random](#) nor any other PRNGs provided by other than Libplanet. Use IActionContext.Random instead. IActionContext.Random guarantees the

same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

`plainValue` IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class UnjailValidator

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
[ActionType("unjail_validator")]
public sealed class UnjailValidator : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← UnjailValidator

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnjailValidator()

```
public UnjailValidator()
```

## Fields

### TypeIdentifier

```
public const string TypeIdentifier = "unjail_validator"
```

## Field Value

[string](#)

# Properties

## PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

## Property Value

### IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

## See Also

[LoadPlainValue\(IValue\)](#)

# Methods

## Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned `Libplanet.Action.State.IWorld` object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

## Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See `Libplanet.Action.IActionContext` for details.

## Returns

### `IWorld`

A map of changed states (so-called "dirty").

## Remarks

This method should be deterministic: for structurally (member-wise) equal actions and `Libplanet.Action.IActionContexts`, the same result should be returned. Side effects should be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than `Libplanet`. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

IActionContext

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data Libplanet.Action.IAction.PlainValue property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

### Parameters

`plainValue IValue`

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

## See Also

PlainValue

# Class UpdateValidators

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class UpdateValidators : ActionBase, IAction
```

## Inheritance

[object](#) ← [ActionBase](#) ← UpdateValidators

## Implements

IAction

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UpdateValidators()

```
public UpdateValidators()
```

## Properties

### PlainValue

Serializes values bound to an action, which is held by properties (or fields) of an action, so that they can be transmitted over network or saved to persistent storage.

Serialized values are deserialized by `Libplanet.Action.IAction.LoadPlainValue(Bencodex.Types.IValue)` method later.

```
public override IValue PlainValue { get; }
```

### Property Value

## IValue

A Bencodex value which encodes this action's bound values (held by properties or fields).

### See Also

[LoadPlainValue\(IValue\)](#)

## Methods

### Execute(IActionContext)

Executes the main game logic of an action. This should be *deterministic*.

Through the `context` object, it receives information such as a transaction signer, its states immediately before the execution, and a deterministic random seed.

Other “bound” information resides in the action object in itself, as its properties (or fields).

A returned Libplanet.Action.State.IWorld object functions as a delta which shifts from previous states to next states.

```
public override IWorld Execute(IActionContext context)
```

### Parameters

`context` IActionContext

A context object containing addresses that signed the transaction, states immediately before the execution, and a PRNG object which produces deterministic random numbers. See Libplanet.Action.IActionContext for details.

### Returns

IWorld

A map of changed states (so-called "dirty").

### Remarks

This method should be deterministic: for structurally (member-wise) equal actions and Libplanet.Action.IActionContexts, the same result should be returned. Side effects should

be avoided, because an action's `Libplanet.Action.IAction.Execute(Libplanet.Action.IActionContext)` method can be called more than once, the time it's called is difficult to predict.

For changing in-memory game states or drawing graphics, implement the `Blockchain.Renderers.IRenderer` interface separately and attach it to a `Blockchain.BlockChain` instance.

For randomness, never use [Random](#) nor any other PRNGs provided by other than Libplanet. Use `IActionContext.Random` instead. `IActionContext.Random` guarantees the same action has the consistent result for every node in the network.

Also do not perform I/O operations such as file system access or networking. These bring an action indeterministic. You maybe fine to log messages for debugging purpose, but equivalent messages could be logged multiple times.

Although it might be surprising, [floating-point arithmetics are underspecified so that it can make different results on different machines, platforms, runtimes, compilers, and builds](#).

Lastly, you need to be aware and keep in mind that there is a global state named [CurrentCulture](#) on .NET; if you format numbers, dates and times, currencies, or other such things into strings and parse these strings back these can rely on [CurrentCulture](#), so that the same action make different results on two differently configured systems like Thai language and French language. In order to make these types of conversions deterministic, you have to explicitly pass [InvariantCulture](#).

For more on determinism in general, please read also [Tendermint ABCI's docs on determinism](#).

Lastly, you can conduct static analysis on your code using [Libplanet.Analyzers](#). The analyzer can be enabled by adding its NuGet package into your project as a dependency.

## See Also

`IActionContext`

## LoadPlainValue(IValue)

Deserializes serialized data (i.e., data `Libplanet.Action.IAction.PlainValue` property made), and then fills this action's properties (or fields) with the deserialized values.

```
public override void LoadPlainValue(IValue plainValue)
```

## Parameters

## plainValue IValue

Data (made by Libplanet.Action.IAction.PlainValue property) to be deserialized and assigned to this action's properties (or fields).

### See Also

[PlainValue](#)

# Class ValidatorConfig

Namespace: [Nekoyume.Action.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public static class ValidatorConfig
```

## Inheritance

[object](#) ← ValidatorConfig

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### PlanetariumValidatorAddress

```
public static readonly Address PlanetariumValidatorAddress
```

Field Value

Address

### PlanetariumValidatorPublicKey

```
public static readonly PublicKey PlanetariumValidatorPublicKey
```

Field Value

PublicKey

# Namespace Nekoyume.Arena

## Classes

### [ArenaHelper](#)

There are only things that don't change

### [ArenaSimulator](#)

Changed at <https://github.com/planetarium/lib9c/pull/2229> ↗

### [ArenaSimulator2](#)

This class is only for previewnet.

## Interfaces

### [IArenaSimulator](#)

# Class ArenaHelper

Namespace: [Nekoyume.Arena](#)

Assembly: Lib9c.dll

There are only things that don't change

```
public static class ArenaHelper
```

## Inheritance

[object](#) ← ArenaHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### ScoreLimits

```
public static readonly IReadOnlyDictionary<ArenaType, (int upper, int lower)> ScoreLimits
```

Field Value

[IReadOnlyDictionary](#)<[ArenaType](#), ([int](#), [int](#))>

### ScoreLimitsV1

```
[Obsolete("Use `ScoreLimits` instead.")]
public static readonly IReadOnlyDictionary<ArenaType, (int, int)> ScoreLimitsV1
```

Field Value

[IReadOnlyDictionary](#)<[ArenaType](#), ([int](#), [int](#))>

## ScoreLimitsV2

```
[Obsolete("Use `ScoreLimits` instead.")]
public static readonly IReadOnlyDictionary<ArenaType, (int upper, int
lower)> ScoreLimitsV2
```

### Field Value

[IReadOnlyDictionary](#)<[ArenaType](#), ([int](#), [int](#))>

## ScoreLimitsV3

```
[Obsolete("Use `ScoreLimits` instead.")]
public static readonly IReadOnlyDictionary<ArenaType, (int upper, int
lower)> ScoreLimitsV3
```

### Field Value

[IReadOnlyDictionary](#)<[ArenaType](#), ([int](#), [int](#))>

## Methods

### DeriveArenaAddress(int, int)

```
public static Address DeriveArenaAddress(int championshipId, int round)
```

### Parameters

championshipId [int](#)

round [int](#)

### Returns

Address

## GetCurrentTicketResetCount(long, long, int)

```
public static int GetCurrentTicketResetCount(long currentBlockIndex, long roundStartBlockIndex, int interval)
```

### Parameters

currentBlockIndex [long](#)

roundStartBlockIndex [long](#)

interval [int](#)

### Returns

[int](#)

## GetEntranceFee(RoundData, long, int)

```
public static FungibleAssetValue GetEntranceFee(ArenaSheet.RoundData roundData, long currentBlockIndex, int avatarLevel)
```

### Parameters

roundData [ArenaSheet.RoundData](#)

currentBlockIndex [long](#)

avatarLevel [int](#)

### Returns

FungibleAssetValue

## GetMaxPurchasedTicketCount(RoundData)

```
[Obsolete("not use since v100320, battle_arena6. Use  
`ArenaSheet.RoundData.MaxPurchaseCount`")]
public static long GetMaxPurchasedTicketCount(ArenaSheet.RoundData roundData)
```

Parameters

roundData [ArenaSheet.RoundData](#)

Returns

[long](#)

## GetMedalTotalCount(Row, AvatarState)

```
public static int GetMedalTotalCount(ArenaSheet.Row row, AvatarState avatarState)
```

Parameters

row [ArenaSheet.Row](#)

avatarState [AvatarState](#)

Returns

[int](#)

## GetRewardCount(int)

```
public static int GetRewardCount(int score)
```

Parameters

score [int](#)

Returns

[int](#)

## GetScores(int, int)

```
public static (int myWinScore, int myDefeatScore, int enemyDefeatScore)
GetScores(int myScore, int enemyScore)
```

### Parameters

myScore [int](#)

enemyScore [int](#)

### Returns

([int](#) [myWinScore](#), [int](#) [myDefeatScore](#), [int](#) [enemyDefeatScore](#))

## GetScoresV1(int, int)

```
[Obsolete("Use `GetScores()` instead.")]
public static (int myWinScore, int myDefeatScore, int enemyDefeatScore)
GetScoresV1(int myScore, int enemyScore)
```

### Parameters

myScore [int](#)

enemyScore [int](#)

### Returns

([int](#) [myWinScore](#), [int](#) [myDefeatScore](#), [int](#) [enemyDefeatScore](#))

## GetTicketPrice(RoundData, ArenaInformation, Currency)

```
public static FungibleAssetValue GetTicketPrice(ArenaSheet.RoundData roundData,
ArenaInformation arenaInformation, Currency currency)
```

## Parameters

roundData [ArenaSheet.RoundData](#)

arenaInformation [ArenaInformation](#)

currency Currency

## Returns

FungibleAssetValue

## ValidateScoreDifference(IReadOnlyDictionary<ArenaType, (int, int)>, ArenaType, int, int)

```
public static bool ValidateScoreDifference(IReadOnlyDictionary<ArenaType, (int, int)> scoreLimits, ArenaType arenaType, int myScore, int enemyScore)
```

## Parameters

scoreLimits [IReadOnlyDictionary<ArenaType, \(int, int\)>](#)

arenaType [ArenaType](#)

myScore [int](#)

enemyScore [int](#)

## Returns

[bool](#)

## ValidateScoreDifferenceV1(IReadOnlyDictionary<ArenaType, (int, int)>, ArenaType, int, int)

```
[Obsolete("Use `ValidateScoreDifference()` instead.")]
public static bool ValidateScoreDifferenceV1(IReadOnlyDictionary<ArenaType, (int, int)> scoreLimits, ArenaType arenaType, int myScore, int enemyScore)
```

## Parameters

scoreLimits [IReadOnlyDictionary](#)<ArenaType, (int, int)>

arenaType [ArenaType](#)

myScore [int](#)

enemyScore [int](#)

## Returns

[bool](#)

## ValidateScoreDifferenceV2(IReadOnlyDictionary<ArenaType, (int, int)>, ArenaType, int, int)

```
[Obsolete("Use `ValidateScoreDifference()` instead.")]
public static bool ValidateScoreDifferenceV2(IReadOnlyDictionary<ArenaType, (int, int)> scoreLimits, ArenaType arenaType, int myScore, int enemyScore)
```

## Parameters

scoreLimits [IReadOnlyDictionary](#)<ArenaType, (int, int)>

arenaType [ArenaType](#)

myScore [int](#)

enemyScore [int](#)

## Returns

[bool](#)

# Class ArenaSimulator

Namespace: [Nekoyume.Arena](#)

Assembly: Lib9c.dll

Changed at <https://github.com/planetarium/lib9c/pull/2229>

```
public class ArenaSimulator : IArenaSimulator
```

## Inheritance

[object](#) ← ArenaSimulator

## Implements

[IArenaSimulator](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaSimulator(IRandom, int, long)

```
public ArenaSimulator(IRandom random, int hpModifier = 2, long  
shatterStrikeMaxDamage = 400000)
```

## Parameters

random IRandom

hpModifier [int](#)

shatterStrikeMaxDamage [long](#)

## Properties

## BuffLinkSheet

```
public BuffLinkSheet BuffLinkSheet { get; set; }
```

Property Value

[BuffLinkSheet](#)

## DeBuffLimitSheet

```
public DeBuffLimitSheet DeBuffLimitSheet { get; }
```

Property Value

[DeBuffLimitSheet](#)

## HpModifier

```
public int HpModifier { get; }
```

Property Value

[int](#)

## Log

```
public ArenaLog Log { get; }
```

Property Value

[ArenaLog](#)

## Random

```
public IRandom Random { get; }
```

Property Value

IRandom

## ShatterStrikeMaxDamage

```
public long ShatterStrikeMaxDamage { get; }
```

Property Value

[long](#)

## Turn

```
public int Turn { get; }
```

Property Value

[int](#)

## Methods

Simulate(ArenaPlayerDigest, ArenaPlayerDigest, ArenaSimulatorSheets, List<StatModifier>, List<StatModifier>, DeBuffLimitSheet, BuffLinkSheet, bool)

```
public ArenaLog Simulate(ArenaPlayerDigest challenger, ArenaPlayerDigest enemy, ArenaSimulatorSheets sheets, List<StatModifier> challengerCollectionModifiers, List<StatModifier> enemyCollectionModifiers, DeBuffLimitSheet deBuffLimitSheet, BuffLinkSheet buffLinkSheet, bool setExtraValueBuffBeforeGetBuffs = false)
```

## Parameters

challenger [ArenaPlayerDigest](#)

enemy [ArenaPlayerDigest](#)

sheets [ArenaSimulatorSheets](#)

challengerCollectionModifiers [List](#)<[StatModifier](#)>

enemyCollectionModifiers [List](#)<[StatModifier](#)>

deBuffLimitSheet [DeBuffLimitSheet](#)

buffLinkSheet [BuffLinkSheet](#)

setExtraValueBuffBeforeGetBuffs [bool](#)

## Returns

[ArenaLog](#)

# Class ArenaSimulator2

Namespace: [Nekoyume.Arena](#)

Assembly: Lib9c.dll

This class is only for previewnet.

```
public class ArenaSimulator2 : Simulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← ArenaSimulator2

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaSimulator2(IRandom, ArenaPlayerDigest, ArenaPlayerDigest, ArenaSimulatorSheetsV1)

```
public ArenaSimulator2(IRandom random, ArenaPlayerDigest myDigest, ArenaPlayerDigest enemyDigest, ArenaSimulatorSheetsV1 simulatorSheets)
```

## Parameters

random [IRandom](#)

myDigest [ArenaPlayerDigest](#)

enemyDigest [ArenaPlayerDigest](#)

simulatorSheets [ArenaSimulatorSheetsV1](#)

## Properties

### Reward

do not use it

```
public override IEnumerable<ItemBase> Reward { get; }
```

## Property Value

[IEnumerable](#)<[ItemBase](#)>

## Methods

### Simulate()

```
public Player Simulate()
```

## Returns

[Player](#)

# Interface IArenaSimulator

Namespace: [Nekoyume.Arena](#)

Assembly: Lib9c.dll

```
public interface IArenaSimulator
```

## Properties

### BuffLinkSheet

```
BuffLinkSheet BuffLinkSheet { get; set; }
```

Property Value

[BuffLinkSheet](#)

### DeBuffLimitSheet

```
DeBuffLimitSheet DeBuffLimitSheet { get; }
```

Property Value

[DeBuffLimitSheet](#)

### Log

```
ArenaLog Log { get; }
```

Property Value

[ArenaLog](#)

## Random

```
IRandom Random { get; }
```

Property Value

IRandom

## ShatterStrikeMaxDamage

```
long ShatterStrikeMaxDamage { get; }
```

Property Value

[long](#) ↗

## Turn

```
int Turn { get; }
```

Property Value

[int](#) ↗

# Namespace Nekoyume.Battle

## Classes

[ArenaScoreHelper](#)

[AttackCountHelper](#)

[CPHelper](#)

[CriticalHelper](#)

[HitHelper](#)

[InvalidCountException](#)

[ListEmptyException](#)

[RaidSimulator](#)

[RaidSimulatorV1](#)

[RaidSimulatorV2](#)

[RankingSimulator](#)

[RankingSimulatorV1](#)

[RewardSelector](#)

[Simulator](#)

[StageRewardExpHelper](#)

[StageSimulator](#)

[StageSimulatorV1](#)

[StageSimulatorV2](#)

[StageSimulatorV3](#)

[TestSimulator](#)

[Wave](#)

[WeightedSelector<T>](#)

## Structs

[EnemyPlayerDigest](#)

## Interfaces

[ISimulator](#)

[IStageSimulator](#)

# Class ArenaScoreHelper

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class ArenaScoreHelper
```

## Inheritance

[object](#) ← ArenaScoreHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### DifferLowerLimit

```
[Obsolete("Only to used for V1")]
public const int DifferLowerLimit = -1000
```

#### Field Value

[int](#)

### DifferUpperLimit

```
[Obsolete("Only to used for V1")]
public const int DifferUpperLimit = 1000
```

#### Field Value

[int](#)

## LoseScoreMax

```
[Obsolete("Only to used for V1")]
public const int LoseScoreMax = -30
```

Field Value

[int](#)

## LoseScoreMin

```
[Obsolete("Only to used for V1")]
public const int LoseScoreMin = -5
```

Field Value

[int](#)

## WinScoreMax

```
[Obsolete("Only to used for V1")]
public const int WinScoreMax = 60
```

Field Value

[int](#)

## WinScoreMin

```
[Obsolete("Only to used for V1")]
public const int WinScoreMin = 1
```

Field Value

[int](#)

# Methods

## GetScore(int, int, Result)

```
public static (int challengerScoreDelta, int defenderScoreDelta) GetScore(int  
challengerRating, int defenderRating, BattleLog.Result result)
```

### Parameters

challengerRating [int](#)

defenderRating [int](#)

result [BattleLog.Result](#)

### Returns

([int](#) [challengerScoreDelta](#), [int](#) [defenderScoreDelta](#))

## GetScoreV1(int, int, Result)

```
[Obsolete("Use GetScore())")]  
public static int GetScoreV1(int challengerRating, int defenderRating,  
BattleLog.Result result)
```

### Parameters

challengerRating [int](#)

defenderRating [int](#)

result [BattleLog.Result](#)

### Returns

[int](#)

## GetScoreV2(int, int, Result)

```
[Obsolete("Use GetScore()")]
public static int GetScoreV2(int challengerRating, int defenderRating,
BattleLog.Result result)
```

### Parameters

challengerRating [int](#)

defenderRating [int](#)

result [BattleLog.Result](#)

### Returns

[int](#)

## GetScoreV3(int, int, Result)

```
[Obsolete("Use GetScore()")]
public static (int challengerScore, int defenderScore) GetScoreV3(int
challengerRating, int defenderRating, BattleLog.Result result)
```

### Parameters

challengerRating [int](#)

defenderRating [int](#)

result [BattleLog.Result](#)

### Returns

([int](#) [challengerScoreDelta](#), [int](#) [defenderScoreDelta](#))

## GetScoreV4(int, int, Result)

```
[Obsolete("Use GetScore())")]
public static (int challengerScoreDelta, int defenderScoreDelta) GetScoreV4(int
challengerRating, int defenderRating, BattleLog.Result result)
```

## Parameters

challengerRating [int](#)

defenderRating [int](#)

result [BattleLog.Result](#)

## Returns

([int](#) challengerScoreDelta, [int](#) defenderScoreDelta)

# Class AttackCountHelper

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class AttackCountHelper
```

## Inheritance

[object](#) ← AttackCountHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### CountMaxLowerLimit

```
public const int CountMaxLowerLimit = 2
```

Field Value

[int](#)

### CountMaxUpperLimit

```
public const int CountMaxUpperLimit = 5
```

Field Value

[int](#)

## Methods

## GetAdditionalCriticalChance(int, int)

```
public static int GetAdditionalCriticalChance(int attackCount, int attackCountMax)
```

Parameters

attackCount [int ↗](#)

attackCountMax [int ↗](#)

Returns

[int ↗](#)

## GetCountMax(int)

```
public static int GetCountMax(int level)
```

Parameters

level [int ↗](#)

Returns

[int ↗](#)

## GetDamageMultiplier(int, int)

```
public static int GetDamageMultiplier(int attackCount, int attackCountMax)
```

Parameters

attackCount [int ↗](#)

attackCountMax [int ↗](#)

Returns

[int ↗](#)

# Class CHelper

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class CHelper
```

## Inheritance

[object](#) ← CHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ConvertCpToStat(StatType, decimal, int)

```
public static decimal ConvertCpToStat(StatType statType, decimal cp,  
int characterLevel)
```

#### Parameters

statType [StatType](#)

cp [decimal](#)

characterLevel [int](#)

#### Returns

[decimal](#)

### DecimalToInt(decimal)

```
public static int DecimalToInt(decimal value)
```

Parameters

value [decimal](#)

Returns

[int](#)

## GetCP(Enemy)

```
public static int GetCP(Enemy enemy)
```

Parameters

enemy [Enemy](#)

Returns

[int](#)

## GetCP(Costume, CostumeStatSheet)

```
public static int GetCP(Costume costume, CostumeStatSheet sheet)
```

Parameters

costume [Costume](#)

sheet [CostumeStatSheet](#)

Returns

[int](#)

## GetCP(INonFungibleItem, CostumeStatSheet)

```
[Obsolete("Use GetCp")]
public static int GetCP(INonFungibleItem tradableItem, CostumeStatSheet sheet)
```

### Parameters

tradableItem [INonFungibleItem](#)

sheet [CostumeStatSheet](#)

### Returns

[int](#)

## GetCP(ItemUsable)

```
public static int GetCP(ItemUsable itemUsable)
```

### Parameters

itemUsable [ItemUsable](#)

### Returns

[int](#)

## GetCP(Player, CostumeStatSheet)

```
public static int GetCP(Player player, CostumeStatSheet costumeStatSheet)
```

### Parameters

player [Player](#)

costumeStatSheet [CostumeStatSheet](#)

Returns

[int](#)

## GetCP(AvatarState, CharacterSheet)

```
[Obsolete("Use TotalCP")]
public static int GetCP(AvatarState avatarState, CharacterSheet characterSheet)
```

Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

Returns

[int](#)

## GetCPOfATK(decimal)

```
public static decimal GetCPOfATK(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetCPOfArmorPenetration(decimal)

```
public static decimal GetCPOfArmorPenetration(decimal value)
```

Parameters

`value decimal`

Returns

`decimal`

## GetCPOfCDMG(decimal, int)

```
public static decimal GetCPOfCDMG(decimal value, int characterLevel)
```

Parameters

`value decimal`

`characterLevel int`

Returns

`decimal`

## GetCPOfCRI(decimal, int)

```
public static decimal GetCPOfCRI(decimal value, int characterLevel)
```

Parameters

`value decimal`

`characterLevel int`

Returns

`decimal`

## GetCPOfDEF(decimal)

```
public static decimal GetCPOfDEF(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetCPOfDRR(decimal, int)

```
public static decimal GetCPOfDRR(decimal value, int characterLevel)
```

Parameters

value [decimal](#)

characterLevel [int](#)

Returns

[decimal](#)

## GetCPOfDRV(decimal)

```
public static decimal GetCPOfDRV(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetCPOfHIT(decimal)

```
public static decimal GetCPOfHIT(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetCPOfHP(decimal)

```
public static decimal GetCPOfHP(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetCPOfSPD(decimal)

```
public static decimal GetCPOfSPD(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetCPOfThorn(decimal)

```
public static decimal GetCPOfThorn(decimal value)
```

Parameters

[value](#) [decimal](#)

Returns

[decimal](#)

## GetCPV2(AvatarState, CharacterSheet, CostumeStatSheet)

```
[Obsolete("Use TotalCP")]
public static int GetCPV2(AvatarState avatarState, CharacterSheet characterSheet,
CostumeStatSheet costumeStatSheet)
```

Parameters

[avatarState](#) [AvatarState](#)

[characterSheet](#) [CharacterSheet](#)

[costumeStatSheet](#) [CostumeStatSheet](#)

Returns

[int](#)

## GetSkillsMultiplier(int)

```
public static decimal GetSkillsMultiplier(int skillsCount)
```

Parameters

`skillsCount` [int](#)

Returns

[decimal](#)

## GetStatCP(StatType, decimal, int)

```
public static decimal GetStatCP(StatType statType, decimal statValue, int  
characterLevel = 1)
```

Parameters

`statType` [StatType](#)

`statValue` [decimal](#)

`characterLevel` [int](#)

Returns

[decimal](#)

## GetStatsCP(IStats, int)

```
public static decimal GetStatsCP(IStats stats, int characterLevel = 1)
```

Parameters

`stats` [IStats](#)

`characterLevel` [int](#)

Returns

[decimal](#)

TotalCP(IReadOnlyCollection<Equipment>,  
IReadOnlyCollection<Costume>,  
IReadOnlyCollection<RuneOptionInfo>, int, Row,  
CostumeStatSheet, List<StatModifier>, int)

```
public static int TotalCP(IReadOnlyCollection<Equipment> equipments,  
IReadOnlyCollection<Costume> costumes,  
IReadOnlyCollection<RuneOptionSheet.Row.RuneOptionInfo> runeOptions, int level,  
CharacterSheet.Row row, CostumeStatSheet costumeStatSheet, List<StatModifier>  
collectionStatModifiers, int runeLevelBonus)
```

## Parameters

equipments [IReadOnlyCollection](#)<[Equipment](#)>

costumes [IReadOnlyCollection](#)<[Costume](#)>

runeOptions [IReadOnlyCollection](#)<[RuneOptionSheet.Row.RuneOptionInfo](#)>

level [int](#)

row [CharacterSheet.Row](#)

costumeStatSheet [CostumeStatSheet](#)

collectionStatModifiers [List](#)<[StatModifier](#)>

runeLevelBonus [int](#)

## Returns

[int](#)

# Class CriticalHelper

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class CriticalHelper
```

## Inheritance

[object](#) ← CriticalHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetCriticalDamage(CharacterBase, long)

```
public static long GetCriticalDamage(CharacterBase caster, long originalDamage)
```

#### Parameters

caster [CharacterBase](#)

originalDamage [long](#)

#### Returns

[long](#)

### GetCriticalDamageForArena(ArenaCharacter, long)

```
public static long GetCriticalDamageForArena(ArenaCharacter caster,  
long originalDamage)
```

## Parameters

caster [ArenaCharacter](#)

originalDamage [long](#) ↴

## Returns

[long](#) ↴

# Struct EnemyPlayerDigest

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public readonly struct EnemyPlayerDigest : IState
```

## Implements

[IState](#)

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### EnemyPlayerDigest(List)

```
public EnemyPlayerDigest(List serialized)
```

#### Parameters

serialized List

### EnemyPlayerDigest(AvatarState)

```
public EnemyPlayerDigest(AvatarState enemyAvatarState)
```

#### Parameters

enemyAvatarState [AvatarState](#)

## Fields

## CharacterId

```
public readonly int CharacterId
```

### Field Value

[int](#)

## Costumes

```
public readonly IReadOnlyList<Costume> Costumes
```

### Field Value

[IReadOnlyList](#)<[Costume](#)>

## EarIndex

```
public readonly int EarIndex
```

### Field Value

[int](#)

## Equipments

```
public readonly IReadOnlyList<Equipment> Equipments
```

### Field Value

[IReadOnlyList](#)<[Equipment](#)>

## HairIndex

```
public readonly int HairIndex
```

Field Value

[int](#)

## LensIndex

```
public readonly int LensIndex
```

Field Value

[int](#)

## Level

```
public readonly int Level
```

Field Value

[int](#)

## NameWithHash

```
public readonly string NameWithHash
```

Field Value

[string](#)

## TailIndex

```
public readonly int TailIndex
```

Field Value

[int](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class HitHelper

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class HitHelper
```

## Inheritance

[object](#) ← HitHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### GetHitStep1CorrectionMax

```
public const long GetHitStep1CorrectionMax = 50
```

Field Value

[long](#)

### GetHitStep1CorrectionMin

```
public const long GetHitStep1CorrectionMin = -5
```

Field Value

[long](#)

### GetHitStep1LevelDiffMax

```
public const long GetHitStep1LevelDiffMax = 10
```

Field Value

[long](#) ↗

## GetHitStep1LevelDiffMin

```
public const long GetHitStep1LevelDiffMin = -14
```

Field Value

[long](#) ↗

## GetHitStep2AdditionalCorrectionMax

```
public const long GetHitStep2AdditionalCorrectionMax = 50
```

Field Value

[long](#) ↗

## GetHitStep2AdditionalCorrectionMin

```
public const long GetHitStep2AdditionalCorrectionMin = 0
```

Field Value

[long](#) ↗

## GetHitStep3CorrectionMax

```
public const long GetHitStep3CorrectionMax = 90
```

Field Value

[long](#) ↗

## GetHitStep3CorrectionMin

```
public const long GetHitStep3CorrectionMin = 10
```

Field Value

[long](#) ↗

## Methods

### GetHitStep1(long, long)

```
public static long GetHitStep1(long attackerLevel, long defenderLevel)
```

Parameters

attackerLevel [long](#) ↗

defenderLevel [long](#) ↗

Returns

[long](#) ↗

### GetHitStep2(long, long)

```
public static long GetHitStep2(long attackerHit, long defenderHit)
```

Parameters

attackerHit [long](#)

defenderHit [long](#)

Returns

[long](#)

## GetHitStep3(long)

```
public static long GetHitStep3(long correction)
```

Parameters

correction [long](#)

Returns

[long](#)

## GetHitStep4(long, long)

```
public static bool GetHitStep4(long lowLimitChance, long correction)
```

Parameters

lowLimitChance [long](#)

correction [long](#)

Returns

[bool](#)

## IsHit(long, long, long, long, long)

```
public static bool IsHit(long attackerLevel, long attackerHit, long defenderLevel,  
long defenderHit, long lowLimitChance)
```

## Parameters

attackerLevel [long](#)

attackerHit [long](#)

defenderLevel [long](#)

defenderHit [long](#)

lowLimitChance [long](#)

## Returns

[bool](#)

## IsHitWithoutLevelCorrection(long, long, long, long, long)

```
public static bool IsHitWithoutLevelCorrection(long attackerLevel, long attackerHit,  
long defenderLevel, long defenderHit, long lowLimitChance)
```

## Parameters

attackerLevel [long](#)

attackerHit [long](#)

defenderLevel [long](#)

defenderHit [long](#)

lowLimitChance [long](#)

## Returns

[bool](#)

# Interface ISimulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public interface ISimulator
```

## Properties

### Characters

```
SimplePriorityQueue<CharacterBase, decimal> Characters { get; }
```

### Property Value

[SimplePriorityQueue<CharacterBase, decimal>](#)

### Log

```
BattleLog Log { get; }
```

### Property Value

[BattleLog](#)

### LogEvent

```
bool LogEvent { get; }
```

### Property Value

[bool](#)

## Player

```
Player Player { get; }
```

Property Value

[Player](#)

## Reward

```
IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#) <[ItemBase](#)>

## WaveNumber

```
int WaveNumber { get; }
```

Property Value

[int](#)

## WaveTurn

```
int WaveTurn { get; }
```

Property Value

[int](#)

# Interface IStageSimulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public interface IStageSimulator : ISimulator
```

## Inherited Members

[ISimulator.Player](#) , [ISimulator.Log](#) , [ISimulator.Characters](#) , [ISimulator.Reward](#) ,  
[ISimulator.WaveNumber](#) , [ISimulator.WaveTurn](#) , [ISimulator.LogEvent](#)

## Properties

### EnemySkillSheet

```
EnemySkillSheet EnemySkillSheet { get; }
```

Property Value

[EnemySkillSheet](#)

### ItemMap

```
CollectionMap ItemMap { get; }
```

Property Value

[CollectionMap](#)

### StageId

```
int StageId { get; }
```

Property Value

[int](#) ↗

# Class InvalidCountException

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class InvalidCountException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
InvalidCountException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidCountException()

```
public InvalidCountException()
```

# Class ListEmptyException

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class ListEmptyException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
ListEmptyException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ListEmptyException(string)

```
public ListEmptyException(string message = "list is empty. add value first")
```

## Parameters

message [string](#)

# Class RaidSimulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class RaidSimulator : Simulator, ISimulator
```

## Inheritance

[object](#) ↳ [Simulator](#) ↳ RaidSimulator

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) ↳ , [object.Equals\(object, object\)](#) ↳ , [object.GetHashCode\(\)](#) ↳ ,  
[object.GetType\(\)](#) ↳ , [object.MemberwiseClone\(\)](#) ↳ , [object.ReferenceEquals\(object, object\)](#) ↳ ,  
[object.ToString\(\)](#) ↳

## Constructors

RaidSimulator(int, IRandom, AvatarState, List<Guid>, AllRuneState, RuneSlotState, RaidSimulatorSheets, CostumeStatSheet, List<StatModifier>, DeBuffLimitSheet, BuffLinkSheet, long)

```
public RaidSimulator(int bossId, IRandom random, AvatarState avatarState, List<Guid> foods, AllRuneState runeStates, RuneSlotState runeSlotState, RaidSimulatorSheets simulatorSheets, CostumeStatSheet costumeStatSheet, List<StatModifier> collectionModifiers, DeBuffLimitSheet deBuffLimitSheet, BuffLinkSheet buffLinkSheet, long shatterStrikeMaxDamage = 400000)
```

## Parameters

bossId [int](#)

random [IRandom](#)

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

runeStates [AllRuneState](#)

runeSlotState [RuneSlotState](#)

simulatorSheets [RaidSimulatorSheets](#)

costumeStatSheet [CostumeStatSheet](#)

collectionModifiers [List](#)<[StatModifier](#)>

deBuffLimitSheet [DeBuffLimitSheet](#)

buffLinkSheet [BuffLinkSheet](#)

shatterStrikeMaxDamage [long](#)

## Properties

### AssetReward

```
public List<FungibleAssetValue> AssetReward { get; }
```

## Property Value

[List](#)<FungibleAssetValue>

## BossId

```
public int BossId { get; }
```

Property Value

[int](#)

## DamageDealt

```
public long DamageDealt { get; }
```

Property Value

[long](#)

## Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## Methods

### Simulate()

```
public BattleLog Simulate()
```

Returns

[BattleLog](#)

## SpawnBoss(RaidBoss)

```
public void SpawnBoss(RaidBoss raidBoss)
```

### Parameters

raidBoss [RaidBoss](#)

# Class RaidSimulatorV1

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class RaidSimulatorV1 : Simulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← RaidSimulatorV1

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

RaidSimulatorV1(int, IRandom, AvatarState, List<Guid>, RaidSimulatorSheetsV1, CostumeStatSheet)

```
public RaidSimulatorV1(int bossId, IRandom random, AvatarState
avatarState, List<Guid> foods, RaidSimulatorSheetsV1 simulatorSheets,
CostumeStatSheet costumeStatSheet)
```

## Parameters

bossId [int](#)

random [IRandom](#)

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

simulatorSheets [RaidSimulatorSheetsV1](#)

costumeStatSheet [CostumeStatSheet](#)

## Properties

### AssetReward

```
public List<FungibleAssetValue> AssetReward { get; }
```

#### Property Value

[List](#)<FungibleAssetValue>

### BossId

```
public int BossId { get; }
```

#### Property Value

[int](#)

## DamageDealt

```
public long DamageDealt { get; }
```

Property Value

[long](#)

## Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## Methods

### Simulate()

```
public BattleLog Simulate()
```

Returns

[BattleLog](#)

### SpawnBoss(RaidBoss)

```
public void SpawnBoss(RaidBoss raidBoss)
```

Parameters

raidBoss [RaidBoss](#)

# Class RaidSimulatorV2

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class RaidSimulatorV2 : Simulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← RaidSimulatorV2

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

RaidSimulatorV2(int, IRandom, AvatarState, List<Guid>, List<RuneState>, RaidSimulatorSheets, CostumeStatSheet)

```
public RaidSimulatorV2(int bossId, IRandom random, AvatarState avatarState,
List<Guid> foods, List<RuneState> runeStates, RaidSimulatorSheets simulatorSheets,
CostumeStatSheet costumeStatSheet)
```

## Parameters

bossId [int](#)

random [IRandom](#)

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

runeStates [List](#)<[RuneState](#)>

simulatorSheets [RaidSimulatorSheets](#)

costumeStatSheet [CostumeStatSheet](#)

## Properties

### AssetReward

```
public List<FungibleAssetValue> AssetReward { get; }
```

### Property Value

[List](#)<FungibleAssetValue>

### BossId

```
public int BossId { get; }
```

### Property Value

[int](#)

## DamageDealt

```
public long DamageDealt { get; }
```

Property Value

[long](#)

## Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## Methods

### Simulate()

```
public BattleLog Simulate()
```

Returns

[BattleLog](#)

### SpawnBoss(RaidBoss)

```
public void SpawnBoss(RaidBoss raidBoss)
```

Parameters

raidBoss [RaidBoss](#)

# Class RankingSimulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class RankingSimulator : Simulator, ISimulator
```

## Inheritance

[object](#) ↳ [Simulator](#) ↳ RankingSimulator

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) ↳ , [object.Equals\(object, object\)](#) ↳ , [object.GetHashCode\(\)](#) ↳ ,  
[object.GetType\(\)](#) ↳ , [object.MemberwiseClone\(\)](#) ↳ , [object.ReferenceEquals\(object, object\)](#) ↳ ,  
[object.ToString\(\)](#) ↳

## Constructors

RankingSimulator(IRandom, Player, EnemyPlayerDigest, List<Guid>, RankingSimulatorSheetsV1, int, CostumeStatSheet)

```
public RankingSimulator(IRandom random, Player player, EnemyPlayerDigest  
enemyPlayerDigest, List<Guid> foods, RankingSimulatorSheetsV1  
rankingSimulatorSheets, int stageId, CostumeStatSheet costumeStatSheet)
```

## Parameters

random IRandom

player [Player](#)

enemyPlayerDigest [EnemyPlayerDigest](#)

foods [List](#)<[Guid](#)>

rankingSimulatorSheets [RankingSimulatorSheetsV1](#)

stageId [int](#)

costumeStatSheet [CostumeStatSheet](#)

## RankingSimulator(IRandom, AvatarState, AvatarState, List<Guid>, RankingSimulatorSheetsV1, int)

```
public RankingSimulator(IRandom random, AvatarState avatarState, AvatarState  
enemyAvatarState, List<Guid> foods, RankingSimulatorSheetsV1 rankingSimulatorSheets,  
int stageId)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

enemyAvatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

rankingSimulatorSheets [RankingSimulatorSheetsV1](#)

stageId [int](#)

**RankingSimulator(IRandom, AvatarState, AvatarState, List<Guid>, RankingSimulatorSheetsV1, int, CostumeStatSheet)**

```
public RankingSimulator(IRandom random, AvatarState avatarState, AvatarState enemyAvatarState, List<Guid> foods, RankingSimulatorSheetsV1 rankingSimulatorSheets, int stageId, CostumeStatSheet costumeStatSheet)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

enemyAvatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

rankingSimulatorSheets [RankingSimulatorSheetsV1](#)

stageId [int](#)

costumeStatSheet [CostumeStatSheet](#)

## Properties

### Reward

Use this property after invoke the [PostSimulate\(rewards\)](#) function. This property will returns [null](#) basically.

```
public override IEnumerable<ItemBase> Reward { get; }
```

## Property Value

[IEnumerable](#)<[ItemBase](#)>

## Methods

## PostSimulate(List<ItemBase>, int, int)

This function should invoke after the `Simulate()` function invoked.

```
public void PostSimulate(List<ItemBase> rewards, int challengerScoreDelta,  
int challengerScore)
```

### Parameters

`rewards` [List](#)<[ItemBase](#)>

`challengerScoreDelta` [int](#)

`challengerScore` [int](#)

## Simulate()

```
public Player Simulate()
```

### Returns

[Player](#)

# Class RankingSimulatorV1

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class RankingSimulatorV1 : Simulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← RankingSimulatorV1

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

RankingSimulatorV1(IRandom, Player,  
EnemyPlayerDigest, List<Guid>,  
RankingSimulatorSheetsV1, int, ArenaInfo, ArenaInfo,  
CostumeStatSheet)

```
public RankingSimulatorV1(IRandom random, Player player, EnemyPlayerDigest enemyPlayerDigest, List<Guid> foods, RankingSimulatorSheetsV1 rankingSimulatorSheets, int stageId, ArenaInfo arenaInfo, ArenaInfo enemyInfo, CostumeStatSheet costumeStatSheet)
```

## Parameters

random IRandom

player [Player](#)

enemyPlayerDigest [EnemyPlayerDigest](#)

foods [List](#)<[Guid](#)>

rankingSimulatorSheets [RankingSimulatorSheetsV1](#)

stageId [int](#)

arenaInfo [ArenaInfo](#)

enemyInfo [ArenaInfo](#)

costumeStatSheet [CostumeStatSheet](#)

RankingSimulatorV1(IRandom, AvatarState, AvatarState, List<Guid>, RankingSimulatorSheetsV1, int, ArenaInfo, ArenaInfo)

```
public RankingSimulatorV1(IRandom random, AvatarState avatarState, AvatarState enemyAvatarState, List<Guid> foods, RankingSimulatorSheetsV1 rankingSimulatorSheets, int stageId, ArenaInfo arenaInfo, ArenaInfo enemyInfo)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

enemyAvatarState [AvatarState](#)

foods [List<Guid>](#)

rankingSimulatorSheets [RankingSimulatorSheetsV1](#)

stageId [int](#)

arenaInfo [ArenaInfo](#)

enemyInfo [ArenaInfo](#)

RankingSimulatorV1(IRandom, AvatarState,  
AvatarState, List<Guid>, RankingSimulatorSheetsV1,  
int, ArenaInfo, ArenaInfo, CostumeStatSheet)

```
public RankingSimulatorV1(IRandom random, AvatarState avatarState,  
AvatarState enemyAvatarState, List<Guid> foods, RankingSimulatorSheetsV1  
rankingSimulatorSheets, int stageId, ArenaInfo arenaInfo, ArenaInfo enemyInfo,  
CostumeStatSheet costumeStatSheet)
```

## Parameters

random [IRandom](#)

avatarState [AvatarState](#)

enemyAvatarState [AvatarState](#)

foods [List<Guid>](#)

rankingSimulatorSheets [RankingSimulatorSheetsV1](#)

stageId [int](#)

arenaInfo [ArenaInfo](#)

enemyInfo [ArenaInfo](#)

costumeStatSheet [CostumeStatSheet](#)

## Fields

# WeeklyArenaRewardSheet

```
public readonly WeeklyArenaRewardSheet WeeklyArenaRewardSheet
```

## Field Value

[WeeklyArenaRewardSheet](#)

## Properties

### Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

## Property Value

[IEnumerable](#)<[ItemBase](#)>

## Methods

### Simulate()

```
public Player Simulate()
```

## Returns

[Player](#)

### SimulateV1()

```
[Obsolete("Use Simulate")]
public Player SimulateV1()
```

## Returns

[Player](#)

## SimulateV2()

```
[Obsolete("Use Simulate")]
public Player SimulateV2()
```

Returns

[Player](#)

## SimulateV3()

```
[Obsolete("Use Simulate")]
public Player SimulateV3()
```

Returns

[Player](#)

## SimulateV4()

```
[Obsolete("use Simulate")]
public Player SimulateV4()
```

Returns

[Player](#)

# Class RewardSelector

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class RewardSelector
```

## Inheritance

[object](#) ← RewardSelector

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

Select(IRandom, WeeklyArenaRewardSheet, MaterialItemSheet, int, int)

```
public static List<ItemBase> Select(IRandom random, WeeklyArenaRewardSheet  
weeklyArenaRewardSheet, MaterialItemSheet materialItemSheet, int playerLevel, int  
maxCount = 1)
```

## Parameters

random IRandom

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

materialItemSheet [MaterialItemSheet](#)

playerLevel [int](#)

maxCount [int](#)

## Returns

[List](#) <ItemBase>

# Class Simulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public abstract class Simulator : ISimulator
```

## Inheritance

[object](#) ← Simulator

## Implements

[ISimulator](#)

## Derived

[ArenaSimulator2](#), [AdventureBossSimulator](#), [RaidSimulator](#), [RaidSimulatorV1](#),  
[RaidSimulatorV2](#), [RankingSimulator](#), [RankingSimulatorV1](#), [StageSimulator](#),  
[StageSimulatorV1](#), [StageSimulatorV2](#), [StageSimulatorV3](#), [TestSimulator](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

Simulator(IRandom, Player, List<Guid>,

SimulatorSheets)

```
protected Simulator(IRandom random, Player player, List<Guid> foods,  
SimulatorSheets simulatorSheets)
```

## Parameters

random IRandom

player [Player](#)

foods [List](#)<[Guid](#)>

simulatorSheets [SimulatorSheets](#)

## Simulator(IRandom, Player, List<Guid>, SimulatorSheetsV1)

```
protected Simulator(IRandom random, Player player, List<Guid> foods,  
SimulatorSheetsV1 simulatorSheets)
```

### Parameters

random IRandom

player [Player](#)

foods [List](#)<[Guid](#)>

simulatorSheets [SimulatorSheetsV1](#)

## Simulator(IRandom, AvatarState, List<Guid>, SimulatorSheets, bool, long)

```
protected Simulator(IRandom random, AvatarState avatarState, List<Guid> foods,  
SimulatorSheets simulatorSheets, bool logEvent = true, long shatterStrikeMaxDamage  
= 400000)
```

### Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

simulatorSheets [SimulatorSheets](#)

logEvent [bool](#)

shatterStrikeMaxDamage [long](#)

# Simulator(IRandom, AvatarState, List<Guid>, SimulatorSheetsV1, bool)

```
protected Simulator(IRandom random, AvatarState avatarState, List<Guid> foods, SimulatorSheetsV1 simulatorSheets, bool logEvent = true)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

simulatorSheets [SimulatorSheetsV1](#)

logEvent [bool](#)

## Fields

### ActionBuffSheet

```
public readonly ActionBuffSheet ActionBuffSheet
```

#### Field Value

[ActionBuffSheet](#)

### CharacterLevelSheet

```
public readonly CharacterLevelSheet CharacterLevelSheet
```

#### Field Value

[CharacterLevelSheet](#)

# CharacterSheet

```
public readonly CharacterSheet CharacterSheet
```

Field Value

[CharacterSheet](#)

# EquipmentItemSetEffectSheet

```
public readonly EquipmentItemSetEffectSheet EquipmentItemSetEffectSheet
```

Field Value

[EquipmentItemSetEffectSheet](#)

# MaterialItemSheet

```
public readonly MaterialItemSheet MaterialItemSheet
```

Field Value

[MaterialItemSheet](#)

# MaxTurn

```
protected const int MaxTurn = 200
```

Field Value

[int](#)

# Random

```
public readonly IRandom Random
```

Field Value

[IRandom](#)

## SkillActionBuffSheet

```
public readonly SkillActionBuffSheet SkillActionBuffSheet
```

Field Value

[SkillActionBuffSheet](#)

## SkillBuffSheet

```
public readonly SkillBuffSheet SkillBuffSheet
```

Field Value

[SkillBuffSheet](#)

## SkillSheet

```
public readonly SkillSheet SkillSheet
```

Field Value

[SkillSheet](#)

## StatBuffSheet

```
public readonly StatBuffSheet StatBuffSheet
```

Field Value

[StatBuffSheet](#)

## TurnNumber

```
public int TurnNumber
```

Field Value

[int](#)

## TurnPriority

```
public const decimal TurnPriority = 100
```

Field Value

[decimal](#)

## Properties

### BuffLinkSheet

```
public BuffLinkSheet BuffLinkSheet { get; set; }
```

Property Value

[BuffLinkSheet](#)

## Characters

```
public SimplePriorityQueue<CharacterBase, decimal> Characters { get; set; }
```

Property Value

[SimplePriorityQueue<CharacterBase, decimal>](#)

## DeBuffLimitSheet

```
public DeBuffLimitSheet DeBuffLimitSheet { get; protected set; }
```

Property Value

[DeBuffLimitSheet](#)

## Log

```
public BattleLog Log { get; }
```

Property Value

[BattleLog](#)

## LogEvent

```
public bool LogEvent { get; protected set; }
```

Property Value

[bool](#)

## Player

```
public Player Player { get; }
```

Property Value

[Player](#)

## Result

```
public BattleLog.Result Result { get; protected set; }
```

Property Value

[BattleLog.Result](#)

## Reward

```
public abstract IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## ShatterStrikeMaxDamage

```
public long ShatterStrikeMaxDamage { get; }
```

Property Value

[long](#)

## WaveNumber

```
public int WaveNumber { get; protected set; }
```

Property Value

[int](#)

## WaveTurn

```
public int WaveTurn { get; set; }
```

Property Value

[int](#)

## Methods

### SetReward(WeightedSelector<RewardData>, int, IRandom, MaterialItemSheet)

```
public static List<ItemBase> SetReward(WeightedSelector<StageSheet.RewardData>
itemSelector, int maxCount, IRandom random, MaterialItemSheet materialItemSheet)
```

Parameters

itemSelector [WeightedSelector<StageSheet.RewardData>](#)

maxCount [int](#)

random IRandom

materialItemSheet [MaterialItemSheet](#)

Returns

[List](#)<ItemBase>

## SetRewardV2(WeightedSelector<RewardData>, int, IRandom, MaterialItemSheet)

```
public static List<ItemBase> SetRewardV2(WeightedSelector<StageSheet.RewardData>
itemSelector, int maxCount, IRandom random, MaterialItemSheet materialItemSheet)
```

### Parameters

itemSelector [WeightedSelector<StageSheet.RewardData>](#)

maxCount [int](#)

random [IRandom](#)

materialItemSheet [MaterialItemSheet](#)

### Returns

[List](#)<[ItemBase](#)>

# Class StageRewardExpHelper

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public static class StageRewardExpHelper
```

## Inheritance

[object](#) ← StageRewardExpHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### CachedExp

key: differ (stage number - character level) value: exp

```
public static readonly IReadOnlyDictionary<int, int> CachedExp
```

### Field Value

[IReadOnlyDictionary](#)<[int](#), [int](#)>

### DifferLowerLimit

```
public const int DifferLowerLimit = -15
```

### Field Value

[int](#)

## DifferUpperLimit

```
public const int DifferUpperLimit = 10
```

Field Value

[int ↗](#)

## RewardExpMax

```
public const int RewardExpMax = 300
```

Field Value

[int ↗](#)

## RewardExpMin

```
public const int RewardExpMin = 0
```

Field Value

[int ↗](#)

## Methods

### GetExp(int, int)

```
public static int GetExp(int characterLevel, int stageNumber)
```

Parameters

characterLevel [int ↗](#)

stageNumber [int ↗](#)

## Returns

[int](#) ↗

# Class StageSimulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class StageSimulator : Simulator, IStageSimulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← StageSimulator

## Implements

[IStageSimulator](#), [ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

StageSimulator(IRandom, AvatarState, List<Guid>,  
AllRuneState, RuneSlotState, List<Skill>, int, int, Row,  
Row, bool, int, SimulatorSheets, EnemySkillSheet,  
CostumeStatSheet, List<ItemBase>,

# List<StatModifier>, DeBuffLimitSheet, BuffLinkSheet, bool, long)

```
public StageSimulator(IRandom random, AvatarState avatarState, List<Guid> foods,
AllRuneState runeStates, RuneSlotState runeSlotState, List<Skill> skillsOnWaveStart,
int worldId, int stageId, StageSheet.Row stageRow, StageWaveSheet.Row stageWaveRow,
bool isCleared, int exp, SimulatorSheets simulatorSheets, EnemySkillSheet
enemySkillSheet, CostumeStatSheet costumeStatSheet, List<ItemBase> waveRewards,
List<StatModifier> collectionModifiers, DeBuffLimitSheet deBuffLimitSheet,
BuffLinkSheet buffLinkSheet, bool logEvent = true, long shatterStrikeMaxDamage
= 400000)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List<Guid>](#)

runeStates [AllRuneState](#)

runeSlotState [RuneSlotState](#)

skillsOnWaveStart [List<Skill>](#)

worldId [int](#)

stageId [int](#)

stageRow [StageSheet.Row](#)

stageWaveRow [StageWaveSheet.Row](#)

isCleared [bool](#)

exp [int](#)

simulatorSheets [SimulatorSheets](#)

enemySkillSheet [EnemySkillSheet](#)

costumeStatSheet [CostumeStatSheet](#)

waveRewards [List<ItemBase>](#)

collectionModifiers [List](#)<[StatModifier](#)>

deBuffLimitSheet [DeBuffLimitSheet](#)

buffLinkSheet [BuffLinkSheet](#)

logEvent [bool](#)

shatterStrikeMaxDamage [long](#)

## Properties

### EnemySkillSheet

`public EnemySkillSheet EnemySkillSheet { get; }`

Property Value

[EnemySkillSheet](#)

### ItemMap

`public CollectionMap ItemMap { get; }`

Property Value

[CollectionMap](#)

### Reward

`public override IEnumerable<ItemBase> Reward { get; }`

Property Value

[IEnumerable](#)<[ItemBase](#)>

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

### GetWaveRewards(IRandom, Row, MaterialItemSheet, int)

```
public static List<ItemBase> GetWaveRewards(IRandom random, StageSheet.Row stageRow,  
MaterialItemSheet materialItemSheet, int playCount = 1)
```

Parameters

random [IRandom](#)

stageRow [StageSheet.Row](#)

materialItemSheet [MaterialItemSheet](#)

playCount [int](#)

Returns

[List](#)<[ItemBase](#)>

## Simulate()

```
public Player Simulate()
```

Returns

[Player](#)



# Class StageSimulatorV1

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class StageSimulatorV1 : Simulator, IStageSimulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← StageSimulatorV1

## Implements

[IStageSimulator](#), [ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

StageSimulatorV1(IRandom, AvatarState, List<Guid>,  
List<Skill>, int, int, StageSimulatorSheetsV1,  
CostumeStatSheet, int, int)

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods, List<Skill> skillsOnWaveStart, int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets, CostumeStatSheet costumeStatSheet, int constructorVersion, int playCount = 1)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

skillsOnWaveStart [List](#)<[Skill](#)>

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

costumeStatSheet [CostumeStatSheet](#)

constructorVersion [int](#)

playCount [int](#)

## StageSimulatorV1(IRandom, AvatarState, List<Guid>, int, int, StageSimulatorSheetsV1)

Do not use anymore since v100025.

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods, int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List<Guid>](#)

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

StageSimulatorV1(IRandom, AvatarState, List<Guid>, int, int, StageSimulatorSheetsV1, Skill)

Do not use anymore since v100025.

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods,
int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets, Skill skill)
```

## Parameters

random [IRandom](#)

avatarState [AvatarState](#)

foods [List<Guid>](#)

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

skill [Skill](#)

StageSimulatorV1(IRandom, AvatarState, List<Guid>, int, int, StageSimulatorSheetsV1, Skill, int, int)

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods,
int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets, Skill skill,
int constructorVersion, int playCount)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

skill [Skill](#)

constructorVersion [int](#)

playCount [int](#)

**StageSimulatorV1(IRandom, AvatarState, List<Guid>, int, int, StageSimulatorSheetsV1, CostumeStatSheet)**

Do not use anymore since v100025.

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods,
int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets,
CostumeStatSheet costumeStatSheet)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

costumeStatSheet [CostumeStatSheet](#)

StageSimulatorV1(IRandom, AvatarState, List<Guid>, int, int, StageSimulatorSheetsV1, CostumeStatSheet, int, int)

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods, int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets, CostumeStatSheet costumeStatSheet, int constructorVersion, int playCount = 1)
```

Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

costumeStatSheet [CostumeStatSheet](#)

constructorVersion [int](#)

playCount [int](#)

StageSimulatorV1(IRandom, AvatarState, List<Guid>, int, int, StageSimulatorSheetsV1, int, int)

```
public StageSimulatorV1(IRandom random, AvatarState avatarState, List<Guid> foods, int worldId, int stageId, StageSimulatorSheetsV1 stageSimulatorSheets, int constructorVersion, int playCount)
```

Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

worldId [int](#)

stageId [int](#)

stageSimulatorSheets [StageSimulatorSheetsV1](#)

constructorVersion [int](#)

playCount [int](#)

## Fields

### ConstructorVersionDefault

```
public const int ConstructorVersionDefault = 1
```

Field Value

[int](#)

### ConstructorVersionV100025

```
public const int ConstructorVersionV100025 = 2
```

Field Value

[int](#)

### ConstructorVersionV100080

```
public const int ConstructorVersionV100080 = 3
```

Field Value

[int](#)

## Properties

### EnemySkillSheet

```
public EnemySkillSheet EnemySkillSheet { get; }
```

Property Value

[EnemySkillSheet](#)

### ItemMap

```
public CollectionMap ItemMap { get; }
```

Property Value

[CollectionMap](#)

### Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

### StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

### SetItemSelector(StageSheet.Row, IRandom)

```
public static WeightedSelector<StageSheet.RewardData> SetItemSelector(StageSheet.Row  
stageRow, IRandom random)
```

Parameters

stageRow [StageSheet.Row](#)

random IRandom

Returns

[WeightedSelector<StageSheet.RewardData>](#)

### Simulate(int)

```
public Player Simulate(int playCount)
```

Parameters

playCount [int](#)

Returns

[Player](#)

### SimulateV1()

```
[Obsolete("Use Simulate")]
```

```
public Player SimulateV1()
```

Returns

[Player](#)

## SimulateV2()

```
[Obsolete("Use Simulate")]
public Player SimulateV2()
```

Returns

[Player](#)

## SimulateV3()

```
[Obsolete("Use Simulate")]
public Player SimulateV3()
```

Returns

[Player](#)

## SimulateV4()

```
[Obsolete("Use Simulate")]
public Player SimulateV4()
```

Returns

[Player](#)

## SimulateV5(int)

```
[Obsolete("Use Simulate")]
public Player SimulateV5(int playCount)
```

## Parameters

playCount [int](#)

## Returns

[Player](#)

# Class StageSimulatorV2

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class StageSimulatorV2 : Simulator, IStageSimulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← StageSimulatorV2

## Implements

[IStageSimulator](#), [ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

StageSimulatorV2(IRandom, AvatarState, List<Guid>, List<Skill>, int, int, Row, Row, bool, int, SimulatorSheetsV1, EnemySkillSheet, CostumeStatSheet, List<ItemBase>)

```
public StageSimulatorV2(IRandom random, AvatarState avatarState, List<Guid> foods, List<Skill> skillsOnWaveStart, int worldId, int stageId, StageSheet.Row stageRow, StageWaveSheet.Row stageWaveRow, bool isCleared, int exp, SimulatorSheetsV1 simulatorSheets, EnemySkillSheet enemySkillSheet, CostumeStatSheet costumeStatSheet, List<ItemBase> waveRewards)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

skillsOnWaveStart [List](#)<[Skill](#)>

worldId [int](#)

stageId [int](#)

stageRow [StageSheet.Row](#)

stageWaveRow [StageWaveSheet.Row](#)

isCleared [bool](#)

exp [int](#)

simulatorSheets [SimulatorSheetsV1](#)

enemySkillSheet [EnemySkillSheet](#)

costumeStatSheet [CostumeStatSheet](#)

waveRewards [List](#)<[ItemBase](#)>

## Properties

### EnemySkillSheet

```
public EnemySkillSheet EnemySkillSheet { get; }
```

Property Value

[EnemySkillSheet](#)

## ItemMap

```
public CollectionMap ItemMap { get; }
```

Property Value

[CollectionMap](#)

## Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

GetWaveRewards(IRandom, Row, MaterialItemSheet, int)

```
public static List<ItemBase> GetWaveRewards(IRandom random, StageSheet.Row stageRow,  
MaterialItemSheet materialItemSheet, int playCount = 1)
```

## Parameters

random IRandom

stageRow [StageSheet.Row](#)

materialItemSheet [MaterialItemSheet](#)

playCount [int](#)

## Returns

[List](#)<[ItemBase](#)>

## Simulate()

```
public Player Simulate()
```

## Returns

[Player](#)

# Class StageSimulatorV3

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class StageSimulatorV3 : Simulator, IStageSimulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← StageSimulatorV3

## Implements

[IStageSimulator](#), [ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

StageSimulatorV3(IRandom, AvatarState, List<Guid>,  
List<RuneState>, List<Skill>, int, int, Row, Row, bool,  
int, SimulatorSheets, EnemySkillSheet,  
CostumeStatSheet, List<ItemBase>)

```
public StageSimulatorV3(IRandom random, AvatarState avatarState, List<Guid> foods,
List<RuneState> runeStates, List<Skill> skillsOnWaveStart, int worldId, int stageId,
StageSheet.Row stageRow, StageWaveSheet.Row stageWaveRow, bool isCleared, int exp,
SimulatorSheets simulatorSheets, EnemySkillSheet enemySkillSheet, CostumeStatSheet
costumeStatSheet, List<ItemBase> waveRewards)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

runeStates [List](#)<[RuneState](#)>

skillsOnWaveStart [List](#)<[Skill](#)>

worldId [int](#)

stageId [int](#)

stageRow [StageSheet.Row](#)

stageWaveRow [StageWaveSheet.Row](#)

isCleared [bool](#)

exp [int](#)

simulatorSheets [SimulatorSheets](#)

enemySkillSheet [EnemySkillSheet](#)

costumeStatSheet [CostumeStatSheet](#)

waveRewards [List](#)<[ItemBase](#)>

## Properties

EnemySkillSheet

```
public EnemySkillSheet EnemySkillSheet { get; }
```

Property Value

[EnemySkillSheet](#)

## ItemMap

```
public CollectionMap ItemMap { get; }
```

Property Value

[CollectionMap](#)

## Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

## GetWaveRewards(IRandom, Row, MaterialItemSheet, int)

```
public static List<ItemBase> GetWaveRewards(IRandom random, StageSheet.Row stageRow, MaterialItemSheet materialItemSheet, int playCount = 1)
```

### Parameters

random IRandom

stageRow [StageSheet.Row](#)

materialItemSheet [MaterialItemSheet](#)

playCount [int](#)

### Returns

[List](#)<[ItemBase](#)>

## Simulate()

```
public Player Simulate()
```

### Returns

[Player](#)

# Class TestSimulator

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class TestSimulator : Simulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← TestSimulator

## Implements

[ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom,](#)  
[MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

TestSimulator(IRandom, AvatarState, List<Guid>, SimulatorSheets)

```
public TestSimulator(IRandom random, AvatarState avatarState, List<Guid> foods,  
SimulatorSheets simulatorSheets)
```

## Parameters

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

simulatorSheets [SimulatorSheets](#)

## Properties

### Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

### Property Value

[IEnumerable](#)<[ItemBase](#)>

# Class Wave

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class Wave
```

## Inheritance

[object](#) ← Wave

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### HasBoss

```
public bool HasBoss
```

Field Value

[bool](#)

## Methods

### Add(Enemy)

```
public void Add(Enemy enemy)
```

Parameters

enemy [Enemy](#)

## Spawn(ISimulator)

```
public void Spawn(ISimulator simulator)
```

Parameters

simulator [ISimulator](#)

## SpawnV1(ISimulator)

```
[Obsolete("Use Spawn")]
public void SpawnV1(ISimulator simulator)
```

Parameters

simulator [ISimulator](#)

## SpawnV2(ISimulator)

```
[Obsolete("Use Spawn")]
public void SpawnV2(ISimulator simulator)
```

Parameters

simulator [ISimulator](#)

# Class WeightedSelector<T>

Namespace: [Nekoyume.Battle](#)

Assembly: Lib9c.dll

```
public class WeightedSelector<T>
```

## Type Parameters

T

### Inheritance

[object](#) ← WeightedSelector<T>

### Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### WeightedSelector(IRandom)

```
public WeightedSelector(IRandom random)
```

## Parameters

random IRandom

## Properties

### Count

```
public int Count { get; }
```

Property Value

[int](#)

## Methods

### Add(T, decimal)

```
public void Add(T item, decimal weight)
```

Parameters

item T

weight [decimal](#)

### Clear()

```
public void Clear()
```

### Select(int)

```
public IEnumerable<T> Select(int count)
```

Parameters

count [int](#)

Returns

[IEnumerable](#)<T>

### SelectV1(int)

```
[Obsolete("Use Select")]
public IEnumerable<T> SelectV1(int count)
```

## Parameters

count [int](#)

## Returns

[IEnumerable](#)<T>

## SelectV2(int)

```
[Obsolete("Use Select")]
public IEnumerable<T> SelectV2(int count)
```

## Parameters

count [int](#)

## Returns

[IEnumerable](#)<T>

# Namespace Nekoyume.Battle.Adventure Boss Classes

[AdventureBossSimulator](#)

# Class AdventureBossSimulator

Namespace: [Nekoyume.Battle.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class AdventureBossSimulator : Simulator, IStageSimulator, ISimulator
```

## Inheritance

[object](#) ← [Simulator](#) ← AdventureBossSimulator

## Implements

[IStageSimulator](#), [ISimulator](#)

## Inherited Members

[Simulator.Random](#) , [Simulator.Log](#) , [Simulator.Player](#) , [Simulator.Result](#) ,  
[Simulator.Characters](#) , [Simulator.TurnPriority](#) , [Simulator.MaterialItemSheet](#) ,  
[Simulator.SkillSheet](#) , [Simulator.SkillBuffSheet](#) , [Simulator.StatBuffSheet](#) ,  
[Simulator.SkillActionBuffSheet](#) , [Simulator.ActionBuffSheet](#) , [Simulator.CharacterSheet](#) ,  
[Simulator.CharacterLevelSheet](#) , [Simulator.EquipmentItemSetEffectSheet](#) ,  
[Simulator.DeBuffLimitSheet](#) , [Simulator.BuffLinkSheet](#) ,  
[Simulator.ShatterStrikeMaxDamage](#) , [Simulator.MaxTurn](#) , [Simulator.TurnNumber](#) ,  
[Simulator.WaveNumber](#) , [Simulator.WaveTurn](#) , [Simulator.LogError](#) ,  
[Simulator.SetReward\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[Simulator.SetRewardV2\(WeightedSelector<StageSheet.RewardData>, int, IRandom, MaterialItemSheet\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

AdventureBossSimulator(int, int, IRandom, AvatarState, SimulatorSheets, bool, long)

```
public AdventureBossSimulator(int bossId, int floorId, IRandom random, AvatarState  
avatarState, SimulatorSheets simulatorSheets, bool logEvent = true, long
```

```
shatterStrikeMaxDamage = 400000)
```

## Parameters

bossId [int](#)

floorId [int](#)

random IRandom

avatarState [AvatarState](#)

simulatorSheets [SimulatorSheets](#)

logEvent [bool](#)

shatterStrikeMaxDamage [long](#)

AdventureBossSimulator(int, int, IRandom, AvatarState, List<Guid>, AllRuneState, RuneSlotState, Row, Row, SimulatorSheets, EnemySkillSheet, CostumeStatSheet, List<ItemBase>, List<StatModifier>, DeBuffLimitSheet, BuffLinkSheet, bool, long)

```
public AdventureBossSimulator(int bossId, int floorId, IRandom random, AvatarState
avatarState, List<Guid> foods, AllRuneState runeStates, RuneSlotState runeSlotState,
AdventureBossFloorSheet.Row floorRow, AdventureBossFloorWaveSheet.Row floorWaveRow,
SimulatorSheets simulatorSheets, EnemySkillSheet enemySkillSheet, CostumeStatSheet
costumeStatSheet, List<ItemBase> waveRewards, List<StatModifier>
collectionModifiers, DeBuffLimitSheet deBuffLimitSheet, BuffLinkSheet buffLinkSheet,
bool logEvent = true, long shatterStrikeMaxDamage = 400000)
```

## Parameters

bossId [int](#)

floorId [int](#)

random IRandom

avatarState [AvatarState](#)

foods [List](#)<[Guid](#)>

runeStates [AllRuneState](#)

runeSlotState [RuneSlotState](#)

floorRow [AdventureBossFloorSheet.Row](#)

floorWaveRow [AdventureBossFloorWaveSheet.Row](#)

simulatorSheets [SimulatorSheets](#)

enemySkillSheet [EnemySkillSheet](#)

costumeStatSheet [CostumeStatSheet](#)

waveRewards [List](#)<[ItemBase](#)>

collectionModifiers [List](#)<[StatModifier](#)>

deBuffLimitSheet [DeBuffLimitSheet](#)

buffLinkSheet [BuffLinkSheet](#)

logEvent [bool](#)

shatterStrikeMaxDamage [long](#)

## Properties

### BossId

```
public int BossId { get; }
```

Property Value

[int](#)

## EnemySkillSheet

```
public EnemySkillSheet EnemySkillSheet { get; }
```

Property Value

[EnemySkillSheet](#)

## FloorId

```
public int FloorId { get; }
```

Property Value

[int](#)

## ItemMap

```
public CollectionMap ItemMap { get; }
```

Property Value

[CollectionMap](#)

## Reward

```
public override IEnumerable<ItemBase> Reward { get; }
```

Property Value

[IEnumerable](#)<[ItemBase](#)>

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

AddBreakthrough(List<int>, AdventureBossFloorWaveSheet)

```
public void AddBreakthrough(List<int> floorIdList, AdventureBossFloorWaveSheet  
adventureBossFloorWaveSheet)
```

Parameters

`floorIdList` [List](#)<[int](#)>

`adventureBossFloorWaveSheet` [AdventureBossFloorWaveSheet](#)

GetWaveRewards(IRandom, Row, MaterialItemSheet, int)

```
public static List<ItemBase> GetWaveRewards(IRandom random,  
AdventureBossFloorSheet.Row floorRow, MaterialItemSheet materialItemSheet, int  
playCount = 1)
```

Parameters

`random` IRandom

`floorRow` [AdventureBossFloorSheet.Row](#)

`materialItemSheet` [MaterialItemSheet](#)

`playCount` [int](#)

Returns

[List](#)<ItemBase>

## SetItemSelector(Row, IRandom)

```
public static WeightedSelector<AdventureBossFloorSheet.RewardData>
SetItemSelector(AdventureBossFloorSheet.Row floorRow, IRandom random)
```

Parameters

floorRow [AdventureBossFloorSheet.Row](#)

random IRandom

Returns

[WeightedSelector](#)<AdventureBossFloorSheet.RewardData>

## SetReward(WeightedSelector<RewardData>, int, IRandom, MaterialItemSheet)

```
public static List<ItemBase>
SetReward(WeightedSelector<AdventureBossFloorSheet.RewardData> itemSelector, int
maxCount, IRandom random, MaterialItemSheet materialItemSheet)
```

Parameters

itemSelector [WeightedSelector](#)<AdventureBossFloorSheet.RewardData>

maxCount [int](#)

random IRandom

materialItemSheet [MaterialItemSheet](#)

Returns

[List](#)<ItemBase>

## Simulate()

```
public Player Simulate()
```

Returns

[Player](#)

# Namespace Nekoyume.Data

## Classes

[AdventureBossGameData](#)

## Structs

[AdventureBossGameData.ClivableReward](#)

# Class AdventureBossGameData

Namespace: [Nekoyume.Data](#)

Assembly: Lib9c.dll

```
public static class AdventureBossGameData
```

## Inheritance

[object](#) ← AdventureBossGameData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### AdventureBossOperationalAddress

```
public static readonly Address AdventureBossOperationalAddress
```

#### Field Value

Address

# Struct AdventureBossGameData.ClaimableReward

Namespace: [Nekoyume.Data](#)

Assembly: Lib9c.dll

```
public struct AdventureBossGameData.ClaimableReward
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Fields

### FavReward

```
public Dictionary<int, int> FavReward
```

#### Field Value

[Dictionary<int, int>](#)

### ItemReward

```
public Dictionary<int, int> ItemReward
```

#### Field Value

[Dictionary<int, int>](#)

# NcgReward

```
public FungibleAssetValue? NcgReward
```

## Field Value

FungibleAssetValue?

## Methods

### IsEmpty()

```
public bool IsEmpty()
```

## Returns

[bool](#)

# Namespace Nekoyume.Delegation

## Classes

[Bond](#)

[DelegateeMetadata](#)

[Delegatee<T, TSelf>](#)

[DelegationAddress](#)

[DelegationRepository](#)

[DelegatorMetadata](#)

[Delegator<T, TSelf>](#)

[LumpSumRewardsRecord](#)

[RebondGrace](#)

[UnbondLockIn](#)

[UnbondingEntry](#)

[UnbondingEntry.Comparer](#)

[UnbondingFactory](#)

[UnbondingRef](#)

[UnbondingSet](#)

## Interfaces

[IDelegatee](#)

[IDelegateeMetadata](#)

[IDelegationRepository](#)

[IDelegator](#)

[IDelegatorMetadata](#)

[IUnbonding](#)

## Enums

[UnbondingType](#)

# Class Bond

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public sealed class Bond : IBencodable, IEquatable<Bond>
```

## Inheritance

[object](#) ← Bond

## Implements

IBencodable, [IEquatable](#)<[Bond](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Bond(Address)

```
public Bond(Address address)
```

#### Parameters

address Address

### Bond(Address, IValue)

```
public Bond(Address address, IValue bencoded)
```

#### Parameters

address Address

bencoded IValue

## Bond(Address, List)

```
public Bond(Address address, List bencoded)
```

### Parameters

address Address

bencoded List

## Properties

### Address

```
public Address Address { get; }
```

### Property Value

Address

### Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

### IsEmpty

```
public bool IsEmpty { get; }
```

Property Value

[bool](#) ↴

## LastDistributeHeight

`public long? LastDistributeHeight { get; }`

Property Value

[long](#) ↴?

## Share

`public BigInteger Share { get; }`

Property Value

[BigInteger](#) ↴

## Methods

### Equals(Bond?)

Indicates whether the current object is equal to another object of the same type.

`public bool Equals(Bond? other)`

Parameters

`other` [Bond](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

### Parameters

[obj](#) [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

### Returns

[int](#)

A hash code for the current object.

# Class DelegateeMetadata

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public class DelegateeMetadata : IDelegateeMetadata, IBencodable
```

## Inheritance

[object](#) ← DelegateeMetadata

## Implements

[IDelegateeMetadata](#), IBencodable

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### DelegateeMetadata(Address, Address, IValue)

```
public DelegateeMetadata(Address delegateeAddress, Address delegateeAccountAddress,  
IValue bencoded)
```

#### Parameters

delegateeAddress Address

delegateeAccountAddress Address

bencoded IValue

### DelegateeMetadata(Address, Address, List)

```
public DelegateeMetadata(Address address, Address accountAddress, List bencoded)
```

## Parameters

address Address

accountAddress Address

bencoded List

## DelegateeMetadata(Address, Address, Currency, IEnumerable<Currency>, Address, Address, Address, Address, long, int, int)

```
public DelegateeMetadata(Address delegateeAddress, Address delegateeAccountAddress,
Currency delegationCurrency, IEnumerable<Currency> rewardCurrencies, Address
delegationPoolAddress, Address rewardPoolAddress, Address
rewardRemainderPoolAddress, Address slashedPoolAddress, long unbondingPeriod, int
maxUnbondLockInEntries, int maxRebondGraceEntries)
```

## Parameters

delegateeAddress Address

delegateeAccountAddress Address

delegationCurrency Currency

rewardCurrencies [IEnumerable](#)<Currency>

delegationPoolAddress Address

rewardPoolAddress Address

rewardRemainderPoolAddress Address

slashedPoolAddress Address

unbondingPeriod [long](#)

maxUnbondLockInEntries [int](#)

maxRebondGraceEntries [int](#)

# Properties

## Address

```
public Address Address { get; }
```

Property Value

Address

## Bencoded

```
public List Bencoded { get; }
```

Property Value

List

## DelegateeAccountAddress

```
public Address DelegateeAccountAddress { get; }
```

Property Value

Address

## DelegateeAddress

```
public Address DelegateeAddress { get; }
```

Property Value

Address

## DelegationCurrency

```
public Currency DelegationCurrency { get; }
```

Property Value

Currency

## DelegationPoolAddress

```
public Address DelegationPoolAddress { get; }
```

Property Value

Address

## Delegators

```
public ImmutableSortedSet<Address> Delegators { get; }
```

Property Value

[ImmutableSortedSet](#)<Address>

## Jailed

```
public bool Jailed { get; }
```

Property Value

[bool](#)

## JailedUntil

```
public long JailedUntil { get; }
```

Property Value

[long](#)

## MaxRebondGraceEntries

```
public int MaxRebondGraceEntries { get; }
```

Property Value

[int](#)

## MaxUnbondLockInEntries

```
public int MaxUnbondLockInEntries { get; }
```

Property Value

[int](#)

## RewardCurrencies

```
public ImmutableSortedSet<Currency> RewardCurrencies { get; }
```

Property Value

[ImmutableSortedSet](#)<Currency>

## RewardPoolAddress

```
public Address RewardPoolAddress { get; }
```

Property Value

Address

## RewardRemainderPoolAddress

```
public Address RewardRemainderPoolAddress { get; }
```

Property Value

Address

## SlashedPoolAddress

```
public Address SlashedPoolAddress { get; }
```

Property Value

Address

## Tombstoned

```
public bool Tombstoned { get; }
```

Property Value

[bool](#) ↗

## TotalDelegatedFAV

```
public FungibleAssetValue TotalDelegatedFAV { get; }
```

Property Value

FungibleAssetValue

## TotalShares

```
public BigInteger TotalShares { get; }
```

Property Value

[BigInteger](#)

## UnbondingPeriod

```
public long UnbondingPeriod { get; }
```

Property Value

[long](#)

## UnbondingRefs

```
public ImmutableSortedSet<UnbondingRef> UnbondingRefs { get; }
```

Property Value

[ImmutableSortedSet](#)<[UnbondingRef](#)>

## Methods

AddDelegatedFAV(FungibleAssetValue)

```
public void AddDelegatedFAV(FungibleAssetValue fav)
```

Parameters

**fav** FungibleAssetValue

## AddDelegator(Address)

```
public void AddDelegator(Address delegatorAddress)
```

Parameters

**delegatorAddress** Address

## AddShare(BigInteger)

```
public void AddShare(BigInteger share)
```

Parameters

**share** [BigInteger](#)

## AddUnbondingRef(UnbondingRef)

```
public void AddUnbondingRef(UnbondingRef unbondingRef)
```

Parameters

**unbondingRef** [UnbondingRef](#)

## BondAddress(Address)

```
public Address BondAddress(Address delegatorAddress)
```

Parameters

**delegatorAddress** Address

Returns

Address

## CurrentLumpSumRewardsRecordAddress()

```
public virtual Address CurrentLumpSumRewardsRecordAddress()
```

Returns

Address

## Equals(IDelegateeMetadata?)

```
public virtual bool Equals(IDelegateeMetadata? other)
```

Parameters

**other** [IDelegateeMetadata](#)

Returns

[bool](#)

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

## Parameters

**obj** [object](#)

The object to compare with the current object.

## Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## FAVFromShare(BigInteger)

```
public FungibleAssetValue FAVFromShare(BigInteger share)
```

## Parameters

**share** [BigInteger](#)

## Returns

FungibleAssetValue

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

## Returns

[int](#)

A hash code for the current object.

## LumpSumRewardsRecordAddress(long)

```
public virtual Address LumpSumRewardsRecordAddress(long height)
```

Parameters

[height](#) [long](#)

Returns

Address

## RebondGraceAddress(Address)

```
public virtual Address RebondGraceAddress(Address delegatorAddress)
```

Parameters

[delegatorAddress](#) Address

Returns

Address

## RemoveDelegatedFAV(FungibleAssetValue)

```
public void RemoveDelegatedFAV(FungibleAssetValue fav)
```

Parameters

[fav](#) FungibleAssetValue

## RemoveDelegator(Address)

```
public void RemoveDelegator(Address delegatorAddress)
```

Parameters

**delegatorAddress** Address

## RemoveShare(BigInteger)

```
public void RemoveShare(BigInteger share)
```

Parameters

**share** [BigInteger](#)

## RemoveUnbondingRef(UnbondingRef)

```
public void RemoveUnbondingRef(UnbondingRef unbondingRef)
```

Parameters

**unbondingRef** [UnbondingRef](#)

## ShareFromFAV(FungibleAssetValue)

```
public BigInteger ShareFromFAV(FungibleAssetValue fav)
```

Parameters

**fav** FungibleAssetValue

Returns

[BigInteger](#)

## UnbondLockInAddress(Address)

```
public Address UnbondLockInAddress(Address delegatorAddress)
```

Parameters

delegatorAddress Address

Returns

Address

## UpdateUnbondingPeriod(long)

```
public void UpdateUnbondingPeriod(long unbondingPeriod)
```

Parameters

unbondingPeriod [long](#)

# Class Delegatee<T, TSelf>

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public abstract class Delegatee<T, TSelf> : IDelegatee where T : Delegator<TSelf, T>
where TSelf : Delegatee<T, TSelf>
```

## Type Parameters

T

TSelf

### Inheritance

[object](#) ← Delegatee<T, TSelf>

### Implements

[IDelegatee](#)

### Derived

[GuildDelegatee](#), [ValidatorDelegatee](#)

### Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

Delegatee(Address, Address, Currency,  
IEnumerable<Currency>, Address, Address, Address,  
Address, long, int, int, IDelegationRepository)

```
public Delegatee(Address address, Address accountAddress, Currency
delegationCurrency, IEnumerable<Currency> rewardCurrencies, Address
delegationPoolAddress, Address rewardPoolAddress, Address
```

```
rewardRemainderPoolAddress, Address slashedPoolAddress, long unbondingPeriod, int  
maxUnbondLockInEntries, int maxRebondGraceEntries, IDelegationRepository repository)
```

## Parameters

address Address

accountAddress Address

delegationCurrency Currency

rewardCurrencies [IEnumerable](#)<Currency>

delegationPoolAddress Address

rewardPoolAddress Address

rewardRemainderPoolAddress Address

slashedPoolAddress Address

unbondingPeriod [long](#)

maxUnbondLockInEntries [int](#)

maxRebondGraceEntries [int](#)

repository [IDelegationRepository](#)

## Delegatee(Address, IDelegationRepository)

```
public Delegatee(Address address, IDelegationRepository repository)
```

## Parameters

address Address

repository [IDelegationRepository](#)

## Properties

## AccountAddress

```
public Address AccountAddress { get; }
```

Property Value

Address

## Address

```
public Address Address { get; }
```

Property Value

Address

## DelegationCurrency

```
public Currency DelegationCurrency { get; }
```

Property Value

Currency

## DelegationPoolAddress

```
public Address DelegationPoolAddress { get; }
```

Property Value

Address

## Delegators

```
public ImmutableSortedSet<Address> Delegators { get; }
```

Property Value

[ImmutableSortedSet](#) <Address>

## Jailed

```
public bool Jailed { get; }
```

Property Value

[bool](#)

## JailedUntil

```
public long JailedUntil { get; }
```

Property Value

[long](#)

## MaxRebondGraceEntries

```
public int MaxRebondGraceEntries { get; }
```

Property Value

[int](#)

## MaxUnbondLockInEntries

```
public int MaxUnbondLockInEntries { get; }
```

Property Value

[int](#)

## Metadata

```
public DelegateeMetadata Metadata { get; }
```

Property Value

[DelegateeMetadata](#)

## MetadataAddress

```
public Address MetadataAddress { get; }
```

Property Value

Address

## MetadataBencoded

```
public List MetadataBencoded { get; }
```

Property Value

List

## Repository

```
public IDelegationRepository Repository { get; }
```

Property Value

[IDelegationRepository](#)

## RewardCurrencies

```
public ImmutableSortedSet<Currency> RewardCurrencies { get; }
```

Property Value

[ImmutableSortedSet](#)<Currency>

## RewardPoolAddress

```
public Address RewardPoolAddress { get; }
```

Property Value

Address

## RewardRemainderPoolAddress

```
public Address RewardRemainderPoolAddress { get; }
```

Property Value

Address

## SlashedPoolAddress

```
public Address SlashedPoolAddress { get; }
```

Property Value

Address

## Tombstoned

```
public bool Tombstoned { get; }
```

Property Value

[bool](#)

## TotalDelegated

```
public FungibleAssetValue TotalDelegated { get; }
```

Property Value

FungibleAssetValue

## TotalShares

```
public BigInteger TotalShares { get; }
```

Property Value

[BigInteger](#)

## UnbondingPeriod

```
public long UnbondingPeriod { get; }
```

Property Value

[long](#) ↗

## Methods

### AddUnbondingRef(UnbondingRef)

```
public void AddUnbondingRef(UnbondingRef reference)
```

Parameters

reference [UnbondingRef](#)

### Bond(IDelegator, FungibleAssetValue, long)

```
public BigInteger Bond(IDelegator delegator, FungibleAssetValue fav, long height)
```

Parameters

delegator [IDelegator](#)

fav FungibleAssetValue

height [long](#) ↗

Returns

[BigInteger](#) ↗

### Bond(T, FungibleAssetValue, long)

```
public virtual BigInteger Bond(T delegator, FungibleAssetValue fav, long height)
```

Parameters

**delegator** T

**fav** FungibleAssetValue

**height** [long](#)

Returns

[BigInteger](#)

## BondAddress(Address)

**public** Address **BondAddress**(Address delegatorAddress)

Parameters

**delegatorAddress** Address

Returns

Address

## CalculateReward(BigInteger, IEnumerable<LumpSumRewardsRecord>)

**public** ImmutableDictionary<Currency, FungibleAssetValue> **CalculateReward**(BigInteger share, IEnumerable<LumpSumRewardsRecord> lumpSumRewardsRecords)

Parameters

**share** [BigInteger](#)

**lumpSumRewardsRecords** [IEnumerable](#)<LumpSumRewardsRecord>

Returns

[ImmutableDictionary](#)<Currency, FungibleAssetValue>

## CollectRewards(long)

```
public void CollectRewards(long height)
```

### Parameters

height [long](#)

## CurrentLumpSumRewardsRecordAddress()

```
public Address CurrentLumpSumRewardsRecordAddress()
```

### Returns

Address

## DistributeReward(IDelegator, long)

```
public void DistributeReward(IDelegator delegator, long height)
```

### Parameters

delegator [IDelegator](#)

height [long](#)

## DistributeReward(T, long)

```
public void DistributeReward(T delegator, long height)
```

### Parameters

delegator T

height [long](#)

## FAVFromShare(BigInteger)

```
public FungibleAssetValue FAVFromShare(BigInteger share)
```

Parameters

share [BigInteger](#)

Returns

FungibleAssetValue

## Jail(long)

```
public void Jail(long releaseHeight)
```

Parameters

releaseHeight [long](#)

## LumpSumRewardsRecordAddress(long)

```
public Address LumpSumRewardsRecordAddress(long height)
```

Parameters

height [long](#)

Returns

Address

## RebondGraceAddress(Address)

```
public Address RebondGraceAddress(Address delegatorAddress)
```

### Parameters

delegatorAddress Address

### Returns

Address

## RemoveUnbondingRef(UnbondingRef)

```
public void RemoveUnbondingRef(UnbondingRef reference)
```

### Parameters

reference [UnbondingRef](#)

## ShareFromFAV(FungibleAssetValue)

```
public BigInteger ShareFromFAV(FungibleAssetValue fav)
```

### Parameters

fav FungibleAssetValue

### Returns

[BigInteger](#)

## Slash(BigInteger, long, long)

```
public void Slash(BigInteger slashFactor, long infractionHeight, long height)
```

Parameters

slashFactor [BigInteger](#)

infractionHeight [long](#)

height [long](#)

## Tombstone()

```
public void Tombstone()
```

## Unbond(IDelegator, BigInteger, long)

```
public FungibleAssetValue Unbond(IDelegator delegator, BigInteger share,  
long height)
```

Parameters

delegator [IDelegator](#)

share [BigInteger](#)

height [long](#)

Returns

FungibleAssetValue

## Unbond(T, BigInteger, long)

```
public FungibleAssetValue Unbond(T delegator, BigInteger share, long height)
```

Parameters

delegator T

share [BigInteger](#)

height [long](#)

Returns

FungibleAssetValue

## UnbondLockInAddress(Address)

```
public Address UnbondLockInAddress(Address delegatorAddress)
```

Parameters

delegatorAddress Address

Returns

Address

## Unjail(long)

```
public void Unjail(long height)
```

Parameters

height [long](#)

## Events

### DelegationChanged

```
public event EventHandler<long>? DelegationChanged
```

Event Type

[EventHandler](#)<[long](#)>

## Enjailed

`public event EventHandler<long> Enjailed`

Event Type

[EventHandler](#)

## Unjailed

`public event EventHandler<long> Unjailed`

Event Type

[EventHandler](#)

# Class DelegationAddress

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public static class DelegationAddress
```

## Inheritance

[object](#) ← DelegationAddress

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### BondAddress(Address, Address)

```
public static Address BondAddress(Address delegateeMetadataAddress,  
Address delegatorAddress)
```

#### Parameters

delegateeMetadataAddress Address

delegatorAddress Address

#### Returns

Address

### BondAddress(Address, Address, Address)

```
public static Address BondAddress(Address delegateeAddress, Address  
delegateeAccountAddress, Address delegatorAddress)
```

## Parameters

`delegateeAddress` Address

`delegateeAccountAddress` Address

`delegatorAddress` Address

## Returns

Address

## CurrentLumpSumRewardsRecordAddress(Address)

```
public static Address CurrentLumpSumRewardsRecordAddress(Address  
delegateeMetadataAddress)
```

## Parameters

`delegateeMetadataAddress` Address

## Returns

Address

## CurrentLumpSumRewardsRecordAddress(Address, Address)

```
public static Address CurrentLumpSumRewardsRecordAddress(Address delegateeAddress,  
Address delegateeAccountAddress)
```

## Parameters

`delegateeAddress` Address

`delegateeAccountAddress` Address

## Returns

Address

## DelegateeMetadataAddress(Address, Address)

```
public static Address DelegateeMetadataAddress(Address delegateeAddress,  
Address delegateeAccountAddress)
```

Parameters

**delegateeAddress** Address

**delegateeAccountAddress** Address

Returns

Address

## DelegationPoolAddress(Address)

```
public static Address DelegationPoolAddress(Address delegateeMetadataAddress)
```

Parameters

**delegateeMetadataAddress** Address

Returns

Address

## DelegationPoolAddress(Address, Address)

```
public static Address DelegationPoolAddress(Address delegateeAddress,  
Address delegateeAccountAddress)
```

Parameters

`delegateeAddress` Address

`delegateeAccountAddress` Address

Returns

Address

## DelegatorMetadataAddress(Address, Address)

```
public static Address DelegatorMetadataAddress(Address delegatorAddress,  
Address delegatorAccountAddress)
```

Parameters

`delegatorAddress` Address

`delegatorAccountAddress` Address

Returns

Address

## LumpSumRewardsRecordAddress(Address, Address, long)

```
public static Address LumpSumRewardsRecordAddress(Address delegateeAddress, Address  
delegateeAccountAddress, long height)
```

Parameters

`delegateeAddress` Address

`delegateeAccountAddress` Address

`height` [long](#)

Returns

Address

## LumpSumRewardsRecordAddress(Address, long)

```
public static Address LumpSumRewardsRecordAddress(Address delegateeMetadataAddress,  
long height)
```

Parameters

delegateeMetadataAddress Address

height [long](#)

Returns

Address

## RebondGraceAddress(Address, Address)

```
public static Address RebondGraceAddress(Address delegateeMetadataAddress,  
Address delegatorAddress)
```

Parameters

delegateeMetadataAddress Address

delegatorAddress Address

Returns

Address

## RebondGraceAddress(Address, Address, Address)

```
public static Address RebondGraceAddress(Address delegateeAddress, Address  
delegateeAccountAddress, Address delegatorAddress)
```

## Parameters

**delegateeAddress** Address

**delegateeAccountAddress** Address

**delegatorAddress** Address

## Returns

Address

## RewardPoolAddress(Address)

```
public static Address RewardPoolAddress(Address delegateeMetadataAddress)
```

## Parameters

**delegateeMetadataAddress** Address

## Returns

Address

## RewardPoolAddress(Address, Address)

```
public static Address RewardPoolAddress(Address delegateeAddress,  
Address delegateeAccountAddress)
```

## Parameters

**delegateeAddress** Address

**delegateeAccountAddress** Address

## Returns

Address

## UnbondLockInAddress(Address, Address)

```
public static Address UnbondLockInAddress(Address delegateeMetadataAddress,  
Address delegatorAddress)
```

### Parameters

**delegateeMetadataAddress** Address

**delegatorAddress** Address

### Returns

Address

## UnbondLockInAddress(Address, Address, Address)

```
public static Address UnbondLockInAddress(Address delegateeAddress, Address  
delegateeAccountAddress, Address delegatorAddress)
```

### Parameters

**delegateeAddress** Address

**delegateeAccountAddress** Address

**delegatorAddress** Address

### Returns

Address

# Class DelegationRepository

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public abstract class DelegationRepository : IDelegationRepository
```

## Inheritance

[object](#) ← DelegationRepository

## Implements

[IDelegationRepository](#)

## Derived

[GuildRepository](#), [ValidatorRepository](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

DelegationRepository(IWorld, IActionContext, Address, Address, Address, Address, Address, Address, Address, Address, Address)

```
public DelegationRepository(IWorld world, IActionContext actionContext, Address  
delegateeAccountAddress, Address delegatorAccountAddress, Address  
delegateeMetadataAccountAddress, Address delegatorMetadataAccountAddress, Address  
bondAccountAddress, Address unbondLockInAccountAddress, Address  
rebondGraceAccountAddress, Address unbondingSetAccountAddress, Address  
lumpSumRewardRecordAccountAddress)
```

## Parameters

**world** IWorld

```
actionContext IActionContext  
  
delegateeAccountAddress Address  
  
delegatorAccountAddress Address  
  
delegateeMetadataAccountAddress Address  
  
delegatorMetadataAccountAddress Address  
  
bondAccountAddress Address  
  
unbondLockInAccountAddress Address  
  
rebondGraceAccountAddress Address  
  
unbondingSetAccountAddress Address  
  
lumpSumRewardRecordAccountAddress Address
```

## Fields

### bondAccount

```
protected IAccount bondAccount
```

Field Value

IAccount

### delegateeAccount

```
protected IAccount delegateeAccount
```

Field Value

IAccount

## delegateeMetadataAccount

```
protected IAccount delegateeMetadataAccount
```

Field Value

IAccount

## delegatorAccount

```
protected IAccount delegatorAccount
```

Field Value

IAccount

## delegatorMetadataAccount

```
protected IAccount delegatorMetadataAccount
```

Field Value

IAccount

## lumpSumRewardsRecordAccount

```
protected IAccount lumpSumRewardsRecordAccount
```

Field Value

IAccount

## previousWorld

```
protected IWorld previousWorld
```

Field Value

IWorld

## rebondGraceAccount

```
protected IAccount rebondGraceAccount
```

Field Value

IAccount

## unbondLockInAccount

```
protected IAccount unbondLockInAccount
```

Field Value

IAccount

## unbondingSetAccount

```
protected IAccount unbondingSetAccount
```

Field Value

IAccount

## Properties

### ActionContext

```
public IActionContext ActionContext { get; }
```

Property Value

IActionContext

## DelegateeAccountAddress

```
public Address DelegateeAccountAddress { get; }
```

Property Value

Address

## DelegatorAccountAddress

```
public Address DelegatorAccountAddress { get; }
```

Property Value

Address

## World

```
public virtual IWorld World { get; }
```

Property Value

IWorld

## Methods

### GetBalance(Address, Currency)

```
public FungibleAssetValue GetBalance(Address address, Currency currency)
```

## Parameters

address Address

currency Currency

## Returns

FungibleAssetValue

## GetBond(IDelegatee, Address)

```
public Bond GetBond(IDelegatee delegatee, Address delegatorAddress)
```

## Parameters

delegatee [IDelegatee](#)

delegatorAddress Address

## Returns

[Bond](#)

## GetCurrentLumpSumRewardsRecord(IDelegatee)

```
public LumpSumRewardsRecord? GetCurrentLumpSumRewardsRecord(IDelegatee delegatee)
```

## Parameters

delegatee [IDelegatee](#)

## Returns

[LumpSumRewardsRecord](#)

## GetDelegatee(Address)

```
public abstract IDelegatee GetDelegatee(Address address)
```

Parameters

address Address

Returns

[IDelegatee](#)

## GetDelegateeMetadata(Address)

```
public DelegateeMetadata GetDelegateeMetadata(Address delegateeAddress)
```

Parameters

delegateeAddress Address

Returns

[DelegateeMetadata](#)

## GetDelegator(Address)

```
public abstract IDegenerator GetDelegator(Address address)
```

Parameters

address Address

Returns

## [IDelegator](#)

### GetDelegatorMetadata(Address)

```
public DelegatorMetadata GetDelegatorMetadata(Address delegatorAddress)
```

#### Parameters

delegatorAddress Address

#### Returns

[DelegatorMetadata](#)

### GetLumpSumRewardsRecord(IDelegatee, long)

```
public LumpSumRewardsRecord? GetLumpSumRewardsRecord(IDelegatee delegatee,  
long height)
```

#### Parameters

delegatee [IDelegatee](#)

height [long](#)

#### Returns

[LumpSumRewardsRecord](#)

### GetRebondGrace(IDelegatee, Address)

```
public RebondGrace GetRebondGrace(IDelegatee delegatee, Address delegatorAddress)
```

#### Parameters

delegatee [IDelegatee](#)

`delegatorAddress` Address

Returns

[RebondGrace](#)

## GetUnbondLockIn(IDelegatee, Address)

`public` `UnbondLockIn` `GetUnbondLockIn`(`IDelegatee` `delegatee`, `Address` `delegatorAddress`)

Parameters

`delegatee` [IDelegatee](#)

`delegatorAddress` Address

Returns

[UnbondLockIn](#)

## GetUnbondingSet()

`public` `UnbondingSet` `GetUnbondingSet`()

Returns

[UnbondingSet](#)

## GetUnlimitedRebondGrace(Address)

`public` `RebondGrace` `GetUnlimitedRebondGrace`(`Address` `address`)

Parameters

`address` Address

Returns

[RebondGrace](#)

## GetUnlimitedUnbondLockIn(Address)

```
public UnbondLockIn GetUnlimitedUnbondLockIn(Address address)
```

Parameters

address Address

Returns

[UnbondLockIn](#)

## SetBond(Bond)

```
public void SetBond(Bond bond)
```

Parameters

bond [Bond](#)

## SetDelegatee(IDelegatee)

```
public abstract void SetDelegatee(IDelegatee delegatee)
```

Parameters

delegatee [IDelegatee](#)

## SetDelegateeMetadata(DelegateeMetadata)

```
public void SetDelegateeMetadata(DelegateeMetadata delegateeMetadata)
```

Parameters

delegateeMetadata [DelegateeMetadata](#)

## SetDelegator(IDelegator)

```
public abstract void SetDelegator(IDelegator delegator)
```

Parameters

delegator [IDelegator](#)

## SetDelegatorMetadata(DelegatorMetadata)

```
public void SetDelegatorMetadata(DelegatorMetadata delegatorMetadata)
```

Parameters

delegatorMetadata [DelegatorMetadata](#)

## SetLumpSumRewardsRecord(LumpSumRewardsRecord)

```
public void SetLumpSumRewardsRecord(LumpSumRewardsRecord lumpSumRewardsRecord)
```

Parameters

lumpSumRewardsRecord [LumpSumRewardsRecord](#)

## SetRebondGrace(RebondGrace)

```
public void SetRebondGrace(RebondGrace rebondGrace)
```

Parameters

**rebondGrace** [RebondGrace](#)

## SetUnbondLockIn(UnbondLockIn)

```
public void SetUnbondLockIn(UnbondLockIn unbondLockIn)
```

Parameters

**unbondLockIn** [UnbondLockIn](#)

## SetUnbondingSet(UnbondingSet)

```
public void SetUnbondingSet(UnbondingSet unbondingSet)
```

Parameters

**unbondingSet** [UnbondingSet](#)

## TransferAsset(Address, Address, FungibleAssetValue)

```
public void TransferAsset(Address sender, Address recipient,  
FungibleAssetValue value)
```

Parameters

**sender** Address

**recipient** Address

**value** FungibleAssetValue

## UpdateWorld(IWorld)

```
public virtual void UpdateWorld(IWorld world)
```

### Parameters

`world` IWorld

# Class DelegatorMetadata

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public class DelegatorMetadata : IDelegatorMetadata, IBencodable
```

## Inheritance

[object](#) ← DelegatorMetadata

## Implements

[IDelegatorMetadata](#), [IBencodable](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### DelegatorMetadata(Address, Address, IValue)

```
public DelegatorMetadata(Address address, Address accountAddress, IValue bencoded)
```

#### Parameters

address Address

accountAddress Address

bencoded IValue

### DelegatorMetadata(Address, Address, List)

```
public DelegatorMetadata(Address address, Address accountAddress, List bencoded)
```

#### Parameters

`address` Address

`accountAddress` Address

`bencoded` List

## DelegatorMetadata(Address, Address, Address, Address)

```
public DelegatorMetadata(Address address, Address accountAddress, Address  
delegationPoolAddress, Address rewardAddress)
```

### Parameters

`address` Address

`accountAddress` Address

`delegationPoolAddress` Address

`rewardAddress` Address

## Properties

### Address

```
public Address Address { get; }
```

### Property Value

Address

### Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

## Delegatees

```
public ImmutableSortedSet<Address> Delegatees { get; }
```

Property Value

[ImmutableSortedSet](#)<Address>

## DelegationPoolAddress

```
public Address DelegationPoolAddress { get; }
```

Property Value

Address

## DelegatorAccountAddress

```
public Address DelegatorAccountAddress { get; }
```

Property Value

Address

## DelegatorAddress

```
public Address DelegatorAddress { get; }
```

Property Value

Address

# RewardAddress

```
public Address RewardAddress { get; }
```

Property Value

Address

## Methods

### AddDelegatee(Address)

```
public void AddDelegatee(Address delegatee)
```

Parameters

**delegatee** Address

### Equals(IDelegator?)

```
public virtual bool Equals(IDelegator? other)
```

Parameters

**other** [IDelegator](#)

Returns

[bool](#)

### Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

## Parameters

**obj** [object](#)

The object to compare with the current object.

## Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

## Returns

[int](#)

A hash code for the current object.

## RemoveDelegatee(Address)

```
public void RemoveDelegatee(Address delegatee)
```

## Parameters

**delegatee** Address

# Class Delegator<T, TSelf>

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public abstract class Delegator<T, TSelf> : IDelegator where T : Delegatee<TSelf, T>
where TSelf : Delegator<T, TSelf>
```

## Type Parameters

T

TSelf

## Inheritance

[object](#) ← Delegator<T, TSelf>

## Implements

[IDelegator](#)

## Derived

[GuildDelegator](#), [ValidatorDelegator](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

Delegator(Address, Address, Address, Address,  
IDelegationRepository)

```
public Delegator(Address address, Address accountAddress, Address
delegationPoolAddress, Address rewardAddress, IDelegationRepository repository)
```

## Parameters

`address` Address

`accountAddress` Address

`delegationPoolAddress` Address

`rewardAddress` Address

`repository` [IDelegationRepository](#)

## Delegator(Address, ID*elegationRepository*)

```
public Delegator(Address address, IDelegationRepository repository)
```

### Parameters

`address` Address

`repository` [IDelegationRepository](#)

## Properties

### AccountAddress

```
public Address AccountAddress { get; }
```

### Property Value

Address

### Address

```
public Address Address { get; }
```

### Property Value

Address

## Delegatees

```
public ImmutableSortedSet<Address> Delegatees { get; }
```

Property Value

[ImmutableSortedSet](#)<Address>

## DelegationPoolAddress

```
public Address DelegationPoolAddress { get; }
```

Property Value

Address

## Metadata

```
public DelegatorMetadata Metadata { get; }
```

Property Value

[DelegatorMetadata](#)

## MetadataAddress

```
public Address MetadataAddress { get; }
```

Property Value

Address

## MetadataBencoded

```
public List<MetadataBencoded> get();
```

Property Value

List

## Repository

```
public IDelegationRepository Repository() { return null; }
```

Property Value

[IDelegationRepository](#)

## RewardAddress

```
public Address RewardAddress() { return null; }
```

Property Value

Address

## Methods

### CancelUndelegate(T, FungibleAssetValue, long)

```
public void CancelUndelegate(T delegatee, FungibleAssetValue fav, long height)
```

Parameters

`delegatee` T

`fav` FungibleAssetValue

height [long](#)

## ClaimReward(T, long)

```
public void ClaimReward(T delegatee, long height)
```

### Parameters

delegatee T

height [long](#)

## Delegate(T, FungibleAssetValue, long)

```
public virtual void Delegate(T delegatee, FungibleAssetValue fav, long height)
```

### Parameters

delegatee T

fav FungibleAssetValue

height [long](#)

## Redelegate(T, T, BigInteger, long)

```
public virtual void Redelegate(T srcDelegatee, T dstDelegatee, BigInteger share,  
long height)
```

### Parameters

srcDelegatee T

dstDelegatee T

share [BigInteger](#)

height [long](#)

## Undelegate(T, BigInteger, long)

```
public virtual void Undelegate(T delegatee, BigInteger share, long height)
```

### Parameters

delegatee T

share [BigInteger](#)

height [long](#)

# Interface IDelegatee

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public interface IDelegatee
```

## Properties

### AccountAddress

```
Address AccountAddress { get; }
```

#### Property Value

Address

### Address

```
Address Address { get; }
```

#### Property Value

Address

### DelegationCurrency

```
Currency DelegationCurrency { get; }
```

#### Property Value

Currency

## DelegationPoolAddress

Address DelegationPoolAddress { [get](#); }

Property Value

Address

## Delegators

ImmutableSortedSet<Address> Delegators { [get](#); }

Property Value

[ImmutableSortedSet](#)<Address>

## Jailed

bool Jailed { [get](#); }

Property Value

[bool](#)

## JailedUntil

long JailedUntil { [get](#); }

Property Value

[long](#)

## MaxRebondGraceEntries

```
int MaxRebondGraceEntries { get; }
```

Property Value

[int](#)

## MaxUnbondLockInEntries

```
int MaxUnbondLockInEntries { get; }
```

Property Value

[int](#)

## RewardCurrencies

```
ImmutableSortedSet<Currency> RewardCurrencies { get; }
```

Property Value

[ImmutableSortedSet](#)<Currency>

## RewardPoolAddress

```
Address RewardPoolAddress { get; }
```

Property Value

Address

## RewardRemainderPoolAddress

```
Address RewardRemainderPoolAddress { get; }
```

Property Value

Address

## Tombstoned

```
bool Tombstoned { get; }
```

Property Value

[bool](#)

## TotalDelegated

```
FungibleAssetValue TotalDelegated { get; }
```

Property Value

FungibleAssetValue

## TotalShares

```
BigInteger TotalShares { get; }
```

Property Value

[BigInteger](#)

## UnbondingPeriod

```
long UnbondingPeriod { get; }
```

Property Value

[long](#)

## Methods

### Bond(IDelegator, FungibleAssetValue, long)

```
BigInteger Bond(IDelegator delegator, FungibleAssetValue fav, long height)
```

Parameters

`delegator` [IDelegator](#)

`fav` FungibleAssetValue

`height` [long](#)

Returns

[BigInteger](#)

### BondAddress(Address)

```
Address BondAddress(Address delegatorAddress)
```

Parameters

`delegatorAddress` Address

Returns

Address

## CollectRewards(long)

void CollectRewards(long height)

### Parameters

height [long](#)

## CurrentLumpSumRewardsRecordAddress()

Address CurrentLumpSumRewardsRecordAddress()

### Returns

Address

## DistributeReward(IDelegator, long)

void DistributeReward(IDelegator delegator, long height)

### Parameters

delegator [IDelegator](#)

height [long](#)

## FAVFromShare(BigInteger)

FungibleAssetValue FAVFromShare(BigInteger share)

### Parameters

share [BigInteger](#)

Returns

FungibleAssetValue

## Jail(long)

`void Jail(long releaseHeight)`

Parameters

`releaseHeight long`

## LumpSumRewardsRecordAddress(long)

`Address LumpSumRewardsRecordAddress(long height)`

Parameters

`height long`

Returns

Address

## RebondGraceAddress(Address)

`Address RebondGraceAddress(Address delegatorAddress)`

Parameters

`delegatorAddress Address`

Returns

Address

## ShareFromFAV(FungibleAssetValue)

```
BigInteger ShareFromFAV(FungibleAssetValue fav)
```

### Parameters

**fav** FungibleAssetValue

### Returns

[BigInteger](#)

## Slash(BigInteger, long, long)

```
void Slash(BigInteger slashFactor, long infractionHeight, long height)
```

### Parameters

**slashFactor** [BigInteger](#)

**infractionHeight** [long](#)

**height** [long](#)

## Tombstone()

```
void Tombstone()
```

## Unbond(IDelegator, BigInteger, long)

```
FungibleAssetValue Unbond(IDelegator delegator, BigInteger share, long height)
```

### Parameters

**delegator** [IDelegator](#)

share [BigInteger](#)

height [long](#)

Returns

FungibleAssetValue

## UnbondLockInAddress(Address)

Address [UnbondLockInAddress](#)(Address delegatorAddress)

Parameters

delegatorAddress Address

Returns

Address

## Unjail(long)

void [Unjail](#)(long height)

Parameters

height [long](#)

## Events

### DelegationChanged

event EventHandler<long>? DelegationChanged

Event Type

[EventHandler](#)<[long](#)>

# Interface IDelegateeMetadata

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public interface IDelegateeMetadata : IBencodable
```

## Inherited Members

IBencodable.Bencoded

## Properties

### Address

```
Address Address { get; }
```

Property Value

Address

### DelegateeAccountAddress

```
Address DelegateeAccountAddress { get; }
```

Property Value

Address

### DelegateeAddress

```
Address DelegateeAddress { get; }
```

Property Value

Address

## DelegationCurrency

```
Currency DelegationCurrency { get; }
```

Property Value

Currency

## DelegationPoolAddress

```
Address DelegationPoolAddress { get; }
```

Property Value

Address

## Delegators

```
ImmutableSortedSet<Address> Delegators { get; }
```

Property Value

[ImmutableSortedSet](#)<Address>

## Jailed

```
bool Jailed { get; }
```

Property Value

[bool](#)

## JailedUntil

```
long JailedUntil { get; }
```

Property Value

[long](#) ↗

## MaxRebondGraceEntries

```
int MaxRebondGraceEntries { get; }
```

Property Value

[int](#) ↗

## MaxUnbondLockInEntries

```
int MaxUnbondLockInEntries { get; }
```

Property Value

[int](#) ↗

## RewardCurrencies

```
ImmutableSortedSet<Currency> RewardCurrencies { get; }
```

Property Value

[ImmutableSortedSet](#) ↗ <Currency>

## RewardPoolAddress

```
Address RewardPoolAddress { get; }
```

Property Value

Address

## RewardRemainderPoolAddress

```
Address RewardRemainderPoolAddress { get; }
```

Property Value

Address

## Tombstoned

```
bool Tombstoned { get; }
```

Property Value

[bool](#)

## TotalDelegatedFAV

```
FungibleAssetValue TotalDelegatedFAV { get; }
```

Property Value

FungibleAssetValue

## TotalShares

```
BigInteger TotalShares { get; }
```

Property Value

[BigInteger](#)

## UnbondingPeriod

```
long UnbondingPeriod { get; }
```

Property Value

[long](#)

## Methods

### FAVFromShare(BigInteger)

```
FungibleAssetValue FAVFromShare(BigInteger share)
```

Parameters

share [BigInteger](#)

Returns

FungibleAssetValue

### ShareFromFAV(FungibleAssetValue)

```
BigInteger ShareFromFAV(FungibleAssetValue fav)
```

Parameters

`fav` FungibleAssetValue

Returns

[BigInteger](#)

# Interface IDelegationRepository

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public interface IDelegationRepository
```

## Properties

### ActionContext

```
IActionContext ActionContext { get; }
```

#### Property Value

IActionContext

### DelegateeAccountAddress

```
Address DelegateeAccountAddress { get; }
```

#### Property Value

Address

### DelegatorAccountAddress

```
Address DelegatorAccountAddress { get; }
```

#### Property Value

Address

# World

```
IWorld World { get; }
```

Property Value

IWorld

## Methods

### GetBalance(Address, Currency)

```
FungibleAssetValue GetBalance(Address address, Currency currency)
```

Parameters

**address** Address

**currency** Currency

Returns

FungibleAssetValue

### GetBond(IDelegatee, Address)

```
Bond GetBond(IDelegatee delegatee, Address delegatorAddress)
```

Parameters

**delegatee** [IDelegatee](#)

**delegatorAddress** Address

Returns

[Bond](#)

## GetCurrentLumpSumRewardsRecord(IDelegatee)

LumpSumRewardsRecord? GetCurrentLumpSumRewardsRecord(IDelegatee delegatee)

### Parameters

delegatee [IDelegatee](#)

### Returns

[LumpSumRewardsRecord](#)

## GetDelegatee(Address)

IDelegatee [GetDelegatee](#)(Address address)

### Parameters

address Address

### Returns

[IDelegatee](#)

## GetDelegateeMetadata(Address)

DelegateeMetadata [GetDelegateeMetadata](#)(Address delegateeAddress)

### Parameters

delegateeAddress Address

### Returns

## [DelegateeMetadata](#)

### GetDelegator(Address)

IDelegator **GetDelegator**(Address address)

Parameters

address Address

Returns

[IDelegator](#)

### GetDelegatorMetadata(Address)

DelegatorMetadata **GetDelegatorMetadata**(Address delegatorAddress)

Parameters

delegatorAddress Address

Returns

[DelegatorMetadata](#)

### GetLumpSumRewardsRecord(IDelegatee, long)

LumpSumRewardsRecord? **GetLumpSumRewardsRecord**(IDelegatee delegatee, **long** height)

Parameters

delegatee [IDelegatee](#)

height [long](#)

Returns

[LumpSumRewardsRecord](#)

## GetRebondGrace(IDelegatee, Address)

RebondGrace **GetRebondGrace**(IDelegatee delegatee, Address delegatorAddress)

Parameters

delegatee [IDelegatee](#)

delegatorAddress Address

Returns

[RebondGrace](#)

## GetUnbondLockIn(IDelegatee, Address)

UnbondLockIn **GetUnbondLockIn**(IDelegatee delegatee, Address delegatorAddress)

Parameters

delegatee [IDelegatee](#)

delegatorAddress Address

Returns

[UnbondLockIn](#)

## GetUnbondingSet()

UnbondingSet **GetUnbondingSet**()

Returns

[UnbondingSet](#)

## GetUnlimitedRebondGrace(Address)

RebondGrace **GetUnlimitedRebondGrace**(Address address)

Parameters

address Address

Returns

[RebondGrace](#)

## GetUnlimitedUnbondLockIn(Address)

UnbondLockIn **GetUnlimitedUnbondLockIn**(Address address)

Parameters

address Address

Returns

[UnbondLockIn](#)

## SetBond(Bond)

void **SetBond**(Bond bond)

Parameters

bond [Bond](#)

## SetDelegatee(IDelegatee)

```
void SetDelegatee(IDelegatee delegatee)
```

### Parameters

delegatee [IDelegatee](#)

## SetDelegateeMetadata(DelegateeMetadata)

```
void SetDelegateeMetadata(DelegateeMetadata delegateeMetadata)
```

### Parameters

delegateeMetadata [DelegateeMetadata](#)

## SetDelegator(IDelegator)

```
void SetDelegator(IDelegator delegator)
```

### Parameters

delegator [IDelegator](#)

## SetDelegatorMetadata(DelegatorMetadata)

```
void SetDelegatorMetadata(DelegatorMetadata delegatorMetadata)
```

### Parameters

delegatorMetadata [DelegatorMetadata](#)

## SetLumpSumRewardsRecord(LumpSumRewardsRecord)

```
void SetLumpSumRewardsRecord(LumpSumRewardsRecord lumpSumRewardsRecord)
```

Parameters

lumpSumRewardsRecord [LumpSumRewardsRecord](#)

## SetRebondGrace(RebondGrace)

```
void SetRebondGrace(RebondGrace rebondGrace)
```

Parameters

rebondGrace [RebondGrace](#)

## SetUnbondLockIn(UnbondLockIn)

```
void SetUnbondLockIn(UnbondLockIn unbondLockIn)
```

Parameters

unbondLockIn [UnbondLockIn](#)

## SetUnbondingSet(UnbondingSet)

```
void SetUnbondingSet(UnbondingSet unbondingSet)
```

Parameters

unbondingSet [UnbondingSet](#)

## TransferAsset(Address, Address, FungibleAssetValue)

```
void TransferAsset(Address sender, Address recipient, FungibleAssetValue value)
```

## Parameters

**sender** Address

**recipient** Address

**value** FungibleAssetValue

## UpdateWorld(IWorld)

```
void UpdateWorld(IWorld world)
```

## Parameters

**world** IWorld

# Interface IDelegator

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public interface IDelegator
```

## Properties

### AccountAddress

Address AccountAddress { [get](#); }

#### Property Value

Address

### Address

Address Address { [get](#); }

#### Property Value

Address

### Delegatees

ImmutableSortedSet<Address> Delegatees { [get](#); }

#### Property Value

[ImmutableSortedSet](#)<Address>

## DelegationPoolAddress

Address DelegationPoolAddress { [get](#); }

Property Value

Address

## RewardAddress

Address RewardAddress { [get](#); }

Property Value

Address

## Methods

### CancelUndelegate(IDelegatee, FungibleAssetValue, long)

[void CancelUndelegate\(IDelegatee delegatee, FungibleAssetValue fav, long height\)](#)

Parameters

[delegatee](#) [IDelegatee](#)

[fav](#) FungibleAssetValue

[height](#) [long](#) ↗

### ClaimReward(IDelegatee, long)

[void ClaimReward\(IDelegatee delegatee, long height\)](#)

Parameters

delegatee [IDelegatee](#)

height [long](#)

## Delegate(IDelegatee, FungibleAssetValue, long)

```
void Delegate(IDelegatee delegatee, FungibleAssetValue fav, long height)
```

Parameters

delegatee [IDelegatee](#)

fav FungibleAssetValue

height [long](#)

## Redelegate(IDelegatee, IDelegatee, BigInteger, long)

```
void Redelegate(IDelegatee srcDelegatee, IDelegatee dstDelegatee, BigInteger share,
long height)
```

Parameters

srcDelegatee [IDelegatee](#)

dstDelegatee [IDelegatee](#)

share [BigInteger](#)

height [long](#)

## Undelegate(IDelegatee, BigInteger, long)

```
void Undelegate(IDelegatee delegatee, BigInteger share, long height)
```

## Parameters

delegatee [IDelegatee](#)

share [BigInteger](#)

height [long](#)

# Interface IDegeneratorMetadata

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public interface IDegeneratorMetadata : IBencodable
```

## Inherited Members

IBencodable.Bencoded

## Properties

### Address

```
Address Address { get; }
```

Property Value

Address

### Delegatees

```
ImmutableSortedSet<Address> Delegatees { get; }
```

Property Value

[ImmutableSortedSet](#)<Address>

### DelegationPoolAddress

```
Address DelegationPoolAddress { get; }
```

Property Value

Address

## DelegatorAccountAddress

```
Address DelegatorAccountAddress { get; }
```

Property Value

Address

## DelegatorAddress

```
Address DelegatorAddress { get; }
```

Property Value

Address

## Methods

### AddDelegatee(Address)

```
void AddDelegatee(Address delegatee)
```

Parameters

**delegatee** Address

### RemoveDelegatee(Address)

```
void RemoveDelegatee(Address delegatee)
```

Parameters

**delegatee** Address

# Interface IUnbonding

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public interface IUnbonding
```

## Properties

### Address

```
Address Address { get; }
```

### Property Value

Address

### IsEmpty

```
bool IsEmpty { get; }
```

### Property Value

[bool](#)

### IsFull

```
bool IsFull { get; }
```

### Property Value

[bool](#)

# LowestExpireHeight

`long LowestExpireHeight { get; }`

Property Value

[long](#)

## Methods

### Release(long)

`IUnbonding Release(long height)`

Parameters

`height` [long](#)

Returns

[IUnbonding](#)

### Slash(BigInteger, long, long, out FungibleAssetValue?)

`IUnbonding Slash(BigInteger slashFactor, long infractionHeight, long height, out FungibleAssetValue? slashedFAV)`

Parameters

`slashFactor` [BigInteger](#)

`infractionHeight` [long](#)

`height` [long](#)

`slashedFAV` [FungibleAssetValue?](#)

## Returns

[IUnbonding](#)

# Class LumpSumRewardsRecord

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public class LumpSumRewardsRecord : IBencodable, IEquatable<LumpSumRewardsRecord>
```

## Inheritance

[object](#) ← LumpSumRewardsRecord

## Implements

IBencodable, [IEquatable](#)<[LumpSumRewardsRecord](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### LumpSumRewardsRecord(Address, IValue)

```
public LumpSumRewardsRecord(Address address, IValue bencoded)
```

#### Parameters

address Address

bencoded IValue

### LumpSumRewardsRecord(Address, List)

```
public LumpSumRewardsRecord(Address address, List bencoded)
```

#### Parameters

address Address

bencoded List

LumpSumRewardsRecord(Address, long, BigInteger,  
ImmutableSortedSet<Address>,  
IEnumerable<Currency>)

```
public LumpSumRewardsRecord(Address address, long startHeight, BigInteger  
totalShares, ImmutableSortedSet<Address> delegators, IEnumerable<Currency>  
currencies)
```

Parameters

address Address

startHeight [long](#)

totalShares [BigInteger](#)

delegators [ImmutableSortedSet](#)<Address>

currencies [IEnumerable](#)<Currency>

LumpSumRewardsRecord(Address, long, BigInteger,  
ImmutableSortedSet<Address>,  
IEnumerable<Currency>, long?)

```
public LumpSumRewardsRecord(Address address, long startHeight, BigInteger  
totalShares, ImmutableSortedSet<Address> delegators, IEnumerable<Currency>  
currencies, long? lastStartHeight)
```

Parameters

address Address

startHeight [long](#)

totalShares [BigInteger](#)

delegators [ImmutableSortedSet](#)<Address>

currencies [IEnumerable](#)<Currency>

lastStartHeight [long](#)?

LumpSumRewardsRecord(Address, long, BigInteger,  
ImmutableSortedSet<Address>,  
IEnumerable<FungibleAssetValue>, long?)

```
public LumpSumRewardsRecord(Address address, long startHeight, BigInteger  
totalShares, ImmutableSortedSet<Address> delegators, IEnumerable<FungibleAssetValue>  
lumpSumRewards, long? lastStartHeight)
```

## Parameters

address Address

startHeight [long](#)

totalShares [BigInteger](#)

delegators [ImmutableSortedSet](#)<Address>

lumpSumRewards [IEnumerable](#)<FungibleAssetValue>

lastStartHeight [long](#)?

## Properties

### Address

```
public Address Address { get; }
```

### Property Value

Address

## Bencoded

```
public List Bencoded { get; }
```

Property Value

List

## Delegators

```
public ImmutableSortedSet<Address> Delegators { get; }
```

Property Value

[ImmutableSortedSet](#)<Address>

## LastStartHeight

```
public long? LastStartHeight { get; }
```

Property Value

[long](#)?

## LumpSumRewards

```
public ImmutableDictionary<Currency, FungibleAssetValue> LumpSumRewards { get; }
```

Property Value

[ImmutableDictionary](#)<Currency, FungibleAssetValue>

## StartHeight

```
public long StartHeight { get; }
```

Property Value

[long](#)

## TotalShares

```
public BigInteger TotalShares { get; }
```

Property Value

[BigInteger](#)

## Methods

### AddLumpSumRewards(FungibleAssetValue)

```
public LumpSumRewardsRecord AddLumpSumRewards(FungibleAssetValue rewards)
```

Parameters

`rewards` FungibleAssetValue

Returns

[LumpSumRewardsRecord](#)

### AddLumpSumRewards(LumpSumRewardsRecord, FungibleAssetValue)

```
public static LumpSumRewardsRecord AddLumpSumRewards(LumpSumRewardsRecord record, FungibleAssetValue rewards)
```

Parameters

record [LumpSumRewardsRecord](#)

rewards FungibleAssetValue

Returns

[LumpSumRewardsRecord](#)

AddLumpSumRewards(IEnumerable<FungibleAssetValue>)

```
public LumpSumRewardsRecord AddLumpSumRewards(IEnumerable<FungibleAssetValue>
rewards)
```

Parameters

rewards [IEnumerable](#)<FungibleAssetValue>

Returns

[LumpSumRewardsRecord](#)

Equals(LumpSumRewardsRecord?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(LumpSumRewardsRecord? other)
```

Parameters

other [LumpSumRewardsRecord](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

### Parameters

[obj](#) [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

### Returns

[int](#)

A hash code for the current object.

## MoveAddress(Address)

```
public LumpSumRewardsRecord MoveAddress(Address address)
```

Parameters

**address** Address

Returns

[LumpSumRewardsRecord](#)

## RmRemoveDelegator(Address)

```
public LumpSumRewardsRecord RemoveDelegator(Address delegator)
```

Parameters

**delegator** Address

Returns

[LumpSumRewardsRecord](#)

## RewardsDuringPeriod(BigInteger)

```
public ImmutableSortedDictionary<Currency, FungibleAssetValue>
RewardsDuringPeriod(BigInteger share)
```

Parameters

**share** [BigInteger](#)

Returns

[ImmutableSortedDictionary](#)<Currency, FungibleAssetValue>

## RewardsDuringPeriod(BigInteger, Currency)

```
public FungibleAssetValue RewardsDuringPeriod(BigInteger share, Currency currency)
```

## Parameters

share [BigInteger](#) ↗

currency [Currency](#)

## Returns

[FungibleAssetValue](#)

# Class RebondGrace

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public sealed class RebondGrace : IUnbonding, IBencodable, IEquatable<RebondGrace>
```

## Inheritance

[object](#) ← RebondGrace

## Implements

[IUnbonding](#), [IBencodable](#), [IEquatable](#)<[RebondGrace](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RebondGrace(Address, int, [IValue](#),  
[IDelegationRepository](#)?)

```
public RebondGrace(Address address, int maxEntries, IValue bencoded,  
IDelegationRepository? repository = null)
```

## Parameters

address Address

maxEntries [int](#)

bencoded [IValue](#)

repository [IDelegationRepository](#)

RebondGrace(Address, int, List, [IDelegationRepository](#)?)

```
public RebondGrace(Address address, int maxEntries, List bencoded,  
IDelegationRepository? repository = null)
```

## Parameters

address Address

maxEntries [int](#)

bencoded List

repository [IDelegationRepository](#)

## RebondGrace(Address, int, IDelegationRepository?)

```
public RebondGrace(Address address, int maxEntries, IDelegationRepository?  
repository = null)
```

## Parameters

address Address

maxEntries [int](#)

repository [IDelegationRepository](#)

## RebondGrace(Address, int, IEnumerable<UnbondingEntry>, IDelegationRepository?)

```
public RebondGrace(Address address, int maxEntries, IEnumerable<UnbondingEntry>  
entries, IDelegationRepository? repository = null)
```

## Parameters

address Address

maxEntries [int](#)

entries [IEnumerable](#)<[UnbondingEntry](#)>

repository [IDelegationRepository](#)

## Properties

### Address

`public Address Address { get; }`

### Property Value

Address

### Bencoded

`public List Bencoded { get; }`

### Property Value

List

### DelegateeAddress

`public Address DelegateeAddress { get; }`

### Property Value

Address

### DelegatorAddress

```
public Address DelegatorAddress { get; }
```

Property Value

Address

## Entries

```
public ImmutableSortedDictionary<long, ImmutableList<UnbondingEntry>> Entries {  
    get; }
```

Property Value

[ImmutableSortedDictionary](#)<[long](#), [ImmutableList](#)<[UnbondingEntry](#)>>

## FlattenedEntries

```
public ImmutableList<UnbondingEntry> FlattenedEntries { get; }
```

Property Value

[ImmutableList](#)<[UnbondingEntry](#)>

## IsEmpty

```
public bool IsEmpty { get; }
```

Property Value

[bool](#)

## IsFull

```
public bool IsFull { get; }
```

Property Value

[bool](#)

## LowestExpireHeight

```
public long LowestExpireHeight { get; }
```

Property Value

[long](#)

## MaxEntries

```
public int MaxEntries { get; }
```

Property Value

[int](#)

## Repository

```
public IDelegationRepository? Repository { get; }
```

Property Value

[IDelegationRepository](#)

## Methods

### Equals(RebondGrace?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(RebondGrace? other)
```

Parameters

**other** [RebondGrace](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

Parameters

**obj** [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Release(long)

```
public RebondGrace Release(long height)
```

Parameters

height [long](#)

Returns

[RebondGrace](#)

## Slash(BigInteger, long, long, out FungibleAssetValue?)

```
public RebondGrace Slash(BigInteger slashFactor, long infractionHeight, long height,  
out FungibleAssetValue? slashedFAV)
```

Parameters

slashFactor [BigInteger](#)

infractionHeight [long](#)

height [long](#)

slashedFAV [FungibleAssetValue?](#)

Returns

[RebondGrace](#)



# Class UnbondLockIn

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public sealed class UnbondLockIn : IUnbonding, IBencodable, IEquatable<UnbondLockIn>
```

## Inheritance

[object](#) ← UnbondLockIn

## Implements

[IUnbonding](#), [IBencodable](#), [IEquatable](#)<[UnbondLockIn](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

UnbondLockIn(Address, int, IValue,  
IDelegationRepository?)

```
public UnbondLockIn(Address address, int maxEntries, IValue bencoded,  
IDelegationRepository? repository = null)
```

## Parameters

address Address

maxEntries [int](#)

bencoded IValue

repository [IDelegationRepository](#)

UnbondLockIn(Address, int, List,

## IDelegationRepository?)

```
public UnbondLockIn(Address address, int maxEntries, List bencoded,  
IDelegationRepository? repository = null)
```

### Parameters

address Address

maxEntries [int](#)

bencoded List

repository [IDelegationRepository](#)

## UnbondLockIn(Address, int, Address, Address, IDelegationRepository?)

```
public UnbondLockIn(Address address, int maxEntries, Address delegateeAddress,  
Address delegatorAddress, IDelegationRepository? repository)
```

### Parameters

address Address

maxEntries [int](#)

delegateeAddress Address

delegatorAddress Address

repository [IDelegationRepository](#)

## UnbondLockIn(Address, int, Address, Address, IEnumerable<UnbondingEntry>, IDelegationRepository?)

```
public UnbondLockIn(Address address, int maxEntries, Address delegateeAddress,  
Address delegatorAddress, IEnumerable<UnbondingEntry> entries,  
IDelegationRepository? repository = null)
```

## Parameters

address Address

maxEntries [int](#)

delegateeAddress Address

delegatorAddress Address

entries [IEnumerable](#)<[UnbondingEntry](#)>

repository [IDelegationRepository](#)

## Properties

### Address

```
public Address Address { get; }
```

### Property Value

Address

### Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

## DelegateeAddress

```
public Address DelegateeAddress { get; }
```

Property Value

Address

## DelegatorAddress

```
public Address DelegatorAddress { get; }
```

Property Value

Address

## Entries

```
public ImmutableSortedDictionary<long, ImmutableList<UnbondingEntry>> Entries {  
    get; }
```

Property Value

[ImmutableSortedDictionary](#)<[long](#), [ImmutableList](#)<[UnbondingEntry](#)>>

## FlattenedEntries

```
public ImmutableList<UnbondingEntry> FlattenedEntries { get; }
```

Property Value

[ImmutableList](#)<[UnbondingEntry](#)>

## IsEmpty

```
public bool IsEmpty { get; }
```

Property Value

[bool](#) ↗

## IsFull

```
public bool IsFull { get; }
```

Property Value

[bool](#) ↗

## LowestExpireHeight

```
public long LowestExpireHeight { get; }
```

Property Value

[long](#) ↗

## MaxEntries

```
public int MaxEntries { get; }
```

Property Value

[int](#) ↗

## Methods

## Equals(UnbondLockIn?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(UnbondLockIn? other)
```

Parameters

`other` [UnbondLockIn](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the `other` parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

Parameters

`obj` [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Release(long)

```
public UnbondLockIn Release(long height)
```

Parameters

height [long](#)

Returns

[UnbondLockIn](#)

## Slash(BigInteger, long, long, out FungibleAssetValue?)

```
public UnbondLockIn Slash(BigInteger slashFactor, long infractionHeight, long height, out FungibleAssetValue? slashedFAV)
```

Parameters

slashFactor [BigInteger](#)

infractionHeight [long](#)

height [long](#)

slashedFAV [FungibleAssetValue?](#)

Returns

[UnbondLockIn](#)



# Class UnbondingEntry

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public class UnbondingEntry : IBencodable, IEquatable<UnbondingEntry>
```

## Inheritance

[object](#) ← UnbondingEntry

## Implements

IBencodable, [IEquatable](#)<[UnbondingEntry](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnbondingEntry(IValue)

```
public UnbondingEntry(IValue bencoded)
```

#### Parameters

bencoded IValue

### UnbondingEntry(Address, FungibleAssetValue, FungibleAssetValue, long, long)

```
public UnbondingEntry(Address unbondeeAddress, FungibleAssetValue  
initialUnbondingFAV, FungibleAssetValue unbondingFAV, long creationHeight,  
long expireHeight)
```

#### Parameters

`unbondeeAddress` Address

`initialUnbondingFAV` FungibleAssetValue

`unbondingFAV` FungibleAssetValue

`creationHeight` [long](#)

`expireHeight` [long](#)

## UnbondingEntry(Address, FungibleAssetValue, long, long)

```
public UnbondingEntry(Address unbondeeAddress, FungibleAssetValue unbondingFAV, long  
creationHeight, long expireHeight)
```

### Parameters

`unbondeeAddress` Address

`unbondingFAV` FungibleAssetValue

`creationHeight` [long](#)

`expireHeight` [long](#)

## Properties

### Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

### CreationHeight

```
public long CreationHeight { get; }
```

Property Value

[long](#) ↗

## ExpireHeight

```
public long ExpireHeight { get; }
```

Property Value

[long](#) ↗

## InitialUnbondingFAV

```
public FungibleAssetValue InitialUnbondingFAV { get; }
```

Property Value

FungibleAssetValue

## UnbondeeAddress

```
public Address UnbondeeAddress { get; }
```

Property Value

Address

## UnbondingFAV

```
public FungibleAssetValue UnbondingFAV { get; }
```

## Property Value

FungibleAssetValue

## Methods

### Equals(UnbondingEntry?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(UnbondingEntry? other)
```

#### Parameters

**other** [UnbondingEntry](#)

An object to compare with this object.

#### Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

### Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

#### Parameters

**obj** [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Slash(BigInteger, long, out FungibleAssetValue)

```
public UnbondingEntry Slash(BigInteger slashFactor, long infractionHeight, out  
FungibleAssetValue slashedFAV)
```

Parameters

slashFactor [BigInteger](#)

infractionHeight [long](#)

slashedFAV [FungibleAssetValue](#)

Returns

[UnbondingEntry](#)

# Class UnbondingEntry.Comparer

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public class UnbondingEntry.Comparer : IComparer<UnbondingEntry>
```

## Inheritance

[object](#) ← UnbondingEntry.Comparer

## Implements

[IComparer](#)<[UnbondingEntry](#)>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Compare(UnbondingEntry?, UnbondingEntry?)

Compares two objects and returns a value indicating whether one is less than, equal to, or greater than the other.

```
public int Compare(UnbondingEntry? x, UnbondingEntry? y)
```

#### Parameters

x [UnbondingEntry](#)

The first object to compare.

y [UnbondingEntry](#)

The second object to compare.

#### Returns

A signed integer that indicates the relative values of  $x$  and  $y$ , as shown in the following table.

Value	Meaning
<b>Less than zero</b>	$x$ is less than $y$ .
<b>Zero</b>	$x$ equals $y$ .
<b>Greater than zero</b>	$x$ is greater than $y$ .

# Class UnbondingFactory

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public static class UnbondingFactory
```

## Inheritance

[object](#) ← UnbondingFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetUnbondingFromRef(IValue, IDelegationRepository)

```
public static IUnbonding GetUnbondingFromRef(IValue bencoded,  
IDelegationRepository repository)
```

#### Parameters

bencoded IValue

repository [IDelegationRepository](#)

#### Returns

[IUnbonding](#)

### GetUnbondingFromRef(UnbondingRef, IDelegationRepository)

```
public static IUnbonding GetUnbondingFromRef(UnbondingRef reference,  
IDelegationRepository repository)
```

## Parameters

reference [UnbondingRef](#)

repository [IDelegationRepository](#)

## Returns

[IUnbonding](#)

## ToReference(IUnbonding)

```
public static UnbondingRef ToReference(IUnbonding unbonding)
```

## Parameters

unbonding [IUnbonding](#)

## Returns

[UnbondingRef](#)

# Class UnbondingRef

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public class UnbondingRef : IEquatable<UnbondingRef>, IComparable<UnbondingRef>,  
IComparable, IBencodable
```

## Inheritance

[object](#) ← UnbondingRef

## Implements

[IEquatable](#)<[UnbondingRef](#)>, [IComparable](#)<[UnbondingRef](#)>, [IComparable](#), [IBencodable](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnbondingRef(IValue)

```
public UnbondingRef(IValue bencoded)
```

#### Parameters

bencoded IValue

### UnbondingRef(List)

```
public UnbondingRef(List list)
```

#### Parameters

list List

# UnbondingRef(Address, UnbondingType)

```
public UnbondingRef(Address address, UnbondingType unbondingType)
```

## Parameters

address Address

unbondingType [UnbondingType](#)

## Properties

### Address

```
public Address Address { get; }
```

### Property Value

Address

### Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

### UnbondingType

```
public UnbondingType UnbondingType { get; }
```

### Property Value

[UnbondingType](#)

# Methods

## CompareTo(UnbondingRef?)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(UnbondingRef? other)
```

### Parameters

**other** [UnbondingRef](#)

An object to compare with this instance.

### Returns

[int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <b>other</b> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <b>other</b> .
<b>Greater than zero</b>	This instance follows <b>other</b> in the sort order.

## CompareTo(object?)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(object? obj)
```

## Parameters

### obj [object](#)

An object to compare with this instance.

## Returns

### [int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <a href="#">obj</a> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <a href="#">obj</a> .
<b>Greater than zero</b>	This instance follows <a href="#">obj</a> in the sort order.

## Exceptions

### [ArgumentException](#)

[obj](#) is not the same type as this instance.

## Equals(UnbondingRef?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(UnbondingRef? other)
```

## Parameters

### other [UnbondingRef](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

Parameters

[obj](#) [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

# Operators

## operator ==(UnbondingRef?, UnbondingRef?)

```
public static bool operator ==(UnbondingRef? left, UnbondingRef? right)
```

### Parameters

left [UnbondingRef](#)

right [UnbondingRef](#)

### Returns

[bool](#) ↗

## operator >(UnbondingRef, UnbondingRef)

```
public static bool operator >(UnbondingRef left, UnbondingRef right)
```

### Parameters

left [UnbondingRef](#)

right [UnbondingRef](#)

### Returns

[bool](#) ↗

## operator >=(UnbondingRef, UnbondingRef)

```
public static bool operator >=(UnbondingRef left, UnbondingRef right)
```

### Parameters

left [UnbondingRef](#)

`right` [UnbondingRef](#)

Returns

[bool](#) ↗

**operator !=(UnbondingRef?, UnbondingRef?)**

```
public static bool operator !=(UnbondingRef? left, UnbondingRef? right)
```

Parameters

`left` [UnbondingRef](#)

`right` [UnbondingRef](#)

Returns

[bool](#) ↗

**operator <(UnbondingRef, UnbondingRef)**

```
public static bool operator <(UnbondingRef left, UnbondingRef right)
```

Parameters

`left` [UnbondingRef](#)

`right` [UnbondingRef](#)

Returns

[bool](#) ↗

**operator <=(UnbondingRef, UnbondingRef)**

```
public static bool operator <=(UnbondingRef left, UnbondingRef right)
```

Parameters

left [UnbondingRef](#)

right [UnbondingRef](#)

Returns

[bool](#)

# Class UnbondingSet

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public sealed class UnbondingSet : IBencodable
```

## Inheritance

[object](#) ← UnbondingSet

## Implements

IBencodable

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnbondingSet(IValue, IDelegationRepository)

```
public UnbondingSet(IValue bencoded, IDelegationRepository repository)
```

#### Parameters

bencoded IValue

repository [IDelegationRepository](#)

### UnbondingSet(List, IDelegationRepository)

```
public UnbondingSet(List bencoded, IDelegationRepository repository)
```

#### Parameters

bencoded List

repository [IDelegationRepository](#)

## UnbondingSet(IDelegationRepository)

```
public UnbondingSet(IDelegationRepository repository)
```

### Parameters

repository [IDelegationRepository](#)

## Properties

### Address

```
public static Address Address { get; }
```

### Property Value

Address

### Bencoded

```
public List Bencoded { get; }
```

### Property Value

List

### FlattenedUnbondingRefs

```
public ImmutableList<UnbondingRef> FlattenedUnbondingRefs { get; }
```

### Property Value

## [ImmutableArray](#) <[UnbondingRef](#)>

### IsEmpty

```
public bool IsEmpty { get; }
```

Property Value

[bool](#)

### Repository

```
public IDelegationRepository Repository { get; }
```

Property Value

[IDelegationRepository](#)

### UnbondingRefs

```
public ImmutableSortedDictionary<long, ImmutableSortedSet<UnbondingRef>>
UnbondingRefs { get; }
```

Property Value

[ImmutableSortedDictionary](#) <[long](#), [ImmutableSortedSet](#) <[UnbondingRef](#)>>

### Methods

#### SetUnbonding(IUnbonding)

```
public UnbondingSet SetUnbonding(IUnbonding unbonding)
```

Parameters

unbonding [IUnbonding](#).

Returns

[UnbondingSet](#)

## SetUnbondings(IEnumerable<IUnbonding>)

```
public UnbondingSet SetUnbondings(IEnumerable<IUnbonding> unbondings)
```

Parameters

unbondings [IEnumerable](#)<[IUnbonding](#)>

Returns

[UnbondingSet](#)

## UnbondingsToRelease(long)

```
public ImmutableArray<IUnbonding> UnbondingsToRelease(long height)
```

Parameters

height [long](#)

Returns

[ImmutableArray](#)<[IUnbonding](#)>

# Enum UnbondingType

Namespace: [Nekoyume.Delegation](#)

Assembly: Lib9c.dll

```
public enum UnbondingType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

RebondGrace = 1

UnbondLockIn = 0

# Namespace Nekoyume.Exceptions

## Classes

[AlreadyFoughtAvatarException](#)

[BencodexTypesExtensions](#)

[InvalidActionFieldException](#)

[InvalidMaterialCountException](#)

[ItemNotFoundException](#)

[MedalIdNotFoundException](#)

[NotEnoughEventDungeonTicketsException](#)

[NotEnoughItemException](#)

[PetCostNotFoundException](#)

[StateNullException](#)

[UnsupportedStateException](#)

# Class AlreadyFoughtAvatarException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AlreadyFoughtAvatarException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AlreadyFoughtAvatarException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AlreadyFoughtAvatarException()

```
public AlreadyFoughtAvatarException()
```

### AlreadyFoughtAvatarException(string)

```
public AlreadyFoughtAvatarException(string message)
```

## Parameters

message [string](#)



# Class BencodexTypesExtensions

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
public static class BencodexTypesExtensions
```

## Inheritance

[object](#) ← BencodexTypesExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Replace(List, int, IValue)

Returns a new list with the value replaced at the specified index.

```
public static List Replace(this List list, int index, IValue value)
```

#### Parameters

**list** List

**index** [int](#)

**value** IValue

#### Returns

List

#### Exceptions

[ArgumentOutOfRangeException](#)



# Class InvalidActionFieldException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidActionFieldException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidActionFieldException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidActionFieldException(SerializationInfo, StreamingContext)

```
protected InvalidActionFieldException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidOperationException(string?)

```
public InvalidOperationException(string? message = null)
```

### Parameters

message [string](#)

## InvalidOperationException(string?, Exception?)

```
public InvalidOperationException(string? message = null, Exception? innerException = null)
```

### Parameters

message [string](#)

innerException [Exception](#)

## InvalidOperationException(string, string, string, string, Exception?)

```
public InvalidOperationException(string actionType, string addressesHex, string fieldName, string message, Exception? innerException = null)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

fieldName [string](#)

message [string](#)

innerException [Exception](#)

# Class InvalidMaterialCountException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidMaterialCountException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidMaterialCountException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidMaterialCountException(string)

```
public InvalidMaterialCountException(string message)
```

#### Parameters

message [string](#)

### InvalidMaterialCountException(string, string, int, int)

```
public InvalidMaterialCountException(string actionType, string addressesHex, int required, int playerHas)
```

## Parameters

actionType [string](#)

addressesHex [string](#)

required [int](#)

playerHas [int](#)

# Class ItemNotFoundException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ItemNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ItemNotFoundException()

```
public ItemNotFoundException()
```

### ItemNotFoundException(Address, HashDigest<SHA256>, Exception?)

```
public ItemNotFoundException(Address inventoryAddr, HashDigest<SHA256> fungibleId,  
Exception? innerException = null)
```

## Parameters

`inventoryAddr` Address  
`fungibleId` HashDigest<[SHA256](#)>  
`innerException` [Exception](#)

## ItemNotFoundException(Address, Guid, Exception?)

```
public ItemNotFoundException(Address inventoryAddr, Guid nonFungibleId, Exception?  
innerException = null)
```

## Parameters

`inventoryAddr` Address  
`nonFungibleId` [Guid](#)  
`innerException` [Exception](#)

## ItemNotFoundException(SerializationInfo, StreamingContext)

```
protected ItemNotFoundException(SerializationInfo info, StreamingContext context)
```

## Parameters

`info` [SerializationInfo](#)  
`context` [StreamingContext](#)

## ItemNotFoundException(string?)

```
public ItemNotFoundException(string? message)
```

Parameters

message [string](#)

## ItemNotFoundException(string?, Exception?)

```
public ItemNotFoundException(string? message, Exception? innerException)
```

Parameters

message [string](#)

innerException [Exception](#)

# Class MedalIdNotFoundException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MedalIdNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← `MedalIdNotFoundException`

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### `MedalIdNotFoundException()`

```
public MedalIdNotFoundException()
```

### `MedalIdNotFoundException(SerializationInfo, StreamingContext)`

```
protected MedalIdNotFoundException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MedalIdNotFoundException(string)

```
public MedalIdNotFoundException(string message)
```

Parameters

message [string](#)

## MedalIdNotFoundException(string, Exception)

```
public MedalIdNotFoundException(string message, Exception innerException)
```

Parameters

message [string](#)

innerException [Exception](#)

## MedalIdNotFoundException(string, string, int, int)

```
public MedalIdNotFoundException(string actionType, string addressesHex, int requiredAmount, int currentAmount)
```

Parameters

actionType [string](#)

addressesHex [string](#)

requiredAmount [int](#)

currentAmount [int](#)

# Class NotEnoughEventDungeonTicketsException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughEventDungeonTicketsException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughEventDungeonTicketsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughEventDungeonTicketsException()

```
public NotEnoughEventDungeonTicketsException()
```

### NotEnoughEventDungeonTicketsException(SerializationInfo, StreamingContext)

```
protected NotEnoughEventDungeonTicketsException(SerializationInfo info,
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughEventDungeonTicketsException(string)

```
public NotEnoughEventDungeonTicketsException(string message)
```

Parameters

message [string](#)

## NotEnoughEventDungeonTicketsException(string, Exception)

```
public NotEnoughEventDungeonTicketsException(string message,  
Exception innerException)
```

Parameters

message [string](#)

innerException [Exception](#)

## NotEnoughEventDungeonTicketsException(string, string, int, int)

```
public NotEnoughEventDungeonTicketsException(string actionType, string addressesHex,  
int requiredAmount, int currentAmount)
```

Parameters

actionType [string](#)

addressesHex [string](#)

requiredAmount [int](#)

currentAmount [int](#)

# Class NotEnoughItemException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughItemException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughItemException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughItemException()

```
public NotEnoughItemException()
```

### NotEnoughItemException(Address, HashDigest<SHA256>, int, int, Exception?)

```
public NotEnoughItemException(Address inventoryAddr, HashDigest<SHA256> fungibleId,  
int expectedCount, int actualCount, Exception? innerException = null)
```

Parameters

`inventoryAddr` Address

`fungibleId` HashDigest<[SHA256](#)>

`expectedCount` [int](#)

`actualCount` [int](#)

`innerException` [Exception](#)

## NotEnoughItemException(SerializationInfo, StreamingContext)

`protected NotEnoughItemException(SerializationInfo info, StreamingContext context)`

Parameters

`info` [SerializationInfo](#)

`context` [StreamingContext](#)

## NotEnoughItemException(string?)

`public NotEnoughItemException(string? message)`

Parameters

`message` [string](#)

## NotEnoughItemException(string?, Exception?)

`public NotEnoughItemException(string? message, Exception? innerException)`

Parameters

message [string](#)

innerException [Exception](#)

# Class PetCostNotFoundException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetCostNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← PetCostNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PetCostNotFoundException(string, Exception)

```
public PetCostNotFoundException(string message, Exception innerException = null)
```

## Parameters

message [string](#)

innerException [Exception](#)

## PetCostNotFoundException(string, string, string, Exception)

```
public PetCostNotFoundException(string actionType, string addressesHex, string message, Exception innerException = null)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

message [string](#)

innerException [Exception](#)

# Class StateNullException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StateNullException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← StateNullException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### StateNullException()

```
public StateNullException()
```

### StateNullException(Address, Address, Exception?)

```
public StateNullException(Address accountAddress, Address address, Exception?
innerException = null)
```

## Parameters

accountAddress Address

address Address

innerException [Exception](#)

## StateNullException(SerializationInfo, StreamingContext)

`protected StateNullException(SerializationInfo info, StreamingContext context)`

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## StateNullException(string?)

`public StateNullException(string? message)`

Parameters

message [string](#)

## StateNullException(string?, Exception?)

`public StateNullException(string? message, Exception? innerException)`

Parameters

message [string](#)

innerException [Exception](#)

# Class UnsupportedStateException

Namespace: [Nekoyume.Exceptions](#)

Assembly: Lib9c.dll

```
[Serializable]
public class UnsupportedStateException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← UnsupportedStateException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### UnsupportedStateException(int, int, Exception)

```
public UnsupportedStateException(int expectedVersion, int actualVersion, Exception  
innerException = null)
```

## Parameters

expectedVersion [int](#)

actualVersion [int](#)

innerException [Exception](#)

## UnsupportedStateException(SerializationInfo, StreamingContext)

```
protected UnsupportedStateException(SerializationInfo info,  
StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## UnsupportedStateException(string)

```
public UnsupportedStateException(string msg)
```

### Parameters

msg [string](#)

## UnsupportedStateException(string, int, string, int, Exception)

```
public UnsupportedStateException(string expectedName, int expectedVersion, string  
actualName, int actualVersion, Exception innerException = null)
```

### Parameters

expectedName [string](#)

expectedVersion [int](#)

actualName [string](#)

actualVersion [int](#)

innerException [Exception](#)



# Namespace Nekoyume.Extensions

## Classes

[ActionContextExtensions](#)

[AgentAddressExtensions](#)

[CombinationSlotStateExtensions](#)

[EquipmentExtensions](#)

[EventDungeonExtensions](#)

[EventRecipeExtensions](#)

[EventScheduleExtensions](#)

[IntegerExtensions](#)

[MaterialItemSheetExtensions](#)

[RandomExtensions](#)

[SheetsExtensions](#)

[WorldExtensions](#)

[WorldInformationExtensions](#)

# Class ActionContextExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class ActionContextExtensions
```

## Inheritance

[object](#) ← ActionContextExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetAgentAddress(IActionContext)

```
public static AgentAddress GetAgentAddress(this IActionContext context)
```

#### Parameters

context IActionContext

#### Returns

[AgentAddress](#)

# Class AgentAddressExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class AgentAddressExtensions
```

## Inheritance

[object](#) ← AgentAddressExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetPledgeAddress(AgentAddress)

```
public static PledgeAddress GetPledgeAddress(this AgentAddress address)
```

#### Parameters

address [AgentAddress](#)

#### Returns

[PledgeAddress](#)

# Class CombinationSlotStateExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class CombinationSlotStateExtensions
```

## Inheritance

[object](#) ← CombinationSlotStateExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

**TryGetMail(CombinationSlotState, long, long, out CombinationMail, out ItemEnhanceMail)**

```
public static bool TryGetMail(this CombinationSlotState state, long  
blockIndex, long requiredBlockIndex, out CombinationMail combinationMail, out  
ItemEnhanceMail itemEnhanceMail)
```

## Parameters

state [CombinationSlotState](#)

blockIndex [long](#)

requiredBlockIndex [long](#)

combinationMail [CombinationMail](#)

itemEnhanceMail [ItemEnhanceMail](#)

## Returns

[bool](#)

## TryGetResultId(CombinationSlotState, out Guid)

```
public static bool TryGetResultId(this CombinationSlotState state, out  
Guid resultId)
```

### Parameters

state [CombinationSlotState](#)

resultId [Guid](#)

### Returns

[bool](#)

## ValidateFromAction(CombinationSlotState, long, AvatarState, int, string, string)

```
public static void ValidateFromAction(this CombinationSlotState slotState,  
long blockIndex, AvatarState avatarState, int slotIndex, string actionTypeText,  
string addressesHex)
```

### Parameters

slotState [CombinationSlotState](#)

blockIndex [long](#)

avatarState [AvatarState](#)

slotIndex [int](#)

actionTypeText [string](#)

addressesHex [string](#)

# Class EquipmentExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class EquipmentExtensions
```

## Inheritance

[object](#) ← EquipmentExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

GetRequirementLevel(Equipment,  
ItemRequirementSheet, EquipmentItemRecipeSheet,  
EquipmentItemSubRecipeSheetV2,  
EquipmentItemOptionSheet)

```
public static int GetRequirementLevel(this Equipment equipment,  
ItemRequirementSheet requirementSheet, EquipmentItemRecipeSheet recipeSheet,  
EquipmentItemSubRecipeSheetV2 subRecipeSheet, EquipmentItemOptionSheet  
itemOptionSheet)
```

## Parameters

equipment [Equipment](#)

requirementSheet [ItemRequirementSheet](#)

recipeSheet [EquipmentItemRecipeSheet](#)

subRecipeSheet [EquipmentItemSubRecipeSheetV2](#)

itemOptionSheet [EquipmentItemOptionSheet](#)

Returns

[int](#)

**IsMadeWithMimisbrunnrRecipe(Equipment,  
EquipmentItemRecipeSheet,  
EquipmentItemSubRecipeSheetV2,  
EquipmentItemOptionSheet)**

```
public static bool IsMadeWithMimisbrunnrRecipe(this Equipment equipment,  
EquipmentItemRecipeSheet recipeSheet, EquipmentItemSubRecipeSheetV2 subRecipeSheet,  
EquipmentItemOptionSheet itemOptionSheet)
```

Parameters

equipment [Equipment](#)

recipeSheet [EquipmentItemRecipeSheet](#)

subRecipeSheet [EquipmentItemSubRecipeSheetV2](#)

itemOptionSheet [EquipmentItemOptionSheet](#)

Returns

[bool](#)

# Class EventDungeonExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class EventDungeonExtensions
```

## Inheritance

[object](#) ← EventDungeonExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetRemainingTicketsConsiderReset(EventDungeonInfo, Row, long)

```
public static int GetRemainingTicketsConsiderReset(this EventDungeonInfo info,  
EventScheduleSheet.Row eventScheduleRow, long currentBlockIndex)
```

#### Parameters

info [EventDungeonInfo](#)

eventScheduleRow [EventScheduleSheet.Row](#)

currentBlockIndex [long](#)

#### Returns

[int](#)

### GetStageNumber(Row)

```
public static int GetStageNumber(this EventDungeonStageSheet.Row row)
```

Parameters

row [EventDungeonStageSheet.Row](#)

Returns

[int](#)

## GetStageRows(EventDungeonStageSheet, int, int)

```
public static List<EventDungeonStageSheet.Row> GetStageRows(this  
EventDungeonStageSheet sheet, int beginningStageId, int endStageId)
```

Parameters

sheet [EventDungeonStageSheet](#)

beginningStageId [int](#)

endStageId [int](#)

Returns

[List](#)<[EventDungeonStageSheet.Row](#)>

## GetStageWaveRows(EventDungeonStageWaveSheet, int, int)

```
public static List<EventDungeonStageWaveSheet.Row> GetStageWaveRows(this  
EventDungeonStageWaveSheet sheet, int beginningStageId, int endStageId)
```

Parameters

sheet [EventDungeonStageWaveSheet](#)

`beginningStageId` [int](#)

`endStageId` [int](#)

Returns

[List](#) <[EventDungeonStageWaveSheet.Row](#)>

## ToEventDungeonStageNumber(int)

`public static int ToEventDungeonStageNumber(this int eventDungeonStageId)`

Parameters

`eventDungeonStageId` [int](#)

Returns

[int](#)

## TryGetRowByEventDungeonStageId(EventDungeonSheet, int, out Row)

`public static bool TryGetRowByEventDungeonStageId(this EventDungeonSheet sheet, int eventDungeonStageId, out EventDungeonSheet.Row row)`

Parameters

`sheet` [EventDungeonSheet](#)

`eventDungeonStageId` [int](#)

`row` [EventDungeonSheet.Row](#)

Returns

[bool](#)

## TryGetRowByEventScheduleId(EventDungeonSheet, int, out Row)

```
public static bool TryGetRowByEventScheduleId(this EventDungeonSheet sheet, int eventScheduleId, out EventDungeonSheet.Row row)
```

### Parameters

sheet [EventDungeonSheet](#)

eventScheduleId [int](#)

row [EventDungeonSheet.Row](#)

### Returns

[bool](#)

## ValidateFromAction(EventDungeonSheet, int, int, string, string)

```
public static EventDungeonSheet.Row ValidateFromAction(this EventDungeonSheet sheet, int eventDungeonId, int eventDungeonStageId, string actionTypeText, string addressesHex)
```

### Parameters

sheet [EventDungeonSheet](#)

eventDungeonId [int](#)

eventDungeonStageId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

### Returns

[EventDungeonSheet.Row](#)

## ValidateFromAction(EventDungeonStageSheet, int, string, string)

```
public static EventDungeonStageSheet.Row ValidateFromAction(this  
EventDungeonStageSheet sheet, int eventDungeonStageId, string actionTypeText,  
string addressesHex)
```

### Parameters

sheet [EventDungeonStageSheet](#)

eventDungeonStageId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

### Returns

[EventDungeonStageSheet.Row](#)

# Class EventRecipeExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class EventRecipeExtensions
```

## Inheritance

[object](#) ← EventRecipeExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetRecipeRows(EventConsumableItemRecipeSheet, int)

```
public static List<EventConsumableItemRecipeSheet.Row> GetRecipeRows(this  
EventConsumableItemRecipeSheet sheet, int eventScheduleId)
```

#### Parameters

sheet [EventConsumableItemRecipeSheet](#)

eventScheduleId [int](#)

#### Returns

[List](#)<[EventConsumableItemRecipeSheet.Row](#)>

### GetRecipeRows(EventMaterialItemRecipeSheet, int)

```
public static List<EventMaterialItemRecipeSheet.Row> GetRecipeRows(this  
EventMaterialItemRecipeSheet sheet, int eventScheduleId)
```

## Parameters

sheet [EventMaterialItemRecipeSheet](#)

eventScheduleId [int](#)

## Returns

[List](#)<[EventMaterialItemRecipeSheet.Row](#)>

## ValidateFromAction(EventConsumableItemRecipeSheet, int, string, string)

```
public static EventConsumableItemRecipeSheet.Row ValidateFromAction(this  
EventConsumableItemRecipeSheet sheet, int eventConsumableItemId, string  
actionTypeText, string addressesHex)
```

## Parameters

sheet [EventConsumableItemRecipeSheet](#)

eventConsumableItemId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

## Returns

[EventConsumableItemRecipeSheet.Row](#)

## ValidateFromAction(EventMaterialItemRecipeSheet, int, string, string)

```
public static EventMaterialItemRecipeSheet.Row ValidateFromAction(this  
EventMaterialItemRecipeSheet sheet, int eventMaterialItemId, string  
actionTypeText, string addressesHex)
```

## Parameters

sheet [EventMaterialItemRecipeSheet](#)

eventMaterialItemRecipeId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

## Returns

[EventMaterialItemRecipeSheet.Row](#)

## ValidateFromAction(Row, MaterialItemSheet, Dictionary<int, int>, string, string)

```
public static void ValidateFromAction(this EventMaterialItemRecipeSheet.Row  
recipeRow, MaterialItemSheet materialItemSheet, Dictionary<int, int> materialsToUse,  
string actionTypeText, string addressesHex)
```

## Parameters

recipeRow [EventMaterialItemRecipeSheet.Row](#)

materialItemSheet [MaterialItemSheet](#)

materialsToUse [Dictionary](#)<[int](#), [int](#)>

actionTypeText [string](#)

addressesHex [string](#)

# Class EventScheduleExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class EventScheduleExtensions
```

## Inheritance

[object](#) ← EventScheduleExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetDungeonTicketCost(Row, int, Currency)

```
public static FungibleAssetValue GetDungeonTicketCost(this EventScheduleSheet.Row  
row, int numberOfTicketPurchases, Currency currency)
```

#### Parameters

row [EventScheduleSheet.Row](#)

numberOfTicketPurchases [int](#)

currency Currency

#### Returns

FungibleAssetValue

### GetDungeonTicketCostV1(Row, int)

```
[Obsolete("Use `GetDungeonTicketCost()` instead.")]
public static long GetDungeonTicketCostV1(this EventScheduleSheet.Row row,
int numberOfTicketPurchases)
```

## Parameters

row [EventScheduleSheet.Row](#)

numberOfTicketPurchases [int](#)

## Returns

[long](#)

## GetStageExp(Row, int, int)

```
public static int GetStageExp(this EventScheduleSheet.Row row, int stageNumber, int
multiplier = 1)
```

## Parameters

row [EventScheduleSheet.Row](#)

stageNumber [int](#)

multiplier [int](#)

## Returns

[int](#)

## ToEventScheduleId(int)

```
public static int ToEventScheduleId(this int eventDungeonOrRecipeId)
```

## Parameters

eventDungeonOrRecipeId [int](#)

Returns

[int](#)

## TryGetRowForDungeon(EventScheduleSheet, long, out Row)

```
public static bool TryGetRowForDungeon(this EventScheduleSheet sheet, long  
blockIndex, out EventScheduleSheet.Row row)
```

Parameters

sheet [EventScheduleSheet](#)

blockIndex [long](#)

row [EventScheduleSheet.Row](#)

Returns

[bool](#)

## TryGetRowForRecipe(EventScheduleSheet, long, out Row)

```
public static bool TryGetRowForRecipe(this EventScheduleSheet sheet, long  
blockIndex, out EventScheduleSheet.Row row)
```

Parameters

sheet [EventScheduleSheet](#)

blockIndex [long](#)

row [EventScheduleSheet.Row](#)

Returns

[bool](#)

## ValidateFromActionForDungeon(EventScheduleSheet, long, int, int, string, string)

```
public static EventScheduleSheet.Row ValidateFromActionForDungeon(this  
EventScheduleSheet scheduleSheet, long blockIndex, int eventScheduleId, int  
eventDungeonId, string actionTypeText, string addressesHex)
```

Parameters

scheduleSheet [EventScheduleSheet](#)

blockIndex [long](#)

eventScheduleId [int](#)

eventDungeonId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

Returns

[EventScheduleSheet.Row](#)

## ValidateFromActionForRecipe(EventScheduleSheet, long, int, int, string, string)

```
public static EventScheduleSheet.Row ValidateFromActionForRecipe(this  
EventScheduleSheet scheduleSheet, long blockIndex, int eventScheduleId, int  
eventRecipeId, string actionTypeText, string addressesHex)
```

Parameters

scheduleSheet [EventScheduleSheet](#)

blockIndex [long](#)

eventScheduleId [int](#)

eventRecipeId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

Returns

[EventScheduleSheet.Row](#)

# Class IntegerExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class IntegerExtensions
```

## Inheritance

[object](#) ← IntegerExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### NormalizeFromTenThousandths(int)

```
public static decimal NormalizeFromTenThousandths(this int value)
```

#### Parameters

value [int](#)

#### Returns

[decimal](#)

# Class MaterialItemSheetExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class MaterialItemSheetExtensions
```

## Inheritance

[object](#) ← MaterialItemSheetExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

**ValidateFromAction(MaterialItemSheet,  
List<MaterialInfo>, Dictionary<int, int>, string)**

```
public static void ValidateFromAction(this MaterialItemSheet materialItemSheet,  
List<EquipmentItemSubRecipeSheet.MaterialInfo> materialInfos, Dictionary<int, int>  
requiredFungibleItems, string addressesHex)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

materialInfos [List](#)<[EquipmentItemSubRecipeSheet.MaterialInfo](#)>

requiredFungibleItems [Dictionary](#)<[int](#), [int](#)>

addressesHex [string](#)

# Class RandomExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class RandomExtensions
```

## Inheritance

[object](#) ← RandomExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GenerateAddress(IRandom)

```
public static Address GenerateAddress(this IRandom random)
```

#### Parameters

random IRandom

#### Returns

Address

# Class SheetsExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class SheetsExtensions
```

## Inheritance

[object](#) ← SheetsExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

**FindLevelByStakedAmount(IStakeRewardSheet,  
Address, FungibleAssetValue)**

```
public static int FindLevelByStakedAmount(this IStakeRewardSheet sheet, Address  
agentAddress, FungibleAssetValue balance)
```

### Parameters

sheet [IStakeRewardSheet](#)

agentAddress Address

balance FungibleAssetValue

### Returns

[int](#)

## FindPreviousRaidIdByBlockIndex(WorldBossListSheet, long)

```
public static int FindPreviousRaidIdByBlockIndex(this WorldBossListSheet sheet,  
long blockIndex)
```

### Parameters

sheet [WorldBossListSheet](#)

blockIndex [long](#)

### Returns

[int](#)

## FindPreviousRowByBlockIndex(WorldBossListSheet, long)

```
public static WorldBossListSheet.Row FindPreviousRowByBlockIndex(this  
WorldBossListSheet sheet, long blockIndex)
```

### Parameters

sheet [WorldBossListSheet](#)

blockIndex [long](#)

### Returns

[WorldBossListSheet.Row](#)

## FindRaidIdByBlockIndex(WorldBossListSheet, long)

```
public static int FindRaidIdByBlockIndex(this WorldBossListSheet sheet,  
long blockIndex)
```

Parameters

sheet [WorldBossListSheet](#)

blockIndex [long](#)

Returns

[int](#)

## FindRowByBlockIndex(WorldBossListSheet, long)

```
public static WorldBossListSheet.Row FindRowByBlockIndex(this WorldBossListSheet  
sheet, long blockIndex)
```

Parameters

sheet [WorldBossListSheet](#)

blockIndex [long](#)

Returns

[WorldBossListSheet.Row](#)

## GetActionPointByStaking(StakeActionPointCoefficientSheet, int, int, int)

```
public static int GetActionPointByStaking(this StakeActionPointCoefficientSheet  
sheet, int originAp, int playCount, int level)
```

Parameters

sheet [StakeActionPointCoefficientSheet](#)

originAp [int](#)

playCount [int](#)

level [int](#)

Returns

[int](#)

## GetAddress(Dictionary<Type, (Address address, ISheet sheet)>, Type)

```
public static Address GetAddress(this Dictionary<Type, (Address address, ISheet sheet)> sheets, Type type)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

type [Type](#)

Returns

Address

## GetAddress<T>(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static Address GetAddress<T>(this Dictionary<Type, (Address address, ISheet sheet)> sheets) where T : ISheet
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

Address

Type Parameters

## GetArenaSimulatorSheets(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static ArenaSimulatorSheets GetArenaSimulatorSheets(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

### Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

### Returns

[ArenaSimulatorSheets](#)

## GetArenaSimulatorSheetsV1(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static ArenaSimulatorSheetsV1 GetArenaSimulatorSheetsV1(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

### Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

### Returns

[ArenaSimulatorSheetsV1](#)

## GetArenaSimulatorSheets\_v100291(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static ArenaSimulatorSheetsV1 GetArenaSimulatorSheets_v100291(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[ArenaSimulatorSheetsV1](#)

**GetAvatarSheets(Dictionary<Type, (Address address, ISheet sheet)>)**

```
public static AvatarSheets GetAvatarSheets(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[AvatarSheets](#)

**GetItemSheet(Dictionary<Type, (Address address, ISheet sheet)>)**

```
public static ItemSheet GetItemSheet(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[ItemSheet](#)

## GetQuestSheet(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static QuestSheet GetQuestSheet(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

### Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

### Returns

[QuestSheet](#)

## GetRaidSimulatorSheets(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static RaidSimulatorSheets GetRaidSimulatorSheets(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

### Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

### Returns

[RaidSimulatorSheets](#)

## GetRaidSimulatorSheetsV1(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static RaidSimulatorSheetsV1 GetRaidSimulatorSheetsV1(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

### Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[RaidSimulatorSheetsV1](#)

**GetRankingSimulatorSheets(Dictionary<Type, (Address address, ISheet sheet)>)**

```
public static RankingSimulatorSheets GetRankingSimulatorSheets(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[RankingSimulatorSheets](#)

**GetRankingSimulatorSheetsV1(Dictionary<Type, (Address address, ISheet sheet)>)**

```
public static RankingSimulatorSheetsV1 GetRankingSimulatorSheetsV1(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[RankingSimulatorSheetsV1](#)

## GetRankingSimulatorSheetsV100291(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static RankingSimulatorSheetsV1 GetRankingSimulatorSheetsV100291(this  
Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

### Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

### Returns

[RankingSimulatorSheetsV1](#)

## GetSheet(Dictionary<Type, (Address address, ISheet sheet)>, Type)

```
public static ISheet GetSheet(this Dictionary<Type, (Address address, ISheet sheet)>  
sheets, Type sheetType)
```

### Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

sheetType [Type](#)

### Returns

[ISheet](#)

## GetSheet<T>(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static T GetSheet<T>(this Dictionary<Type, (Address address, ISheet sheet)>  
sheets) where T : ISheet
```

## Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

## Returns

T

## Type Parameters

T

## GetSimulatorSheets(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static SimulatorSheets GetSimulatorSheets(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

## Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

## Returns

[SimulatorSheets](#)

## GetSimulatorSheetsV1(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static SimulatorSheetsV1 GetSimulatorSheetsV1(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

## Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

## Returns

## [SimulatorSheetsV1](#)

### GetSimulatorSheetsV100291(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static SimulatorSheetsV1 GetSimulatorSheetsV100291(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

#### Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

#### Returns

[SimulatorSheetsV1](#)

### GetStageSimulatorSheets(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static StageSimulatorSheets GetStageSimulatorSheets(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

#### Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

#### Returns

[StageSimulatorSheets](#)

### GetStageSimulatorSheetsV1(Dictionary<Type, (Address address, ISheet sheet)>)

```
public static StageSimulatorSheetsV1 GetStageSimulatorSheetsV1(this Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[StageSimulatorSheetsV1](#)

**GetStageSimulatorSheetsV100291(Dictionary<Type, (Address address, ISheet sheet)>)**

```
public static StageSimulatorSheetsV1 GetStageSimulatorSheetsV100291(this  
Dictionary<Type, (Address address, ISheet sheet)> sheets)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

Returns

[StageSimulatorSheetsV1](#)

**TryGetAddress(Dictionary<Type, (Address address, ISheet sheet)>, Type, out Address)**

```
public static bool TryGetAddress(this Dictionary<Type, (Address address, ISheet  
sheet)> sheets, Type type, out Address address)
```

Parameters

sheets [Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

type [Type](#)

address Address

Returns

[bool](#)

## TryGetAddress<T>(Dictionary<Type, (Address address, ISheet sheet)>, out Address)

```
public static bool TryGetAddress<T>(this Dictionary<Type, (Address address, ISheet sheet)> sheets, out Address address) where T : ISheet
```

### Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

address Address

### Returns

[bool](#)

## Type Parameters

T

## TryGetSheet(Dictionary<Type, (Address address, ISheet sheet)>, Type, out ISheet)

```
public static bool TryGetSheet(this Dictionary<Type, (Address address, ISheet sheet)> sheets, Type type, out ISheet sheet)
```

### Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

type [Type](#)

sheet [ISheet](#)

### Returns

[bool](#)

TryGetSheet<T>(Dictionary<Type, (Address address, ISheet sheet)>, out T)

```
public static bool TryGetSheet<T>(this Dictionary<Type, (Address address, ISheet sheet)> sheets, out T sheet) where T : ISheet
```

Parameters

sheets [Dictionary](#)<[Type](#)>, (Address [address](#), [ISheet](#) [sheet](#))>

sheet T

Returns

[bool](#)

Type Parameters

T

# Class WorldExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class WorldExtensions
```

## Inheritance

[object](#) ← WorldExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

MutateAccount(IWorld, Address, Func<IAccount, IAccount>)

```
public static IWorld MutateAccount(this IWorld world, Address accountAddress,  
Func<IAccount, IAccount> mutateFn)
```

### Parameters

**world** IWorld

**accountAddress** Address

**mutateFn** [Func](#)<IAccount, IAccount>

### Returns

IWorld

# Class WorldInformationExtensions

Namespace: [Nekoyume.Extensions](#)

Assembly: Lib9c.dll

```
public static class WorldInformationExtensions
```

## Inheritance

[object](#) ← WorldInformationExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ValidateFromAction(WorldInformation, int, string, string)

```
public static void ValidateFromAction(this WorldInformation worldInformation, int  
stageId, string actionTypeText, string addressesHex)
```

## Parameters

worldInformation [WorldInformation](#)

stageId [int](#)

actionTypeText [string](#)

addressesHex [string](#)

# Namespace Nekoyume.Helper

## Classes

[AdventureBossHelper](#)

[AvatarStateExtensions](#)

[CrystalCalculator](#)

[CustomCraftHelper](#)

[InventoryExtensions](#)

[ItemOptionHelper](#)

[ItemOptionInfo](#)

[PetHelper](#)

[RuneHelper](#)

[WorldBossHelper](#)

# Class AdventureBossHelper

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class AdventureBossHelper
```

## Inheritance

[object](#) ← AdventureBossHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### FixedRewardRatio

```
public const decimal FixedRewardRatio = 0.7
```

Field Value

[decimal](#)

### RaffleRewardPercent

```
public const int RaffleRewardPercent = 5
```

Field Value

[int](#)

### RandomRewardRatio

```
public const decimal RandomRewardRatio = 0.3
```

Field Value

[decimal](#) ↴

## TotalRewardMultiplier

```
public const decimal TotalRewardMultiplier = 1.2
```

Field Value

[decimal](#) ↴

## Methods

### AddExploreRewards(IActionContext, IWorld, Address, Inventory, IEnumerable<RewardAmountData>)

```
public static IWorld AddExploreRewards(IActionContext context, IWorld states,  
Address avatarAddress, Inventory inventory,  
IEnumerable<AdventureBossSheet.RewardAmountData> rewardList)
```

Parameters

**context** IActionContext

**states** IWorld

**avatarAddress** Address

**inventory** [Inventory](#)

**rewardList** [IEnumerable](#) ↴<[AdventureBossSheet.RewardAmountData](#)>

Returns

## CalculateExploreReward(ClaimableReward, BountyBoard, ExploreBoard, Explorer, Address, AdventureBossNcgRewardRatioSheet, decimal, decimal, bool, out FungibleAssetValue)

Calculate reward for adventure boss explorers. This method only calculates reward for given avatar, not actually give rewards.

```
public static AdventureBossGameData.ClaimableReward  
CalculateExploreReward(AdventureBossGameData.ClaimableReward reward, BountyBoard  
bountyBoard, ExploreBoard exploreBoard, Explorer explorer, Address avatarAddress,  
AdventureBossNcgRewardRatioSheet sheet, decimal ncgApRatio, decimal ncgRuneRatio,  
bool isReal, out FungibleAssetValue ncgReward)
```

### Parameters

**reward** [AdventureBossGameData.ClaimableReward](#)

Claimable reward for this avatar so far.

**bountyBoard** [BountyBoard](#)

Bounty board for this season. NCG reward is based on totalBounty on this board.

**exploreBoard** [ExploreBoard](#)

Explore board for this season. Total reward amount is base on usedApPotion on this board.

**explorer** [Explorer](#)

Target explorer to calculate reward

**avatarAddress** [Address](#)

Target avatar address to calculate reward.

**sheet** [AdventureBossNcgRewardRatioSheet](#)

NCG to reward exchange ratio sheet. Calculate total reward amount based on this sheet.

**ncgApRatio** [decimal](#)

Exchange ratio between used AP potion to NCG. Used to set total reward amount.

**ncgRuneRatio** [decimal](#)

If a reward is rune, use this fixed ratio, not in sheet.

**isReal** [bool](#)

**ncgReward** FungibleAssetValue

out value: calculated NCG reward in this function. We must handle NCG reward separately because NCG reward must be transferred from each season's bounty address.

Returns

[AdventureBossGameData.ClivableReward](#)

Updated Claimable reward after calculation.

## CalculateRaffleReward(BountyBoard)

```
public static FungibleAssetValue CalculateRaffleReward(BountyBoard bountyBoard)
```

Parameters

**bountyBoard** [BountyBoard](#)

Returns

FungibleAssetValue

## CalculateWantedReward(ClivableReward, BountyBoard, Address, AdventureBossNcgRewardRatioSheet, decimal, out FungibleAssetValue)

Calculate reward for adventure boss operators. This method only calculates reward for given avatar, not actually give rewards.

```
public static AdventureBossGameData.ClaimableReward  
CalculateWantedReward(AdventureBossGameData.ClaimableReward reward, BountyBoard  
bountyBoard, Address avatarAddress, AdventureBossNcgRewardRatioSheet sheet, decimal  
ncgRuneRatio, out FungibleAssetValue ncgReward)
```

## Parameters

**reward** [AdventureBossGameData.ClaimableReward](#)

Claimable reward for this avatar so far.

**bountyBoard** [BountyBoard](#)

Bounty board for this season. All the reward amount is based on totalBounty on this board.

**avatarAddress** Address

Target avatar address to calculate reward.

**sheet** [AdventureBossNcgRewardRatioSheet](#)

NCG to reward exchange ratio sheet. Calculate total reward amount based on this sheet.

**ncgRuneRatio** [decimal](#)

If a reward is rune, use this fixed ratio, not in sheet.

**ncgReward** FungibleAssetValue

out value: calculated NCG reward in this function. We must handle NCG reward separately because NCG reward must be transferred from each season's bounty address.

## Returns

[AdventureBossGameData.ClaimableReward](#)

Updated Claimable reward after calculation.

CollectExploreReward(ClaimableReward,  
GameConfigState,  
AdventureBossNcgRewardRatioSheet, SeasonInfo,  
BountyBoard, ExploreBoard, Explorer, long, Address, ref  
ClaimableReward, out FungibleAssetValue)

```
public static bool CollectExploreReward(AdventureBossGameData.ClaimableReward  
reward, GameConfigState gameConfig, AdventureBossNcgRewardRatioSheet  
ncgRewardRatioSheet, SeasonInfo seasonInfo, BountyBoard bountyBoard,  
ExploreBoard exploreBoard, Explorer explorer, long currentBlockIndex, Address  
avatarAddress, ref AdventureBossGameData.ClaimableReward updatedReward, out  
FungibleAssetValue ncgReward)
```

## Parameters

reward [AdventureBossGameData.ClaimableReward](#)

gameConfig [GameConfigState](#)

ncgRewardRatioSheet [AdventureBossNcgRewardRatioSheet](#)

seasonInfo [SeasonInfo](#)

bountyBoard [BountyBoard](#)

exploreBoard [ExploreBoard](#)

explorer [Explorer](#)

currentBlockIndex [long](#)

avatarAddress [Address](#)

updatedReward [AdventureBossGameData.ClaimableReward](#)

ncgReward [FungibleAssetValue](#)

## Returns

[bool](#)

CollectWantedReward(ClaimableReward,  
GameConfigState,  
AdventureBossNcgRewardRatioSheet, SeasonInfo,  
BountyBoard, Investor, long, Address, ref  
ClaimableReward)

```
public static bool CollectWantedReward(AdventureBossGameData.ClaimableReward reward,  
GameConfigState gameConfig, AdventureBossNcgRewardRatioSheet ncgRewardRatioSheet,  
SeasonInfo seasonInfo, BountyBoard bountyBoard, Investor investor, long  
currentBlockIndex, Address avatarAddress, ref AdventureBossGameData.ClaimableReward  
updatedReward)
```

## Parameters

reward [AdventureBossGameData.ClaimableReward](#)

gameConfig [GameConfigState](#)

ncgRewardRatioSheet [AdventureBossNcgRewardRatioSheet](#)

seasonInfo [SeasonInfo](#)

bountyBoard [BountyBoard](#)

investor [Investor](#)

currentBlockIndex [long](#)

avatarAddress Address

updatedReward [AdventureBossGameData.ClaimableReward](#)

## Returns

[bool](#)

## GetSeasonAsAddressForm(long)

```
public static string GetSeasonAsAddressForm(long season)
```

Parameters

season [long](#)

Returns

[string](#)

## PickExploreRaffle(BountyBoard, ExploreBoard, ExplorerList, IRandom)

```
public static ExploreBoard PickExploreRaffle(BountyBoard bountyBoard, ExploreBoard exploreBoard, ExplorerList explorerList, IRandom random)
```

Parameters

bountyBoard [BountyBoard](#)

exploreBoard [ExploreBoard](#)

explorerList [ExplorerList](#)

random IRandom

Returns

[ExploreBoard](#)

## PickReward(IRandom, IEnumerable<RewardRatioData>)

```
public static (int?, int?) PickReward(IRandom random, IEnumerable<AdventureBossSheet.RewardRatioData> rewardData)
```

Parameters

random IRandom

rewardData [IEnumerable](#)<[AdventureBossSheet.RewardRatioData](#)>

## Returns

([int](#)? , [int](#)? )

# Class AvatarStateExtensions

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class AvatarStateExtensions
```

## Inheritance

[object](#) ← AvatarStateExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetLevelAndExp(AvatarState, CharacterLevelSheet, int, int)

```
public static (int, long) GetLevelAndExp(this AvatarState avatarState,  
CharacterLevelSheet characterLevelSheet, int stageId, int repeatCount)
```

#### Parameters

avatarState [AvatarState](#)

characterLevelSheet [CharacterLevelSheet](#)

stageId [int](#)

repeatCount [int](#)

#### Returns

([int](#), [long](#))

## GetLevelAndExpV1(AvatarState, CharacterLevelSheet, int, int)

```
[Obsolete("Use GetLevelAndExp")]
public static (int, long) GetLevelAndExpV1(this AvatarState avatarState,
CharacterLevelSheet characterLevelSheet, int stageId, int repeatCount)
```

### Parameters

avatarState [AvatarState](#)

characterLevelSheet [CharacterLevelSheet](#)

stageId [int](#)

repeatCount [int](#)

### Returns

([int](#), [long](#))

## UpdateExp(AvatarState, int, long)

```
public static void UpdateExp(this AvatarState avatarState, int level, long exp)
```

### Parameters

avatarState [AvatarState](#)

level [int](#)

exp [long](#)

## UpdateInventory(AvatarState, List<ItemBase>)

```
public static void UpdateInventory(this AvatarState avatarState,
List<ItemBase> rewards)
```

## Parameters

avatarState [AvatarState](#)

rewards [List](#)<[ItemBase](#)>

## UpdateMonsterMap(AvatarState, StageWaveSheet, int)

```
public static void UpdateMonsterMap(this AvatarState avatarState, StageWaveSheet  
stageWaveSheet, int stageId)
```

## Parameters

avatarState [AvatarState](#)

stageWaveSheet [StageWaveSheet](#)

stageId [int](#)

## ValidEquipmentAndCostume(AvatarState, IEnumerable<Guid>, List<Guid>, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet, long, string)

```
public static void ValidEquipmentAndCostume(this AvatarState avatarState,  
IEnumerable<Guid> costumeIds, List<Guid> equipmentIds, ItemRequirementSheet  
itemRequirementSheet, EquipmentItemRecipeSheet equipmentItemRecipeSheet,  
EquipmentItemSubRecipeSheetV2 equipmentItemSubRecipeSheetV2,  
EquipmentItemOptionSheet equipmentItemOptionSheet, long blockIndex,  
string addressesHex)
```

## Parameters

avatarState [AvatarState](#)

costumeIds [IEnumerable](#)<[Guid](#)>

equipmentIds [List<Guid>](#)

itemRequirementSheet [ItemRequirementSheet](#)

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

equipmentItemSubRecipeSheetV2 [EquipmentItemSubRecipeSheetV2](#)

equipmentItemOptionSheet [EquipmentItemOptionSheet](#)

blockIndex [long](#)

addressesHex [string](#)

ValidEquipmentAndCostumeV2(AvatarState,  
IEnumerable<Guid>, List<Guid>,  
ItemRequirementSheet, EquipmentItemRecipeSheet,  
EquipmentItemSubRecipeSheetV2,  
EquipmentItemOptionSheet, long, string,  
GameConfigState)

```
public static (List<Equipment> equipments, List<Costume> costumes)
ValidEquipmentAndCostumeV2(this AvatarState avatarState, IEnumerable<Guid>
costumeIds, List<Guid> equipmentIds, ItemRequirementSheet itemRequirementSheet,
EquipmentItemRecipeSheet equipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2
equipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet equipmentItemOptionSheet,
long blockIndex, string addressesHex, GameConfigState gameConfigState)
```

## Parameters

avatarState [AvatarState](#)

costumeIds [IEnumerable<Guid>](#)

equipmentIds [List<Guid>](#)

itemRequirementSheet [ItemRequirementSheet](#)

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

equipmentItemSubRecipeSheetV2 [EquipmentItemSubRecipeSheetV2](#)

equipmentItemOptionSheet [EquipmentItemOptionSheet](#)

blockIndex [long](#)

addressesHex [string](#)

gameConfigState [GameConfigState](#)

Returns

([List](#)<[Equipment](#)> [equipments](#), [List](#)<[Costume](#)> [costumes](#))

# Class CrystalCalculator

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class CrystalCalculator
```

## Inheritance

[object](#) ← CrystalCalculator

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### CRYSTAL

```
public static readonly Currency CRYSTAL
```

#### Field Value

Currency

### CrystalLimit

```
public const long CrystalLimit = 1000000000
```

#### Field Value

[long](#)

### MaxLevelExponent

```
public const int MaxLevelExponent = 5
```

## Field Value

[int](#)

## Methods

### CalculateBuffGachaCost(int, bool, CrystalStageBuffGachaSheet)

```
public static FungibleAssetValue CalculateBuffGachaCost(int stageId, bool  
advancedGacha, CrystalStageBuffGachaSheet stageBuffGachaSheet)
```

#### Parameters

stageId [int](#)

advancedGacha [bool](#)

stageBuffGachaSheet [CrystalStageBuffGachaSheet](#)

#### Returns

FungibleAssetValue

### CalculateCombinationCost(FungibleAssetValue, Row, CrystalCostState, CrystalCostState)

```
public static FungibleAssetValue CalculateCombinationCost(FungibleAssetValue  
crystal, CrystalFluctuationSheet.Row row, CrystalCostState prevWeeklyCostState =  
null, CrystalCostState beforePrevWeeklyCostState = null)
```

#### Parameters

crystal FungibleAssetValue

row [CrystalFluctuationSheet.Row](#)

prevWeeklyCostState [CrystalCostState](#)

beforePrevWeeklyCostState [CrystalCostState](#)

Returns

FungibleAssetValue

CalculateCrystal(Address, IEnumerable<Equipment>,  
FungibleAssetValue, bool,  
CrystalEquipmentGrindingSheet,  
CrystalMonsterCollectionMultiplierSheet,  
StakeRegularRewardSheet)

```
public static FungibleAssetValue CalculateCrystal(Address agentAddress,  
IEnumerable<Equipment> equipmentList, FungibleAssetValue stakedAmount, bool  
enhancementFailed, CrystalEquipmentGrindingSheet crystalEquipmentGrindingSheet,  
CrystalMonsterCollectionMultiplierSheet crystalMonsterCollectionMultiplierSheet,  
StakeRegularRewardSheet stakeRegularRewardSheet)
```

Parameters

agentAddress Address

equipmentList [IEnumerable](#)<Equipment>

stakedAmount FungibleAssetValue

enhancementFailed [bool](#)

crystalEquipmentGrindingSheet [CrystalEquipmentGrindingSheet](#)

crystalMonsterCollectionMultiplierSheet [CrystalMonsterCollectionMultiplierSheet](#)

stakeRegularRewardSheet [StakeRegularRewardSheet](#)

Returns

FungibleAssetValue

**CalculateCrystal(IEnumerable<Equipment>, bool, CrystalEquipmentGrindingSheet, CrystalMonsterCollectionMultiplierSheet, int)**

```
public static FungibleAssetValue CalculateCrystal(IEnumerable<Equipment>
equipmentList, bool enhancementFailed, CrystalEquipmentGrindingSheet
crystalEquipmentGrindingSheet, CrystalMonsterCollectionMultiplierSheet
crystalMonsterCollectionMultiplierSheet, int stakingLevel)
```

Parameters

equipmentList [IEnumerable<Equipment>](#)

enhancementFailed [bool](#)

crystalEquipmentGrindingSheet [CrystalEquipmentGrindingSheet](#)

crystalMonsterCollectionMultiplierSheet [CrystalMonsterCollectionMultiplierSheet](#)

stakingLevel [int](#)

Returns

FungibleAssetValue

**CalculateEntranceFee(int, BigInteger)**

```
public static FungibleAssetValue CalculateEntranceFee(int level,
BigInteger entranceFee)
```

Parameters

level [int](#)

entranceFee [BigInteger](#)

Returns

FungibleAssetValue

## CalculateMaterialCost(int, int, CrystalMaterialCostSheet)

```
public static FungibleAssetValue CalculateMaterialCost(int materialId, int  
materialCount, CrystalMaterialCostSheet crystalMaterialCostSheet)
```

### Parameters

materialId [int](#)

materialCount [int](#)

crystalMaterialCostSheet [CrystalMaterialCostSheet](#)

### Returns

FungibleAssetValue

## CalculateRecipeUnlockCost(IEnumerable<int>, EquipmentItemRecipeSheet)

```
public static FungibleAssetValue CalculateRecipeUnlockCost(IEnumerable<int>  
recipeIds, EquipmentItemRecipeSheet equipmentItemRecipeSheet)
```

### Parameters

recipeIds [IEnumerable](#)<[int](#)>

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

### Returns

FungibleAssetValue

## CalculateWorldUnlockCost(IEnumerable<int>, WorldUnlockSheet)

```
public static FungibleAssetValue CalculateWorldUnlockCost(IEnumerable<int> worldIds,  
WorldUnlockSheet worldUnlockSheet)
```

### Parameters

`worldIds` [IEnumerable<int>](#)

`worldUnlockSheet` [WorldUnlockSheet](#)

### Returns

`FungibleAssetValue`

# Class CustomCraftHelper

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class CustomCraftHelper
```

## Inheritance

[object](#) ← CustomCraftHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

**CalculateAdditionalCost(int,  
CustomEquipmentCraftRelationshipSheet)**

```
public static (BigInteger, IDictionary<int, int>)? CalculateAdditionalCost(int  
relationship, CustomEquipmentCraftRelationshipSheet relationshipSheet)
```

### Parameters

relationship [int](#)

relationshipSheet [CustomEquipmentCraftRelationshipSheet](#)

### Returns

([BigInteger](#), [IDictionary](#)<[int](#), [int](#)>)?

**CalculateCraftCost(int, int, MaterialItemSheet, Row,  
Row, decimal)**

```
public static (BigInteger, IDictionary<int, int>) CalculateCraftCost(int iconId, int relationship, MaterialItemSheet materialItemSheet, CustomEquipmentCraftRecipeSheet.Row recipeRow, CustomEquipmentCraftRelationshipSheet.Row relationshipRow, decimal iconCostMultiplier)
```

## Parameters

iconId [int](#)

relationship [int](#)

materialItemSheet [MaterialItemSheet](#)

recipeRow [CustomEquipmentCraftRecipeSheet.Row](#)

relationshipRow [CustomEquipmentCraftRelationshipSheet.Row](#)

iconCostMultiplier [decimal](#)

## Returns

([BigInteger](#), [IDictionary](#)<[int](#), [int](#)>)

## SelectCp(Row, IRandom)

```
public static int SelectCp(CustomEquipmentCraftRelationshipSheet.Row relationshipRow, IRandom random)
```

## Parameters

relationshipRow [CustomEquipmentCraftRelationshipSheet.Row](#)

random [IRandom](#)

## Returns

[int](#)

# Class InventoryExtensions

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class InventoryExtensions
```

## Inheritance

[object](#) ← InventoryExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetEquippedFullCostumeOrArmorId(Inventory)

```
public static int GetEquippedFullCostumeOrArmorId(this Inventory inventory)
```

#### Parameters

inventory [Inventory](#)

#### Returns

[int](#)

### UseActionPoint(Inventory, long, int, bool, MaterialItemSheet, long)

```
public static long UseActionPoint(this Inventory inventory, long originalAp, int requiredAp, bool chargeAp, MaterialItemSheet materialItemSheet, long blockIndex)
```

## Parameters

inventory [Inventory](#)

originalAp [long](#)

requiredAp [int](#)

chargeAp [bool](#)

materialItemSheet [MaterialItemSheet](#)

blockIndex [long](#)

## Returns

[long](#)

# Class ItemOptionHelper

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class ItemOptionHelper
```

## Inheritance

[object](#) ← ItemOptionHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetStatOptionRows(Row, EquipmentItemOptionSheet, ItemUsable)

```
public static List<EquipmentItemOptionSheet.Row>
GetStatOptionRows(EquipmentItemSubRecipeSheetV2.Row subRecipeRow,
EquipmentItemOptionSheet itemOptionSheet, ItemUsable itemUsable)
```

## Parameters

subRecipeRow [EquipmentItemSubRecipeSheetV2.Row](#)

itemOptionSheet [EquipmentItemOptionSheet](#)

itemUsable [ItemUsable](#)

## Returns

[List](#)<[EquipmentItemOptionSheet.Row](#)>

## GetStatOptionRows(long, ItemUsable, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet)

```
public static List<EquipmentItemOptionSheet.Row> GetStatOptionRows(long subRecipeId,  
ItemUsable itemUsable, EquipmentItemSubRecipeSheetV2 subRecipeSheet,  
EquipmentItemOptionSheet itemOptionSheet)
```

### Parameters

subRecipeId [long](#)

itemUsable [ItemUsable](#)

subRecipeSheet [EquipmentItemSubRecipeSheetV2](#)

itemOptionSheet [EquipmentItemOptionSheet](#)

### Returns

[List](#)<[EquipmentItemOptionSheet](#).Row>

## TryGet(ItemUsable, out ItemOptionInfo)

```
public static bool TryGet(ItemUsable itemUsable, out ItemOptionInfo itemOptionInfo)
```

### Parameters

itemUsable [ItemUsable](#)

itemOptionInfo [ItemOptionInfo](#)

### Returns

[bool](#)

# Class ItemOptionInfo

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public class ItemOptionInfo
```

## Inheritance

[object](#) ← ItemOptionInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemOptionInfo(Equipment)

```
public ItemOptionInfo(Equipment equipment)
```

#### Parameters

equipment [Equipment](#)

### ItemOptionInfo(ItemUsable)

```
public ItemOptionInfo(ItemUsable itemUsable)
```

#### Parameters

itemUsable [ItemUsable](#)

## Fields

## CP

```
public readonly int CP
```

### Field Value

[int](#)

## MainStat

```
public readonly (StatType type, long baseValue, long totalValue) MainStat
```

### Field Value

([StatType](#) [type](#), [long](#) [baseValue](#), [long](#) [totalValue](#))

## OptionCountFromCombination

```
public readonly int OptionCountFromCombination
```

### Field Value

[int](#)

## SkillOptions

```
public readonly List<(SkillSheet.Row skillRow, long power, int chance, int statPowerRatio, StatType refStatType)> SkillOptions
```

### Field Value

[List](#)<([SkillSheet.Row](#) [skillRow](#), [long](#) [power](#), [int](#) [chance](#), [int](#) [statPowerRatio](#), [StatType](#) [refStatType](#))>

# StatOptions

```
public readonly List<(StatType type, long value, int count)> StatOptions
```

## Field Value

[List](#)<(StatType type, long value, int count)>

# Class PetHelper

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class PetHelper
```

## Inheritance

[object](#) ← PetHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

CalculateDiscountedHourglass(long, int, PetState, PetOptionSheet)

```
public static int CalculateDiscountedHourglass(long diff, int hourglassPerBlock,  
PetState petState, PetOptionSheet petOptionSheet)
```

### Parameters

diff [long](#)

hourglassPerBlock [int](#)

petState [PetState](#)

petOptionSheet [PetOptionSheet](#)

### Returns

[int](#)

## CalculateDiscountedMaterialCost(FungibleAssetValue, PetState, PetOptionSheet)

```
public static FungibleAssetValue CalculateDiscountedMaterialCost(FungibleAssetValue originalCost, PetState petState, PetOptionSheet petOptionSheet)
```

### Parameters

originalCost FungibleAssetValue

petState [PetState](#)

petOptionSheet [PetOptionSheet](#)

### Returns

FungibleAssetValue

## CalculateEnhancementCost(PetCostSheet, int, int, int)

```
public static (int ncgQuantity, int soulStoneQuantity)
CalculateEnhancementCost(PetCostSheet costSheet, int petId, int currentLevel,
int targetLevel)
```

### Parameters

costSheet [PetCostSheet](#)

petId [int](#)

currentLevel [int](#)

targetLevel [int](#)

### Returns

([int](#) [challengerScoreDelta](#), [int](#) [defenderScoreDelta](#))

## CalculateReducedBlockOnCraft(long, long, PetState, PetOptionSheet)

```
public static long CalculateReducedBlockOnCraft(long originalBlock, long minimumBlock, PetState petState, PetOptionSheet petOptionSheet)
```

### Parameters

originalBlock [long](#)

minimumBlock [long](#)

petState [PetState](#)

petOptionSheet [PetOptionSheet](#)

### Returns

[long](#)

## GetBonusOptionProbability(int, PetState, PetOptionSheet)

```
public static int GetBonusOptionProbability(int originalRatio, PetState petState, PetOptionSheet petOptionSheet)
```

### Parameters

originalRatio [int](#)

petState [PetState](#)

petOptionSheet [PetOptionSheet](#)

### Returns

[int](#)

## GetSoulstoneCurrency(string)

```
public static Currency GetSoulstoneCurrency(string ticker)
```

### Parameters

**ticker** [string](#)

### Returns

Currency

# Class RuneHelper

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class RuneHelper
```

## Inheritance

[object](#) ← RuneHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### DailyRewardRune

```
public static readonly Currency DailyRewardRune
```

#### Field Value

Currency

### StakeRune

```
public static readonly Currency StakeRune
```

#### Field Value

Currency

## Methods

## CalculateRuneLevelBonus(AllRuneState, RuneListSheet, RuneLevelBonusSheet)

```
public static int CalculateRuneLevelBonus(AllRuneState allRuneState, RuneListSheet  
runeListSheet, RuneLevelBonusSheet runeLevelBonusSheet)
```

### Parameters

allRuneState [AllRuneState](#)

runeListSheet [RuneListSheet](#)

runeLevelBonusSheet [RuneLevelBonusSheet](#)

### Returns

[int](#)

## CalculateStakeReward(FungibleAssetValue, int)

```
public static FungibleAssetValue CalculateStakeReward(FungibleAssetValue  
stakeAmount, int rate)
```

### Parameters

stakeAmount [FungibleAssetValue](#)

rate [int](#)

### Returns

[FungibleAssetValue](#)

## ToCurrency(Row)

```
public static Currency ToCurrency(RuneSheet.Row runeRow)
```

Parameters

runeRow [RuneSheet.Row](#)

Returns

Currency

## ToFungibleAssetValue(Row, int)

```
public static FungibleAssetValue ToFungibleAssetValue(RuneSheet.Row runeRow,  
int quantity)
```

Parameters

runeRow [RuneSheet.Row](#)

quantity [int](#)

Returns

FungibleAssetValue

## TryEnhancement(int, Row, IRandom, int, out LevelUpResult)

```
public static bool TryEnhancement(int startRuneLevel, RuneCostSheet.Row costRow,  
IRandom random, int tryCount, out RuneEnhancement.LevelUpResult levelUpResult)
```

Parameters

startRuneLevel [int](#)

costRow [RuneCostSheet.Row](#)

random IRandom

tryCount [int](#)

`levelUpResult` [RuneEnhancement.LevelUpResult](#)

Returns

[bool](#) ↗

# Class WorldBossHelper

Namespace: [Nekoyume.Helper](#)

Assembly: Lib9c.dll

```
public static class WorldBossHelper
```

## Inheritance

[object](#) ← WorldBossHelper

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### MaxChallengeCount

```
public const int MaxChallengeCount = 3
```

#### Field Value

[int](#)

### RefillInterval

```
[Obsolete("Use GameConfigState.DailyWorldBossInterval")]
public const long RefillInterval = 7200
```

#### Field Value

[long](#)

# Methods

## CalculateRank(Row, long)

```
public static int CalculateRank(WorldBossCharacterSheet.Row row, long score)
```

### Parameters

row [WorldBossCharacterSheet.Row](#)

score [long](#)

### Returns

[int](#)

## CalculateReward(int, int, RuneWeightSheet, IWorldBossRewardSheet, RuneSheet, MaterialItemSheet, IRandom)

```
public static (List<FungibleAssetValue> assets, Dictionary<TradableMaterial, int>  
materials) CalculateReward(int rank, int bossId, RuneWeightSheet sheet,  
IWorldBossRewardSheet rewardSheet, RuneSheet runeSheet, MaterialItemSheet  
materialSheet, IRandom random)
```

### Parameters

rank [int](#)

bossId [int](#)

sheet [RuneWeightSheet](#)

rewardSheet [IWorldBossRewardSheet](#)

runeSheet [RuneSheet](#)

materialSheet [MaterialItemSheet](#)

random [IRandom](#)

Returns

([List](#)<FungibleAssetValue> [assets](#), [Dictionary](#)<TradableMaterial, [int](#)> [materials](#))

## CalculateTicketPrice(Row, RaiderState, Currency)

```
public static FungibleAssetValue CalculateTicketPrice(WorldBossListSheet.Row row,  
RaiderState raiderState, Currency currency)
```

Parameters

row [WorldBossListSheet.Row](#)

raiderState [RaiderState](#)

currency Currency

Returns

FungibleAssetValue

## CanRefillTicket(long, long, long, int)

```
public static bool CanRefillTicket(long blockIndex, long refilledIndex, long  
startedIndex, int refillInterval)
```

Parameters

blockIndex [long](#)

refilledIndex [long](#)

startedIndex [long](#)

refillInterval [int](#)

Returns

[bool](#)

## CanRefillTicketV1(long, long, long)

```
public static bool CanRefillTicketV1(long blockIndex, long refilledIndex,  
long startedIndex)
```

### Parameters

blockIndex [long](#)

refilledIndex [long](#)

startedIndex [long](#)

### Returns

[bool](#)

# Namespace Nekoyume.Model

## Classes

[ArenaCharacter](#)

[ArenaSkills](#)

[CharacterBase](#)

[CollectionMap](#)

[Enemy](#)

[EnemyPlayer](#)

[FailedAddWorldException](#)

[FailedToUnlockWorldException](#)

[Player](#)

[Player.ExpData](#)

[RaidBoss](#)

[Skills](#)

[WorldInformation](#)

## Structs

[ActivationKey](#)

[ArenaPlayerDigest](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1156> ↗

[FungibleItemValue](#)

[WorldInformation.World](#)

## Interfaces

[IArena](#)

[IStage](#)

# Struct ActivationKey

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
public struct ActivationKey
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Fields

### DeriveKey

```
public const string DeriveKey = "activated"
```

#### Field Value

[string](#)

## Properties

### PendingAddress

```
public readonly Address PendingAddress { get; }
```

#### Property Value

Address

### PrivateKey

```
public readonly PrivateKey PrivateKey { get; }
```

Property Value

PrivateKey

## Methods

### Create(PrivateKey, byte[])

```
public static (ActivationKey, PendingActivationState) Create(PrivateKey privateKey,  
byte[] nonce)
```

Parameters

privateKey PrivateKey

nonce [byte](#)[]

Returns

([ActivationKey](#), [PendingActivationState](#))

### Decode(string)

```
public static ActivationKey Decode(string hexWithSlash)
```

Parameters

hexWithSlash [string](#)[]

Returns

[ActivationKey](#)

## Encode()

```
public string Encode()
```

Returns

[string](#)

# Class ArenaCharacter

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
public class ArenaCharacter : ICloneable
```

## Inheritance

[object](#) ← ArenaCharacter

## Implements

[ICloneable](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaCharacter(IArenaSimulator, ArenaPlayerDigest,  
ArenaSimulatorSheets, bool)

```
public ArenaCharacter(IArenaSimulator simulator, ArenaPlayerDigest digest,  
ArenaSimulatorSheets sheets, bool isEnemy = false)
```

## Parameters

simulator [IArenaSimulator](#)

digest [ArenaPlayerDigest](#)

sheets [ArenaSimulatorSheets](#)

isEnemy [bool](#)

`ArenaCharacter(IArenaSimulator, ArenaPlayerDigest, ArenaSimulatorSheets, long, List<StatModifier>, bool, bool)`

```
public ArenaCharacter(IArenaSimulator simulator, ArenaPlayerDigest
digest, ArenaSimulatorSheets sheets, long hpModifier, List<StatModifier>
collectionModifiers, bool isEnemy = false, bool setExtraValueBuffBeforeGetBuffs
= false)
```

## Parameters

`simulator` [IArenaSimulator](#)

`digest` [ArenaPlayerDigest](#)

`sheets` [ArenaSimulatorSheets](#)

`hpModifier` [long](#)

`collectionModifiers` [List](#)<[StatModifier](#)>

`isEnemy` [bool](#)

`setExtraValueBuffBeforeGetBuffs` [bool](#)

## Fields

### CriticalMultiplier

```
public const decimal CriticalMultiplier = 1.5
```

## Field Value

[decimal](#)

### RuneSkillCooldownMap

```
public readonly Dictionary<int, int> RuneSkillCooldownMap
```

Field Value

[Dictionary](#)<[int](#), [int](#)>

## Simulator

`public readonly IArenaSimulator Simulator`

Field Value

[IArenaSimulator](#)

## \_runeSkills

`public readonly ArenaSkills _runeSkills`

Field Value

[ArenaSkills](#)

## usedSkill

`public ArenaSkill usedSkill`

Field Value

[ArenaSkill](#)

## Properties

### ATK

`public long ATK { get; }`

Property Value

[long](#) ↴

## ActionBuffs

```
public IEnumerable<ActionBuff> ActionBuffs { get; }
```

Property Value

[IEnumerable](#) ↴ <[ActionBuff](#)>

## AdditionalHP

```
public long AdditionalHP { get; }
```

Property Value

[long](#) ↴

## ArmorPenetration

```
public long ArmorPenetration { get; }
```

Property Value

[long](#) ↴

## AttackCount

```
public int AttackCount { get; }
```

Property Value

[int](#)

## AttackRange

```
public float AttackRange { get; }
```

Property Value

[float](#)

## Buffs

```
public Dictionary<int, Buff> Buffs { get; }
```

Property Value

[Dictionary](#)<[int](#), [Buff](#)>

## CDMG

```
public long CDMG { get; }
```

Property Value

[long](#)

## CRI

```
public long CRI { get; }
```

Property Value

[long](#)

## CharacterId

```
public int CharacterId { get; }
```

Property Value

[int](#) ↗

## CurrentHP

```
public long CurrentHP { get; set; }
```

Property Value

[long](#) ↗

## DEF

```
public long DEF { get; }
```

Property Value

[long](#) ↗

## DRR

```
public long DRR { get; }
```

Property Value

[long](#) ↗

## DRV

```
public long DRV { get; }
```

Property Value

[long](#) ↗

## DefenseElementalType

```
public ElementalType DefenseElementalType { get; }
```

Property Value

[ElementalType](#)

## HIT

```
public long HIT { get; }
```

Property Value

[long](#) ↗

## HP

```
public long HP { get; }
```

Property Value

[long](#) ↗

## Id

```
public Guid Id { get; }
```

Property Value

[Guid](#)

IsDead

```
public bool IsDead { get; }
```

Property Value

[bool](#)

IsEnemy

```
public bool IsEnemy { get; }
```

Property Value

[bool](#)

Level

```
public int Level { get; set; }
```

Property Value

[int](#)

OffensiveElementalType

```
public ElementalType OffensiveElementalType { get; }
```

Property Value

[ElementalType](#)

## RunSpeed

```
public float RunSpeed { get; }
```

Property Value

[float](#)

## SPD

```
public long SPD { get; }
```

Property Value

[long](#)

## SizeType

```
public SizeType SizeType { get; }
```

Property Value

[SizeType](#)

## SkillLog

```
public ArenaSkill SkillLog { get; }
```

Property Value

[ArenaSkill](#)

## StatBuffs

```
public IEnumerable<StatBuff> StatBuffs { get; }
```

Property Value

[IEnumerable](#)<[StatBuff](#)>

## Stats

```
public CharacterStats Stats { get; }
```

Property Value

[CharacterStats](#)

## Thorn

```
public long Thorn { get; }
```

Property Value

[long](#)

## Methods

AddBuff(Buff, bool)

```
public IEnumerable<Buff> AddBuff(Buff buff, bool updateImmediate = true)
```

Parameters

buff [Buff](#)

updateImmediate [bool](#)

Returns

[IEnumerable](#)<[Buff](#)>

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## GetDamage(long, bool)

```
public long GetDamage(long damage, bool considerAttackCount = true)
```

Parameters

damage [long](#)

considerAttackCount [bool](#)

Returns

[long](#)

## Heal(long)

```
public void Heal(long heal)
```

### Parameters

heal [long](#)

## IsCritical(bool)

```
public bool IsCritical(bool considerAttackCount = true)
```

### Parameters

considerAttackCount [bool](#)

### Returns

[bool](#)

## IsHit(ArenaCharacter)

```
public virtual bool IsHit(ArenaCharacter caster)
```

### Parameters

caster [ArenaCharacter](#)

### Returns

[bool](#)

## OnPostSkill(ArenaSkill)

```
protected virtual void OnPostSkill(ArenaSkill usedSkill)
```

Parameters

`usedSkill` [ArenaSkill](#)

## OnPreSkill()

```
protected virtual bool OnPreSkill()
```

Returns

[bool](#) ↗

## RemoveActionBuff(ActionBuff)

```
public void RemoveActionBuff(ActionBuff removedBuff)
```

Parameters

`removedBuff` [ActionBuff](#)

## RemoveRecentStatBuff()

```
public void RemoveRecentStatBuff()
```

## RemoveStatBuff(StatBuff)

```
public void RemoveStatBuff(StatBuff removedBuff)
```

Parameters

`removedBuff` [StatBuff](#)

## ResetCurrentHP()

```
protected void ResetCurrentHP()
```

## SetRuneSkills(IEnumerable<RuneState>, RuneOptionSheet, SkillSheet)

```
public void SetRuneSkills(IEnumerable<RuneState> runes, RuneOptionSheet  
runeOptionSheet, SkillSheet skillSheet)
```

### Parameters

runes [IEnumerable](#)<RuneState>

runeOptionSheet [RuneOptionSheet](#)

skillSheet [SkillSheet](#)

## SetRuneStats(IEnumerable<RuneState>, RuneOptionSheet, int)

```
public void SetRuneStats(IEnumerable<RuneState> runes, RuneOptionSheet  
runeOptionSheet, int runeLevelBonus)
```

### Parameters

runes [IEnumerable](#)<RuneState>

runeOptionSheet [RuneOptionSheet](#)

runeLevelBonus [int](#)

## SetRuneV1(List<RuneState>, RuneOptionSheet, SkillSheet)

```
[Obsolete("Use SetRune instead.")]
public void SetRuneV1(List<RuneState> runes, RuneOptionSheet runeOptionSheet,
SkillSheet skillSheet)
```

## Parameters

runes [List](#)<[RuneState](#)>

runeOptionSheet [RuneOptionSheet](#)

skillSheet [SkillSheet](#)

## SetRuneV2(List<RuneState>, RuneOptionSheet, SkillSheet)

```
[Obsolete("Use SetRune instead.")]
public void SetRuneV2(List<RuneState> runes, RuneOptionSheet runeOptionSheet,
SkillSheet skillSheet)
```

## Parameters

runes [List](#)<[RuneState](#)>

runeOptionSheet [RuneOptionSheet](#)

skillSheet [SkillSheet](#)

## Spawn(ArenaCharacter)

```
public void Spawn(ArenaCharacter target)
```

## Parameters

target [ArenaCharacter](#)

## SpawnV2(ArenaCharacter)

```
[Obsolete("Use Spawn")]
public void SpawnV2(ArenaCharacter target)
```

## Parameters

target [ArenaCharacter](#)

## Tick()

```
public void Tick()
```

# Struct ArenaPlayerDigest

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1156>

```
public readonly struct ArenaPlayerDigest : IState
```

## Implements

[IState](#)

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaPlayerDigest(List)

```
public ArenaPlayerDigest(List serialized)
```

#### Parameters

serialized List

### ArenaPlayerDigest(AvatarState, AllRuneState, RuneSlotState)

```
public ArenaPlayerDigest(AvatarState avatarState, AllRuneState runes,  
RuneSlotState runeSlotState)
```

#### Parameters

avatarState [AvatarState](#)

runes [AllRuneState](#)

runeSlotState [RuneSlotState](#)

## ArenaPlayerDigest(AvatarState, ArenaAvatarState)

```
[Obsolete("Do not use it")]
public ArenaPlayerDigest(AvatarState avatarState, ArenaAvatarState arenaAvatarState)
```

### Parameters

avatarState [AvatarState](#)

arenaAvatarState [ArenaAvatarState](#)

## ArenaPlayerDigest(AvatarState, List<Costume>, List<Equipment>, AllRuneState, RuneSlotState)

```
public ArenaPlayerDigest(AvatarState avatarState, List<Costume> costumes,
List<Equipment> equipments, AllRuneState runes, RuneSlotState runeSlotState)
```

### Parameters

avatarState [AvatarState](#)

costumes [List](#)<[Costume](#)>

equipments [List](#)<[Equipment](#)>

runes [AllRuneState](#)

runeSlotState [RuneSlotState](#)

## ArenaPlayerDigest(AvatarState, List<Guid>, List<Guid>, AllRuneState, RuneSlotState)

```
public ArenaPlayerDigest(AvatarState avatarState, List<Guid> equipments, List<Guid> costumes, AllRuneState runes, RuneSlotState runeSlotState)
```

## Parameters

avatarState [AvatarState](#)  
equipments [List](#)<[Guid](#)>  
costumes [List](#)<[Guid](#)>  
runes [AllRuneState](#)  
runeSlotState [RuneSlotState](#)

## Fields

### CharacterId

```
public readonly int CharacterId
```

### Field Value

[int](#)

### Costumes

```
public readonly List<Costume> Costumes
```

### Field Value

[List](#)<[Costume](#)>

### EarIndex

```
public readonly int EarIndex
```

Field Value

[int](#)

## Equipments

```
public readonly List<Equipment> Equipments
```

Field Value

[List](#)<[Equipment](#)>

## HairIndex

```
public readonly int HairIndex
```

Field Value

[int](#)

## LensIndex

```
public readonly int LensIndex
```

Field Value

[int](#)

## Level

```
public readonly int Level
```

Field Value

[int](#)

## NameWithHash

```
public readonly string NameWithHash
```

Field Value

[string](#)

## RuneSlotState

```
public readonly RuneSlotState RuneSlotState
```

Field Value

[RuneSlotState](#)

## Runes

```
public readonly AllRuneState Runes
```

Field Value

[AllRuneState](#)

## TailIndex

```
public readonly int TailIndex
```

Field Value

[int](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class ArenaSkills

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaSkills : IEnumerable<ArenaSkill>, IEnumerable
```

## Inheritance

[object](#) ← ArenaSkills

## Implements

[IEnumerable](#)<[ArenaSkill](#)>, [IEnumerable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Methods

### Add(ArenaSkill)

```
public void Add(ArenaSkill skill)
```

#### Parameters

skill [ArenaSkill](#)

### Clear()

```
public void Clear()
```

### GetCooldown(int)

```
public int GetCooldown(int skillId)
```

Parameters

skillId [int](#)

Returns

[int](#)

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumarator<ArenaSkill> GetEnumerator()
```

Returns

[IEnumarator](#)<[ArenaSkill](#)>

An enumerator that can be used to iterate through the collection.

## ReduceCooldown()

```
public void ReduceCooldown()
```

## Select(IRandom)

```
public ArenaSkill Select(IRandom random)
```

Parameters

random IRandom

Returns

## SelectWithoutDefaultAttack(IRandom)

```
public ArenaSkill SelectWithoutDefaultAttack(IRandom random)
```

### Parameters

random IRandom

### Returns

[ArenaSkill](#)

## SetCooldown(int, int)

```
public void SetCooldown(int skillId, int cooldown)
```

### Parameters

skillId [int](#)

cooldown [int](#)

# Class CharacterBase

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class CharacterBase : ICloneable
```

## Inheritance

[object](#) ← CharacterBase

## Implements

[ICloneable](#)

## Derived

[Enemy](#), [Player](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

CharacterBase(Simulator, CharacterStats, int, ElementalType, Row, SizeType, float, float)

```
protected CharacterBase(Simulator simulator, CharacterStats stat, int characterId,
ElementalType elementType, CharacterSheet.Row rowData = null, SizeType sizeType =
SizeType.XL, float attackRange = 4, float runSpeed = 0.3)
```

## Parameters

simulator [Simulator](#)

stat [CharacterStats](#)

characterId [int](#)

elementType [ElementalType](#)

rowData [CharacterSheet.Row](#)

sizeType [SizeType](#)

attackRange [float](#)

runSpeed [float](#)

## CharacterBase(Simulator, CharacterSheet, int, int, IEnumerable<StatModifier>)

```
protected CharacterBase(Simulator simulator, CharacterSheet characterSheet, int characterId, int level, IEnumerable<StatModifier> optionalStatModifiers = null)
```

### Parameters

simulator [Simulator](#)

characterSheet [CharacterSheet](#)

characterId [int](#)

level [int](#)

optionalStatModifiers [IEnumerable](#)<[StatModifier](#)>

## CharacterBase(CharacterBase)

```
protected CharacterBase(CharacterBase value)
```

### Parameters

value [CharacterBase](#)

## CharacterBase(Row)

```
protected CharacterBase(CharacterSheet.Row row)
```

## Parameters

row [CharacterSheet.Row](#)

## Fields

### BuffSkills

```
public readonly Skills BuffSkills
```

#### Field Value

[Skills](#)

### Buffs

```
public readonly Dictionary<int, Buff> Buffs
```

#### Field Value

[Dictionary](#)<[int](#), [Buff](#)>

### CriticalMultiplier

```
public const decimal CriticalMultiplier = 1.5
```

#### Field Value

[decimal](#)

## Id

```
public readonly Guid Id
```

Field Value

[Guid](#)

## Simulator

```
[NonSerialized]  
public Simulator Simulator
```

Field Value

[Simulator](#)

## Skills

```
public readonly Skills Skills
```

Field Value

[Skills](#)

## Targets

```
public readonly List<CharacterBase> Targets
```

Field Value

[List](#) <[CharacterBase](#)>

## atkElementType

```
public ElementalType atkElementType
```

Field Value

[ElementalType](#)

## attackRange

```
public float attackRange
```

Field Value

[float](#)

## defElementType

```
public ElementalType defElementType
```

Field Value

[ElementalType](#)

## Properties

### ATK

```
public long ATK { get; }
```

Property Value

[long](#)

## ActionBuffs

```
public IEnumerable<ActionBuff> ActionBuffs { get; }
```

Property Value

[IEnumerable](#)<[ActionBuff](#)>

## AdditionalHP

```
public long AdditionalHP { get; }
```

Property Value

[long](#)

## ArmorPenetration

```
public long ArmorPenetration { get; }
```

Property Value

[long](#)

## AttackCount

```
public int AttackCount { get; set; }
```

Property Value

[int](#)

## AttackCountMax

```
public int AttackCountMax { get; protected set; }
```

Property Value

[int](#)

## CDMG

```
public long CDMG { get; }
```

Property Value

[long](#)

## CRI

```
public long CRI { get; }
```

Property Value

[long](#)

## CharacterId

```
public int CharacterId { get; }
```

Property Value

[int](#)

## CurrentHP

```
public long CurrentHP { get; set; }
```

Property Value

[long](#) ↗

DEF

```
public long DEF { get; }
```

Property Value

[long](#) ↗

DRR

```
public long DRR { get; }
```

Property Value

[long](#) ↗

DRV

```
public long DRV { get; }
```

Property Value

[long](#) ↗

HIT

```
public long HIT { get; }
```

Property Value

[long](#) ↗

HP

```
public long HP { get; }
```

Property Value

[long](#) ↗

IsDead

```
public bool IsDead { get; }
```

Property Value

[bool](#) ↗

Level

```
public int Level { get; set; }
```

Property Value

[int](#) ↗

RowData

```
public CharacterSheet.Row RowData { get; }
```

Property Value

[CharacterSheet.Row](#)

## RunSpeed

```
public float RunSpeed { get; }
```

Property Value

[float](#)

## SPD

```
public long SPD { get; }
```

Property Value

[long](#)

## SizeType

```
public SizeType SizeType { get; }
```

Property Value

[SizeType](#)

## StatBuffs

```
public IEnumerable<StatBuff> StatBuffs { get; }
```

Property Value

[IEnumerable](#)<[StatBuff](#)>

## Stats

```
public CharacterStats Stats { get; }
```

Property Value

[CharacterStats](#)

## Thorn

```
public long Thorn { get; }
```

Property Value

[long](#)

## usedSkill

```
public Skill usedSkill { get; set; }
```

Property Value

[Skill](#)

## Methods

AddBuff(Buff, bool)

Add buff/debuff to target; it means buff/debuff is used by caster. When `Dispel` is used, it can remove prev. debuffs on target. All the removed debuffs will be returned and saved in battle log.

```
public IEnumerable<Buff> AddBuff(Buff buff, bool updateImmediate = true)
```

## Parameters

`buff` [Buff](#)

`updateImmediate` [bool](#)

## Returns

[IEnumerable](#)<[Buff](#)>

An enumerable of removed debuffs from target. `null` will be returned if nothing eliminated.

## Clone()

Creates a new object that is a copy of the current instance.

```
public abstract object Clone()
```

## Returns

[object](#)

A new object that is a copy of this instance.

## Die()

```
public void Die()
```

## EndTurn()

```
protected virtual void EndTurn()
```

## GetChance(int)

```
public bool GetChance(int chance)
```

Parameters

chance [int](#)

Returns

[bool](#)

## GetDamage(long, bool)

```
public long GetDamage(long damage, bool considerAttackCount = true)
```

Parameters

damage [long](#)

considerAttackCount [bool](#)

Returns

[long](#)

## Heal(long)

```
public void Heal(long heal)
```

Parameters

heal [long](#)

## InitAI()

```
public virtual void InitAI()
```

## InitAIV1()

```
[Obsolete("Use InitAI")]
public void InitAIV1()
```

## InitAIV2()

```
[Obsolete("Use InitAI")]
public void InitAIV2()
```

## IsCritical(bool)

```
public bool IsCritical(bool considerAttackCount = true)
```

### Parameters

`considerAttackCount bool`

### Returns

bool

## IsHit(CharacterBase)

```
public virtual bool IsHit(CharacterBase caster)
```

Parameters

caster [CharacterBase](#)

Returns

[bool](#) ↗

## IsHit(ElementalResult)

```
public bool IsHit(ElementalResult result)
```

Parameters

result [ElementalResult](#)

Returns

[bool](#) ↗

## OnDead()

```
protected virtual void OnDead()
```

## OnPostSkill(Skill)

```
protected virtual void OnPostSkill(Skill usedSkill)
```

Parameters

usedSkill [Skill](#)

## OnPreSkill()

```
protected virtual bool OnPreSkill()
```

Returns

[bool](#)

## ReduceSkillCooldown()

```
protected virtual void ReduceSkillCooldown()
```

## RemoveActionBuff(ActionBuff)

```
public void RemoveActionBuff(ActionBuff removedBuff)
```

Parameters

removedBuff [ActionBuff](#)

## RemoveRecentStatBuff()

```
public void RemoveRecentStatBuff()
```

## RemoveStatBuff(StatBuff)

```
public void RemoveStatBuff(StatBuff removedBuff)
```

Parameters

removedBuff [StatBuff](#)

## ResetCurrentHP()

```
public void ResetCurrentHP()
```

## SetSkill()

```
protected virtual void SetSkill()
```

## Tick()

```
public void Tick()
```

## UseSkill()

```
protected virtual Skill UseSkill()
```

### Returns

[Skill](#)

# Class CollectionMap

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CollectionMap : IState, IDictionary<int, int>,
ICollection<KeyValuePair<int, int>>, IEnumerable<KeyValuePair<int, int>>,
IEnumerable, ISerializable
```

## Inheritance

[object](#) ← CollectionMap

## Implements

[IState](#), [IDictionary](#)<[int](#), [int](#)>, [ICollection](#)<[KeyValuePair](#)<[int](#), [int](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [int](#)>>, [IEnumerable](#), [ISerializable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### CollectionMap()

```
public CollectionMap()
```

### CollectionMap(Dictionary)

```
public CollectionMap(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

# Properties

## Count

Gets the number of elements contained in the [ICollection<T>](#).

```
public int Count { get; }
```

### Property Value

[int](#)

The number of elements contained in the [ICollection<T>](#).

## IsReadOnly

Gets a value indicating whether the [ICollection<T>](#) is read-only.

```
public bool IsReadOnly { get; }
```

### Property Value

[bool](#)

[true](#) if the [ICollection<T>](#) is read-only; otherwise, [false](#).

## this[int]

Gets or sets the element with the specified key.

```
public int this[int key] { get; set; }
```

### Parameters

**key** [int](#)

The key of the element to get or set.

## Property Value

### [int](#)

The element with the specified key.

## Exceptions

### [ArgumentNullException](#)

`key` is [null](#).

### [KeyNotFoundException](#)

The property is retrieved and `key` is not found.

### [NotSupportedException](#)

The property is set and the [IDictionary<TKey, TValue>](#) is read-only.

## Keys

Gets an [ICollection<T>](#) containing the keys of the [IDictionary<TKey, TValue>](#).

```
public ICollection<int> Keys { get; }
```

## Property Value

### [ICollection<int>](#)

An [ICollection<T>](#) containing the keys of the object that implements [IDictionary<TKey, TValue>](#).

## Values

Gets an [ICollection<T>](#) containing the values in the [IDictionary<TKey, TValue>](#).

```
public ICollection<int> Values { get; }
```

## Property Value

## [ICollection<int>](#)

An [ICollection<T>](#) containing the values in the object that implements [IDictionary< TKey, TValue >](#).

## Methods

### Add(KeyValuePair<int, int>)

Adds an item to the [ICollection<T>](#).

```
public void Add(KeyValuePair<int, int> item)
```

#### Parameters

**item** [KeyValuePair<int, int>](#)

The object to add to the [ICollection<T>](#).

#### Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

### Add(int, int)

Adds an element with the provided key and value to the [IDictionary< TKey, TValue >](#).

```
public void Add(int key, int value)
```

#### Parameters

**key** [int](#)

The object to use as the key of the element to add.

**value** [int](#)

The object to use as the value of the element to add.

## Exceptions

### [ArgumentNullException](#)

key is [null](#).

### [ArgumentException](#)

An element with the same key already exists in the [IDictionary<TKey, TValue>](#).

### [NotSupportedException](#)

The [IDictionary<TKey, TValue>](#) is read-only.

## Clear()

Removes all items from the [ICollection<T>](#).

```
public void Clear()
```

## Exceptions

### [NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Contains(KeyValuePair<int, int>)

Determines whether the [ICollection<T>](#) contains a specific value.

```
public bool Contains(KeyValuePair<int, int> item)
```

## Parameters

item [KeyValuePair](#)<[int](#), [int](#)>

The object to locate in the [ICollection<T>](#).

## Returns

[bool](#)

[true](#) if `item` is found in the [ICollection<T>](#); otherwise, [false](#).

## ContainsKey(int)

Determines whether the [IDictionary< TKey, TValue >](#) contains an element with the specified key.

```
public bool ContainsKey(int key)
```

### Parameters

`key` [int](#)

The key to locate in the [IDictionary< TKey, TValue >](#).

### Returns

[bool](#)

[true](#) if the [IDictionary< TKey, TValue >](#) contains an element with the key; otherwise, [false](#).

### Exceptions

[ArgumentNullException](#)

`key` is [null](#).

## CopyTo(KeyValuePair<int, int>[], int)

Copies the elements of the [ICollection<T>](#) to an [Array](#), starting at a particular [Array](#) index.

```
public void CopyTo(KeyValuePair<int, int>[] array, int arrayIndex)
```

### Parameters

`array` [KeyValuePair<int, int>\[\]](#)

The one-dimensional [Array](#) that is the destination of the elements copied from [ICollection<T>](#). The [Array](#) must have zero-based indexing.

`arrayIndex` [int](#)

The zero-based index in `array` at which copying begins.

## Exceptions

[ArgumentNullException](#)

`array` is [null](#).

[ArgumentOutOfRangeException](#)

`arrayIndex` is less than 0.

[ArgumentException](#)

The number of elements in the source [ICollection<T>](#) is greater than the available space from `arrayIndex` to the end of the destination `array`.

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumrator<KeyValuePair<int, int>> GetEnumerator()
```

Returns

[IEnumerator<KeyValuePair<int, int>>](#)

An enumerator that can be used to iterate through the collection.

## GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

`info` [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

`context` [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Remove(KeyValuePair<int, int>)

Removes the first occurrence of a specific object from the [ICollection<T>](#).

```
public bool Remove(KeyValuePair<int, int> item)
```

## Parameters

`item` [KeyValuePair](#)<[int](#), [int](#)>

The object to remove from the [ICollection<T>](#).

## Returns

[bool](#)

[true](#) if `item` was successfully removed from the [ICollection<T>](#); otherwise, [false](#). This method also returns [false](#) if `item` is not found in the original [ICollection<T>](#).

## Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Remove(int)

Removes the element with the specified key from the [IDictionary< TKey, TValue >](#).

```
public bool Remove(int key)
```

### Parameters

key [int](#)

The key of the element to remove.

### Returns

[bool](#)

[true](#) if the element is successfully removed; otherwise, [false](#). This method also returns [false](#) if [key](#) was not found in the original [IDictionary< TKey, TValue >](#).

### Exceptions

[ArgumentNullException](#)

[key](#) is [null](#).

[NotSupportedException](#)

The [IDictionary< TKey, TValue >](#) is read-only.

## Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# TryGetValue(int, out int)

Gets the value associated with the specified key.

```
public bool TryGetValue(int key, out int value)
```

## Parameters

**key** [int](#)

The key whose value to get.

**value** [int](#)

When this method returns, the value associated with the specified key, if the key is found; otherwise, the default value for the type of the **value** parameter. This parameter is passed uninitialized.

## Returns

[bool](#)

[true](#) if the object that implements [IDictionary<TKey, TValue>](#) contains an element with the specified key; otherwise, [false](#).

## Exceptions

[ArgumentNullException](#)

**key** is [null](#).

# Class Enemy

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Enemy : CharacterBase, ICloneable
```

## Inheritance

[object](#) ← [CharacterBase](#) ← Enemy

## Implements

[ICloneable](#)

## Derived

[RaidBoss](#)

## Inherited Members

[CharacterBase.CriticalMultiplier](#) , [CharacterBase.Id](#) , [CharacterBase.Simulator](#) ,  
[CharacterBase.atkElementType](#) , [CharacterBase.attackRange](#) ,  
[CharacterBase.defElementType](#) , [CharacterBase.Skills](#) , [CharacterBase.BuffSkills](#) ,  
[CharacterBase.Buffs](#) , [CharacterBase.StatBuffs](#) , [CharacterBase.ActionBuffs](#) ,  
[CharacterBase.Targets](#) , [CharacterBase.RowData](#) , [CharacterBase.CharacterId](#) ,  
[CharacterBase.SizeType](#) , [CharacterBase.RunSpeed](#) , [CharacterBase.Stats](#) ,  
[CharacterBase.Level](#) , [CharacterBase.HP](#) , [CharacterBase.AdditionalHP](#) ,  
[CharacterBase.ATK](#) , [CharacterBase.DEF](#) , [CharacterBase.CRI](#) , [CharacterBase.HIT](#) ,  
[CharacterBase.SPD](#) , [CharacterBase.DRV](#) , [CharacterBase.DRR](#) , [CharacterBase.CDMG](#) ,  
[CharacterBase.ArmorPenetration](#) , [CharacterBase.Thorn](#) , [CharacterBase.CurrentHP](#) ,  
[CharacterBase.IsDead](#) , [CharacterBase.AttackCount](#) , [CharacterBase.AttackCountMax](#) ,  
[CharacterBase.usedSkill](#) , [CharacterBase.InitAI\(\)](#) , [CharacterBase.InitAIV1\(\)](#) ,  
[CharacterBase.InitAIV2\(\)](#) , [CharacterBase.Tick\(\)](#) , [CharacterBase.ReduceSkillCooldown\(\)](#) ,  
[CharacterBase.UseSkill\(\)](#) , [CharacterBase.EndTurn\(\)](#) , [CharacterBase.AddBuff\(Buff, bool\)](#) ,  
[CharacterBase.RemoveActionBuff\(ActionBuff\)](#) , [CharacterBase.RemoveStatBuff\(StatBuff\)](#) ,  
[CharacterBase.RemoveRecentStatBuff\(\)](#) , [CharacterBase.ResetCurrentHP\(\)](#) ,  
[CharacterBase.IsCritical\(bool\)](#) , [CharacterBase.IsHit\(ElementalResult\)](#) ,  
[CharacterBase.IsHit\(CharacterBase\)](#) , [CharacterBase.GetDamage\(long, bool\)](#) ,  
[CharacterBase.Die\(\)](#) , [CharacterBase.Heal\(long\)](#) , [CharacterBase.GetChance\(int\)](#) ,  
[CharacterBase.OnPreSkill\(\)](#) , [CharacterBase.OnPostSkill\(Skill\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Enemy(CharacterBase, CharacterStats, Row, ElementalType)

```
public Enemy(CharacterBase player, CharacterStats stat, CharacterSheet.Row rowData,  
ElementalType elementType)
```

#### Parameters

player [CharacterBase](#)

stat [CharacterStats](#)

rowData [CharacterSheet.Row](#)

elementType [ElementalType](#)

### Enemy(CharacterBase, CharacterStats, int, ElementalType)

```
public Enemy(CharacterBase player, CharacterStats stat, int characterId,  
ElementalType elementType)
```

#### Parameters

player [CharacterBase](#)

stat [CharacterStats](#)

characterId [int](#)

elementType [ElementalType](#)

### Enemy(CharacterBase, Row, int, IEnumerable<StatModifier>)

```
public Enemy(CharacterBase player, CharacterSheet.Row rowData, int monsterLevel,  
IEnumerable<StatModifier> optionalStatModifiers = null)
```

## Parameters

player [CharacterBase](#)

rowData [CharacterSheet.Row](#)

monsterLevel [int](#)

optionalStatModifiers [IEnumerable](#)<[StatModifier](#)>

## Enemy(Enemy)

```
public Enemy(Enemy value)
```

## Parameters

value [Enemy](#)

## Enemy(Row)

```
public Enemy(CharacterSheet.Row rowData)
```

## Parameters

rowData [CharacterSheet.Row](#)

## Fields

### spawnIndex

```
public int spawnIndex
```

## Field Value

[int](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

### OnDead()

```
protected override void OnDead()
```

### SetSkill()

```
protected override void SetSkill()
```

# Class EnemyPlayer

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EnemyPlayer : Player, ICloneable
```

## Inheritance

[object](#) ← [CharacterBase](#) ← [Player](#) ← EnemyPlayer

## Implements

[ICloneable](#)

## Inherited Members

[Player.Exp](#) , [Player.Inventory](#) , [Player.worldInformation](#) , [Player.weapon](#) , [Player.armor](#) ,  
[Player.belt](#) , [Player.necklace](#) , [Player.ring](#) , [Player.aura](#) , [Player.Grimoire](#) ,  
[Player.monsterMap](#) , [Player.eventMap](#) , [Player.monsterMapForBeforeV100310](#) ,  
[Player.eventMapForBeforeV100310](#) , [Player.hairIndex](#) , [Player.lensIndex](#) , [Player.earIndex](#) ,  
[Player.tailIndex](#) , [Player.characterLevelSheet](#) , [Player.costumes](#) , [Player.equipments](#) ,  
[Player.RuneSkills](#) , [Player.RuneSkillCooldownMap](#) , [Player.Costumes](#) , [Player.Equipments](#) ,  
[Player.IsHit\(CharacterBase\)](#) , [Player.RemoveTarget\(Enemy\)](#) ,  
[Player.RemoveTarget\(EnemyPlayer\)](#) ,  
[Player.SetEquipmentStat\(EquipmentItemSetEffectSheet\)](#) , [Player.GetExp\(long, bool\)](#) ,  
[Player.GetExp2\(long, bool\)](#) , [Player.GetExp3\(long, bool\)](#) ,  
[Player.GetRewards\(List<ItemBase>\)](#) , [Player.GetRewards2\(List<ItemBase>\)](#) ,  
[Player.Use\(List<Guid>\)](#) , [Player.OverrideSkill\(Skill\)](#) , [Player.AddSkill\(Skill\)](#) , [Player.UseSkill\(\)](#) ,  
[Player.ReduceSkillCooldown\(\)](#) , [Player.SetCostumeStat\(CostumeStatSheet\)](#) ,  
[Player.SetRuneStats\(List<RuneState>, RuneOptionSheet, int\)](#) ,  
[Player.SetRuneSkills\(List<RuneState>, RuneOptionSheet, SkillSheet\)](#) ,  
[Player.SetCollections\(IEnumerable<StatModifier>\)](#) ,  
[Player.ConfigureStats\(CostumeStatSheet, List<RuneState>, RuneOptionSheet, int, SkillSheet, List<StatModifier>\)](#) ,  
[Player.SetRuneV1\(List<RuneState>, RuneOptionSheet, SkillSheet\)](#) , [Player.EndTurn\(\)](#) ,  
[CharacterBase.CriticalMultiplier](#) , [CharacterBase.Id](#) , [CharacterBase.Simulator](#) ,  
[CharacterBase.atkElementType](#) , [CharacterBase.attackRange](#) ,  
[CharacterBase.defElementType](#) , [CharacterBase.Skills](#) , [CharacterBase.BuffSkills](#) ,  
[CharacterBase.Buffs](#) , [CharacterBase.StatBuffs](#) , [CharacterBase.ActionBuffs](#) ,  
[CharacterBase.Targets](#) , [CharacterBase.RowData](#) , [CharacterBase.CharacterId](#) ,

[CharacterBase.SizeType](#) , [CharacterBase.RunSpeed](#) , [CharacterBaseStats](#) ,  
[CharacterBase.Level](#) , [CharacterBase.HP](#) , [CharacterBase.AdditionalHP](#) ,  
[CharacterBase.ATK](#) , [CharacterBase.DEF](#) , [CharacterBase.CRI](#) , [CharacterBase.HIT](#) ,  
[CharacterBase.SPD](#) , [CharacterBase.DRV](#) , [CharacterBase.DRR](#) , [CharacterBase.CDMG](#) ,  
[CharacterBase.ArmorPenetration](#) , [CharacterBase.Thorn](#) , [CharacterBase.CurrentHP](#) ,  
[CharacterBase.IsDead](#) , [CharacterBase.AttackCount](#) , [CharacterBase.AttackCountMax](#) ,  
[CharacterBase.usedSkill](#) , [CharacterBase.InitAI\(\)](#) , [CharacterBase.InitAIV1\(\)](#) ,  
[CharacterBase.InitAIV2\(\)](#) , [CharacterBase.Tick\(\)](#) , [CharacterBase.AddBuff\(Buff, bool\)](#) ,  
[CharacterBase.RemoveActionBuff\(ActionBuff\)](#) , [CharacterBase.RemoveStatBuff\(StatBuff\)](#) ,  
[CharacterBase.RemoveRecentStatBuff\(\)](#) , [CharacterBase.ResetCurrentHP\(\)](#) ,  
[CharacterBase.IsCritical\(bool\)](#) , [CharacterBase.IsHit\(ElementalResult\)](#) ,  
[CharacterBase.GetDamage\(long, bool\)](#) , [CharacterBase.Die\(\)](#) , [CharacterBase.Heal\(long\)](#) ,  
[CharacterBase.SetSkill\(\)](#) , [CharacterBase.GetChance\(int\)](#) , [CharacterBase.OnPreSkill\(\)](#) ,  
[CharacterBase.OnPostSkill\(Skill\)](#) , [object.Equals\(object\)↗](#) , [object.Equals\(object, object\)↗](#) ,  
[object.GetHashCode\(\)↗](#) , [object.GetType\(\)↗](#) , [object.MemberwiseClone\(\)↗](#) ,  
[object.ReferenceEquals\(object, object\)↗](#) , [object.ToString\(\)↗](#)

## Constructors

**EnemyPlayer(EnemyPlayerDigest, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet)**

```
public EnemyPlayer(EnemyPlayerDigest enemyPlayerDigest, CharacterSheet  
characterSheet, CharacterLevelSheet levelSheet, EquipmentItemSetEffectSheet  
equipmentItemSetEffectSheet)
```

## Parameters

enemyPlayerDigest [EnemyPlayerDigest](#)

characterSheet [CharacterSheet](#)

levelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

**EnemyPlayer(ArenaPlayerDigest, ArenaSimulatorSheetsV1)**

```
public EnemyPlayer(ArenaPlayerDigest arenaPlayerDigest, ArenaSimulatorSheetsV1  
simulatorSheets)
```

## Parameters

arenaPlayerDigest [ArenaPlayerDigest](#)

simulatorSheets [ArenaSimulatorSheetsV1](#)

## EnemyPlayer(AvatarState, Simulator)

```
public EnemyPlayer(AvatarState avatarState, Simulator simulator)
```

## Parameters

avatarState [AvatarState](#)

simulator [Simulator](#)

## EnemyPlayer(AvatarState, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet)

```
public EnemyPlayer(AvatarState avatarState, CharacterSheet characterSheet,  
CharacterLevelSheet characterLevelSheet, EquipmentItemSetEffectSheet  
equipmentItemSetEffectSheet)
```

## Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

## EnemyPlayer(AvatarState, SimulatorSheetsV1)

```
public EnemyPlayer(AvatarState avatarState, SimulatorSheetsV1 simulatorSheets)
```

### Parameters

avatarState [AvatarState](#)

simulatorSheets [SimulatorSheetsV1](#)

## EnemyPlayer(int, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet)

```
public EnemyPlayer(int level, CharacterSheet characterSheet, CharacterLevelSheet  
characterLevelSheet, EquipmentItemSetEffectSheet equipmentItemSetEffectSheet)
```

### Parameters

level [int](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

## Fields

### NameWithHash

```
public readonly string NameWithHash
```

### Field Value

[string](#)

# Methods

## Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## OnDead()

```
protected override void OnDead()
```

## Spawn()

```
public override void Spawn()
```

## SpawnV1()

```
[Obsolete("Use Spawn")]
public override void SpawnV1()
```

## SpawnV2()

```
[Obsolete("Use Spawn")]
public override void SpawnV2()
```

# Class FailedAddWorldException

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FailedAddWorldException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← FailedAddWorldException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### FailedAddWorldException(SerializationInfo, StreamingContext)

```
protected FailedAddWorldException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## FailedAddWorldException(string)

```
public FailedAddWorldException(string message)
```

### Parameters

message [string](#)

# Class FailedToUnlockWorldException

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FailedToUnlockWorldException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← FailedToUnlockWorldException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### FailedToUnlockWorldException(SerializationInfo, StreamingContext)

```
protected FailedToUnlockWorldException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## FailedToUnlockWorldException(string)

```
public FailedToUnlockWorldException(string message)
```

### Parameters

message [string](#)

# Struct FungibleItemValue

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
public readonly struct FungibleItemValue
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### FungibleItemValue(List)

```
public FungibleItemValue(List bencoded)
```

#### Parameters

bencoded List

### FungibleItemValue(HashDigest<SHA256>, int)

```
public FungibleItemValue(HashDigest<SHA256> id, int count)
```

#### Parameters

id HashDigest<[SHA256](#)>

count [int](#)

## Properties

## Count

```
public int Count { get; }
```

Property Value

[int](#)

## Id

```
public HashDigest<SHA256> Id { get; }
```

Property Value

HashDigest<[SHA256](#)>

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Interface IArena

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
public interface IArena
```

## Methods

`CoAreaAttack(ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)`

```
IEnumerator CoAreaAttack(ArenaCharacter caster,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

### Parameters

`caster` [ArenaCharacter](#)

`skillInfos` [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

`buffInfos` [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

### Returns

[IEnumerator](#)

`CoBlowAttack(ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)`

```
IEnumerator CoBlowAttack(ArenaCharacter caster,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,
```

```
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

## Returns

[IEnumerator](#)

## CoBuff(ArenaCharacter, IEnumerable<ArenaSkillInfo>, IEnumerable<ArenaSkillInfo>)

```
IEnumerator CoBuff(ArenaCharacter caster, IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos, IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

## Returns

[IEnumerator](#)

## CoBuffRemovalAttack(ArenaCharacter, IEnumerable<ArenaSkillInfo>, IEnumerable<ArenaSkillInfo>)

```
IEnumerator CoBuffRemovalAttack(ArenaCharacter caster, IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,
```

IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)

## Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<ArenaSkill.ArenaSkillInfo>

buffInfos [IEnumerable](#)<ArenaSkill.ArenaSkillInfo>

## Returns

[IEnumerator](#)

## CoCustomEvent(ArenaCharacter, ArenaEventBase)

IEnumerator **CoCustomEvent**(ArenaCharacter caster, ArenaEventBase eventBase)

## Parameters

caster [ArenaCharacter](#)

eventBase [ArenaEventBase](#)

## Returns

[IEnumerator](#)

## CoDead(ArenaCharacter)

IEnumerator **CoDead**(ArenaCharacter caster)

## Parameters

caster [ArenaCharacter](#)

## Returns

## [IEnumerator](#)

CoDoubleAttack(ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
IEnumerator CoDoubleAttack(ArenaCharacter caster,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

Returns

[IEnumerator](#)

CoDoubleAttackWithCombo(ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
IEnumerator CoDoubleAttackWithCombo(ArenaCharacter caster,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

Returns

[IEnumerator](#)

## CoHeal(ArenaCharacter, IEnumerable<ArenaSkillInfo>, IEnumerable<ArenaSkillInfo>)

`IEnumerator CoHeal(ArenaCharacter caster, IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos, IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)`

Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

Returns

[IEnumerator](#)

## CoNormalAttack(ArenaCharacter, IEnumerable<ArenaSkillInfo>, IEnumerable<ArenaSkillInfo>)

`IEnumerator CoNormalAttack(ArenaCharacter caster, IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos, IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)`

Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

Returns

[IEnumerator](#)

## CoRemoveBuffs(ArenaCharacter)

IEnumerator **CoRemoveBuffs**(ArenaCharacter caster)

Parameters

caster [ArenaCharacter](#)

Returns

[IEnumerator](#)

## CoShatterStrike(ArenaCharacter, IEnumerable<ArenaSkillInfo>, IEnumerable<ArenaSkillInfo>)

IEnumerator **CoShatterStrike**(ArenaCharacter caster,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)

Parameters

caster [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill](#).[ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill](#).[ArenaSkillInfo](#)>

Returns

[IEnumerator](#)

## CoSpawnCharacter(ArenaCharacter)

IEnumerator **CoSpawnCharacter**(ArenaCharacter character)

Parameters

character [ArenaCharacter](#)

Returns

[IEnumerator](#)

## CoTickDamage(ArenaCharacter, IEnumerable<ArenaSkillInfo>)

IEnumerator **CoTickDamage**(ArenaCharacter affectedCharacter,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos)

Parameters

affectedCharacter [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill](#).[ArenaSkillInfo](#)>

Returns

[IEnumerator](#)

## CoTurnEnd(int)

IEnumerator **CoTurnEnd**(int turnNumber)

Parameters

turnNumber [int](#)

Returns

[IEnumerator](#)

# Interface IStage

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
public interface IStage
```

## Methods

**CoAreaAttack(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)**

```
IEnumerator CoAreaAttack(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

### Parameters

**caster** [CharacterBase](#)

**skillId** [int](#)

**skillInfos** [IEnumerable](#)<[Skill.SkillInfo](#)>

**buffInfos** [IEnumerable](#)<[Skill.SkillInfo](#)>

### Returns

[IEnumerator](#)

**CoBlowAttack(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)**

```
IEnumerator CoBlowAttack(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

**CoBreakthrough(CharacterBase, int, List<MonsterData>)**

[IEnumerator](#) **CoBreakthrough**(CharacterBase character, [int](#) floorId,  
[List](#)<[AdventureBossFloorWaveSheet.MonsterData](#)> monsters)

Parameters

character [CharacterBase](#)

floorId [int](#)

monsters [List](#)<[AdventureBossFloorWaveSheet.MonsterData](#)>

Returns

[IEnumerator](#)

**CoBuff(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)**

[IEnumerator](#) **CoBuff**(CharacterBase caster, [int](#) skillId, [IEnumerable](#)<[Skill.SkillInfo](#)>  
skillInfos, [IEnumerable](#)<[Skill.SkillInfo](#)> buffInfos)

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

**CoBuffRemovalAttack(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)**

```
IEnumerator CoBuffRemovalAttack(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

**CoCustomEvent(CharacterBase, EventBase)**

```
IEnumerator CoCustomEvent(CharacterBase character, EventBase eventBase)
```

Parameters

character [CharacterBase](#)

eventBase [EventBase](#)

Returns

[IEnumerator](#)

## CoDead(CharacterBase)

`IEnumerator CoDead(CharacterBase character)`

Parameters

character [CharacterBase](#)

Returns

[IEnumerator](#)

## CoDoubleAttack(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

`IEnumerator CoDoubleAttack(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)`

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

**CoDoubleAttackWithCombo(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)**

```
IEnumerator CoDoubleAttackWithCombo(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

**CoDropBox(List<ItemBase>)**

```
IEnumerator CoDropBox(List<ItemBase> items)
```

Parameters

items [List](#)<[ItemBase](#)>

Returns

[IEnumerator](#)

**CoGetExp(long)**

```
IEnumerator CoGetExp(long exp)
```

Parameters

exp [long](#)

Returns

[IEnumerator](#)

## CoGetReward(List<ItemBase>)

```
IEnumerator CoGetReward(List<ItemBase> rewards)
```

Parameters

rewards [List](#)<[ItemBase](#)>

Returns

[IEnumerator](#)

## CoHeal(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
IEnumerator CoHeal(CharacterBase caster, int skillId, IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

## CoNormalAttack(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
IEnumerator CoNormalAttack(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

## CoRemoveBuffs(CharacterBase)

```
IEnumerator CoRemoveBuffs(CharacterBase caster)
```

Parameters

caster [CharacterBase](#)

Returns

[IEnumerator](#)

## CoShatterStrike(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
IEnumerator CoShatterStrike(CharacterBase caster, int skillId,  
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

### Parameters

caster [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

### Returns

[IEnumerator](#)

## CoSpawnEnemyPlayer(EnemyPlayer)

```
IEnumerator CoSpawnEnemyPlayer(EnemyPlayer character)
```

### Parameters

character [EnemyPlayer](#)

### Returns

[IEnumerator](#)

## CoSpawnPlayer(Player)

```
IEnumerator CoSpawnPlayer(Player character)
```

### Parameters

character [Player](#)

Returns

[IEnumerator](#)

## CoSpawnWave(int, int, List<Enemy>, bool)

IEnumerator **CoSpawnWave**(int waveNumber, int waveTurn, List<Enemy> enemies, bool hasBoss)

Parameters

waveNumber [int](#)

waveTurn [int](#)

enemies [List](#)<Enemy>

hasBoss [bool](#)

Returns

[IEnumerator](#)

## CoStageBuff(CharacterBase, int, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

IEnumerator **CoStageBuff**(CharacterBase affected, int skillId, IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)

Parameters

affected [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<Skill.SkillInfo>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

## CoTickDamage(CharacterBase, int, IEnumerable<SkillInfo>)

IEnumerator [CoTickDamage](#)(CharacterBase affectedCharacter, [int](#) skillId, [IEnumerable](#)<[Skill.SkillInfo](#)> skillInfos)

Parameters

affectedCharacter [CharacterBase](#)

skillId [int](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

Returns

[IEnumerator](#)

## CoWaveTurnEnd(int, int)

IEnumerator [CoWaveTurnEnd](#)([int](#) turnNumber, [int](#) waveTurn)

Parameters

turnNumber [int](#)

waveTurn [int](#)

Returns

[IEnumerator](#)

# Class Player

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Player : CharacterBase, ICloneable
```

## Inheritance

[object](#) ← [CharacterBase](#) ← Player

## Implements

[ICloneable](#)

## Derived

[EnemyPlayer](#)

## Inherited Members

[CharacterBase.CriticalMultiplier](#) , [CharacterBase.Id](#) , [CharacterBase.Simulator](#) ,  
[CharacterBase.atkElementType](#) , [CharacterBase.attackRange](#) ,  
[CharacterBase.defElementType](#) , [CharacterBase.Skills](#) , [CharacterBase.BuffSkills](#) ,  
[CharacterBase.Buffs](#) , [CharacterBase.StatBuffs](#) , [CharacterBase.ActionBuffs](#) ,  
[CharacterBase.Targets](#) , [CharacterBase.RowData](#) , [CharacterBase.CharacterId](#) ,  
[CharacterBase.SizeType](#) , [CharacterBase.RunSpeed](#) , [CharacterBase.Stats](#) ,  
[CharacterBase.Level](#) , [CharacterBase.HP](#) , [CharacterBase.AdditionalHP](#) ,  
[CharacterBase.ATK](#) , [CharacterBase.DEF](#) , [CharacterBase.CRI](#) , [CharacterBase.HIT](#) ,  
[CharacterBase.SPD](#) , [CharacterBase.DRV](#) , [CharacterBase.DRR](#) , [CharacterBase.CDMG](#) ,  
[CharacterBase.ArmorPenetration](#) , [CharacterBase.Thorn](#) , [CharacterBase.CurrentHP](#) ,  
[CharacterBase.IsDead](#) , [CharacterBase.AttackCount](#) , [CharacterBase.AttackCountMax](#) ,  
[CharacterBase.usedSkill](#) , [CharacterBase.InitAI\(\)](#) , [CharacterBase.InitAIV1\(\)](#) ,  
[CharacterBase.InitAIV2\(\)](#) , [CharacterBase.Tick\(\)](#) , [CharacterBase.AddBuff\(Buff, bool\)](#) ,  
[CharacterBase.RemoveActionBuff\(ActionBuff\)](#) , [CharacterBase.RemoveStatBuff\(StatBuff\)](#) ,  
[CharacterBase.RemoveRecentStatBuff\(\)](#) , [CharacterBase.ResetCurrentHP\(\)](#) ,  
[CharacterBase.IsCritical\(bool\)](#) , [CharacterBase.IsHit\(ElementalResult\)](#) ,  
[CharacterBase.GetDamage\(long, bool\)](#) , [CharacterBase.Die\(\)](#) , [CharacterBase.Heal\(long\)](#) ,  
[CharacterBase.SetSkill\(\)](#) , [CharacterBase.GetChance\(int\)](#) , [CharacterBase.OnPreSkill\(\)](#) ,  
[CharacterBase.OnPostSkill\(Skill\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Player(ArenaPlayerDigest, ArenaSimulatorSheetsV1)

```
public Player(ArenaPlayerDigest enemyArenaPlayerDigest, ArenaSimulatorSheetsV1  
simulatorSheets)
```

#### Parameters

enemyArenaPlayerDigest [ArenaPlayerDigest](#)

simulatorSheets [ArenaSimulatorSheetsV1](#)

### Player(Player)

```
protected Player(Player value)
```

#### Parameters

value [Player](#)

### Player(AvatarState, Simulator)

```
public Player(AvatarState avatarState, Simulator simulator)
```

#### Parameters

avatarState [AvatarState](#)

simulator [Simulator](#)

### Player(AvatarState, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet)

```
public Player(AvatarState avatarState, CharacterSheet characterSheet,  
CharacterLevelSheet characterLevelSheet, EquipmentItemSetEffectSheet  
equipmentItemSetEffectSheet)
```

## Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

## Player(AvatarState, SimulatorSheetsV1)

```
public Player(AvatarState avatarState, SimulatorSheetsV1 simulatorSheets)
```

## Parameters

avatarState [AvatarState](#)

simulatorSheets [SimulatorSheetsV1](#)

## Player(int, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet)

```
public Player(int level, CharacterSheet characterSheet, CharacterLevelSheet  
characterLevelSheet, EquipmentItemSetEffectSheet equipmentItemSetEffectSheet)
```

## Parameters

level [int](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

## Fields

### Exp

```
public readonly Player.ExpData Exp
```

#### Field Value

[Player.ExpData](#)

### Grimoire

```
public Grimoire Grimoire
```

#### Field Value

[Grimoire](#)

### Inventory

```
public readonly Inventory Inventory
```

#### Field Value

[Inventory](#)

### RuneSkillCooldownMap

```
public readonly Dictionary<int, int> RuneSkillCooldownMap
```

#### Field Value

[Dictionary](#)<[int](#), [int](#)>

## RuneSkills

`public readonly Skills RuneSkills`

Field Value

[Skills](#)

## armor

`public Armor armor`

Field Value

[Armor](#)

## aura

`public Aura aura`

Field Value

[Aura](#)

## belt

`public Belt belt`

Field Value

[Belt](#)

## characterLevelSheet

```
public CharacterLevelSheet characterLevelSheet
```

### Field Value

[CharacterLevelSheet](#)

## costumes

```
protected List<Costume> costumes
```

### Field Value

[List](#) <[Costume](#)>

## earIndex

```
public int earIndex
```

### Field Value

[int](#)

## equipments

```
protected List<Equipment> equipments
```

### Field Value

[List](#) <[Equipment](#)>

## eventMap

```
public CollectionMap eventMap
```

Field Value

[CollectionMap](#)

## eventMapForBeforeV100310

```
public CollectionMap eventMapForBeforeV100310
```

Field Value

[CollectionMap](#)

## hairIndex

```
public int hairIndex
```

Field Value

[int](#)

## lensIndex

```
public int lensIndex
```

Field Value

[int](#)

## monsterMap

```
public CollectionMap monsterMap
```

Field Value

[CollectionMap](#)

## monsterMapForBeforeV100310

```
public CollectionMap monsterMapForBeforeV100310
```

Field Value

[CollectionMap](#)

## necklace

```
public Necklace necklace
```

Field Value

[Necklace](#)

## ring

```
public Ring ring
```

Field Value

[Ring](#)

## tailIndex

```
public int tailIndex
```

Field Value

[int](#)

weapon

```
public Weapon weapon
```

Field Value

[Weapon](#)

worldInformation

```
public WorldInformation worldInformation
```

Field Value

[WorldInformation](#)

Properties

Costumes

```
public IReadOnlyList<Costume> Costumes { get; }
```

Property Value

[IReadOnlyList](#)<[Costume](#)>

Equipments

```
public IReadOnlyList<Equipment> Equipments { get; }
```

Property Value

[IReadOnlyList](#)<[Equipment](#)>

## Methods

### AddSkill(Skill)

```
public void AddSkill(Skill skill)
```

Parameters

[skill](#) [Skill](#)

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

### ConfigureStats(CostumeStatSheet, List<RuneState>, RuneOptionSheet, int, SkillSheet, List<StatModifier>)

```
public void ConfigureStats(CostumeStatSheet costumeStatSheet, List<RuneState> runeStates, RuneOptionSheet runeOptionSheet, int runeLevelBonus, SkillSheet skillSheet, List<StatModifier> collectionModifiers)
```

## Parameters

costumeStatSheet [CostumeStatSheet](#)

runeStates [List](#)<[RuneState](#)>

runeOptionSheet [RuneOptionSheet](#)

runeLevelBonus [int](#)

skillSheet [SkillSheet](#)

collectionModifiers [List](#)<[StatModifier](#)>

## EndTurn()

```
protected override void EndTurn()
```

## GetExp(long, bool)

```
public void GetExp(long waveExp, bool log = false)
```

## Parameters

waveExp [long](#)

log [bool](#)

## GetExp2(long, bool)

```
[Obsolete("Use GetExp")]
```

```
public void GetExp2(long waveExp, bool log = false)
```

## Parameters

waveExp [long](#)

log [bool](#)

## GetExp3(long, bool)

```
[Obsolete("Use GetExp")]
public void GetExp3(long waveExp, bool log = false)
```

### Parameters

waveExp [long](#)

log [bool](#)

## GetRewards(List<ItemBase>)

```
public CollectionMap GetRewards(List<ItemBase> items)
```

### Parameters

items [List](#)<[ItemBase](#)>

### Returns

[CollectionMap](#)

## GetRewards2(List<ItemBase>)

```
[Obsolete("Use GetRewards")]
public CollectionMap GetRewards2(List<ItemBase> items)
```

### Parameters

items [List](#)<[ItemBase](#)>

### Returns

## IsHit(CharacterBase)

```
public override bool IsHit(CharacterBase caster)
```

Parameters

`caster` [CharacterBase](#)

Returns

`bool` ↗

## OnDead()

```
protected override void OnDead()
```

## OverrideSkill(Skill)

```
public void OverrideSkill(Skill skill)
```

Parameters

`skill` [Skill](#)

## ReduceSkillCooldown()

```
protected override void ReduceSkillCooldown()
```

## RemoveTarget(Enemy)

```
public void RemoveTarget(Enemy enemy)
```

Parameters

enemy [Enemy](#)

## RemoveTarget(EnemyPlayer)

```
public void RemoveTarget(EnemyPlayer enemy)
```

Parameters

enemy [EnemyPlayer](#)

## SetCollections(IEnumerable<StatModifier>)

```
public void SetCollections(IEnumerable<StatModifier> statModifiers)
```

Parameters

statModifiers [IEnumerable](#)<[StatModifier](#)>

## SetCostumeStat(CostumeStatSheet)

```
public void SetCostumeStat(CostumeStatSheet costumeStatSheet)
```

Parameters

costumeStatSheet [CostumeStatSheet](#)

## SetEquipmentStat(EquipmentItemSetEffectSheet)

```
protected void SetEquipmentStat(EquipmentItemSetEffectSheet sheet)
```

## Parameters

sheet [EquipmentItemSetEffectSheet](#)

## SetRuneSkills(List<RuneState>, RuneOptionSheet, SkillSheet)

Sets the rune skills for the player.

```
public void SetRuneSkills(List<RuneState> runes, RuneOptionSheet runeOptionSheet, SkillSheet skillSheet)
```

## Parameters

runes [List<RuneState>](#)

The list of rune states.

runeOptionSheet [RuneOptionSheet](#)

The rune option sheet.

skillSheet [SkillSheet](#)

The skill sheet.

## SetRuneStats(List<RuneState>, RuneOptionSheet, int)

Sets the rune stats for a player character.

```
public void SetRuneStats(List<RuneState> runes, RuneOptionSheet runeOptionSheet, int runeLevelBonus)
```

## Parameters

runes [List<RuneState>](#)

The AllRuneState for the player character.

#### runeOptionSheet [RuneOptionSheet](#)

The rune option sheet that contains information about rune options.

#### runeLevelBonus [int](#)

The rune level bonus value from RuneLevelBonusSheet. This enhances equipped rune stats.

## SetRuneV1(List<RuneState>, RuneOptionSheet, SkillSheet)

```
[Obsolete("Use SetRune")]
public void SetRuneV1(List<RuneState> runes, RuneOptionSheet runeOptionSheet,
SkillSheet skillSheet)
```

### Parameters

#### runes [List](#)<RuneState>

#### runeOptionSheet [RuneOptionSheet](#)

#### skillSheet [SkillSheet](#)

## Spawn()

```
public virtual void Spawn()
```

## SpawnV1()

```
[Obsolete("Use Spawn")]
public virtual void SpawnV1()
```

## SpawnV2()

```
[Obsolete("Use Spawn")]
public virtual void SpawnV2()
```

## Use(List<Guid>)

```
public void Use(List<Guid> consumableIds)
```

### Parameters

consumableIds [List](#)<[Guid](#)>

## UseSkill()

```
protected override Skill UseSkill()
```

### Returns

[Skill](#)

# Class Player.ExpData

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Player.ExpData : ICloneable
```

## Inheritance

[object](#) ← Player.ExpData

## Implements

[ICloneable](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ExpData()

```
public ExpData()
```

### ExpData(ExpData)

```
protected ExpData(Player.ExpData value)
```

## Parameters

value [Player.ExpData](#)

## Properties

## Current

```
public long Current { get; set; }
```

Property Value

[long](#) ↗

## Max

```
public long Max { get; }
```

Property Value

[long](#) ↗

## Need

```
public long Need { get; }
```

Property Value

[long](#) ↗

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#) ↗

A new object that is a copy of this instance.

## Set(Row)

```
public void Set(CharacterLevelSheet.Row row)
```

Parameters

row [CharacterLevelSheet.Row](#)

# Class RaidBoss

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
public class RaidBoss : Enemy, ICloneable
```

## Inheritance

[object](#) ← [CharacterBase](#) ← [Enemy](#) ← RaidBoss

## Implements

[ICloneable](#)

## Inherited Members

[Enemy.spawnIndex](#) , [Enemy.OnDead\(\)](#) , [CharacterBase.CriticalMultiplier](#) , [CharacterBase.Id](#) ,  
[CharacterBase.Simulator](#) , [CharacterBase.atkElementType](#) , [CharacterBase.attackRange](#) ,  
[CharacterBase.defElementType](#) , [CharacterBase.Skills](#) , [CharacterBase.BuffSkills](#) ,  
[CharacterBase.Buffs](#) , [CharacterBase.StatBuffs](#) , [CharacterBase.ActionBuffs](#) ,  
[CharacterBase.Targets](#) , [CharacterBase.CharacterId](#) , [CharacterBase.SizeType](#) ,  
[CharacterBase.RunSpeed](#) , [CharacterBase.Stats](#) , [CharacterBase.Level](#) , [CharacterBase.HP](#) ,  
[CharacterBase.AdditionalHP](#) , [CharacterBase.ATK](#) , [CharacterBase.DEF](#) , [CharacterBase.CRI](#) ,  
[CharacterBase.HIT](#) , [CharacterBase.SPD](#) , [CharacterBase.DRV](#) , [CharacterBase.DRR](#) ,  
[CharacterBase.CDMG](#) , [CharacterBase.ArmorPenetration](#) , [CharacterBase.Thorn](#) ,  
[CharacterBase.CurrentHP](#) , [CharacterBase.IsDead](#) , [CharacterBase.AttackCount](#) ,  
[CharacterBase.AttackCountMax](#) , [CharacterBase.usedSkill](#) , [CharacterBase.InitAI\(\)](#) ,  
[CharacterBase.InitAIV1\(\)](#) , [CharacterBase.InitAIV2\(\)](#) , [CharacterBase.Tick\(\)](#) ,  
[CharacterBase.ReduceSkillCooldown\(\)](#) , [CharacterBase.AddBuff\(Buff, bool\)](#) ,  
[CharacterBase.RemoveActionBuff\(ActionBuff\)](#) , [CharacterBase.RemoveStatBuff\(StatBuff\)](#) ,  
[CharacterBase.RemoveRecentStatBuff\(\)](#) , [CharacterBase.ResetCurrentHP\(\)](#) ,  
[CharacterBase.IsCritical\(bool\)](#) , [CharacterBase.IsHit\(ElementalResult\)](#) ,  
[CharacterBase.GetDamage\(long, bool\)](#) , [CharacterBase.Die\(\)](#) , [CharacterBase.Heal\(long\)](#) ,  
[CharacterBase.GetChance\(int\)](#) , [CharacterBase.OnPreSkill\(\)](#) ,  
[CharacterBase.OnPostSkill\(Skill\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

## RaidBoss(CharacterBase, Row, Row, WaveStatData, bool)

```
public RaidBoss(CharacterBase player, WorldBossCharacterSheet.Row characterRow,
WorldBossActionPatternSheet.Row patternRow, WorldBossCharacterSheet.WaveStatData
statData, bool ignoreLevelCorrectionOnHit)
```

### Parameters

player [CharacterBase](#)

characterRow [WorldBossCharacterSheet.Row](#)

patternRow [WorldBossActionPatternSheet.Row](#)

statData [WorldBossCharacterSheet.WaveStatData](#)

ignoreLevelCorrectionOnHit [bool](#) ↗

## RaidBoss(RaidBoss)

```
public RaidBoss(RaidBoss value)
```

### Parameters

value [RaidBoss](#)

## Properties

### Enraged

```
public bool Enraged { get; protected set; }
```

### Property Value

[bool](#) ↗

## IgnoreLevelCorrectionOnHit

```
public bool IgnoreLevelCorrectionOnHit { get; protected set; }
```

Property Value

[bool](#)

## PatternRowData

```
public WorldBossActionPatternSheet.Row PatternRowData { get; }
```

Property Value

[WorldBossActionPatternSheet.Row](#)

## RowData

```
public WorldBossCharacterSheet.Row RowData { get; }
```

Property Value

[WorldBossCharacterSheet.Row](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## EndTurn()

```
protected override void EndTurn()
```

## Enrage()

```
public void Enrage()
```

## IsHit(CharacterBase)

```
public override bool IsHit(CharacterBase caster)
```

Parameters

**caster** [CharacterBase](#)

Returns

[bool](#) ↗

## SetSkill()

```
protected override void SetSkill()
```

## UseSkill()

```
protected override Skill UseSkill()
```

Returns



# Class Skills

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Skills : IEnumerable<Skill>, IEnumerable
```

## Inheritance

[object](#) ← Skills

## Implements

[IEnumerable](#)<[Skill](#)>, [IEnumerable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

# Methods

## Add(Skill)

```
public void Add(Skill skill)
```

## Parameters

skill [Skill](#)

## Clear()

```
public void Clear()
```

## GetCooldown(int)

```
public int GetCooldown(int skillId)
```

Parameters

skillId [int](#)

Returns

[int](#)

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumarator<Skill> GetEnumerator()
```

Returns

[IEnumarator](#)<[Skill](#)>

An enumerator that can be used to iterate through the collection.

## ReduceCooldown()

```
public void ReduceCooldown()
```

## ReduceCooldownV1()

```
[Obsolete("ReduceCooldown")]
public void ReduceCooldownV1()
```

## Select(IRandom)

```
public Skill Select(IRandom random)
```

Parameters

random IRandom

Returns

[Skill](#)

## SelectV1(IRandom)

```
[Obsolete("Use Select")]
public Skill SelectV1(IRandom random)
```

Parameters

random IRandom

Returns

[Skill](#)

## SelectV2(IRandom)

```
[Obsolete("Use Select")]
public Skill SelectV2(IRandom random)
```

Parameters

random IRandom

Returns

[Skill](#)

## SelectWithoutDefaultAttack(IRandom)

```
public Skill SelectWithoutDefaultAttack(IRandom random)
```

Parameters

random IRandom

Returns

[Skill](#)

## SetCooldown(int, int)

```
public void SetCooldown(int skillId, int cooldown)
```

Parameters

skillId [int](#)

cooldown [int](#)

# Class WorldInformation

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldInformation : IState, ISerializable, ICloneable
```

## Inheritance

[object](#) ← WorldInformation

## Implements

[IState](#), [ISerializable](#), [ICloneable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Extension Methods

[WorldInformationExtensions.ValidateFromAction\(WorldInformation, int, string, string\)](#)

## Constructors

### WorldInformation(Dictionary)

```
public WorldInformation(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### WorldInformation(long, WorldSheet, bool, string)

```
public WorldInformation(long blockIndex, WorldSheet worldSheet, bool
openAllOfWorldsAndStages = false, string avatarName = "")
```

## Parameters

blockIndex [long](#)

worldSheet [WorldSheet](#)

openAllOfWorldsAndStages [bool](#)

avatarName [string](#)

## WorldInformation(long, WorldSheet, int)

```
public WorldInformation(long blockIndex, WorldSheet worldSheet, int clearStageId  
= 0)
```

## Parameters

blockIndex [long](#)

worldSheet [WorldSheet](#)

clearStageId [int](#)

## Methods

### AddAndUnlockMimisbrunnrWorld(Row, long, WorldSheet, WorldUnlockSheet)

```
public void AddAndUnlockMimisbrunnrWorld(WorldSheet.Row worldRow, long unlockedAt,  
WorldSheet worldSheet, WorldUnlockSheet worldUnlockSheet)
```

## Parameters

worldRow [WorldSheet.Row](#)

unlockedAt [long](#)

worldSheet [WorldSheet](#)

worldUnlockSheet [WorldUnlockSheet](#)

## AddAndUnlockNewWorld(Row, long, WorldSheet)

```
public void AddAndUnlockNewWorld(WorldSheet.Row worldRow, long unlockedAt,  
WorldSheet worldSheet)
```

### Parameters

worldRow [WorldSheet.Row](#)

unlockedAt [long](#)

worldSheet [WorldSheet](#)

## ClearStage(int, int, long, WorldSheet, WorldUnlockSheet)

Clear a specific stage. And consider world unlock.

```
public void ClearStage(int worldId, int stageId, long clearedAt, WorldSheet  
worldSheet, WorldUnlockSheet worldUnlockSheet)
```

### Parameters

worldId [int](#)

stageId [int](#)

clearedAt [long](#)

worldSheet [WorldSheet](#)

worldUnlockSheet [WorldUnlockSheet](#)

### Exceptions

[FailedToUnlockWorldException](#)

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

Parameters

[info](#) [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

[context](#) [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

Exceptions

[SecurityException](#)

The caller does not have the required permission.

## IsStageCleared(int)

```
public bool IsStageCleared(int stageId)
```

Parameters

`stageId` [int](#)

Returns

[bool](#)

## IsWorldUnlocked(int)

```
public bool IsWorldUnlocked(int worldId)
```

Parameters

`worldId` [int](#)

Returns

[bool](#)

## Serialize()

```
public IValue Serialize()
```

Returns

[IValue](#)

## TryAddWorld(Row, out World)

```
public bool TryAddWorld(WorldSheet.Row worldRow, out WorldInformation.World world)
```

Parameters

`worldRow` [WorldSheet.Row](#)

`world` [WorldInformation.World](#)

Returns

[bool](#) ↗

## TryGetFirstWorld(out World)

```
public bool TryGetFirstWorld(out WorldInformation.World world)
```

Parameters

`world` [WorldInformation.World](#)

Returns

[bool](#) ↗

## TryGetLastClearedMimisbrunnrStageId(out int)

```
public bool TryGetLastClearedMimisbrunnrStageId(out int stageId)
```

Parameters

`stageId` [int](#) ↗

Returns

[bool](#) ↗

## TryGetLastClearedStageId(out int)

Get stage id of the most recent cleared.(ignore mimisbrunnr)

```
public bool TryGetLastClearedStageId(out int stageId)
```

Parameters

`stageId` [int](#)

Returns

[bool](#)

## TryGetUnlockedWorldByStageClearedBlockIndex([out World](#))

Get [World](#) object that contains the most recent stage clear.

```
public bool TryGetUnlockedWorldByStageClearedBlockIndex(out WorldInformation.World world\)
```

Parameters

`world` [WorldInformation.World](#)

Returns

[bool](#)

## TryGetWorld([int](#), [out World](#))

Get [World](#) object that equals to `worldId` argument.

```
public bool TryGetWorld(int worldId, out WorldInformation.World world)
```

Parameters

`worldId` [int](#)

`world` [WorldInformation.World](#)

Returns

[bool](#)

## Exceptions

[KeyNotFoundException](#)

## TryGetWorldByStageId(int, out World)

Get `World` object that contains `stageId` argument.

```
public bool TryGetWorldByStageId(int stageId, out WorldInformation.World world)
```

## Parameters

`stageId` [int](#)

`world` [WorldInformation.World](#)

## Returns

[bool](#)

## UnlockWorld(int, long, WorldSheet)

Unlock a specific world.

```
public void UnlockWorld(int worldId, long unlockedAt, WorldSheet worldSheet)
```

## Parameters

`worldId` [int](#)

`unlockedAt` [long](#)

`worldSheet` [WorldSheet](#)

## Exceptions

[FailedToUnlockWorldException](#)

## UpdateWorld(Row)

```
public void UpdateWorld(WorldSheet.Row worldRow)
```

### Parameters

worldRow [WorldSheet.Row](#)

# Struct WorldInformation.World

Namespace: [Nekoyume.Model](#)

Assembly: Lib9c.dll

```
[Serializable]
public struct WorldInformation.World : IState
```

## Implements

[IState](#)

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### World(Dictionary)

```
public World(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### World(World, long)

```
public World(WorldInformation.World world, long unlockedBlockIndex = -1)
```

#### Parameters

**world** [WorldInformation.World](#)

**unlockedBlockIndex** [long](#)

## World(World, long, int)

```
public World(WorldInformation.World world, long stageClearedBlockIndex,  
int stageClearedId)
```

### Parameters

world [WorldInformation.World](#)

stageClearedBlockIndex [long](#)

stageClearedId [int](#)

## World(Row, long, long, int)

```
public World(WorldSheet.Row worldRow, long unlockedBlockIndex = -1, long  
stageClearedBlockIndex = -1, int stageClearedId = -1)
```

### Parameters

worldRow [WorldSheet.Row](#)

unlockedBlockIndex [long](#)

stageClearedBlockIndex [long](#)

stageClearedId [int](#)

## Fields

### Id

```
public readonly int Id
```

### Field Value

[int](#)

## Name

```
public readonly string Name
```

### Field Value

[string](#)

## StageBegin

```
public readonly int StageBegin
```

### Field Value

[int](#)

## StageClearedBlockIndex

```
public readonly long StageClearedBlockIndex
```

### Field Value

[long](#)

## StageClearedId

```
public readonly int StageClearedId
```

### Field Value

[int](#)

## StageEnd

```
public readonly int StageEnd
```

Field Value

[int](#)

## UnlockedBlockIndex

```
public readonly long UnlockedBlockIndex
```

Field Value

[long](#)

## Properties

### IsStageCleared

```
public bool IsStageCleared { get; }
```

Property Value

[bool](#)

### IsUnlocked

```
public bool IsUnlocked { get; }
```

Property Value

[bool](#)

## Methods

## ContainsStageId(int)

```
public bool ContainsStageId(int stageId)
```

Parameters

stageId [int](#)

Returns

[bool](#)

## GetNextStageId()

```
public int GetNextStageId()
```

Returns

[int](#)

## GetNextStageIdForPlay()

```
public int GetNextStageIdForPlay()
```

Returns

[int](#)

## IsPlayable(int)

```
public bool IsPlayable(int stageId)
```

Parameters

stageId [int](#)

Returns

bool ↗

## Serialize()

`public IValue Serialize()`

Returns

IValue

# Namespace Nekoyume.Model.Adventure Boss

## Classes

[BountyBoard](#)

[ExploreBoard](#)

[Explorer](#)

[ExplorerList](#)

[Investor](#)

[SeasonInfo](#)

# Class BountyBoard

Namespace: [Nekoyume.Model.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class BountyBoard
```

## Inheritance

[object](#) ← BountyBoard

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### BountyBoard(List)

```
public BountyBoard(List bencoded)
```

#### Parameters

bencoded List

### BountyBoard(long)

```
public BountyBoard(long season)
```

#### Parameters

season [long](#)

## Fields

## FixedRewardFavId

```
public int? FixedRewardFavId
```

Field Value

[int](#)?

## FixedRewardItemId

```
public int? FixedRewardItemId
```

Field Value

[int](#)?

## Investors

```
public List<Investor> Investors
```

Field Value

[List](#)<[Investor](#)>

## RandomRewardFavId

```
public int? RandomRewardFavId
```

Field Value

[int](#)?

## RandomRewardItemId

```
public int? RandomRewardItemId
```

Field Value

[int](#)?

## Season

```
public long Season
```

Field Value

[long](#)

## Properties

### Bencoded

```
public IValue Bencoded { get; }
```

Property Value

IValue

## Methods

### AddOrUpdate(Address, string, FungibleAssetValue)

```
public void AddOrUpdate(Address avatarAddress, string name,  
FungibleAssetValue price)
```

Parameters

avatarAddress Address

`name` [string](#)

`price` FungibleAssetValue

## SetReward(Row, IRandom)

```
public void SetReward(AdventureBossWantedRewardSheet.Row rewardInfo, IRandom random)
```

### Parameters

`rewardInfo` [AdventureBossWantedRewardSheet.Row](#)

`random` IRandom

## totalBounty()

```
public FungibleAssetValue totalBounty()
```

### Returns

FungibleAssetValue

# Class ExploreBoard

Namespace: [Nekoyume.Model.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class ExploreBoard
```

## Inheritance

[object](#) ← ExploreBoard

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ExploreBoard(List)

```
public ExploreBoard(List bencoded)
```

#### Parameters

bencoded List

### ExploreBoard(long)

```
public ExploreBoard(long season)
```

#### Parameters

season [long](#)

## Fields

## ExplorerCount

`public int ExplorerCount`

### Field Value

[int](#) ↗

## FixedRewardFavId

`public int? FixedRewardFavId`

### Field Value

[int](#) ↗?

## FixedRewardItemId

`public int? FixedRewardItemId`

### Field Value

[int](#) ↗?

## RaffleReward

`public FungibleAssetValue? RaffleReward`

### Field Value

FungibleAssetValue?

## RaffleWinner

```
public Address? RaffleWinner
```

Field Value

Address?

## RaffleWinnerName

```
public string RaffleWinnerName
```

Field Value

[string](#)

## Season

```
public long Season
```

Field Value

[long](#)

## TotalPoint

```
public long TotalPoint
```

Field Value

[long](#)

## UsedApPotion

```
public long UsedApPotion
```

Field Value

[long](#) ↗

## UsedGoldenDust

```
public long UsedGoldenDust
```

Field Value

[long](#) ↗

## UsedNcg

```
public BigInteger UsedNcg
```

Field Value

[BigInteger](#) ↗

## Methods

### Bencoded()

```
public IValue Bencoded()
```

Returns

IValue

### SetReward(Row, IRandom)

```
public void SetReward(AdventureBossContributionRewardSheet.Row rewardInfo,  
IRandom random)
```

## Parameters

rewardInfo [AdventureBossContributionRewardSheet.Row](#)

random IRandom

# Class Explorer

Namespace: [Nekoyume.Model.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class Explorer : IBencodable
```

## Inheritance

[object](#) ← Explorer

## Implements

IBencodable

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### Explorer(IValue)

```
public Explorer(IValue bencoded)
```

#### Parameters

bencoded IValue

### Explorer(Address, string)

```
public Explorer(Address avatarAddress, string name)
```

#### Parameters

avatarAddress Address

name [string](#)

## Fields

### AvatarAddress

`public Address AvatarAddress`

#### Field Value

Address

### Claimed

`public bool Claimed`

#### Field Value

[bool](#)

### Floor

`public int Floor`

#### Field Value

[int](#)

### MaxFloor

`public int MaxFloor`

#### Field Value

[int](#)

## Name

`public string Name`

### Field Value

[string](#)

## Score

`public int Score`

### Field Value

[int](#)

## UsedApPotion

`public int UsedApPotion`

### Field Value

[int](#)

## UsedGoldenDust

`public int UsedGoldenDust`

### Field Value

[int](#)

# UsedNcg

```
public BigInteger UsedNcg
```

## Field Value

[BigInteger](#)

## Properties

### Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

## Property Value

IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

# Class ExplorerList

Namespace: [Nekoyume.Model.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class ExplorerList
```

## Inheritance

[object](#) ← ExplorerList

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ExplorerList(List)

```
public ExplorerList(List bencoded)
```

#### Parameters

bencoded List

### ExplorerList(long)

```
public ExplorerList(long season)
```

#### Parameters

season [long](#)

## Fields

# Explorers

```
public HashSet<(Address, string)> Explorers
```

## Field Value

[HashSet](#)<(Address, [string](#))>

# Season

```
public long Season
```

## Field Value

[long](#)

# Properties

## Bencoded

```
public IValue Bencoded { get; }
```

## Property Value

IValue

# Methods

## AddExplorer(Address, string)

```
public void AddExplorer(Address avatarAddress, string name)
```

## Parameters

avatarAddress Address

name [string](#)

# Class Investor

Namespace: [Nekoyume.Model.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class Investor : IBencodable
```

## Inheritance

[object](#) ← Investor

## Implements

IBencodable

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### Investor(IValue)

```
public Investor(IValue value)
```

#### Parameters

value IValue

### Investor(Address, string, FungibleAssetValue)

```
public Investor(Address avatarAddress, string name, FungibleAssetValue price)
```

#### Parameters

avatarAddress Address

`name` [string](#)

`price` [FungibleAssetValue](#)

## Fields

### AvatarAddress

`public` [Address](#) AvatarAddress

#### Field Value

Address

### Claimed

`public bool` Claimed

#### Field Value

[bool](#)

### Count

`public int` Count

#### Field Value

[int](#)

### MaxInvestmentCount

`public const int` MaxInvestmentCount = 3

## Field Value

[int](#)

## Name

`public string Name`

## Field Value

[string](#)

## Price

`public FungibleAssetValue Price`

## Field Value

FungibleAssetValue

## Properties

### Bencoded

An `Bencodex.Types.IValue` representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

`public IValue Bencoded { get; }`

## Property Value

IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

# Class SeasonInfo

Namespace: [Nekoyume.Model.AdventureBoss](#)

Assembly: Lib9c.dll

```
public class SeasonInfo
```

## Inheritance

[object](#) ← SeasonInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### SeasonInfo(List)

```
public SeasonInfo(List serialized)
```

#### Parameters

serialized List

### SeasonInfo(long, long, long, long, long?, long?)

```
public SeasonInfo(long season, long startBlockIndex, long activeInterval, long  
inactiveInterval, long? endBlockIndex = null, long? nextStartBlockIndex = null)
```

#### Parameters

season [long](#)

startBlockIndex [long](#)

activeInterval [long](#)

inactiveInterval [long](#)

endBlockIndex [long](#)?

nextStartBlockIndex [long](#)?

## Fields

### BossId

`public int BossId`

#### Field Value

[int](#)

### EndBlockIndex

`public readonly long EndBlockIndex`

#### Field Value

[long](#)

### NextStartBlockIndex

`public readonly long NextStartBlockIndex`

#### Field Value

[long](#)

## Season

```
public readonly long Season
```

Field Value

[long](#) ↗

## StartBlockIndex

```
public readonly long StartBlockIndex
```

Field Value

[long](#) ↗

## Properties

### Bencoded

```
public IValue Bencoded { get; }
```

Property Value

IValue

# Namespace Nekoyume.Model.Arena

## Classes

[AddressNotFoundInArenaParticipantsException](#)

[ArenaAvatarStateNotFoundException](#)

[ArenaInformation](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1029>

[ArenaInformationAlreadyContainsException](#)

[ArenaInformationNotFoundException](#)

[ArenaParticipant](#)

This class combines the information from [ArenaScore](#) and [ArenaInformation](#), and brings together information about the arena participant, including additional information.

[ArenaParticipants](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1027>

[ArenaParticipantsNotFoundException](#)

[ArenaScore](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1027>

[ArenaScoreAlreadyContainsException](#)

[ArenaScoreNotFoundException](#)

[CoolDownBlockException](#)

[ExceedPlayCountException](#)

[ExceedTicketPurchaseLimitDuringIntervalException](#)

[ExceedTicketPurchaseLimitException](#)

[InvalidSeasonException](#)

[NotEnoughTicketException](#)

[RoundNotFoundException](#)

[ThisArenasClosedException](#)

[TicketPurchaseLimitExceedException](#)

[ValidateScoreDifferenceException](#)

# Class AddressNotFoundInArenaParticipantsException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AddressNotFoundInArenaParticipantsException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← AddressNotFoundInArenaParticipantsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### AddressNotFoundInArenaParticipantsException(SerializationInfo, StreamingContext)

```
protected AddressNotFoundInArenaParticipantsException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## AddressNotFoundInArenaParticipantsException(string)

public AddressNotFoundInArenaParticipantsException([string](#) message)

### Parameters

message [string](#)

# Class ArenaAvatarStateNotFoundException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaAvatarStateNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ArenaAvatarStateNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaAvatarStateNotFoundException(SerializationInfo, StreamingContext)

```
protected ArenaAvatarStateNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaAvatarStateNotFoundException(string)

```
public ArenaAvatarStateNotFoundException(string message)
```

### Parameters

message [string](#)

# Class ArenaInformation

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1029>

```
public class ArenaInformation : IState
```

## Inheritance

[object](#) ← ArenaInformation

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaInformation(List)

```
public ArenaInformation(List serialized)
```

#### Parameters

serialized List

### ArenaInformation(Address, int, int)

```
public ArenaInformation(Address avatarAddress, int championshipId, int round)
```

#### Parameters

avatarAddress Address

championshipId [int↗](#)

round [int↗](#)

## Fields

### Address

`public Address Address`

Field Value

Address

### MaxTicketCount

`public const int MaxTicketCount = 8`

Field Value

[int↗](#)

## Properties

### Lose

`public int Lose { get; }`

Property Value

[int↗](#)

## PurchasedTicketCount

```
public int PurchasedTicketCount { get; }
```

Property Value

[int ↗](#)

## Ticket

```
public int Ticket { get; }
```

Property Value

[int ↗](#)

## TicketResetCount

```
public int TicketResetCount { get; }
```

Property Value

[int ↗](#)

## Win

```
public int Win { get; }
```

Property Value

[int ↗](#)

## Methods

## BuyTicket(long)

```
public void BuyTicket(long maxCount)
```

### Parameters

maxCount [long](#)

## DeriveAddress(Address, int, int)

```
public static Address DeriveAddress(Address avatarAddress, int championshipId,  
int round)
```

### Parameters

avatarAddress Address

championshipId [int](#)

round [int](#)

### Returns

Address

## ResetTicket(int)

```
public void ResetTicket(int resetCount)
```

### Parameters

resetCount [int](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## UpdateRecord(int, int)

```
public void UpdateRecord(int win, int lose)
```

Parameters

win [int](#)

lose [int](#)

## UseTicket(int)

```
public void UseTicket(int ticketCount)
```

Parameters

ticketCount [int](#)

# Class ArenaInformationAlreadyContainsException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaInformationAlreadyContainsException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ArenaInformationAlreadyContainsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaInformationAlreadyContainsException(SerializationInfo, StreamingContext)

```
protected ArenaInformationAlreadyContainsException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaInformationAlreadyContainsException(string)

```
public ArenaInformationAlreadyContainsException(string message)
```

### Parameters

message [string](#)

# Class ArenaInformationNotFoundException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaInformationNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ArenaInformationNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaInformationNotFoundException(SerializationInfo, StreamingContext)

```
protected ArenaInformationNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaInformationNotFoundException(string)

```
public ArenaInformationNotFoundException(string message)
```

### Parameters

message [string](#)

# Class ArenaParticipant

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

This class combines the information from [ArenaScore](#) and [ArenaInformation](#), and brings together information about the arena participant, including additional information.

```
public class ArenaParticipant : IBencodable, IState
```

## Inheritance

[object](#) ← ArenaParticipant

## Implements

IBencodable, [IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaParticipant(IValue)

```
public ArenaParticipant(IValue bencoded)
```

#### Parameters

bencoded IValue

### ArenaParticipant(Address)

```
public ArenaParticipant(Address avatarAddr)
```

#### Parameters

**avatarAddr** Address

## Fields

### AvatarAddr

```
public readonly Address AvatarAddr
```

#### Field Value

Address

### Cp

```
public int Cp
```

#### Field Value

[int](#)

### DefaultScore

```
public const int DefaultScore = 1000
```

#### Field Value

[int](#)

### LastBattleBlockIndex

```
public long LastBattleBlockIndex
```

#### Field Value

[long](#) ↴

## Level

`public int Level`

### Field Value

[int](#) ↴

## Lose

`public int Lose`

### Field Value

[int](#) ↴

## MaxTicketCount

`public const int MaxTicketCount = 8`

### Field Value

[int](#) ↴

## Name

If you need to know [NameWithHash](#), check [PostConstructor\(\)](#) method of the [AvatarState](#) class and you can find the relevant information there. It provides a formatted string that includes the avatar's [name](#) and a shortened version of their address.

`public string Name`

Field Value

[string](#)

Examples

```
 ${name} <size=80%><color=#A68F7E>#${address.ToHex().Substring(0, 4)}  
</color></size>";
```

PortraitId

```
public int PortraitId
```

Field Value

[int](#)

PurchasedTicketCount

```
public int PurchasedTicketCount
```

Field Value

[int](#)

Score

```
public int Score
```

Field Value

[int](#)

## Ticket

```
public int Ticket
```

Field Value

[int](#)

## TicketResetCount

```
public int TicketResetCount
```

Field Value

[int](#)

## Win

```
public int Win
```

Field Value

[int](#)

## Properties

### Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

## Property Value

### IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

## Methods

### AddScore(int)

```
public void AddScore(int score)
```

#### Parameters

score [int](#)

### BuyTicket(long)

```
public void BuyTicket(long maxCount)
```

#### Parameters

maxCount [long](#)

### ResetTicket(int)

```
public void ResetTicket(int resetCount)
```

Parameters

resetCount [int](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## UpdateRecord(int, int)

```
public void UpdateRecord(int win, int lose)
```

Parameters

win [int](#)

lose [int](#)

## UseTicket(int)

```
public void UseTicket(int ticketCount)
```

Parameters

ticketCount [int](#)

# Class ArenaParticipants

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1027>

```
public class ArenaParticipants : IState
```

## Inheritance

[object](#) ← ArenaParticipants

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaParticipants(List)

```
public ArenaParticipants(List serialized)
```

#### Parameters

serialized List

### ArenaParticipants(int, int)

```
public ArenaParticipants(int championshipId, int round)
```

#### Parameters

championshipId [int](#)

round [int](#)

## Fields

### Address

`public Address Address`

#### Field Value

Address

### AvatarAddresses

`public readonly List<Address> AvatarAddresses`

#### Field Value

[List](#)<Address>

## Methods

### Add(Address)

`public void Add(Address address)`

#### Parameters

address Address

### DeriveAddress(int, int)

```
public static Address DeriveAddress(int championshipId, int round)
```

Parameters

championshipId [int](#)

round [int](#)

Returns

Address

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class ArenaParticipantsNotFoundException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaParticipantsNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ArenaParticipantsNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaParticipantsNotFoundException(SerializationInfo, StreamingContext)

```
protected ArenaParticipantsNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaParticipantsNotFoundException(string)

```
public ArenaParticipantsNotFoundException(string message)
```

### Parameters

message [string](#)

# Class ArenaScore

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1027>

```
public class ArenaScore : IState
```

## Inheritance

[object](#) ← ArenaScore

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaScore(List)

```
public ArenaScore(List serialized)
```

#### Parameters

serialized List

### ArenaScore(Address, int, int, int)

```
public ArenaScore(Address avatarAddress, int championshipId, int round, int score  
= 1000)
```

#### Parameters

avatarAddress Address

championshipId [int](#)

round [int](#)

score [int](#)

## Fields

### Address

[public](#) Address Address

### Field Value

Address

### ArenaScoreDefault

[public const int](#) ArenaScoreDefault = 1000

### Field Value

[int](#)

## Properties

### Score

[public int](#) Score { [get](#); }

### Property Value

[int](#)

# Methods

## AddScore(int)

```
public void AddScore(int score)
```

### Parameters

score [int](#)

## DeriveAddress(Address, int, int)

```
public static Address DeriveAddress(Address avatarAddress, int championshipId,  
int round)
```

### Parameters

avatarAddress Address

championshipId [int](#)

round [int](#)

### Returns

Address

## Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Class ArenaScoreAlreadyContainsException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaScoreAlreadyContainsException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ArenaScoreAlreadyContainsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaScoreAlreadyContainsException(SerializationInfo, StreamingContext)

```
protected ArenaScoreAlreadyContainsException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaScoreAlreadyContainsException(string)

```
public ArenaScoreAlreadyContainsException(string message)
```

### Parameters

message [string](#)

# Class ArenaScoreNotFoundException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaScoreNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ArenaScoreNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ArenaScoreNotFoundException(SerializationInfo, StreamingContext)

```
protected ArenaScoreNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ArenaScoreNotFoundException(string)

```
public ArenaScoreNotFoundException(string message)
```

### Parameters

message [string](#)

# Class CoolDownBlockException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CoolDownBlockException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← CoolDownBlockException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### CoolDownBlockException(SerializationInfo, StreamingContext)

```
protected CoolDownBlockException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## CoolDownBlockException(string)

```
public CoolDownBlockException(string message)
```

### Parameters

message [string](#)

# Class ExceedPlayCountException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ExceedPlayCountException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ExceedPlayCountException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ExceedPlayCountException(SerializationInfo, StreamingContext)

```
protected ExceedPlayCountException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ExceedPlayCountException(string)

```
public ExceedPlayCountException(string message)
```

### Parameters

message [string](#)

# Class ExceedTicketPurchaseLimitDuringIntervalException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ExceedTicketPurchaseLimitDuringIntervalException :
Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ExceedTicketPurchaseLimitDuringIntervalException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ExceedTicketPurchaseLimitDuringIntervalException(SerializationInfo, StreamingContext)

```
protected ExceedTicketPurchaseLimitDuringIntervalException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ExceedTicketPurchaseLimitDuringIntervalException(string)

public ExceedTicketPurchaseLimitDuringIntervalException([string](#) message)

### Parameters

message [string](#)

# Class ExceedTicketPurchaseLimitException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ExceedTicketPurchaseLimitException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ExceedTicketPurchaseLimitException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ExceedTicketPurchaseLimitException(SerializationInfo, StreamingContext)

```
protected ExceedTicketPurchaseLimitException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ExceedTicketPurchaseLimitException(string)

```
public ExceedTicketPurchaseLimitException(string message)
```

### Parameters

message [string](#)

# Class InvalidSeasonException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidSeasonException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← InvalidSeasonException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidSeasonException(SerializationInfo, StreamingContext)

```
protected InvalidSeasonException(SerializationInfo info, StreamingContext context)
```

#### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidSeasonException(string)

```
public InvalidSeasonException(string message)
```

### Parameters

message [string](#)

# Class NotEnoughTicketException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NotEnoughTicketException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← NotEnoughTicketException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NotEnoughTicketException(SerializationInfo, StreamingContext)

```
protected NotEnoughTicketException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## NotEnoughTicketException(string)

```
public NotEnoughTicketException(string message)
```

### Parameters

message [string](#)

# Class RoundNotFoundException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RoundNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RoundNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RoundNotFoundException(SerializationInfo, StreamingContext)

```
protected RoundNotFoundException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RoundNotFoundException(string)

```
public RoundNotFoundException(string message)
```

### Parameters

message [string](#)

# Class ThisArenalsClosedException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ThisArenaIsClosedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ThisArenalsClosedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ThisArenalsClosedException(SerializationInfo, StreamingContext)

```
protected ThisArenaIsClosedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ThisArenalsClosedException(string)

```
public ThisArenaIsClosedException(string message)
```

### Parameters

message [string](#)

# Class TicketPurchaseLimitExceedException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
public class TicketPurchaseLimitExceedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← TicketPurchaseLimitExceedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### TicketPurchaseLimitExceedException(SerializationInfo, StreamingContext)

```
protected TicketPurchaseLimitExceedException(SerializationInfo info,  
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## TicketPurchaseLimitExceedException(string)

```
public TicketPurchaseLimitExceedException(string message)
```

### Parameters

message [string](#)

# Class ValidateScoreDifferenceException

Namespace: [Nekoyume.Model.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ValidateScoreDifferenceException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ValidateScoreDifferenceException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ValidateScoreDifferenceException(SerializationInfo, StreamingContext)

```
protected ValidateScoreDifferenceException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ValidateScoreDifferenceException(string)

```
public ValidateScoreDifferenceException(string message)
```

### Parameters

message [string](#)

# Namespace Nekoyume.Model.BattleStatus

## Classes

[AreaAttack](#)

[BattleLog](#)

[BlowAttack](#)

[Buff](#)

[BuffRemovalAttack](#)

[Dead](#)

[DoubleAttack](#)

[DoubleAttackWithCombo](#)

[DropBox](#)

[EventBase](#)

[GetExp](#)

[GetReward](#)

[HealSkill](#)

[NormalAttack](#)

[RemoveBuffs](#)

[ShatterStrike](#)

[Skill](#)

[Skill.SkillInfo](#)

[SpawnEnemyPlayer](#)

[SpawnPlayer](#)

[SpawnWave](#)

[Tick](#)

[TickDamage](#)

[WaveTurnEnd](#)

## Enums

[BattleLog.Result](#)

# Class AreaAttack

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AreaAttack : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← AreaAttack

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

AreaAttack(int, CharacterBase, IEnumerable<SkillInfo>,  
IEnumerable<SkillInfo>)

```
public AreaAttack(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class BattleLog

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BattleLog : IEnumerable<EventBase>, IEnumerable
```

## Inheritance

[object](#) ← BattleLog

## Implements

[IEnumerable](#)<[EventBase](#)>, [IEnumerable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Fields

### clearedWaveNumber

```
public int clearedWaveNumber
```

#### Field Value

[int](#)

### diffScore

```
public int diffScore
```

#### Field Value

[int](#)

## events

```
public List<EventBase> events
```

### Field Value

[List](#)<[EventBase](#)>

## id

```
public Guid id
```

### Field Value

[Guid](#)

## newlyCleared

```
public bool newlyCleared
```

### Field Value

[bool](#)

## result

```
public BattleLog.Result result
```

### Field Value

[BattleLog.Result](#)

## score

```
public int score
```

### Field Value

[int ↗](#)

## stageId

```
public int stageId
```

### Field Value

[int ↗](#)

## waveCount

```
public int waveCount
```

### Field Value

[int ↗](#)

## worldId

```
public int worldId
```

### Field Value

[int ↗](#)

## Properties

## Count

```
public int Count { get; }
```

Property Value

[int](#)

## IsClear

```
public bool IsClear { get; }
```

Property Value

[bool](#)

## Methods

### Add(EventBase)

```
public void Add(EventBase e)
```

Parameters

e [EventBase](#)

### GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumarator<EventBase> GetEnumerator()
```

Returns

[IEnumarator](#) <[EventBase](#)>

An enumerator that can be used to iterate through the collection.

# Enum BattleLog.Result

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
public enum BattleLog.Result
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Lose = 1

TimeOver = 2

Win = 0

# Class BlowAttack

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BlowAttack : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← BlowAttack

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

BlowAttack(int, CharacterBase, IEnumerable<SkillInfo>,  
IEnumerable<SkillInfo>)

```
public BlowAttack(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class Buff

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Buff : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← [Buff](#)

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Buff(int, CharacterBase, IEnumerable<SkillInfo>)

```
public Buff(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#) ↗

# Class BuffRemovalAttack

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuffRemovalAttack : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← BuffRemovalAttack

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

BuffRemovalAttack(int, CharacterBase,  
IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public BuffRemovalAttack(int skillId, CharacterBase character,
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class Dead

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Dead : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← Dead

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Dead(CharacterBase)

```
public Dead(CharacterBase character)
```

## Parameters

character [CharacterBase](#)

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

## Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class DoubleAttack

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DoubleAttack : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← DoubleAttack

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

DoubleAttack(int, CharacterBase,  
IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public DoubleAttack(int skillId, CharacterBase character,
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class DoubleAttackWithCombo

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DoubleAttackWithCombo : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← DoubleAttackWithCombo

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

DoubleAttackWithCombo(int, CharacterBase,  
IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public DoubleAttackWithCombo(int skillId, CharacterBase character,
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class DropBox

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DropBox : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← DropBox

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

DropBox(CharacterBase, List<ItemBase>)

```
public DropBox(CharacterBase character, List<ItemBase> items)
```

## Parameters

character [CharacterBase](#)

items [List](#)<[ItemBase](#)>

## Fields

### Items

```
public readonly List<ItemBase> Items
```

## Field Value

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

#### Parameters

stage [IStage](#)

#### Returns

[IEnumerator](#)

# Class EventBase

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class EventBase
```

## Inheritance

[object](#) ← EventBase

## Derived

[Breakthrough](#), [Dead](#), [DropBox](#), [GetExp](#), [GetReward](#), [RemoveBuffs](#), [Skill](#), [SpawnEnemyPlayer](#),  
[SpawnPlayer](#), [SpawnWave](#), [WaveTurnEnd](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### EventBase(CharacterBase)

```
protected EventBase(CharacterBase character)
```

## Parameters

character [CharacterBase](#)

## Fields

### Character

```
public readonly CharacterBase Character
```

Field Value

[CharacterBase](#)

## Methods

### CoExecute(IStage)

```
public abstract IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class GetExp

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GetExp : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← GetExp

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GetExp(CharacterBase, long)

```
public GetExp(CharacterBase character, long exp)
```

## Parameters

character [CharacterBase](#)

exp [long](#)

## Properties

### Exp

```
public long Exp { get; }
```

## Property Value

[long](#) ↴

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

#### Parameters

stage [IStage](#)

#### Returns

[IEnumerator](#) ↴

# Class GetReward

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GetReward : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← GetReward

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GetReward(CharacterBase, List<ItemBase>)

```
public GetReward(CharacterBase character, List<ItemBase> rewards)
```

## Parameters

character [CharacterBase](#)

rewards [List](#)<[ItemBase](#)>

## Fields

### Rewards

```
public readonly List<ItemBase> Rewards
```

## Field Value

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

#### Parameters

stage [IStage](#)

#### Returns

[IEnumerator](#)

# Class HealSkill

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class HealSkill : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← HealSkill

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

HealSkill(int, CharacterBase, IEnumerable<SkillInfo>,  
IEnumerable<SkillInfo>)

```
public HealSkill(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class NormalAttack

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NormalAttack : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← NormalAttack

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

NormalAttack(int, CharacterBase,  
IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public NormalAttack(int skillId, CharacterBase character,
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class RemoveBuffs

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RemoveBuffs : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← RemoveBuffs

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RemoveBuffs(CharacterBase)

```
public RemoveBuffs(CharacterBase character)
```

## Parameters

character [CharacterBase](#)

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

## Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class ShatterStrike

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ShatterStrike : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← ShatterStrike

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

ShatterStrike(int, CharacterBase,  
IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public ShatterStrike(int skillId, CharacterBase character,
IEnumerable<Skill.SkillInfo> skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class Skill

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Skill : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← Skill

## Derived

[StageBuff](#), [AreaAttack](#), [BlowAttack](#), [Buff](#), [BuffRemovalAttack](#), [DoubleAttack](#),  
[DoubleAttackWithCombo](#), [HealSkill](#), [NormalAttack](#), [ShatterStrike](#), [Tick](#), [TickDamage](#)

## Inherited Members

[EventBase.Character](#) , [EventBase.CoExecute\(IStage\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

Skill(int, CharacterBase, IEnumerable<SkillInfo>,  
IEnumerable<SkillInfo>)

```
protected Skill(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

# Fields

## BuffInfos

```
public readonly IEnumerable<Skill.SkillInfo>? BuffInfos
```

### Field Value

[IEnumerable](#) <[Skill.SkillInfo](#)>

## SkillId

```
public readonly int SkillId
```

### Field Value

[int](#)

## SkillInfos

```
public readonly IEnumerable<Skill.SkillInfo> SkillInfos
```

### Field Value

[IEnumerable](#) <[Skill.SkillInfo](#)>

# Class Skill.SkillInfo

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Skill.SkillInfo
```

## Inheritance

[object](#) ← Skill.SkillInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

SkillInfo(Guid, bool, long, long, bool, SkillCategory, int, ElementType, SkillTargetType, Buff?, CharacterBase?, bool, IEnumerable<Buff>?, IceShield?)

```
public SkillInfo(Guid characterId, bool isDead, long thorn, long effect, bool
critical, SkillCategory skillCategory, int waveTurn, ElementType elementType =
ElementType.Normal, SkillTargetType targetType = SkillTargetType.Enemy, Buff? buff
= null, CharacterBase? target = null, bool affected = true, IEnumerable<Buff>?
dispellList = null, IceShield? iceShield = null)
```

## Parameters

characterId [Guid](#)

isDead [bool](#)

thorn [long](#)

effect [long](#)

critical [bool](#)

skillCategory [SkillCategory](#)

waveTurn [int](#)

elementType [ElementalType](#)

targetType [SkillTargetType](#)

buff [Buff](#)

target [CharacterBase](#)

affected [bool](#)

dispelList [IEnumerable](#)<[Buff](#)>

iceShield [IceShield](#)

## Fields

### Affected

public readonly bool Affected

### Field Value

[bool](#)

## Buff

public readonly Buff? Buff

### Field Value

[Buff](#)

## CharacterId

```
public readonly Guid CharacterId
```

### Field Value

[Guid](#) ↴

## Critical

```
public readonly bool Critical
```

### Field Value

[bool](#) ↴

## DispellList

```
public readonly IEnumerable<Buff>? DispellList
```

### Field Value

[IEnumerable](#) ↴ <[Buff](#)>

## Effect

```
public readonly long Effect
```

### Field Value

[long](#) ↴

## ElementalType

```
public readonly ElementType ElementType
```

Field Value

[ElementType](#)

## **IceShield**

```
public readonly IceShield? IceShield
```

Field Value

[IceShield](#)

## **IsDead**

```
public readonly bool IsDead
```

Field Value

[bool](#) ↗

## **SkillCategory**

```
public readonly SkillCategory SkillCategory
```

Field Value

[SkillCategory](#)

## **SkillTargetType**

```
public readonly SkillTargetType SkillTargetType
```

Field Value

[SkillTargetType](#)

## Target

```
public readonly CharacterBase? Target
```

Field Value

[CharacterBase](#)

## Thorn

```
public readonly long Thorn
```

Field Value

[long](#)

## WaveTurn

```
public readonly int WaveTurn
```

Field Value

[int](#)

# Class SpawnEnemyPlayer

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SpawnEnemyPlayer : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← SpawnEnemyPlayer

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SpawnEnemyPlayer(CharacterBase)

```
public SpawnEnemyPlayer(CharacterBase character)
```

## Parameters

character [CharacterBase](#)

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

## Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class SpawnPlayer

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SpawnPlayer : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← SpawnPlayer

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SpawnPlayer(CharacterBase)

```
public SpawnPlayer(CharacterBase character)
```

## Parameters

character [CharacterBase](#)

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

## Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class SpawnWave

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SpawnWave : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← SpawnWave

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

SpawnWave(CharacterBase, int, int, List<Enemy>, bool)

```
public SpawnWave(CharacterBase character, int waveNumber, int waveTurn, List<Enemy>
enemies, bool hasBoss)
```

## Parameters

character [CharacterBase](#)

waveNumber [int](#)

waveTurn [int](#)

enemies [List](#)<Enemy>

hasBoss [bool](#)

## Fields

## Enemies

```
public readonly List<Enemy> Enemies
```

Field Value

[List](#) <[Enemy](#)>

## HasBoss

```
public readonly bool HasBoss
```

Field Value

[bool](#)

## WaveNumber

```
public readonly int WaveNumber
```

Field Value

[int](#)

## WaveTurn

```
public readonly int WaveTurn
```

Field Value

[int](#)

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class Tick

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Tick : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← Tick

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Tick(CharacterBase)

```
public Tick(CharacterBase character)
```

#### Parameters

character [CharacterBase](#)

### Tick(int, CharacterBase, IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public Tick(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

#### Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

#### Parameters

stage [IStage](#)

#### Returns

[IEnumerator](#)

# Class TickDamage

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TickDamage : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← TickDamage

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

TickDamage(int, CharacterBase,  
IEnumerable<SkillInfo>, IEnumerable<SkillInfo>)

```
public TickDamage(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Class WaveTurnEnd

Namespace: [Nekoyume.Model.BattleStatus](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WaveTurnEnd : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← WaveTurnEnd

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### WaveTurnEnd(CharacterBase, int, int)

```
public WaveTurnEnd(CharacterBase character, int turnNumber, int waveTurn)
```

## Parameters

character [CharacterBase](#)

turnNumber [int](#)

waveTurn [int](#)

## Fields

### TurnNumber

```
public readonly int TurnNumber
```

Field Value

[int](#) ↗

## WaveTurn

```
public readonly int WaveTurn
```

Field Value

[int](#) ↗

## Methods

### CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#) ↗

# Namespace Nekoyume.Model.Battle Status.AdventureBoss

## Classes

[Breakthrough](#)

[StageBuff](#)

# Class Breakthrough

Namespace: [Nekoyume.Model.BattleStatus.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Breakthrough : EventBase
```

## Inheritance

[object](#) ← [EventBase](#) ← Breakthrough

## Inherited Members

[EventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

Breakthrough(CharacterBase, int, List<MonsterData>)

```
public Breakthrough(CharacterBase character, int floorId,
List<AdventureBossFloorWaveSheet.MonsterData> monsters)
```

## Parameters

character [CharacterBase](#)

floorId [int](#)

monsters [List](#)<[AdventureBossFloorWaveSheet.MonsterData](#)>

## Fields

### FloorId

```
public readonly int FloorId
```

Field Value

[int](#)

## Monsters

`public readonly List<AdventureBossFloorWaveSheet.MonsterData> Monsters`

Field Value

[List](#)<[AdventureBossFloorWaveSheet](#).[MonsterData](#)>

## Methods

### CoExecute(IStage)

`public override IEnumerator CoExecute(IStage stage)`

Parameters

`stage` [IStage](#)

Returns

[IEnumerator](#)

# Class StageBuff

Namespace: [Nekoyume.Model.BattleStatus.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageBuff : Skill
```

## Inheritance

[object](#) ← [EventBase](#) ← [Skill](#) ← StageBuff

## Inherited Members

[Skill.SkillId](#) , [Skill.SkillInfos](#) , [Skill.BuffInfos](#) , [EventBase.Character](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

StageBuff(int, CharacterBase, IEnumerable<SkillInfo>,  
IEnumerable<SkillInfo>)

```
public StageBuff(int skillId, CharacterBase character, IEnumerable<Skill.SkillInfo>
skillInfos, IEnumerable<Skill.SkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [CharacterBase](#)

skillInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

buffInfos [IEnumerable](#)<[Skill.SkillInfo](#)>

## Methods

## CoExecute(IStage)

```
public override IEnumerator CoExecute(IStage stage)
```

Parameters

stage [IStage](#)

Returns

[IEnumerator](#)

# Namespace Nekoyume.Model.Battle Status.Arena

## Classes

[ArenaAreaAttack](#)

[ArenaBlowAttack](#)

[ArenaBuff](#)

[ArenaBuffRemovalAttack](#)

[ArenaDead](#)

[ArenaDoubleAttack](#)

[ArenaDoubleAttackWithCombo](#)

[ArenaEventBase](#)

[ArenaHeal](#)

[ArenaLog](#)

[ArenaNormalAttack](#)

[ArenaRemoveBuffs](#)

[ArenaShatterStrike](#)

[ArenaSkill](#)

[ArenaSkill.ArenaSkillInfo](#)

[ArenaSpawnCharacter](#)

[ArenaTick](#)

[ArenaTickDamage](#)

[ArenaTurnEnd](#)

## Enums

[ArenaLog.ArenaResult](#)

# Class ArenaAreaAttack

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaAreaAttack : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaAreaAttack

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaAreaAttack(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaAreaAttack(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaBlowAttack

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaBlowAttack : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaBlowAttack

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaBlowAttack(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaBlowAttack(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaBuff

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaBuff : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaBuff

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaBuff(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaBuff(int skillId, ArenaCharacter character,
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

## Methods

CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

## Parameters

arena [IArena](#)

## Returns

[IEnumerator](#)

# Class ArenaBuffRemovalAttack

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaBuffRemovalAttack : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaBuffRemovalAttack

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaBuffRemovalAttack(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaBuffRemovalAttack(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaDead

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaDead : ArenaEventBase
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← ArenaDead

## Inherited Members

[ArenaEventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaDead(ArenaCharacter)

```
public ArenaDead(ArenaCharacter character)
```

## Parameters

character [ArenaCharacter](#)

## Methods

### CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

## Parameters

arena [IArena](#)

Returns

[IEnumerator](#)

# Class ArenaDoubleAttack

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaDoubleAttack : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaDoubleAttack

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaDoubleAttack(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaDoubleAttack(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaDoubleAttackWithCombo

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaDoubleAttackWithCombo : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaDoubleAttackWithCombo

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaDoubleAttackWithCombo(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaDoubleAttackWithCombo(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaEventBase

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ArenaEventBase
```

## Inheritance

[object](#) ← ArenaEventBase

## Derived

[ArenaDead](#), [ArenaRemoveBuffs](#), [ArenaSkill](#), [ArenaSpawnCharacter](#), [ArenaTurnEnd](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### ArenaEventBase(ArenaCharacter)

```
protected ArenaEventBase(ArenaCharacter character)
```

## Parameters

character [ArenaCharacter](#)

## Fields

### Character

```
public readonly ArenaCharacter Character
```

## Field Value

[ArenaCharacter](#)

## Methods

### CoExecute(IArena)

```
public abstract IEnumerator CoExecute(IArena arena)
```

#### Parameters

arena [IArena](#)

#### Returns

[IEnumerator](#)

# Class ArenaHeal

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaHeal : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaHeal

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaHeal(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaHeal(int skillId, ArenaCharacter character,
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaLog

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaLog : IEnumerable<ArenaEventBase>, IEnumerable
```

## Inheritance

[object](#) ← ArenaLog

## Implements

[IEnumerable](#)<[ArenaEventBase](#)>, [IEnumerable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Fields

### Events

```
public List<ArenaEventBase> Events
```

### Field Value

[List](#)<[ArenaEventBase](#)>

### Id

```
public Guid Id
```

### Field Value

[Guid](#)

# Result

```
public ArenaLog.ArenaResult Result
```

## Field Value

[ArenaLog.ArenaResult](#)

# Score

```
public int Score
```

## Field Value

[int](#)

# Methods

## Add(ArenaEventBase)

```
public void Add(ArenaEventBase e)
```

## Parameters

e [ArenaEventBase](#)

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<ArenaEventBase> GetEnumerator()
```

## Returns

[IEnumerator](#) <[ArenaEventBase](#)>

An enumerator that can be used to iterate through the collection.

# Enum ArenaLog.ArenaResult

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
public enum ArenaLog.ArenaResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Lose = 1

Win = 0

# Class ArenaNormalAttack

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaNormalAttack : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaNormalAttack

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaNormalAttack(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaNormalAttack(int skillId, ArenaCharacter character,
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaRemoveBuffs

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaRemoveBuffs : ArenaEventBase
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← ArenaRemoveBuffs

## Inherited Members

[ArenaEventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaRemoveBuffs(ArenaCharacter)

```
public ArenaRemoveBuffs(ArenaCharacter character)
```

## Parameters

character [ArenaCharacter](#)

## Methods

### CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

## Parameters

arena [IArena](#)

Returns

[IEnumerator](#)

# Class ArenaShatterStrike

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaShatterStrike : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaShatterStrike

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaShatterStrike(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaShatterStrike(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaSkill

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ArenaSkill : ArenaEventBase
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← ArenaSkill

## Derived

[ArenaAreaAttack](#), [ArenaBlowAttack](#), [ArenaBuff](#), [ArenaBuffRemovalAttack](#),  
[ArenaDoubleAttack](#), [ArenaDoubleAttackWithCombo](#), [ArenaHeal](#), [ArenaNormalAttack](#),  
[ArenaShatterStrike](#), [ArenaTick](#), [ArenaTickDamage](#)

## Inherited Members

[ArenaEventBase.Character](#), [ArenaEventBase.CoExecute\(IArena\)](#), [object.Equals\(object\)](#),  
[object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

ArenaSkill(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
protected ArenaSkill(int skillId, ArenaCharacter character,
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill](#).[ArenaSkillInfo](#)>

## Fields

### BuffInfos

`public readonly IEnumerable<ArenaSkill.ArenaSkillInfo>? BuffInfos`

#### Field Value

[IEnumerable](#)<[ArenaSkill](#).[ArenaSkillInfo](#)>

### SkillId

`public readonly int SkillId`

#### Field Value

[int](#)

### SkillInfos

`public readonly IEnumerable<ArenaSkill.ArenaSkillInfo> SkillInfos`

#### Field Value

[IEnumerable](#)<[ArenaSkill](#).[ArenaSkillInfo](#)>

# Class ArenaSkill.ArenaSkillInfo

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaSkill.ArenaSkillInfo
```

## Inheritance

[object](#) ← ArenaSkill.ArenaSkillInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaSkillInfo(ArenaCharacter, long, bool, SkillCategory, int, ElementalType, SkillTargetType, Buff?, bool, IEnumerable<Buff>?, IceShield?)

```
public ArenaSkillInfo(ArenaCharacter character, long effect, bool critical,
SkillCategory skillCategory, int turn, ElementalType elementType =
ElementalType.Normal, SkillTargetType targetType = SkillTargetType.Enemy, Buff? buff
= null, bool affected = true, IEnumerable<Buff>? dispelList = null, IceShield?
iceShield = null)
```

## Parameters

character [ArenaCharacter](#)

effect [long](#)

critical [bool](#)

skillCategory [SkillCategory](#)

turn [int](#)

elementType [ElementalType](#)

targetType [SkillTargetType](#)

buff [Buff](#)

affected [bool](#)

dispelList [IEnumerable](#)<[Buff](#)>

iceShield [IceShield](#)

## Fields

### Affected

public readonly bool Affected

### Field Value

[bool](#)

### Buff

public readonly Buff? Buff

### Field Value

[Buff](#)

### Critical

public readonly bool Critical

### Field Value

[bool](#)

## DispelList

```
public readonly IEnumerable<Buff>? DispellList
```

Field Value

[IEnumerable](#) <[Buff](#)>

## Effect

```
public readonly long Effect
```

Field Value

[long](#)

## ElementType

```
public readonly ElementType ElementType
```

Field Value

[ElementType](#)

## IceShield

```
public readonly IceShield? IceShield
```

Field Value

[IceShield](#)

## SkillCategory

```
public readonly SkillCategory SkillCategory
```

Field Value

[SkillCategory](#)

## SkillTargetType

```
public readonly SkillTargetType SkillTargetType
```

Field Value

[SkillTargetType](#)

## Target

```
public readonly ArenaCharacter Target
```

Field Value

[ArenaCharacter](#)

## Turn

```
public readonly int Turn
```

Field Value

[int](#)

# Class ArenaSpawnCharacter

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
public class ArenaSpawnCharacter : ArenaEventBase
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← ArenaSpawnCharacter

## Inherited Members

[ArenaEventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaSpawnCharacter(ArenaCharacter)

```
public ArenaSpawnCharacter(ArenaCharacter character)
```

## Parameters

character [ArenaCharacter](#)

## Methods

### CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

## Parameters

arena [IArena](#)

Returns

[IEnumerator](#)

# Class ArenaTick

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaTick : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaTick

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaTick(ArenaCharacter)

```
public ArenaTick(ArenaCharacter character)
```

#### Parameters

character [ArenaCharacter](#)

### ArenaTick(int, ArenaCharacter, IEnumerable<ArenaSkillInfo>, IEnumerable<ArenaSkillInfo>)

```
public ArenaTick(int skillId, ArenaCharacter character,  
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,  
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

## Methods

### CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

## Parameters

arena [IArena](#)

## Returns

[IEnumerator](#)

# Class ArenaTickDamage

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaTickDamage : ArenaSkill
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← [ArenaSkill](#) ← ArenaTickDamage

## Inherited Members

[ArenaSkill.SkillId](#) , [ArenaSkill.SkillInfos](#) , [ArenaSkill.BuffInfos](#) , [ArenaEventBase.Character](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

ArenaTickDamage(int, ArenaCharacter,  
IEnumerable<ArenaSkillInfo>,  
IEnumerable<ArenaSkillInfo>)

```
public ArenaTickDamage(int skillId, ArenaCharacter character,
IEnumerable<ArenaSkill.ArenaSkillInfo> skillInfos,
IEnumerable<ArenaSkill.ArenaSkillInfo> buffInfos)
```

## Parameters

skillId [int](#)

character [ArenaCharacter](#)

skillInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

buffInfos [IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Methods

## CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

### Parameters

arena [IArena](#)

### Returns

[IEnumerator](#)

# Class ArenaTurnEnd

Namespace: [Nekoyume.Model.BattleStatus.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaTurnEnd : ArenaEventBase
```

## Inheritance

[object](#) ← [ArenaEventBase](#) ← ArenaTurnEnd

## Inherited Members

[ArenaEventBase.Character](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaTurnEnd(ArenaCharacter, int)

```
public ArenaTurnEnd(ArenaCharacter character, int turnNumber)
```

## Parameters

character [ArenaCharacter](#)

turnNumber [int](#)

## Fields

### TurnNumber

```
public readonly int TurnNumber
```

## Field Value

## Methods

### CoExecute(IArena)

```
public override IEnumerator CoExecute(IArena arena)
```

#### Parameters

arena [IArena](#)

#### Returns

[IEnumerator](#) ↗

# Namespace Nekoyume.Model.Buff

## Classes

[ActionBuff](#)

[Bleed](#)

[Buff](#)

[BuffFactory](#)

[Dispel](#)

[Focus](#)

[IceShield](#)

[StatBuff](#)

[Stun](#)

[Vampiric](#)

## Structs

[BuffInfo](#)

# Class ActionBuff

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ActionBuff : Buff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← ActionBuff

## Implements

[ICloneable](#)

## Derived

[Bleed](#), [Dispel](#), [Focus](#), [IceShield](#), [Stun](#), [Vampiric](#)

## Inherited Members

[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [Buff.Clone\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ActionBuff(ActionBuff)

```
protected ActionBuff(ActionBuff value)
```

#### Parameters

value [ActionBuff](#)

### ActionBuff(SkillCustomField, Row)

```
public ActionBuff(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## ActionBuff(Row)

```
public ActionBuff(ActionBuffSheet.Row row)
```

## Parameters

row [ActionBuffSheet.Row](#)

## Properties

### CustomField

```
public SkillCustomField? CustomField { get; }
```

#### Property Value

[SkillCustomField?](#)

### RowData

```
public ActionBuffSheet.Row RowData { get; }
```

#### Property Value

[ActionBuffSheet.Row](#)

## Methods

## IsBuff()

```
public override bool IsBuff()
```

Returns

bool ↗

## IsDebuff()

```
public override bool IsDebuff()
```

Returns

bool ↗

# Class Bleed

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Bleed : ActionBuff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← [ActionBuff](#) ← Bleed

## Implements

[ICloneable](#)

## Inherited Members

[ActionBuff.RowData](#) , [ActionBuff.CustomField](#) , [ActionBuff.IsBuff\(\)](#) , [ActionBuff.IsDebuff\(\)](#) ,  
[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## Bleed(Bleed)

```
protected Bleed(Bleed value)
```

## Parameters

value [Bleed](#)

## Bleed(SkillCustomField, Row)

```
public Bleed(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## Bleed(Row, long)

```
public Bleed(ActionBuffSheet.Row row, long power)
```

## Parameters

row [ActionBuffSheet.Row](#)

power [long](#)

## Properties

### Power

```
public long Power { get; }
```

## Property Value

[long](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

## Returns

## [object](#)

A new object that is a copy of this instance.

## GiveEffect(CharacterBase, int, bool)

```
public Skill GiveEffect(CharacterBase affectedCharacter, int simulatorWaveTurn, bool  
copyCharacter = true)
```

### Parameters

affectedCharacter [CharacterBase](#)

simulatorWaveTurn [int](#)

copyCharacter [bool](#)

### Returns

[Skill](#)

## GiveEffectForArena(ArenaCharacter, int)

```
public ArenaSkill GiveEffectForArena(ArenaCharacter affectedCharacter,  
int simulatorWaveTurn)
```

### Parameters

affectedCharacter [ArenaCharacter](#)

simulatorWaveTurn [int](#)

### Returns

[ArenaSkill](#)

# Class Buff

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Buff : ICloneable
```

## Inheritance

[object](#) ← Buff

## Implements

[ICloneable](#)

## Derived

[ActionBuff](#), [StatBuff](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

# Constructors

## Buff(Buff)

```
protected Buff(Buff value)
```

### Parameters

value [Buff](#)

## Buff(BuffInfo)

```
protected Buff(BuffInfo buffInfo)
```

## Parameters

`buffInfo` [BuffInfo](#)

## Properties

### BuffInfo

`public BuffInfo BuffInfo { get; }`

Property Value

[BuffInfo](#)

### OriginalDuration

`public int OriginalDuration { get; }`

Property Value

[int](#)

### RemainedDuration

`public int RemainedDuration { get; set; }`

Property Value

[int](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public abstract object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## GetTarget(CharacterBase)

```
public virtual IEnumerable<CharacterBase> GetTarget(CharacterBase caster)
```

Parameters

caster [CharacterBase](#)

Returns

[IEnumerable](#)<CharacterBase>

## IsBuff()

```
public abstract bool IsBuff()
```

Returns

[bool](#)

## IsDebuff()

```
public abstract bool IsDebuff()
```

Returns

bool ↴

# Class BuffFactory

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
public static class BuffFactory
```

## Inheritance

[object](#) ← BuffFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetActionBuff(Stats, Row)

```
public static ActionBuff GetActionBuff(Stats stat, ActionBuffSheet.Row row)
```

#### Parameters

stat [Stats](#)

row [ActionBuffSheet.Row](#)

#### Returns

[ActionBuff](#)

### GetBuffs(CharacterStats, ISkill, SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet, ActionBuffSheet, bool)

```
public static IList<Buff> GetBuffs(CharacterStats stats, ISkill skill,  
SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet, SkillActionBuffSheet  
skillActionBuffSheet, ActionBuffSheet actionBuffSheet, bool hasExtraValueBuff  
= false)
```

## Parameters

stats [CharacterStats](#)

skill [ISkill](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

hasExtraValueBuff [bool](#)

## Returns

[IList](#)<[Buff](#)>

## GetCustomActionBuff(SkillCustomField, Row)

```
public static ActionBuff GetCustomActionBuff(SkillCustomField customField,  
ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## Returns

[ActionBuff](#)

## GetCustomStatBuff(Row, SkillCustomField)

```
public static StatBuff GetCustomStatBuff(StatBuffSheet.Row row,  
SkillCustomField customField)
```

### Parameters

row [StatBuffSheet.Row](#)

customField [SkillCustomField](#)

### Returns

[StatBuff](#)

## GetStatBuff(Row)

```
public static StatBuff GetStatBuff(StatBuffSheet.Row row)
```

### Parameters

row [StatBuffSheet.Row](#)

### Returns

[StatBuff](#)

# Struct BuffInfo

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public struct BuffInfo
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### BuffInfo(int, int, int, int, SkillTargetType)

```
public BuffInfo(int id, int groupId, int chance, int duration,
SkillTargetType skillTargetType)
```

## Parameters

`id` [int](#)

`groupId` [int](#)

`chance` [int](#)

`duration` [int](#)

`skillTargetType` [SkillTargetType](#)

## Fields

### Chance

```
public int Chance
```

Field Value

[int](#)

## Duration

```
public int Duration
```

Field Value

[int](#)

## GroupId

```
public int GroupId
```

Field Value

[int](#)

## Id

```
public int Id
```

Field Value

[int](#)

## SkillTargetType

```
public SkillTargetType SkillTargetType
```

Field Value

[SkillTargetType](#)

# Class Dispel

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Dispel : ActionBuff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← [ActionBuff](#) ← Dispel

## Implements

[ICloneable](#)

## Inherited Members

[ActionBuff.RowData](#) , [ActionBuff.CustomField](#) , [ActionBuff.IsBuff\(\)](#) , [ActionBuff.IsDebuff\(\)](#) ,  
[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Dispel(ActionBuff)

```
protected Dispel(ActionBuff value)
```

## Parameters

value [ActionBuff](#)

### Dispel(SkillCustomField, Row)

```
public Dispel(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## Dispel(Row)

```
public Dispel(ActionBuffSheet.Row row)
```

## Parameters

row [ActionBuffSheet.Row](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

## Returns

[object](#)

A new object that is a copy of this instance.

# Class Focus

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Focus : ActionBuff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← [ActionBuff](#) ← [Focus](#)

## Implements

[ICloneable](#)

## Inherited Members

[ActionBuff.RowData](#) , [ActionBuff.CustomField](#) , [ActionBuff.IsBuff\(\)](#) , [ActionBuff.IsDebuff\(\)](#) ,  
[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## Focus(Focus)

```
protected Focus(Focus value)
```

## Parameters

value [Focus](#)

## Focus(SkillCustomField, Row)

```
public Focus(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## Focus(Row)

```
public Focus(ActionBuffSheet.Row row)
```

## Parameters

row [ActionBuffSheet.Row](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

## Returns

[object](#)

A new object that is a copy of this instance.

# Class IceShield

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class IceShield : ActionBuff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← [ActionBuff](#) ← [IceShield](#)

## Implements

[ICloneable](#)

## Inherited Members

[ActionBuff.RowData](#) , [ActionBuff.CustomField](#) , [ActionBuff.IsBuff\(\)](#) , [ActionBuff.IsDebuff\(\)](#) ,  
[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### [IceShield\(IceShield\)](#)

```
protected IceShield(IceShield value)
```

#### Parameters

[value](#) [IceShield](#)

### [IceShield\(SkillCustomField, Row\)](#)

```
public IceShield(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## **IceShield(Row)**

```
public IceShield(ActionBuffSheet.Row row)
```

## Parameters

row [ActionBuffSheet.Row](#)

## Methods

### **Clone()**

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

## Returns

[object](#)

A new object that is a copy of this instance.

## **FrostBite(StatBuffSheet, BuffLinkSheet)**

```
public StatBuff FrostBite(StatBuffSheet statBuffSheet, BuffLinkSheet buffLinkSheet)
```

## Parameters

statBuffSheet [StatBuffSheet](#)

`buffLinkSheet` [BuffLinkSheet](#)

Returns

[StatBuff](#)

# Class StatBuff

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatBuff : Buff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← StatBuff

## Implements

[ICloneable](#)

## Inherited Members

[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### StatBuff(StatBuff)

```
protected StatBuff(StatBuff value)
```

#### Parameters

value [StatBuff](#)

### StatBuff(SkillCustomField, Row)

```
public StatBuff(SkillCustomField customField, StatBuffSheet.Row row)
```

#### Parameters

`customField` [SkillCustomField](#)

`row` [StatBuffSheet.Row](#)

## StatBuff(Row)

`public StatBuff(StatBuffSheet.Row row)`

### Parameters

`row` [StatBuffSheet.Row](#)

## Properties

### CustomField

`public SkillCustomField? CustomField { get; }`

### Property Value

[SkillCustomField?](#)

### RowData

`public StatBuffSheet.Row RowData { get; }`

### Property Value

[StatBuffSheet.Row](#)

### Stack

`public int Stack { get; }`

Property Value

[int](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

### GetModifier()

```
public StatModifier GetModifier()
```

Returns

[StatModifier](#)

### IsBuff()

```
public override bool IsBuff()
```

Returns

[bool](#)

### IsDebuff()

```
public override bool IsDebuff()
```

Returns

[bool](#)

## SetStack(int)

```
public void SetStack(int stack)
```

Parameters

stack [int](#)

# Class Stun

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Stun : ActionBuff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← [ActionBuff](#) ← Stun

## Implements

[ICloneable](#)

## Inherited Members

[ActionBuff.RowData](#) , [ActionBuff.CustomField](#) , [ActionBuff.IsBuff\(\)](#) , [ActionBuff.IsDebuff\(\)](#) ,  
[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## Stun(ActionBuff)

```
protected Stun(ActionBuff value)
```

## Parameters

value [ActionBuff](#)

## Stun(SkillCustomField, Row)

```
public Stun(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

`customField` [SkillCustomField](#)

`row` [ActionBuffSheet.Row](#)

## Stun(Row)

```
public Stun(ActionBuffSheet.Row row)
```

## Parameters

`row` [ActionBuffSheet.Row](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

## Returns

[object](#)

A new object that is a copy of this instance.

# Class Vampiric

Namespace: [Nekoyume.Model.Buff](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Vampiric : ActionBuff, ICloneable
```

## Inheritance

[object](#) ← [Buff](#) ← [ActionBuff](#) ← [Vampiric](#)

## Implements

[ICloneable](#)

## Inherited Members

[ActionBuff.RowData](#) , [ActionBuff.CustomField](#) , [ActionBuff.IsBuff\(\)](#) , [ActionBuff.IsDebuff\(\)](#) ,  
[Buff.BuffInfo](#) , [Buff.OriginalDuration](#) , [Buff.RemainedDuration](#) ,  
[Buff.GetTarget\(CharacterBase\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Vampiric(Vampiric)

```
protected Vampiric(Vampiric value)
```

#### Parameters

value [Vampiric](#)

### Vampiric(SkillCustomField, Row)

```
public Vampiric(SkillCustomField customField, ActionBuffSheet.Row row)
```

## Parameters

customField [SkillCustomField](#)

row [ActionBuffSheet.Row](#)

## Vampiric(Row, long)

```
public Vampiric(ActionBuffSheet.Row row, long basisPoint)
```

## Parameters

row [ActionBuffSheet.Row](#)

basisPoint [long](#)

## Properties

### BasisPoint

```
public long BasisPoint { get; }
```

## Property Value

[long](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

## Returns

## [object](#)

A new object that is a copy of this instance.

## GiveEffect(CharacterBase, SkillInfo, int, bool)

```
public Skill GiveEffect(CharacterBase affectedCharacter, Skill.SkillInfo skillInfo,  
int simulatorWaveTurn, bool copyCharacter = true)
```

### Parameters

affectedCharacter [CharacterBase](#)

skillInfo [Skill.SkillInfo](#)

simulatorWaveTurn [int](#)

copyCharacter [bool](#)

### Returns

[Skill](#)

## GiveEffectForArena(ArenaCharacter, ArenaSkillInfo, int)

```
public ArenaSkill GiveEffectForArena(ArenaCharacter affectedCharacter,  
ArenaSkill.ArenaSkillInfo skillInfo, int simulatorWaveTurn)
```

### Parameters

affectedCharacter [ArenaCharacter](#)

skillInfo [ArenaSkill.ArenaSkillInfo](#)

simulatorWaveTurn [int](#)

### Returns

[ArenaSkill](#)



# Namespace Nekoyume.Model.Character

## Enums

[SizeType](#)

# Enum SizeType

Namespace: [Nekoyume.Model.Character](#)

Assembly: Lib9c.dll

```
public enum SizeType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

L = 3

M = 2

S = 1

XL = 4

XS = 0

# Namespace Nekoyume.Model.Collection

## Classes

[CollectionFactory](#)

[FungibleCollectionMaterial](#)

[NonFungibleCollectionMaterial](#)

## Interfaces

[ICollectionMaterial](#)

## Enums

[MaterialType](#)

# Class CollectionFactory

Namespace: [Nekoyume.Model.Collection](#)

Assembly: Lib9c.dll

```
public static class CollectionFactory
```

## Inheritance

[object](#) ← CollectionFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### DeserializeMaterial(List)

```
public static ICollectionMaterial DeserializeMaterial(List serialized)
```

#### Parameters

**serialized** List

#### Returns

[ICollectionMaterial](#)

# Class FungibleCollectionMaterial

Namespace: [Nekoyume.Model.Collection](#)

Assembly: Lib9c.dll

```
public class FungibleCollectionMaterial : ICollectionMaterial, IBencodable
```

## Inheritance

[object](#) ← FungibleCollectionMaterial

## Implements

[ICollectionMaterial](#), [IBencodable](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### FungibleCollectionMaterial()

```
public FungibleCollectionMaterial()
```

### FungibleCollectionMaterial(IValue)

```
public FungibleCollectionMaterial(IValue bencoded)
```

## Parameters

bencoded IValue

### FungibleCollectionMaterial(List)

```
public FungibleCollectionMaterial(List serialized)
```

## Parameters

**serialized** List

## Properties

### Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

### Property Value

IValue

### Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

## ItemCount

```
public int ItemCount { get; set; }
```

### Property Value

[int](#)

## ItemId

```
public int ItemId { get; set; }
```

### Property Value

[int](#)

## Type

```
public MaterialType Type { get; }
```

### Property Value

[MaterialType](#)

## Methods

### BurnMaterial(ItemSheet.Row itemRow, Inventory inventory, long blockIndex)

Burns the material from the inventory based on the given item row, inventory, and block index.

```
public void BurnMaterial(ItemSheet.Row itemRow, Inventory inventory,  
long blockIndex)
```

### Parameters

**itemRow** [ItemSheet.Row](#)

The item row from the item sheet.

**inventory** [Inventory](#)

The inventory object.

blockIndex [long](#)

The block index.

# Interface ICollectionMaterial

Namespace: [Nekoyume.Model.Collection](#)

Assembly: Lib9c.dll

```
public interface ICollectionMaterial : IBencodable
```

## Inherited Members

IBencodable.Bencoded

## Properties

### ItemCount

```
int ItemCount { get; set; }
```

Property Value

[int](#)

### ItemId

```
int ItemId { get; set; }
```

Property Value

[int](#)

### Type

```
MaterialType Type { get; }
```

Property Value

## MaterialType

# Enum MaterialType

Namespace: [Nekoyume.Model.Collection](#)

Assembly: Lib9c.dll

```
public enum MaterialType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Fungible = 0

NonFungible = 1

# Class NonFungibleCollectionMaterial

Namespace: [Nekoyume.Model.Collection](#)

Assembly: Lib9c.dll

```
public class NonFungibleCollectionMaterial : ICollectionMaterial, IBencodable
```

## Inheritance

[object](#) ← NonFungibleCollectionMaterial

## Implements

[ICollectionMaterial](#), [IBencodable](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### NonFungibleCollectionMaterial()

```
public NonFungibleCollectionMaterial()
```

### NonFungibleCollectionMaterial(List)

```
public NonFungibleCollectionMaterial(List serialized)
```

## Parameters

serialized List

## Properties

# Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

## Property Value

IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

# ItemCount

```
public int ItemCount { get; set; }
```

## Property Value

[int](#)

# ItemId

```
public int ItemId { get; set; }
```

## Property Value

[int](#)

## Level

```
public int Level { get; set; }
```

Property Value

[int ↗](#)

## NonFungibleId

```
public Guid NonFungibleId { get; set; }
```

Property Value

[Guid ↗](#)

## SkillContains

```
public bool SkillContains { get; set; }
```

Property Value

[bool ↗](#)

## Type

```
public MaterialType Type { get; }
```

Property Value

[MaterialType](#)

## Methods

# BurnMaterial(ItemSheet.Row, Inventory, RequiredMaterial, long)

Burns the specified material from the inventory based on the item type.

```
public void BurnMaterial(ItemSheet.Row itemRow, Inventory inventory,  
CollectionSheet.RequiredMaterial materialInfo, long blockIndex)
```

## Parameters

**itemRow** [ItemSheet.Row](#)

The [ItemSheet.Row](#) object representing the item.

**inventory** [Inventory](#)

The [Inventory](#) object representing the player's inventory.

**materialInfo** [CollectionSheet.RequiredMaterial](#)

The [CollectionSheet.RequiredMaterial](#) object representing the material info.

**blockIndex** [long](#)

The block index where the burn operation is taking place.

## Exceptions

[ItemDoesNotExistException](#)

Thrown when the material item does not exist in the inventory.

[InvalidItemTypeException](#)

Thrown when the item type is not supported by [NonFungibleCollectionMaterial](#).

# Namespace Nekoyume.Model.Coupons

## Structs

[Coupon](#)

[RewardSet](#)

[RewardSet.Comparer](#)

# Struct Coupon

Namespace: [Nekoyume.Model.Coupons](#)

Assembly: Lib9c.dll

```
public readonly struct Coupon : IOrderedEnumerable<(int ItemId, uint Quantity)>,
IEnumerable<(int ItemId, uint Quantity)>, IEnumerable, IEquatable<Coupon>
```

## Implements

[IOrderedEnumerable<\(int ItemId, uint Quantity\)>](#),  
[IEnumerable<\(int ItemId, uint Quantity\)>](#), [IEnumerable](#), [IEquatable<Coupon>](#)

## Inherited Members

[ValueType.ToString\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### Coupon(IValue)

```
public Coupon(IValue serialized)
```

#### Parameters

serialized [IValue](#)

### Coupon(Guid, in RewardSet)

```
public Coupon(Guid id, in RewardSet rewards)
```

#### Parameters

id [Guid](#)

rewards [RewardSet](#)

## Coupon(Guid, IEnumerable<(int ItemId, uint Quantity)>)

```
public Coupon(Guid id, IEnumerable<(int ItemId, uint Quantity)> rewards)
```

### Parameters

`id` [Guid](#)

`rewards` [IEnumerable](#)<(int [ItemId](#), uint [Quantity](#))>

## Coupon(Guid, params (int ItemId, uint Quantity)[])

```
public Coupon(Guid id, params (int ItemId, uint Quantity)[] rewards)
```

### Parameters

`id` [Guid](#)

`rewards` (int [ItemId](#), uint [Quantity](#))[]

## Fields

### Id

```
public readonly Guid Id
```

### Field Value

[Guid](#)

## Rewards

```
public readonly RewardSet Rewards
```

## Field Value

[RewardSet](#)

## Methods

### Equals(Coupon)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(Coupon other)
```

#### Parameters

**other** [Coupon](#)

An object to compare with this object.

#### Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

### Equals(object?)

Indicates whether this instance and a specified object are equal.

```
public override bool Equals(object? obj)
```

#### Parameters

**obj** [object](#)

The object to compare with the current instance.

#### Returns

[bool](#)

[true](#) if `obj` and this instance are the same type and represent the same value; otherwise, [false](#).

## GetHashCode()

Returns the hash code for this instance.

```
public override int GetHashCode()
```

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Struct RewardSet

Namespace: [Nekoyume.Model.Coupons](#)

Assembly: Lib9c.dll

```
public readonly struct RewardSet : IImmutableDictionary<int, uint>,
IReadOnlyDictionary<int, uint>, IReadOnlyCollection<KeyValuePair<int, uint>>,
IEnumerable<KeyValuePair<int, uint>>, IEnumerable, IEquatable<RewardSet>
```

## Implements

[IImmutableDictionary](#)<[int](#), [uint](#)>, [IReadOnlyDictionary](#)<[int](#), [uint](#)>,  
[IReadOnlyCollection](#)<[KeyValuePair](#)<[int](#), [uint](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [uint](#)>>, [IEnumerable](#), [IEquatable](#)<[RewardSet](#)>

## Inherited Members

[ValueType.ToString\(\)](#), [object.Equals\(object, object\)](#), [object.GetType\(\)](#),  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### RewardSet(Dictionary)

```
public RewardSet(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### RewardSet(IEnumerable<(int ItemId, uint Quantity)>)

```
public RewardSet(IEnumerable<(int ItemId, uint Quantity)> rewards)
```

#### Parameters

rewards [IEnumerable](#)<([int](#) [ItemId](#), [uint](#) [Quantity](#))>

## RewardSet(in ImmutableDictionary<int, uint>)

```
public RewardSet(in ImmutableDictionary<int, uint> rewards)
```

### Parameters

rewards [ImmutableDictionary<int, uint>](#)

## RewardSet(params (int ItemId, uint Quantity)[])

```
public RewardSet(params (int ItemId, uint Quantity)[] rewards)
```

### Parameters

rewards [\(int ItemId, uint Quantity\)\[\]](#)

## Properties

### Count

Gets the number of elements in the collection.

```
public int Count { get; }
```

### Property Value

[int](#)

The number of elements in the collection.

### this[int]

Gets the element that has the specified key in the read-only dictionary.

```
public uint this[int key] { get; }
```

## Parameters

**key** [int](#)

The key to locate.

## Property Value

[uint](#)

The element that has the specified key in the read-only dictionary.

## Exceptions

[ArgumentNullException](#)

**key** is [null](#).

[KeyNotFoundException](#)

The property is retrieved and **key** is not found.

## Keys

Gets an enumerable collection that contains the keys in the read-only dictionary.

```
public IEnumerable<int> Keys { get; }
```

## Property Value

[IEnumerable](#)<[int](#)>

An enumerable collection that contains the keys in the read-only dictionary.

## Values

Gets an enumerable collection that contains the values in the read-only dictionary.

```
public IEnumerable<uint> Values { get; }
```

# Property Value

## [IEnumerable<uint>](#)

An enumerable collection that contains the values in the read-only dictionary.

# Methods

## Add(int, uint)

Adds an element with the specified key and value to the dictionary.

```
public IImmutableDictionary<int, uint> Add(int key, uint value)
```

### Parameters

#### key [int](#)

The key of the element to add.

#### value [uint](#)

The value of the element to add.

### Returns

## [IImmutableDictionary<int, uint>](#)

A new immutable dictionary that contains the additional key/value pair.

### Exceptions

#### [ArgumentException](#)

The given key already exists in the dictionary but has a different value.

## AddRange(IEnumerable<KeyValuePair<int, uint>>)

Adds the specified key/value pairs to the dictionary.

```
public IIImmutableDictionary<int, uint> AddRange(IEnumerable<KeyValuePair<int, uint>> pairs)
```

## Parameters

`pairs` [IEnumerable<KeyValuePair<int, uint>>](#)

The key/value pairs to add.

## Returns

[IIImmutableDictionary<int, uint>](#)

A new immutable dictionary that contains the additional key/value pairs.

## Exceptions

[ArgumentException](#)

One of the given keys already exists in the dictionary but has a different value.

## Clear()

Retrieves an empty dictionary that has the same ordering and key/value comparison rules as this dictionary instance.

```
public IIImmutableDictionary<int, uint> Clear()
```

## Returns

[IIImmutableDictionary<int, uint>](#)

An empty dictionary with equivalent ordering and key/value comparison rules.

## Contains(KeyValuePair<int, uint>)

Determines whether the immutable dictionary contains the specified key/value pair.

```
public bool Contains(KeyValuePair<int, uint> pair)
```

## Parameters

**pair** [KeyValuePair<int, uint>](#)

The key/value pair to locate.

## Returns

[bool](#)

[true](#) if the specified key/value pair is found in the dictionary; otherwise, [false](#).

## ContainsKey(int)

Determines whether the read-only dictionary contains an element that has the specified key.

```
public bool ContainsKey(int key)
```

## Parameters

**key** [int](#)

The key to locate.

## Returns

[bool](#)

[true](#) if the read-only dictionary contains an element that has the specified key; otherwise, [false](#).

## Exceptions

[ArgumentNullException](#)

**key** is [null](#).

## Equals(RewardSet)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(RewardSet other)
```

Parameters

`other` [RewardSet](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the `other` parameter; otherwise, [false](#).

## Equals(object?)

Indicates whether this instance and a specified object are equal.

```
public override bool Equals(object? obj)
```

Parameters

`obj` [object](#)

The object to compare with the current instance.

Returns

[bool](#)

[true](#) if `obj` and this instance are the same type and represent the same value; otherwise, [false](#).

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumarator<KeyValuePair<int, uint>> GetEnumerator()
```

Returns

[IEnumarator](#)<[KeyValuePair](#)<[int](#), [uint](#)>>

An enumerator that can be used to iterate through the collection.

## GetHashCode()

Returns the hash code for this instance.

```
public override int GetHashCode()
```

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## Remove(int)

Removes the element with the specified key from the immutable dictionary.

```
public IImmutableDictionary<int, uint> Remove(int key)
```

Parameters

key [int](#)

The key of the element to remove.

Returns

[IImmutableDictionary](#)<[int](#), [uint](#)>

A new immutable dictionary with the specified element removed; or this instance if the specified key cannot be found in the dictionary.

## RemoveRange(IEnumerable<int>)

Removes the elements with the specified keys from the immutable dictionary.

```
public IImmutableDictionary<int, uint> RemoveRange(IEnumerable<int> keys)
```

### Parameters

keys [IEnumerable<int>](#)

The keys of the elements to remove.

### Returns

[IImmutableDictionary<int, uint>](#)

A new immutable dictionary with the specified keys removed; or this instance if the specified keys cannot be found in the dictionary.

## Serialize()

```
public Dictionary Serialize()
```

### Returns

Dictionary

## SetItem(int, uint)

Sets the specified key and value in the immutable dictionary, possibly overwriting an existing value for the key.

```
public IImmutableDictionary<int, uint> SetItem(int key, uint value)
```

## Parameters

**key** `int`

The key of the entry to add.

**value** `uint`

The key value to set.

## Returns

`IImmutableDictionary<int, uint>`

A new immutable dictionary that contains the specified key/value pair.

## SetItems(IEnumerable<KeyValuePair<int, uint>>)

Sets the specified key/value pairs in the immutable dictionary, possibly overwriting existing values for the keys.

```
public IImmutableDictionary<int, uint> SetItems(IEnumerable<KeyValuePair<int, uint>> items)
```

## Parameters

**items** `IEnumerable<KeyValuePair<int, uint>>`

The key/value pairs to set in the dictionary. If any of the keys already exist in the dictionary, this method will overwrite their previous values.

## Returns

`IImmutableDictionary<int, uint>`

A new immutable dictionary that contains the specified key/value pairs.

## TryGetKey(int, out int)

Determines whether this dictionary contains a specified key.

```
public bool TryGetKey(int equalKey, out int actualKey)
```

## Parameters

**equalKey** [int](#)

The key to search for.

**actualKey** [int](#)

The matching key located in the dictionary if found, or **equalkey** if no match is found.

## Returns

[bool](#)

[true](#) if a match for **equalKey** is found; otherwise, [false](#).

## TryGetValue(int, out uint)

Gets the value that is associated with the specified key.

```
public bool TryGetValue(int key, out uint value)
```

## Parameters

**key** [int](#)

The key to locate.

**value** [uint](#)

When this method returns, the value associated with the specified key, if the key is found; otherwise, the default value for the type of the **value** parameter. This parameter is passed uninitialized.

## Returns

[bool](#)

[true](#) if the object that implements the [IReadOnlyDictionary<TKey, TValue>](#) interface contains an element that has the specified key; otherwise, [false](#).

## Exceptions

### [ArgumentNullException](#)

key is [null](#).

# Struct RewardSet.Comparer

Namespace: [Nekoyume.Model.Coupons](#)

Assembly: Lib9c.dll

```
public struct RewardSet.Comparer : IComparer<RewardSet>
```

## Implements

[IComparer](#)<[RewardSet](#)>

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Methods

### Compare(RewardSet, RewardSet)

Compares two objects and returns a value indicating whether one is less than, equal to, or greater than the other.

```
public int Compare(RewardSet x, RewardSet y)
```

#### Parameters

x [RewardSet](#)

The first object to compare.

y [RewardSet](#)

The second object to compare.

#### Returns

[int](#)

A signed integer that indicates the relative values of  $x$  and  $y$ , as shown in the following table.

Value	Meaning
<b>Less than zero</b>	$x$ is less than $y$ .
<b>Zero</b>	$x$ equals $y$ .
<b>Greater than zero</b>	$x$ is greater than $y$ .

# Namespace Nekoyume.Model.Elemental

## Classes

[ElementalTypeComparer](#)

[ElementalTypeExtension](#)

## Enums

[ElementalResult](#)

[ElementalType](#)

# Enum ElementalResult

Namespace: [Nekoyume.Model.Elemental](#)

Assembly: Lib9c.dll

```
public enum ElementalResult
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Draw = 1

Lose = 2

Win = 0

# Enum ElementalType

Namespace: [Nekoyume.Model.Elemental](#)

Assembly: Lib9c.dll

```
public enum ElementalType
```

## Extension Methods

[ElementalTypeExtension.GetBattleResult\(ElementalType, ElementalType\)](#) ,  
[ElementalTypeExtension.GetDamage\(ElementalType, ElementalType, long\)](#) ,  
[ElementalTypeExtension.GetMultiplier\(ElementalType, ElementalType\)](#) ,  
[ElementalTypeExtension.TryGetLoseCase\(ElementalType, out ElementalType\)](#) ,  
[ElementalTypeExtension.TryGetWinCase\(ElementalType, out ElementalType\)](#) ,  
[StateExtensions.Serialize\(Enum\)](#).

## Fields

Fire = 1

Land = 3

Normal = 0

Water = 2

Wind = 4

# Class ElementalTypeComparer

Namespace: [Nekoyume.Model.Elemental](#)

Assembly: Lib9c.dll

```
public class ElementalTypeComparer : IEqualityComparer<ElementalType>
```

## Inheritance

[object](#) ← ElementalTypeComparer

## Implements

[IEqualityComparer](#)<[ElementalType](#)>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Instance

```
public static readonly ElementalTypeComparer Instance
```

### Field Value

[ElementalTypeComparer](#)

## Methods

### Equals(ElementalType, ElementalType)

Determines whether the specified objects are equal.

```
public bool Equals(ElementalType x, ElementalType y)
```

## Parameters

x [ElementalType](#)

The first object of type T to compare.

y [ElementalType](#)

The second object of type T to compare.

## Returns

[bool](#)

[true](#) if the specified objects are equal; otherwise, [false](#).

## GetHashCode(ElementalType)

Returns a hash code for the specified object.

```
public int GetHashCode(ElementalType obj)
```

## Parameters

obj [ElementalType](#)

The [object](#) for which a hash code is to be returned.

## Returns

[int](#)

A hash code for the specified object.

## Exceptions

[ArgumentNullException](#)

The type of obj is a reference type and obj is [null](#).

# Class ElementTypeExtension

Namespace: [Nekoyume.Model.Elemental](#)

Assembly: Lib9c.dll

```
public static class ElementTypeExtension
```

## Inheritance

[object](#) ← ElementTypeExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### WinMultiplier

```
public const decimal WinMultiplier = 1.2
```

Field Value

[decimal](#)

## Methods

### GetAllTypes()

```
public static List<ElementType> GetAllTypes()
```

Returns

[List](#)<[ElementType](#)>

## GetBattleResult(ElementalType, ElementalType)

```
public static ElementalResult GetBattleResult(this ElementalType from,  
ElementalType to)
```

### Parameters

from [ElementalType](#)

to [ElementalType](#)

### Returns

[ElementalResult](#)

1: Win 0: Draw -1: Lose

## GetDamage(ElementalType, ElementalType, long)

```
public static long GetDamage(this ElementalType from, ElementalType to, long damage)
```

### Parameters

from [ElementalType](#)

to [ElementalType](#)

damage [long](#)

### Returns

[long](#)

## GetMultiplier(ElementalType, ElementalType)

```
public static decimal GetMultiplier(this ElementalType from, ElementalType to)
```

Parameters

`from ElementType`  
`to ElementType`

Returns

`decimal` ↴

## TryGetLoseCase(ElementalType, out ElementalType)

```
public static bool TryGetLoseCase(this ElementalType lose, out ElementalType win)
```

Parameters

`lose ElementType`  
`win ElementType`

Returns

`bool` ↴

## TryGetWinCase(ElementalType, out ElementalType)

```
public static bool TryGetWinCase(this ElementalType win, out ElementalType lose)
```

Parameters

`win ElementType`  
`lose ElementType`

Returns

`bool` ↴

# Namespace Nekoyume.Model.EnumType

## Classes

[BattleTypeExtensions](#)

## Enums

[ArenaType](#)

[BattleType](#)

[CraftType](#)

[Grade](#)

[RuneSlotType](#)

[RuneType](#)

[RuneUsePlace](#)

[StatReferenceType](#)

[TradeType](#)

# Enum ArenaType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum ArenaType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Championship = 2

OffSeason = 0

Season = 1

# Enum BattleType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum BattleType
```

## Extension Methods

[BattleTypeExtensions.IsEquippableRune\(BattleType, RuneUsePlace\)](#) ,  
[StateExtensions.Serialize\(Enum\)](#).

## Fields

Adventure = 1

Arena = 2

End = 4

Raid = 3

# Class BattleTypeExtensions

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public static class BattleTypeExtensions
```

## Inheritance

[object](#) ← BattleTypeExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### IsEquippableRune(BattleType, RuneUsePlace)

```
public static bool IsEquippableRune(this BattleType battleType,  
RuneUsePlace runePlace)
```

#### Parameters

battleType [BattleType](#)

runePlace [RuneUsePlace](#)

#### Returns

[bool](#)

# Enum CraftType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum CraftType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Mimisbrunnr = 2

Normal = 0

Premium = 1

# Enum Grade

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum Grade
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Divinity = 6

Epic = 3

Legendary = 5

Normal = 1

Rare = 2

Unique = 4

# Enum RuneSlotType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum RuneSlotType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Crystal = 4

Default = 1

Ncg = 2

Stake = 3

# Enum RuneType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum RuneType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Skill = 2

Stat = 1

# Enum RuneUsePlace

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum RuneUsePlace
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Adventure = 1

AdventureAndArena = 3

All = 7

Arena = 2

Raid = 4

RaidAndAdventure = 5

RaidAndArena = 6

# Enum StatReferenceType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum StatReferenceType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Caster = 0

Target = 1

# Enum TradeType

Namespace: [Nekoyume.Model.EnumType](#)

Assembly: Lib9c.dll

```
public enum TradeType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Buy = 0

Sell = 1

# Namespace Nekoyume.Model.Event

## Classes

[EventDungeonInfo](#)

# Class EventDungeonInfo

Namespace: [Nekoyume.Model.Event](#)

Assembly: Lib9c.dll

```
public class EventDungeonInfo : IState
```

## Inheritance

[object](#) ← EventDungeonInfo

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventDungeonExtensions.GetRemainingTicketsConsiderReset\(EventDungeonInfo, EventScheduleSheet.Row, long\)](#)

## Constructors

### EventDungeonInfo(IValue)

```
public EventDungeonInfo(IValue serialized)
```

#### Parameters

serialized IValue

### EventDungeonInfo(List)

```
public EventDungeonInfo(List serialized)
```

#### Parameters

**serialized** List

## EventDungeonInfo(int, int, int, int)

```
public EventDungeonInfo(int resetTicketsInterval = 0, int remainingTickets = 0, int  
numberOfTicketPurchases = 0, int clearedStageId = 0)
```

### Parameters

resetTicketsInterval [int](#)

remainingTickets [int](#)

numberOfTicketPurchases [int](#)

clearedStageId [int](#)

### Properties

#### ClearedStageId

```
public int ClearedStageId { get; }
```

#### Property Value

[int](#)

#### NumberOfTicketPurchases

```
public int NumberOfTicketPurchases { get; }
```

#### Property Value

[int](#)

## RemainingTickets

```
public int RemainingTickets { get; }
```

Property Value

[int](#)

## ResetTicketsInterval

```
public int ResetTicketsInterval { get; }
```

Property Value

[int](#)

## Methods

### ClearStage(int)

```
public void ClearStage(int stageId)
```

Parameters

stageId [int](#)

### DeriveAddress(Address, int)

```
public static Address DeriveAddress(Address address, int dungeonId)
```

Parameters

address Address

dungeonId [int](#)

Returns

Address

## Equals(EventDungeonInfo)

`protected bool Equals(EventDungeonInfo other)`

Parameters

`other` [EventDungeonInfo](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

`public override bool Equals(object obj)`

Parameters

`obj` [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## HasTickets(int)

```
public bool HasTickets(int tickets)
```

Parameters

[tickets](#) [int](#)

Returns

[bool](#)

## IncreaseNumberOfTicketPurchases()

```
public void IncreaseNumberOfTicketPurchases()
```

## IsCleared(int)

```
public bool IsCleared(int stageId)
```

Parameters

[stageId](#) [int](#)

Returns

[bool](#)

## ResetTickets(int, int)

```
public void ResetTickets(int interval, int tickets)
```

Parameters

interval [int](#)

tickets [int](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## TryUseTickets(int)

```
public bool TryUseTickets(int tickets)
```

Parameters

tickets [int](#)

Returns

[bool](#)

# Namespace Nekoyume.Model.Faucet

## Classes

[FaucetRunelInfo](#)

# Class FaucetRuneInfo

Namespace: [Nekoyume.Model.Faucet](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FaucetRuneInfo
```

## Inheritance

[object](#) ← FaucetRuneInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### FaucetRuneInfo(List)

```
public FaucetRuneInfo(List serialized)
```

#### Parameters

serialized List

### FaucetRuneInfo(int, int)

```
public FaucetRuneInfo(int runeId, int amount)
```

#### Parameters

runeId [int](#)

amount [int](#)

# Properties

## Amount

```
public int Amount { get; }
```

### Property Value

[int](#)

## RunelId

```
public int RuneId { get; }
```

### Property Value

[int](#)

# Methods

## Equals(FaucetRunelInfo)

```
protected bool Equals(FaucetRunelInfo other)
```

### Parameters

**other** [FaucetRunelInfo](#)

### Returns

[bool](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Namespace Nekoyume.Model.Garages

## Classes

[FungibleItemGarage](#)

## Interfaces

[IFungibleItemGarage](#)

[IGarage<T1, T2>](#)

# Class FungibleItemGarage

Namespace: [Nekoyume.Model.Garages](#)

Assembly: Lib9c.dll

```
public class FungibleItemGarage : IFungibleItemGarage, IGarage<IFungibleItemGarage,  
int>
```

## Inheritance

[object](#) ← FungibleItemGarage

## Implements

[IFungibleItemGarage](#), [IGarage<IFungibleItemGarage, int>](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### FungibleItemGarage(IValue?)

```
public FungibleItemGarage(IValue? serialized)
```

#### Parameters

serialized [IValue](#)

### FungibleItemGarage(IFungibleItem, int)

```
public FungibleItemGarage(IFungibleItem item, int count)
```

#### Parameters

item [IFungibleItem](#)

count [int](#)

## Properties

### Count

```
public int Count { get; }
```

#### Property Value

[int](#)

## Item

```
public IFungibleItem Item { get; }
```

#### Property Value

[IFungibleItem](#)

## Methods

### Deliver(IFungibleItemGarage, int)

```
public void Deliver(IFungibleItemGarage to, int count)
```

#### Parameters

to [IFungibleItemGarage](#)

count [int](#)

### Load(int)

```
public void Load(int count)
```

Parameters

count [int](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Unload(int)

```
public void Unload(int count)
```

Parameters

count [int](#)

# Interface IFungibleItemGarage

Namespace: [Nekoyume.Model.Garages](#)

Assembly: Lib9c.dll

```
public interface IFungibleItemGarage : IGarage<IFungibleItemGarage, int>
```

## Inherited Members

[IGarage<IFungibleItemGarage, int>.Load\(int\)](#) ,  
[IGarage<IFungibleItemGarage, int>.Deliver\(IFungibleItemGarage, int\)](#) ,  
[IGarage<IFungibleItemGarage, int>.Unload\(int\)](#).

## Properties

### Count

```
int Count { get; }
```

#### Property Value

[int](#)

### Item

```
IFungibleItem Item { get; }
```

#### Property Value

[IFungibleItem](#)

# Interface IGarage<T1, T2>

Namespace: [Nekoyume.Model.Garages](#)

Assembly: Lib9c.dll

```
public interface IGarage<in T1, in T2> where T1 : IGarage<in T1, in T2>
```

## Type Parameters

T1

T2

## Methods

### Deliver(T1, T2)

```
void Deliver(T1 to, T2 amount)
```

#### Parameters

to T1

amount T2

### Load(T2)

```
void Load(T2 amount)
```

#### Parameters

amount T2

### Unload(T2)

```
void Unload(T2 amount)
```

## Parameters

amount T2

# Namespace Nekoyume.Model.GrandFinale

## Classes

[GrandFinaleInformation](#)

# Class GrandFinaleInformation

Namespace: [Nekoyume.Model.GrandFinale](#)

Assembly: Lib9c.dll

```
public class GrandFinaleInformation : IState
```

## Inheritance

[object](#) ← GrandFinaleInformation

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GrandFinaleInformation(List)

```
public GrandFinaleInformation(List serialized)
```

#### Parameters

serialized List

### GrandFinaleInformation(Address, int)

```
public GrandFinaleInformation(Address avatarAddress, int grandFinaleId)
```

#### Parameters

avatarAddress Address

grandFinaleId [int](#)

## Fields

### Address

[public](#) Address Address

### Field Value

Address

## Methods

### DeriveAddress(Address, int)

[public static](#) Address [DeriveAddress](#)(Address avatarAddress, [int](#) grandFinaleId)

### Parameters

avatarAddress Address

grandFinaleId [int](#)

### Returns

Address

### GetBattleRecordList()

[public](#) List<KeyValuePair<Address, [bool](#)>> GetBattleRecordList()

### Returns

[List](#)<[KeyValuePair](#)<Address, [bool](#)>>

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## TryGetBattleRecord(Address, out bool)

```
public bool TryGetBattleRecord(Address enemyAddress, out bool win)
```

Parameters

enemyAddress Address

win [bool](#)

Returns

[bool](#)

## UpdateRecord(Address, bool)

```
public void UpdateRecord(Address enemyAddress, bool win)
```

Parameters

enemyAddress Address

win [bool](#)

# Namespace Nekoyume.Model.Guild

## Classes

[Guild](#)

[GuildDelegatee](#)

[GuildDelegator](#)

[GuildParticipant](#)

[GuildRejoinCooldown](#)

[GuildRepository](#)

# Class Guild

Namespace: [Nekoyume.Model.Guild](#)

Assembly: Lib9c.dll

```
public class Guild : IBencodable, IEquatable<Guild>
```

## Inheritance

[object](#) ← Guild

## Implements

IBencodable, [IEquatable](#)<[Guild](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Guild(GuildAddress, IValue, GuildRepository)

```
public Guild(GuildAddress address, IValue bencoded, GuildRepository repository)
```

#### Parameters

address [GuildAddress](#)

bencoded IValue

repository [GuildRepository](#)

### Guild(GuildAddress, AgentAddress, Address, GuildRepository)

```
public Guild(GuildAddress address, AgentAddress guildMasterAddress, Address  
validatorAddress, GuildRepository repository)
```

## Parameters

address [GuildAddress](#)

guildMasterAddress [AgentAddress](#)

validatorAddress Address

repository [GuildRepository](#)

## Fields

### GuildMasterAddress

```
public readonly AgentAddress GuildMasterAddress
```

Field Value

[AgentAddress](#)

### ValidatorAddress

```
public readonly Address ValidatorAddress
```

Field Value

Address

## Properties

### Address

```
public GuildAddress Address { get; }
```

Property Value

## [GuildAddress](#)

### Bencoded

```
public List Bencoded { get; }
```

Property Value

List

### Repository

```
public GuildRepository Repository { get; }
```

Property Value

[GuildRepository](#)

### Methods

#### ClaimReward(Address, long)

```
public void ClaimReward(Address validatorAddress, long height)
```

Parameters

validatorAddress Address

height [long](#)

#### Equals(Guild?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(Guild? other)
```

## Parameters

**other** [Guild](#)

An object to compare with this object.

## Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

## Parameters

**obj** [object](#)

The object to compare with the current object.

## Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

## Returns

[int](#) ↗

A hash code for the current object.

# Class GuildDelegatee

Namespace: [Nekoyume.Model.Guild](#)

Assembly: Lib9c.dll

```
public class GuildDelegatee : Delegatee<GuildDelegator, GuildDelegatee>,  
IDelegatee, IEquatable<GuildDelegatee>
```

## Inheritance

[object](#) ← [Delegatee<GuildDelegator, GuildDelegatee>](#) ← [GuildDelegatee](#)

## Implements

[IDelegatee](#), [IEquatable<GuildDelegatee>](#)

## Inherited Members

[Delegatee<GuildDelegator, GuildDelegatee>.DelegationChanged](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Enjailed](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Unjailed](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Metadata](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Repository](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Address](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.AccountAddress](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.MetadataAddress](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.DelegationCurrency](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.RewardCurrencies](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.DelegationPoolAddress](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.RewardPoolAddress](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.RewardRemainderPoolAddress](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.SlashedPoolAddress](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.UnbondingPeriod](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.MaxUnbondLockInEntries](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.MaxRebondGraceEntries](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Delegators](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.TotalDelegated](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.TotalShares](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Jailed](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.JailedUntil](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Tombstoned](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.MetadataBencoded](#) ,

[Delegatee<GuildDelegator, GuildDelegatee>.ShareFromFAV\(FungibleAssetValue\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.FAVFromShare\(BigInteger\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Bond\(IDelegate, FungibleAssetValue, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Unbond\(IDelegate, BigInteger, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.DistributeReward\(IDelegate, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Jail\(long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Unjail\(long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Tombstone\(\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.BondAddress\(Address\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.UnbondLockInAddress\(Address\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.RebondGraceAddress\(Address\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.CurrentLumpSumRewardsRecordAddress\(\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.LumpSumRewardsRecordAddress\(long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Bond\(GuildDelegator, FungibleAssetValue, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Unbond\(GuildDelegator, BigInteger, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.DistributeReward\(GuildDelegator, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.CollectRewards\(long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.Slash\(BigInteger, long, long\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.AddUnbondingRef\(UnbondingRef\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.RemoveUnbondingRef\(UnbondingRef\)](#) ,  
[Delegatee<GuildDelegator, GuildDelegatee>.CalculateReward\(BigInteger, IEnumerable<LumpSumRewardsRecord>\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GuildDelegatee(Address, GuildRepository)

```
public GuildDelegatee(Address address, GuildRepository repository)
```

## Parameters

address Address

repository [GuildRepository](#)

# GuildDelegatee(Address, IEnumerable<Currency>, GuildRepository)

```
public GuildDelegatee(Address address, IEnumerable<Currency> rewardCurrencies,  
GuildRepository repository)
```

## Parameters

address Address

rewardCurrencies [IEnumerable](#)<Currency>

repository [GuildRepository](#)

## Methods

### Activate()

```
public void Activate()
```

### Deactivate()

```
public void Deactivate()
```

### Equals(GuildDelegatee?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(GuildDelegatee? other)
```

## Parameters

other [GuildDelegatee](#)

An object to compare with this object.

## Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

# Class GuildDelegator

Namespace: [Nekoyume.Model.Guild](#)

Assembly: Lib9c.dll

```
public class GuildDelegator : Delegator<GuildDelegatee, GuildDelegator>,
IDelegate, IEquatable<GuildDelegator>
```

## Inheritance

[object](#) ← [Delegator<GuildDelegatee, GuildDelegator>](#) ← [GuildDelegator](#)

## Implements

[IDelegate](#), [IEquatable<GuildDelegator>](#)

## Inherited Members

[Delegator<GuildDelegatee, GuildDelegator>.Metadata](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.Repository](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.Address](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.AccountAddress](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.MetadataAddress](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.DelegationPoolAddress](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.RewardAddress](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.Delegatees](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.MetadataBencoded](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.Delegate\(GuildDelegatee, FungibleAssetValue, long\)](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.Undelegate\(GuildDelegatee, BigInteger, long\)](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.Redelegate\(GuildDelegatee, GuildDelegatee, BigInteger, long\)](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.CancelUndelegate\(GuildDelegatee, FungibleAssetValue, long\)](#) ,  
[Delegator<GuildDelegatee, GuildDelegator>.ClaimReward\(GuildDelegatee, long\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

## GuildDelegator(Address, Address, GuildRepository)

```
public GuildDelegator(Address address, Address delegationPoolAddress,  
GuildRepository repository)
```

### Parameters

address Address

delegationPoolAddress Address

repository [GuildRepository](#)

## GuildDelegator(Address, GuildRepository)

```
public GuildDelegator(Address address, GuildRepository repository)
```

### Parameters

address Address

repository [GuildRepository](#)

## Methods

### Equals(GuildDelegator?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(GuildDelegator? other)
```

### Parameters

other [GuildDelegator](#)

An object to compare with this object.

## Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

# Class GuildParticipant

Namespace: [Nekoyume.Model.Guild](#)

Assembly: Lib9c.dll

```
public class GuildParticipant : IBencodable, IEquatable<GuildParticipant>
```

## Inheritance

[object](#) ← GuildParticipant

## Implements

IBencodable, [IEquatable](#)<[GuildParticipant](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GuildParticipant(AgentAddress, IValue, GuildRepository)

```
public GuildParticipant(AgentAddress address, IValue bencoded,  
GuildRepository repository)
```

#### Parameters

address [AgentAddress](#)

bencoded IValue

repository [GuildRepository](#)

### GuildParticipant(AgentAddress, GuildAddress, GuildRepository)

```
public GuildParticipant(AgentAddress address, GuildAddress guildAddress,
```

```
GuildRepository repository)
```

## Parameters

address [AgentAddress](#)

guildAddress [GuildAddress](#)

repository [GuildRepository](#)

## Fields

### GuildAddress

```
public readonly GuildAddress GuildAddress
```

#### Field Value

[GuildAddress](#)

## Properties

### Address

```
public AgentAddress Address { get; }
```

#### Property Value

[AgentAddress](#)

### Bencoded

```
public List Bencoded { get; }
```

#### Property Value

List

## DelegationPoolAddress

```
public Address DelegationPoolAddress { get; }
```

Property Value

Address

## Repository

```
public GuildRepository Repository { get; }
```

Property Value

[GuildRepository](#)

## RewardAddress

```
public Address RewardAddress { get; }
```

Property Value

Address

## Methods

### ClaimReward(Guild, long)

```
public void ClaimReward(Guild guild, long height)
```

Parameters

guild [Guild](#)

height [long](#)

## Delegate(Guild, FungibleAssetValue, long)

```
public void Delegate(Guild guild, FungibleAssetValue fav, long height)
```

### Parameters

guild [Guild](#)

fav FungibleAssetValue

height [long](#)

## Equals(GuildParticipant?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(GuildParticipant? other)
```

### Parameters

other [GuildParticipant](#)

An object to compare with this object.

### Returns

[bool](#)

[true](#) if the current object is equal to the `other` parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

## Parameters

**obj** [object](#)

The object to compare with the current object.

## Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

## Returns

[int](#)

A hash code for the current object.

## Redelegate(Guild, Guild, BigInteger, long)

```
public void Redelegate(Guild srcGuild, Guild dstGuild, BigInteger share,  
long height)
```

## Parameters

**srcGuild** [Guild](#)

**dstGuild** [Guild](#)

**share** [BigInteger](#)

height [long](#)

## Undelegate(Guild, BigInteger, long)

```
public void Undelegate(Guild guild, BigInteger share, long height)
```

### Parameters

guild [Guild](#)

share [BigInteger](#)

height [long](#)

# Class GuildRejoinCooldown

Namespace: [Nekoyume.Model.Guild](#)

Assembly: Lib9c.dll

```
public class GuildRejoinCooldown : IBencodable, IEquatable<GuildRejoinCooldown>
```

## Inheritance

[object](#) ← GuildRejoinCooldown

## Implements

IBencodable, [IEquatable](#)<[GuildRejoinCooldown](#)>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GuildRejoinCooldown(AgentAddress, IValue)

```
public GuildRejoinCooldown(AgentAddress agentAddress, IValue bencoded)
```

#### Parameters

agentAddress [AgentAddress](#)

bencoded IValue

### GuildRejoinCooldown(AgentAddress, Integer)

```
public GuildRejoinCooldown(AgentAddress agentAddress, Integer bencoded)
```

#### Parameters

agentAddress [AgentAddress](#)

bencoded Integer

## GuildRejoinCooldown(AgentAddress, long)

`public GuildRejoinCooldown(AgentAddress agentAddress, long quitHeight)`

Parameters

agentAddress [AgentAddress](#)

quitHeight [long](#)

## Properties

### AgentAddress

`public AgentAddress AgentAddress { get; }`

Property Value

[AgentAddress](#)

### Bencoded

`public Integer Bencoded { get; }`

Property Value

Integer

### ReleaseHeight

```
public long ReleaseHeight { get; }
```

Property Value

[long](#)

## Methods

### Cooldown(long)

```
public long Cooldown(long currentHeight)
```

Parameters

currentHeight [long](#)

Returns

[long](#)

### Equals(GuildRejoinCooldown?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(GuildRejoinCooldown? other)
```

Parameters

other [GuildRejoinCooldown](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the `other` parameter; otherwise, [false](#).

# Class GuildRepository

Namespace: [Nekoyume.Model.Guild](#)

Assembly: Lib9c.dll

```
public class GuildRepository : DelegationRepository, IDelegationRepository
```

## Inheritance

[object](#) ← [DelegationRepository](#) ← GuildRepository

## Implements

[IDelegationRepository](#)

## Inherited Members

[DelegationRepository.previousWorld](#) , [DelegationRepository.delegateeAccount](#) ,  
[DelegationRepository.delegatorAccount](#) , [DelegationRepository.delegateeMetadataAccount](#) ,  
[DelegationRepository.delegatorMetadataAccount](#) , [DelegationRepository.bondAccount](#) ,  
[DelegationRepository.unbondLockInAccount](#) , [DelegationRepository.rebondGraceAccount](#) ,  
[DelegationRepository.unbondingSetAccount](#) ,  
[DelegationRepository.lumpSumRewardsRecordAccount](#) ,  
[DelegationRepository.ActionContext](#) , [DelegationRepository.DelegateeAccountAddress](#) ,  
[DelegationRepository.DelegatorAccountAddress](#) ,  
[DelegationRepository.GetDelegateeMetadata\(Address\)](#) ,  
[DelegationRepository.GetDelegatorMetadata\(Address\)](#) ,  
[DelegationRepository.GetBond\(IDelegatee, Address\)](#) ,  
[DelegationRepository.GetUnbondLockIn\(IDelegatee, Address\)](#) ,  
[DelegationRepository.GetUnlimitedUnbondLockIn\(Address\)](#) ,  
[DelegationRepository.GetRebondGrace\(IDelegatee, Address\)](#) ,  
[DelegationRepository.GetUnlimitedRebondGrace\(Address\)](#) ,  
[DelegationRepository.GetUnbondingSet\(\)](#) ,  
[DelegationRepository.GetLumpSumRewardsRecord\(IDelegatee, long\)](#) ,  
[DelegationRepository.GetCurrentLumpSumRewardsRecord\(IDelegatee\)](#) ,  
[DelegationRepository.GetBalance\(Address, Currency\)](#) ,  
[DelegationRepository.SetDelegateeMetadata\(DelegateeMetadata\)](#) ,  
[DelegationRepository.SetDelegatorMetadata\(DelegatorMetadata\)](#) ,  
[DelegationRepository.SetBond\(Bond\)](#) ,  
[DelegationRepository.SetUnbondLockIn\(UnbondLockIn\)](#) ,  
[DelegationRepository.SetRebondGrace\(RebondGrace\)](#) ,  
[DelegationRepository.SetUnbondingSet\(UnbondingSet\)](#) ,

[DelegationRepository.SetLumpSumRewardsRecord\(LumpSumRewardsRecord\)](#) ,  
[DelegationRepository.TransferAsset\(Address, Address, FungibleAssetValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Extension Methods

[GuildBanModule.Ban\(GuildRepository, GuildAddress, AgentAddress, AgentAddress\)](#) ,  
[GuildBanModule.IsBanned\(GuildRepository, GuildAddress, Address\)](#) ,  
[GuildBanModule.RemoveBanList\(GuildRepository, GuildAddress\)](#) ,  
[GuildBanModule.Unban\(GuildRepository, GuildAddress, AgentAddress, Address\)](#) ,  
[GuildDelegateeModule.CreateGuildDelegatee\(GuildRepository, Address\)](#) ,  
[GuildDelegateeModule.TryGetGuildDelegatee\(GuildRepository, Address, out GuildDelegatee?\)](#) ,  
[GuildMemberCounterModule.DecreaseGuildMemberCount\(GuildRepository, GuildAddress\)](#) ,  
[GuildMemberCounterModule.GetGuildMemberCount\(GuildRepository, GuildAddress\)](#) ,  
[GuildMemberCounterModule.IncreaseGuildMemberCount\(GuildRepository, GuildAddress\)](#) ,  
[GuildModule.MakeGuild\(GuildRepository, GuildAddress, Address\)](#) ,  
[GuildModule.RemoveGuild\(GuildRepository\)](#) ,  
[GuildModule.TryGetGuild\(GuildRepository, GuildAddress, out Guild?\)](#) ,  
[GuildParticipantModule.Delegate\(GuildRepository, AgentAddress, FungibleAssetValue\)](#) ,  
[GuildParticipantModule.GetJoinedGuild\(GuildRepository, AgentAddress\)](#) ,  
[GuildParticipantModule.JoinGuild\(GuildRepository, GuildAddress, AgentAddress\)](#) ,  
[GuildParticipantModule.LeaveGuild\(GuildRepository, AgentAddress\)](#) ,  
[GuildParticipantModule.MoveGuild\(GuildRepository, AgentAddress, GuildAddress\)](#) ,  
[GuildParticipantModule.Redelegate\(GuildRepository, AgentAddress, GuildAddress\)](#) ,  
[GuildParticipantModule.TryGetGuildParticipant\(GuildRepository, AgentAddress, out GuildParticipant?\)](#)

## Constructors

### GuildRepository(IWorld, IActionContext)

```
public GuildRepository(IWorld world, IActionContext actionContext)
```

## Parameters

**world** IWorld

`actionContext IActionContext`

## GuildRepository(IDelegationRepository)

`public GuildRepository(IDelegationRepository repository)`

### Parameters

`repository` [IDelegationRepository](#)

## Properties

### World

`public override IWorld World { get; }`

### Property Value

`IWorld`

## Methods

### GetDelegatee(Address)

`public override IDegatee GetDelegatee(Address address)`

### Parameters

`address` `Address`

### Returns

[IDegatee](#)

## GetDelegator(Address)

```
public override IDelegate GetDelegator(Address address)
```

Parameters

address Address

Returns

[IDelegate](#)

## GetGuild(Address)

```
public Guild GetGuild(Address address)
```

Parameters

address Address

Returns

[Guild](#)

## GetGuildDelegatee(Address)

```
public GuildDelegatee GetGuildDelegatee(Address address)
```

Parameters

address Address

Returns

[GuildDelegatee](#)

## GetGuildDelegator(Address)

```
public GuildDelegator GetGuildDelegator(Address address)
```

Parameters

address Address

Returns

[GuildDelegator](#)

## GetGuildParticipant(Address)

```
public GuildParticipant GetGuildParticipant(Address address)
```

Parameters

address Address

Returns

[GuildParticipant](#)

## RemoveGuildParticipant(Address)

```
public void RemoveGuildParticipant(Address guildParticipantAddress)
```

Parameters

guildParticipantAddress Address

## SetDelegatee(IDelegatee)

```
public override void SetDelegatee(IDelegatee delegatee)
```

Parameters

delegatee [IDelegatee](#)

## SetDelegator(IDelegator)

```
public override void SetDelegator(IDelegator delegator)
```

Parameters

delegator [IDelegator](#)

## SetGuild(Guild)

```
public void SetGuild(Guild guild)
```

Parameters

guild [Guild](#)

## SetGuildDelegator(GuildDelegator)

```
public void SetGuildDelegator(GuildDelegator guildDelegator)
```

Parameters

guildDelegator [GuildDelegator](#)

## SetGuildDelegatee(GuildDelegatee)

```
public void SetGuildDelegatee(GuildDelegatee guildDelegatee)
```

Parameters

guildDelegatee [GuildDelegatee](#)

## SetGuildParticipant(GuildParticipant)

```
public void SetGuildParticipant(GuildParticipant guildParticipant)
```

### Parameters

guildParticipant [GuildParticipant](#)

## UpdateWorld(IWorld)

```
public override void UpdateWorld(IWorld world)
```

### Parameters

world IWorld

# Namespace Nekoyume.Model.Item

## Classes

[Armor](#)

[Aura](#)

[Belt](#)

[Consumable](#)

[Costume](#)

[Equipment](#)

[Grimoire](#)

[Inventory](#)

[Inventory.Item](#)

[ItemBase](#)

[ItemFactory](#)

[ItemSubTypeComparer](#)

[ItemTypeComparer](#)

[ItemUsable](#)

[Material](#)

[Necklace](#)

[Ring](#)

[ShopItem](#)

[TradableMaterial](#)

[Weapon](#)

## Structs

[OrderLock](#)

# Interfaces

[IEquipableItem](#)

[IFungibleItem](#)

[IItem](#)

[ILock](#)

[INonFungibleItem](#)

[ITradableFungibleItem](#)

[ITradableItem](#)

# Enums

[ItemSubType](#)

[ItemType](#)

[LockType](#)

# Class Armor

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Armor : Equipment, INonFungibleItem, IEquippableItem, ITradableItem,
IItem, IState
```

## Inheritance

[object](#) ↳ [ItemBase](#) ↳ [ItemUsable](#) ↳ [Equipment](#) ↳ Armor

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#), [Equipment.level](#), [Equipment.Exp](#),  
[Equipment.optionCountFromCombination](#), [Equipment.IconId](#), [Equipment.ByCustomCraft](#),  
[Equipment.CraftWithRandom](#), [Equipment.HasRandomOnlyIcon](#), [Equipment.Stat](#),  
[Equipment.SetId](#), [Equipment.SpineResourcePath](#),  
[Equipment.MadeWithMimisbrunnrRecipe](#), [Equipment.UniqueStatType](#),  
[Equipment.Equipped](#), [Equipment.GetIncrementAmountOfEnhancement\(\)](#),  
[Equipment.Serialize\(\)](#), [Equipment.Equip\(\)](#), [Equipment.Unequip\(\)](#), [Equipment.LevelUpV1\(\)](#),  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#),  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#),  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#),  
[Equipment.GetOptions\(\)](#), [Equipment.Equals\(Equipment\)](#), [Equipment.Equals\(object\)](#),  
[Equipment.GetHashCode\(\)](#), [ItemUsable.ItemId](#), [ItemUsable.NonFungibleId](#),  
[ItemUsable.StatsMap](#), [ItemUsable.Skills](#), [ItemUsable.BuffSkills](#),  
[ItemUsable.RequiredBlockIndex](#), [ItemUsable.Equals\(ItemUsable\)](#),  
[ItemUsable.GetOptionCount\(\)](#), [ItemUsable.Update\(long\)](#), [ItemBase.Codec](#), [ItemBase.Id](#),  
[ItemBase.Grade](#), [ItemBase.ItemType](#), [ItemBase.ItemSubType](#), [ItemBase.ElementalType](#),  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#),  
[ItemBase.Equals\(ItemBase\)](#), [ItemBase.ToString\(\)](#), [object.Equals\(object, object\)](#) ↳,  
[object.GetType\(\)](#) ↳, [object.MemberwiseClone\(\)](#) ↳, [object.ReferenceEquals\(object, object\)](#) ↳

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#) ,  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

## Constructors

### Armor(Dictionary)

```
public Armor(Dictionary<string, object> serialized)
```

#### Parameters

serialized Dictionary<string, object>

### Armor(Row, Guid, long, bool)

```
public Armor(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool madeWithMimisbrunnrRecipe = false)
```

#### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

### Armor(SerializationInfo, StreamingContext)

```
protected Armor(SerializationInfo info, StreamingContext _)
```

#### Parameters

info [SerializationInfo](#)

  | [StreamingContext](#)

## Properties

### TradableId

`public Guid TradableId { get; }`

Property Value

[Guid](#)

# Class Aura

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Aura : Equipment, INonFungibleItem, IEquippableItem, IItem, IState
```

## Inheritance

[object](#) ← [ItemBase](#) ← [ItemUsable](#) ← [Equipment](#) ← Aura

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#), [Equipment.level](#), [Equipment.Exp](#),  
[Equipment.optionCountFromCombination](#), [Equipment.IconId](#), [Equipment.ByCustomCraft](#),  
[Equipment.CraftWithRandom](#), [Equipment.HasRandomOnlyIcon](#), [Equipment.Stat](#),  
[Equipment.SetId](#), [Equipment.SpineResourcePath](#),  
[Equipment.MadeWithMimisbrunnrRecipe](#), [Equipment.UniqueStatType](#),  
[Equipment.Equipped](#), [Equipment.GetIncrementAmountOfEnhancement\(\)](#),  
[Equipment.Serialize\(\)](#), [Equipment.Equip\(\)](#), [Equipment.Unequip\(\)](#), [Equipment.LevelUpV1\(\)](#),  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#),  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#),  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#),  
[Equipment.GetOptions\(\)](#), [Equipment.Equals\(Equipment\)](#), [Equipment.Equals\(object\)](#),  
[Equipment.GetHashCode\(\)](#), [ItemUsable.ItemId](#), [ItemUsable.NonFungibleId](#),  
[ItemUsable.StatsMap](#), [ItemUsable.Skills](#), [ItemUsable.BuffSkills](#),  
[ItemUsable.RequiredBlockIndex](#), [ItemUsable.Equals\(ItemUsable\)](#),  
[ItemUsable.GetOptionCount\(\)](#), [ItemUsable.Update\(long\)](#), [ItemBase.Codec](#), [ItemBase.Id](#),  
[ItemBase.Grade](#), [ItemBase.ItemType](#), [ItemBase.ItemSubType](#), [ItemBase.ElementalType](#),  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#),  
[ItemBase.Equals\(ItemBase\)](#), [ItemBase.ToString\(\)](#), [object.Equals\(object, object\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#),  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

# Constructors

## Aura(Dictionary)

```
public Aura(Dictionary serialized)
```

### Parameters

serialized Dictionary

## Aura(Row, Guid, long, bool)

```
public Aura(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool  
madeWithMimisbrunnrRecipe = false)
```

### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

## Aura(SerializationInfo, StreamingContext)

```
protected Aura(SerializationInfo info, StreamingContext _)
```

### Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

# Class Belt

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Belt : Equipment, INonFungibleItem, IEquippableItem, ITradableItem,
IItem, IState
```

## Inheritance

[object](#) ↵ [ItemBase](#) ↵ [ItemUsable](#) ↵ [Equipment](#) ↵ Belt

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#) , [Equipment.level](#) , [Equipment.Exp](#) ,  
[Equipment.optionCountFromCombination](#) , [Equipment.IconId](#) , [Equipment.ByCustomCraft](#) ,  
[Equipment.CraftWithRandom](#) , [Equipment.HasRandomOnlyIcon](#) , [Equipment.Stat](#) ,  
[Equipment.SetId](#) , [Equipment.SpineResourcePath](#) ,  
[Equipment.MadeWithMimisbrunnrRecipe](#) , [Equipment.UniqueStatType](#) ,  
[Equipment.Equipped](#) , [Equipment.GetIncrementAmountOfEnhancement\(\)](#) ,  
[Equipment.Serialize\(\)](#) , [Equipment.Equip\(\)](#) , [Equipment.Unequip\(\)](#) , [Equipment.LevelUpV1\(\)](#) ,  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#) ,  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#) ,  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#) ,  
[Equipment.GetOptions\(\)](#) , [Equipment.Equals\(Equipment\)](#) , [Equipment.Equals\(object\)](#) ,  
[Equipment.GetHashCode\(\)](#) , [ItemUsable.ItemId](#) , [ItemUsable.NonFungibleId](#) ,  
[ItemUsable.StatsMap](#) , [ItemUsable.Skills](#) , [ItemUsable.BuffSkills](#) ,  
[ItemUsable.RequiredBlockIndex](#) , [ItemUsable.Equals\(ItemUsable\)](#) ,  
[ItemUsable.GetOptionCount\(\)](#) , [ItemUsable.Update\(long\)](#) , [ItemBase.Codec](#) , [ItemBase.Id](#) ,  
[ItemBase.Grade](#) , [ItemBase.ItemType](#) , [ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [ItemBase.ToString\(\)](#) , [object.Equals\(object, object\)](#) ↵ ,  
[object.GetType\(\)](#) ↵ , [object.MemberwiseClone\(\)](#) ↵ , [object.ReferenceEquals\(object, object\)](#) ↵

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#) ,  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

## Constructors

### Belt(Dictionary)

```
public Belt(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Belt(Row, Guid, long, bool)

```
public Belt(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool madeWithMimisbrunnrRecipe = false)
```

#### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

### Belt(SerializationInfo, StreamingContext)

```
protected Belt(SerializationInfo info, StreamingContext _)
```

#### Parameters

info [SerializationInfo](#)

  | [StreamingContext](#)

## Properties

### TradableId

`public Guid TradableId { get; }`

Property Value

[Guid](#)

# Class Consumable

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Consumable : ItemUsable, INonFungibleItem, ITradableItem, IItem, IState
```

## Inheritance

[object](#) ← [ItemBase](#) ← [ItemUsable](#) ← Consumable

## Implements

[INonFungibleItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[ItemUsable.ItemId](#), [ItemUsable.NonFungibleId](#), [ItemUsable.StatsMap](#), [ItemUsable.Skills](#),  
[ItemUsable.BuffSkills](#), [ItemUsable.RequiredBlockIndex](#), [ItemUsable.Equals\(ItemUsable\)](#),  
[ItemUsable.Equals\(object\)](#), [ItemUsable.GetHashCode\(\)](#), [ItemUsable.GetOptionCount\(\)](#),  
[ItemUsable.Update\(long\)](#), [ItemBase.Codec](#), [ItemBase.Id](#), [ItemBase.Grade](#),  
[ItemBase.ItemType](#), [ItemBase.ItemSubType](#), [ItemBase.ElementalType](#),  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#),  
[ItemBase.Equals\(ItemBase\)](#), [ItemBase.ToString\(\)](#), [object.Equals\(object, object\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Constructors

### Consumable(Dictionary)

```
public Consumable(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### Consumable(Row, Guid, long)

```
public Consumable(ConsumableItemSheet.Row data, Guid id, long requiredBlockIndex)
```

## Parameters

data [ConsumableItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Consumable(SerializationInfo, StreamingContext)

```
protected Consumable(SerializationInfo info, StreamingContext _)
```

## Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

## Properties

### MainStat

```
public StatType MainStat { get; }
```

#### Property Value

[StatType](#)

### Stats

```
public List<DecimalStat> Stats { get; }
```

#### Property Value

## TradableId

```
public Guid TradableId { get; }
```

Property Value

[Guid](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class Costume

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Costume : ItemBase, INonFungibleItem, IEquippableItem, ITradableItem,
IItem, IState
```

## Inheritance

[object](#) ← [ItemBase](#) ← Costume

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[ItemBase.Codec](#) , [ItemBase.Id](#) , [ItemBase.Grade](#) , [ItemBase.ItemType](#) ,  
[ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [ItemBase.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### Costume(Dictionary)

```
public Costume(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### Costume(Row, Guid)

```
public Costume(CostumeItemSheet.Row data, Guid itemId)
```

## Parameters

data [CostumeItemSheet.Row](#)

itemId [Guid](#)

## Costume(SerializationInfo, StreamingContext)

```
protected Costume(SerializationInfo info, StreamingContext _)
```

## Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

## Fields

### equipped

```
public bool equipped
```

## Field Value

[bool](#)

## Properties

### Equipped

```
public bool Equipped { get; }
```

## Property Value

[bool](#)

## ItemId

```
public Guid ItemId { get; }
```

Property Value

[Guid](#) ↗

## NonFungibleId

```
public Guid NonFungibleId { get; }
```

Property Value

[Guid](#) ↗

## RequiredBlockIndex

```
public long RequiredBlockIndex { get; set; }
```

Property Value

[long](#) ↗

## SpineResourcePath

```
public string SpineResourcePath { get; }
```

Property Value

[string](#) ↗

## TradableId

```
public Guid TradableId { get; }
```

Property Value

[Guid](#)

## Methods

### Equals(Costume)

```
protected bool Equals(Costume other)
```

Parameters

other [Costume](#)

Returns

[bool](#)

### Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## Equip()

```
public void Equip()
```

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int ↗](#)

A hash code for the current object.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Unequip()

```
public void Unequip()
```

## Update(long)

```
public void Update(long blockIndex)
```

Parameters

blockIndex [long](#)

# Class Equipment

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Equipment : ItemUsable, INonFungibleItem, IEquippableItem,
IItem, IState
```

## Inheritance

[object](#) ↵ [ItemBase](#) ↵ [ItemUsable](#) ↵ Equipment

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [IItem](#), [IState](#)

## Derived

[Armor](#), [Aura](#), [Belt](#), [Grimoire](#), [Necklace](#), [Ring](#), [Weapon](#)

## Inherited Members

[ItemUsable.ItemId](#) , [ItemUsable.NonFungibleId](#) , [ItemUsable.StatsMap](#) , [ItemUsable.Skills](#) ,  
[ItemUsable.BuffSkills](#) , [ItemUsable.RequiredBlockIndex](#) , [ItemUsable.Equals\(ItemUsable\)](#) ,  
[ItemUsable.GetOptionCount\(\)](#) , [ItemUsable.Update\(long\)](#) , [ItemBase.Codec](#) , [ItemBase.Id](#) ,  
[ItemBase.Grade](#) , [ItemBase.ItemType](#) , [ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [ItemBase.ToString\(\)](#) , [object.Equals\(object, object\)](#) ↵ ,  
[object.GetType\(\)](#) ↵ , [object.MemberwiseClone\(\)](#) ↵ , [object.ReferenceEquals\(object, object\)](#) ↵

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#) ,  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

## Constructors

### Equipment(Dictionary)

```
public Equipment(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Equipment(Row, Guid, long, bool, int)

```
public Equipment(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool  
madeWithMimisbrunnrRecipe = false, int iconId = 0)
```

## Parameters

**data** [EquipmentItemSheet.Row](#)

**id** [Guid](#)

**requiredBlockIndex** [long](#)

**madeWithMimisbrunnrRecipe** [bool](#)

**iconId** [int](#)

## Equipment(SerializationInfo, StreamingContext)

```
protected Equipment(SerializationInfo info, StreamingContext _)
```

## Parameters

**info** [SerializationInfo](#)

**\_** [StreamingContext](#)

## Fields

### ByCustomCraft

```
public bool ByCustomCraft
```

Field Value

[bool](#) ↗

## CraftWithRandom

```
public bool CraftWithRandom
```

Field Value

[bool](#) ↗

## Exp

```
public long Exp
```

Field Value

[long](#) ↗

## HasRandomOnlyIcon

```
public bool HasRandomOnlyIcon
```

Field Value

[bool](#) ↗

## IconId

```
public int IconId
```

Field Value

[int](#)

## equipped

`public bool equipped`

Field Value

[bool](#)

## level

`public int level`

Field Value

[int](#)

## optionCountFromCombination

`public int optionCountFromCombination`

Field Value

[int](#)

## Properties

### Equipped

`public bool Equipped { get; }`

Property Value

[bool](#)

## MadeWithMimisbrunnrRecipe

`public bool MadeWithMimisbrunnrRecipe { get; }`

Property Value

[bool](#)

## SetId

`public int SetId { get; }`

Property Value

[int](#)

## SpineResourcePath

`public string SpineResourcePath { get; }`

Property Value

[string](#)

## Stat

`public DecimalStat Stat { get; }`

Property Value

[DecimalStat](#)

# UniqueStatType

```
public StatType UniqueStatType { get; }
```

Property Value

[StatType](#)

## Methods

### Equals(Equipment)

```
protected bool Equals(Equipment other)
```

Parameters

other [Equipment](#)

Returns

[bool](#)

### Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## Equip()

```
public void Equip()
```

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetIncrementAmountOfEnhancement()

```
public decimal GetIncrementAmountOfEnhancement()
```

Returns

[decimal](#)

## GetOptions()

```
public List<object> GetOptions()
```

Returns

[List](#)<[object](#)>

## GetRealExp(EquipmentItemSheet, EnhancementCostSheetV3)

```
public long GetRealExp(EquipmentItemSheet itemSheet, EnhancementCostSheetV3  
costSheet)
```

### Parameters

itemSheet [EquipmentItemSheet](#)

costSheet [EnhancementCostSheetV3](#)

### Returns

[long](#)

## LevelUp(IRandom, Row, bool)

```
[Obsolete("Since ItemEnhancement12, Use `SetLevel` instead.")]  
public void LevelUp(IRandom random, EnhancementCostSheetV2.Row row,  
bool isGreatSuccess)
```

### Parameters

random [IRandom](#)

row [EnhancementCostSheetV2.Row](#)

isGreatSuccess [bool](#)

## LevelUpV1()

```
[Obsolete("Use LevelUp")]  
public void LevelUpV1()
```

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SetLevel(IRandom, int, EnhancementCostSheetV3)

```
public void SetLevel(IRandom random, int targetLevel, EnhancementCostSheetV3 sheet)
```

Parameters

random IRandom

targetLevel [int](#)

sheet [EnhancementCostSheetV3](#)

## Unequip()

```
public void Unequip()
```

# Class Grimoire

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Grimoire : Equipment, INonFungibleItem, IEquippableItem, IItem, IState
```

## Inheritance

[object](#) ← [ItemBase](#) ← [ItemUsable](#) ← [Equipment](#) ← Grimoire

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#), [Equipment.level](#), [Equipment.Exp](#),  
[Equipment.optionCountFromCombination](#), [Equipment.iconId](#), [Equipment.ByCustomCraft](#),  
[Equipment.CraftWithRandom](#), [Equipment.HasRandomOnlyIcon](#), [Equipment.Stat](#),  
[Equipment.SetId](#), [Equipment.SpineResourcePath](#),  
[Equipment.MadeWithMimisbrunnrRecipe](#), [Equipment.UniqueStatType](#),  
[Equipment.Equipped](#), [Equipment.GetIncrementAmountOfEnhancement\(\)](#),  
[Equipment.Serialize\(\)](#), [Equipment.Equip\(\)](#), [Equipment.Unequip\(\)](#), [Equipment.LevelUpV1\(\)](#),  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#),  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#),  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#),  
[Equipment.GetOptions\(\)](#), [Equipment.Equals\(Equipment\)](#), [Equipment.Equals\(object\)](#),  
[Equipment.GetHashCode\(\)](#), [ItemUsable.ItemId](#), [ItemUsable.NonFungibleId](#),  
[ItemUsable.StatsMap](#), [ItemUsable.Skills](#), [ItemUsable.BuffSkills](#),  
[ItemUsable.RequiredBlockIndex](#), [ItemUsable.Equals\(ItemUsable\)](#),  
[ItemUsable.GetOptionCount\(\)](#), [ItemUsable.Update\(long\)](#), [ItemBase.Codec](#), [ItemBase.Id](#),  
[ItemBase.Grade](#), [ItemBase.ItemType](#), [ItemBase.ItemSubType](#), [ItemBase.ElementalType](#),  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#),  
[ItemBase.Equals\(ItemBase\)](#), [ItemBase.ToString\(\)](#), [object.Equals\(object, object\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#),  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

# Constructors

## Grimoire(Dictionary)

```
public Grimoire(Dictionary serialized)
```

### Parameters

serialized Dictionary

## Grimoire(Row, Guid, long, bool)

```
public Grimoire(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool  
madeWithMimisbrunnrRecipe = false)
```

### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

## Grimoire(SerializationInfo, StreamingContext)

```
protected Grimoire(SerializationInfo info, StreamingContext _)
```

### Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

# Interface IEquipableItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface IEquipableItem : IItem, IState
```

## Inherited Members

[IItem.ItemType](#) , [IItem.ItemSubType](#) , [IState.Serialize\(\)](#).

## Properties

### Equipped

```
bool Equipped { get; }
```

Property Value

[bool](#) ↗

## Methods

### Equip()

```
void Equip()
```

### Unequip()

```
void Unequip()
```

# Interface IFungibleItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface IFungibleItem : IItem, IState
```

## Inherited Members

[IItem.ItemType](#) , [IItem.ItemSubType](#) , [IState.Serialize\(\)](#).

## Properties

### FungibleId

```
HashDigest<SHA256> FungibleId { get; }
```

Property Value

HashDigest<[SHA256](#)>

# Interface IItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface IItem : IState
```

## Inherited Members

[IState.Serialize\(\)](#)

## Properties

### ItemSubType

```
ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

### ItemType

```
ItemType ItemType { get; }
```

Property Value

[ItemType](#)

# Interface ILock

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface ILock
```

## Properties

### Type

```
LockType Type { get; }
```

Property Value

[LockType](#)

## Methods

### Serialize()

```
IValue Serialize()
```

Returns

IValue

# Interface INonFungibleItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface INonFungibleItem : IItem, IState
```

## Inherited Members

[IItem.ItemType](#) , [IItem.ItemSubType](#) , [IState.Serialize\(\)](#).

## Properties

### NonFungibleId

```
Guid NonFungibleId { get; }
```

Property Value

[Guid](#)

### RequiredBlockIndex

```
long RequiredBlockIndex { get; set; }
```

Property Value

[long](#)

# Interface ITradableFungibleItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface ITradableFungibleItem : ITradableItem, IFungibleItem, IItem,  
IState, ICloneable
```

## Inherited Members

[ITradableItem.TradableId](#) , [ITradableItem.RequiredBlockIndex](#) , [IFungibleItem.FungibleId](#) ,  
[IItem.ItemType](#) , [IItem.ItemSubType](#) , [IState.Serialize\(\)](#) , [ICloneable.Clone\(\)](#) ↗

# Interface ITradableItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public interface ITradableItem : IItem, IState
```

## Inherited Members

[IItem.ItemType](#) , [IItem.ItemSubType](#) , [IState.Serialize\(\)](#).

## Properties

### RequiredBlockIndex

```
long RequiredBlockIndex { get; set; }
```

Property Value

[long](#) ↗

### TradableId

```
Guid TradableId { get; }
```

Property Value

[Guid](#) ↗

# Class Inventory

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Inventory : IState, ICloneable
```

## Inheritance

[object](#) ← Inventory

## Implements

[IState](#), [ICloneable](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[InventoryExtensions.GetEquippedFullCostumeOrArmworld\(Inventory\)](#) ,  
[InventoryExtensions.UseActionPoint\(Inventory, long, int, bool, MaterialItemSheet, long\)](#)

## Constructors

### Inventory()

```
public Inventory()
```

### Inventory(List)

```
public Inventory(List serialized)
```

## Parameters

serialized List

# Properties

## Consumables

```
public IEnumerable<Consumable> Consumables { get; }
```

Property Value

[IEnumerable](#)<[Consumable](#)>

## Costumes

```
public IEnumerable<Costume> Costumes { get; }
```

Property Value

[IEnumerable](#)<[Costume](#)>

## Equipments

```
public IEnumerable<Equipment> Equipments { get; }
```

Property Value

[IEnumerable](#)<[Equipment](#)>

## Items

```
public IReadOnlyList<Inventory.Item> Items { get; }
```

Property Value

[IReadOnlyList](#)<[Inventory](#).[Item](#)>

# Materials

```
public IEnumerable<Material> Materials { get; }
```

Property Value

[IEnumerable](#)<Material>

## Methods

### AddFungibleItem(ItemBase, int, ILock)

```
public Inventory.Item AddFungibleItem(ItemBase itemBase, int count = 1, ILock iLock = null)
```

Parameters

itemBase [ItemBase](#)

count [int](#)

iLock [ILock](#)

Returns

[Inventory.Item](#)

### AddFungibleItem2(ItemBase, int, ILock)

```
[Obsolete("Use AddFungibleItem")]
public Inventory.Item AddFungibleItem2(ItemBase itemBase, int count = 1, ILock iLock = null)
```

Parameters

itemBase [ItemBase](#)

count [int](#)

iLock [ILock](#)

Returns

[Inventory.Item](#)

## AddItem(ItemBase, int, ILock)

```
public KeyValuePair<int, int> AddItem(ItemBase itemBase, int count = 1, ILock iLock  
= null)
```

Parameters

itemBase [ItemBase](#)

count [int](#)

iLock [ILock](#)

Returns

[KeyValuePair](#)<[int](#), [int](#)>

## AddItem2(ItemBase, int, ILock)

```
[Obsolete("Use AddItem")]  
public KeyValuePair<int, int> AddItem2(ItemBase itemBase, int count = 1, ILock iLock  
= null)
```

Parameters

itemBase [ItemBase](#)

count [int](#)

iLock [ILock](#)

Returns

[KeyValuePair](#)<[int](#), [int](#)>

## AddNonFungibleItem(ItemBase, ILock)

`public Inventory.Item AddNonFungibleItem(ItemBase itemBase, ILock iLock = null)`

Parameters

`itemBase` [ItemBase](#)

`iLock` [ILock](#)

Returns

[Inventory.Item](#)

## AddTradableFungibleItem(ITradableFungibleItem, int)

Add tradable fungible item. If there is same item which is not locked, add count to it. The same item is item which has same item sheet id, fungible id, tradable id and same required block index. If there is no same item, add new item.

`public Inventory.Item AddTradableFungibleItem(ITradableFungibleItem tradableFungibleItem, int count)`

Parameters

`tradableFungibleItem` [ITradableFungibleItem](#)

`count` [int](#)

Returns

[Inventory.Item](#)

Exceptions

[ArgumentNullException](#)

[ArgumentOutOfRangeException](#)

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## Equals(Inventory)

```
protected bool Equals(Inventory other)
```

Parameters

other [Inventory](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## FilterConsumables(int, long)

```
public List<Consumable> FilterConsumables(int id, long blockIndex)
```

Parameters

[id](#) [int](#)

[blockIndex](#) [long](#)

Returns

[List](#)<[Consumable](#)>

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## HasFungibleItem(HashDigest<SHA256>, long, int)

```
public bool HasFungibleItem(HashDigest<SHA256> fungibleId, long blockIndex, int count = 1)
```

## Parameters

fungibleId HashDigest<[SHA256](#)>

blockIndex [long](#)

count [int](#)

## Returns

[bool](#)

## HasItem(int, int)

```
public bool HasItem(int rowId, int count = 1)
```

## Parameters

rowId [int](#)

count [int](#)

## Returns

[bool](#)

## HasNonFungibleItem(Guid)

```
public bool HasNonFungibleItem(Guid nonFungibleId)
```

## Parameters

nonFungibleId [Guid](#)

Returns

[bool](#)

## HasTradableFungibleItem(HashDigest<SHA256>, long?, long?, int)

```
public bool HasTradableFungibleItem(HashDigest<SHA256> fungibleId, long?  
requiredBlockIndex, long? blockIndex, int count)
```

Parameters

fungibleId HashDigest<[SHA256](#)>

requiredBlockIndex [long](#)?

blockIndex [long](#)?

count [int](#)

Returns

[bool](#)

## HasTradableItem(Guid, long, int)

```
public bool HasTradableItem(Guid tradableId, long blockIndex, int count)
```

Parameters

tradableId [Guid](#)

blockIndex [long](#)

count [int](#)

Returns

[bool](#)

## IsMaterialRemovable(Item, int)

Checks if the given item is non-tradable material and can be removed from the inventory.

```
public static bool IsMaterialRemovable(Inventory.Item item, int id)
```

### Parameters

`item` [Inventory.Item](#)

The item to check for removal.

`id` [int](#)

The ID of the material to remove.

### Returns

[bool](#)

Returns a boolean indicating whether the item is removable.

## IsMaterialRemovable(Item, int, long)

Checks if the given item is material and can be removed from the inventory.

```
public static bool IsMaterialRemovable(Inventory.Item item, int id, long blockIndex)
```

### Parameters

`item` [Inventory.Item](#)

The item to check for removal.

`id` [int](#)

The ID of the material to remove.

`blockIndex` [long](#)

The block index.

Returns

[bool](#)

Returns a boolean indicating whether the item is removable.

## LockByReferringToDigestList(OrderDigestListState, Guid, long)

```
public void LockByReferringToDigestList(OrderDigestListState digestList, Guid  
tradableId, long blockIndex)
```

Parameters

digestList [OrderDigestListState](#)

tradableId [Guid](#)

blockIndex [long](#)

## ReconfigureFungibleItem(OrderDigestListState, Guid)

```
public void ReconfigureFungibleItem(OrderDigestListState digestList,  
Guid tradableId)
```

Parameters

digestList [OrderDigestListState](#)

tradableId [Guid](#)

## RemoveConsumable(int, long, int)

```
public bool RemoveConsumable(int id, long blockIndex, int count = 1)
```

Parameters

`id int`

`blockIndex long`

`count int`

Returns

`bool`

**RemoveFungibleItem(HashDigest<SHA256>, long, int, bool)**

```
public bool RemoveFungibleItem(HashDigest<SHA256> fungibleId, long blockIndex, int  
count = 1, bool onlyTradableItem = false)
```

Parameters

`fungibleId HashDigest<SHA256>`

`blockIndex long`

`count int`

`onlyTradableItem bool`

Returns

`bool`

**RemoveFungibleItem(IFungibleItem, long, int, bool)**

```
public bool RemoveFungibleItem(IFungibleItem fungibleItem, long blockIndex, int  
count = 1, bool onlyTradableItem = false)
```

Parameters

`fungibleItem` [IFungibleItem](#)

`blockIndex` [long](#)

`count` [int](#)

`onlyTradableItem` [bool](#)

Returns

[bool](#)

## RemoveFungibleItem2(HashDigest<SHA256>, int, bool)

```
[Obsolete("Use RemoveFungibleItem")]
public bool RemoveFungibleItem2(HashDigest<SHA256> fungibleId, int count = 1, bool
onlyTradableItem = false)
```

Parameters

`fungibleId` HashDigest<[SHA256](#)>

`count` [int](#)

`onlyTradableItem` [bool](#)

Returns

[bool](#)

## RemoveFungibleItem2(IFungibleItem, int, bool)

```
[Obsolete("Use RemoveFungibleItem")]
public bool RemoveFungibleItem2(IFungibleItem fungibleItem, int count = 1, bool
onlyTradableItem = false)
```

Parameters

`fungibleItem` [IFungibleItem](#)

`count` [int](#)

`onlyTradableItem` [bool](#)

Returns

[bool](#)

## RemoveItem(Item)

```
public void RemoveItem(Inventory.Item item)
```

Parameters

`item` [Inventory.Item](#)

## RemoveMaterial(int, long, int)

Remove a material from the inventory. contains tradable materials.

```
public bool RemoveMaterial(int id, long blockIndex, int count = 1)
```

Parameters

`id` [int](#)

The ID of the material item to remove.

`blockIndex` [long](#)

The block index.

`count` [int](#)

The number of materials to remove. Default value is 1.

Returns

[bool](#)

True if the material is successfully removed, false otherwise.

## RemoveNonFungibleItem(INonFungibleItem)

```
public bool RemoveNonFungibleItem(INonFungibleItem nonFungibleItem)
```

Parameters

nonFungibleItem [INonFungibleItem](#)

Returns

[bool](#)

## RemoveNonFungibleItem(Guid)

```
public bool RemoveNonFungibleItem(Guid nonFungibleId)
```

Parameters

nonFungibleId [Guid](#)

Returns

[bool](#)

## RemoveNonFungibleItem2(Guid)

```
[Obsolete("Use RemoveNonFungibleItem(Guid nonFungibleId)")]
public bool RemoveNonFungibleItem2(Guid nonFungibleId)
```

Parameters

nonFungibleId [Guid](#)

Returns

[bool](#)

## RemoveNonTradableMaterial(int, int)

Remove a non-tradable material from the inventory.

```
public bool RemoveNonTradableMaterial(int id, int count = 1)
```

Parameters

[id](#) [int](#)

The ID of the material item to remove.

[count](#) [int](#)

The number of materials to remove. Default value is 1.

Returns

[bool](#)

True if the material is successfully removed, false otherwise.

## RemoveTradableFungibleItem(HashDigest<SHA256>, long?, long?, int)

Remove tradable item from inventory. Use the requiredBlockIndex when you want to remove the item that has same requiredBlockIndex. Use the blockIndex when you want to remove the item that has requiredBlockIndex less than or equal to blockIndex. Do not use both requiredBlockIndex and blockIndex at the same time.

```
public bool RemoveTradableFungibleItem(HashDigest<SHA256> fungibleId, long?  
requiredBlockIndex, long? blockIndex, int count)
```

Parameters

`fungibleId HashDigest<SHA256>`

`requiredBlockIndex long?`

`blockIndex long?`

`count int`

Returns

`bool`

Exceptions

[ArgumentNullException](#)

## RemoveTradableItem(Guid, long, int)

`public bool RemoveTradableItem(Guid tradableId, long blockIndex, int count = 1)`

Parameters

`tradableId Guid`

`blockIndex long`

`count int`

Returns

`bool`

## RemoveTradableItemV1(ITradableItem, int)

`[Obsolete("Use RemoveTradableItem())")]`

`public bool RemoveTradableItemV1(ITradableItem tradableItem, int count = 1)`

Parameters

tradableItem [ITradableItem](#)

count [int](#)

Returns

[bool](#)

## RemoveTradableItemV1(Guid, long, int)

```
[Obsolete("Use RemoveTradableItem())")]
public bool RemoveTradableItemV1(Guid tradableId, long blockIndex, int count = 1)
```

Parameters

tradableId [Guid](#)

blockIndex [long](#)

count [int](#)

Returns

[bool](#)

## RemoveTradableMaterial(int, long, int)

Remove a tradable material from the inventory.

```
public bool RemoveTradableMaterial(int materialId, long blockIndex, int count = 1)
```

Parameters

materialId [int](#)

The ID of the material item to remove.

blockIndex [long](#)

The block index.

`count` [int](#)

The number of materials to remove. Default value is 1.

Returns

[bool](#)

True if the material is successfully removed, false otherwise.

## SellItem(Guid, long, int)

```
public ITradableItem SellItem(Guid tradableId, long blockIndex, int count)
```

Parameters

`tradableId` [Guid](#)

`blockIndex` [long](#)

`count` [int](#)

Returns

[ITradableItem](#)

## Serialize()

```
public IValue Serialize()
```

Returns

[IValue](#)

## TryGetCostume(int, out Costume)

```
[Obsolete("Use public bool TryGetNonFungibleItem<T>(Guid itemId, out  
T outNonFungibleItem)")]
public bool TryGetCostume(int rowId, out Costume outCostume)
```

## Parameters

rowId [int](#)

outCostume [Costume](#)

## Returns

[bool](#)

## TryGetFungibleItems(HashDigest<SHA256>, out List<Item>)

```
public bool TryGetFungibleItems(HashDigest<SHA256> fungibleId, out  
List<Inventory.Item> outItems)
```

## Parameters

fungibleId HashDigest<[SHA256](#)>

outItems [List](#)<[Inventory.Item](#)>

## Returns

[bool](#)

## TryGetItem(int, out Item)

```
public bool TryGetItem(int rowId, out Inventory.Item outItem)
```

## Parameters

rowId [int](#)

`outItem` [Inventory.Item](#)

Returns

`bool` ↗

## TryGetLockedItem(ILock, out Item)

```
public bool TryGetLockedItem(ILock iLock, out Inventory.Item outItem)
```

Parameters

`iLock` [ILock](#)

`outItem` [Inventory.Item](#)

Returns

`bool` ↗

## TryGetNonFungibleItem(Guid, out Item)

```
public bool TryGetNonFungibleItem(Guid nonFungibleId, out Inventory.Item outItem)
```

Parameters

`nonFungibleId` [Guid](#) ↗

`outItem` [Inventory.Item](#)

Returns

`bool` ↗

## TryGetNonFungibleItem<T>(Guid, out T, long)

Tries to get a non-fungible item from the inventory with the specified item ID.

```
public bool TryGetNonFungibleItem<T>(Guid itemId, out T outNonFungibleItem, long  
blockIndex = 9223372036854775807) where T : INonFungibleItem
```

## Parameters

**itemId** [Guid](#)

The ID of the non-fungible item to retrieve.

**outNonFungibleItem** T

When this method returns, contains the non-fungible item with the specified ID, if found; otherwise, the default value for the type.

**blockIndex** [long](#)

The block index for the non-fungible item to be considered valid. Defaults to [MaxValue](#).

## Returns

[bool](#)

**true** if a non-fungible item with the specified ID is found in the inventory; otherwise, **false**.

## Type Parameters

T

The type of non-fungible item to retrieve.

## TryGetNonFungibleItem<T>(T, out T)

```
public bool TryGetNonFungibleItem<T>(T nonFungibleItem, out T outNonFungibleItem)  
where T : INonFungibleItem
```

## Parameters

**nonFungibleItem** T

**outNonFungibleItem** T

Returns

[bool](#)

Type Parameters

T

**TryGetTradableFungibleItems(HashDigest<SHA256>, long?, long?, out IEnumerable<Item>)**

Get ITradableFungibleItem from the inventory. This method check the item is not locked, and the item's required block index is less than or equal to the given block index. Do not use both requiredBlockIndex and blockIndex at the same time.

```
public bool TryGetTradableFungibleItems(HashDigest<SHA256> fungibleId, long?  
requiredBlockIndex, long? blockIndex, out IEnumerable<Inventory.Item> outItems)
```

Parameters

fungibleId HashDigest<[SHA256](#)>

requiredBlockIndex [long](#)?

blockIndex [long](#)?

outItems [IEnumerable](#)<[Inventory.Item](#)>

Returns

[bool](#)

Exceptions

[ArgumentNullException](#)

**TryGetTradableItem(Guid, long, int, out Item)**

```
public bool TryGetTradableItem(Guid tradeId, long blockIndex, int count, out Inventory.Item outItem)
```

## Parameters

tradeId [Guid](#)

blockIndex [long](#)

count [int](#)

outItem [Inventory.Item](#)

## Returns

[bool](#)

## TryGetTradableItems(Guid, long, int, out List<Item>)

```
public bool TryGetTradableItems(Guid tradeId, long blockIndex, int count, out List<Inventory.Item> outItem)
```

## Parameters

tradeId [Guid](#)

blockIndex [long](#)

count [int](#)

outItem [List](#)<[Inventory.Item](#)>

## Returns

[bool](#)

## UnlockInvalidSlot(OrderDigestListState, Address, Address)

```
public void UnlockInvalidSlot(OrderDigestListState digestListState, Address agentAddress, Address avatarAddress)
```

## Parameters

digestListState [OrderDigestListState](#)

agentAddress Address

avatarAddress Address

## UpdateTradableItem(Guid, long, int, long)

```
public ITradableItem UpdateTradableItem(Guid tradableId, long blockIndex, int count, long requiredBlockIndex)
```

## Parameters

tradableId [Guid](#)

blockIndex [long](#)

count [int](#)

requiredBlockIndex [long](#)

## Returns

[ITradableItem](#)

# Class Inventory.Item

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Inventory.Item : IState, IComparer<Inventory.Item>,
IComparable<Inventory.Item>
```

## Inheritance

[object](#) ← Inventory.Item

## Implements

[IState](#), [IComparer](#)<[Inventory.Item](#)>, [IComparable](#)<[Inventory.Item](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Item(Dictionary)

```
public Item(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Item(ItemBase, int)

```
public Item(ItemBase itemBase, int count = 1)
```

#### Parameters

itemBase [ItemBase](#)

count [int](#)

## Fields

### Lock

`public ILock Lock`

#### Field Value

[ILock](#)

### count

`public int count`

#### Field Value

[int](#)

### item

`public ItemBase item`

#### Field Value

[ItemBase](#)

## Properties

### Locked

`public bool Locked { get; }`

## Property Value

[bool](#) ↗

## Methods

### Compare(Item, Item)

Compares two objects and returns a value indicating whether one is less than, equal to, or greater than the other.

```
public int Compare(Inventory.Item x, Inventory.Item y)
```

#### Parameters

x [Inventory.Item](#)

The first object to compare.

y [Inventory.Item](#)

The second object to compare.

#### Returns

[int](#) ↗

A signed integer that indicates the relative values of x and y, as shown in the following table.

Value	Meaning
<b>Less than zero</b>	x is less than y.
<b>Zero</b>	x equals y.
<b>Greater than zero</b>	x is greater than y.

### CompareTo(Item)

Compares the current instance with another object of the same type and returns an integer that indicates whether the current instance precedes, follows, or occurs in the same position in the sort order as the other object.

```
public int CompareTo(Inventory.Item other)
```

## Parameters

**other** [Inventory.Item](#)

An object to compare with this instance.

## Returns

[int](#)

A value that indicates the relative order of the objects being compared. The return value has these meanings:

Value	Meaning
<b>Less than zero</b>	This instance precedes <b>other</b> in the sort order.
<b>Zero</b>	This instance occurs in the same position in the sort order as <b>other</b> .
<b>Greater than zero</b>	This instance follows <b>other</b> in the sort order.

## Equals(Item)

```
protected bool Equals(Inventory.Item other)
```

## Parameters

**other** [Inventory.Item](#)

## Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

[obj](#) [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## LockUp(ILock)

```
public void LockUp(ILock iLock)
```

Parameters

iLock [ILock](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Unlock()

```
public void Unlock()
```

# Class ItemBase

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ItemBase : IItem, IState
```

## Inheritance

[object](#) ← ItemBase

## Implements

[IItem](#), [IState](#)

## Derived

[Costume](#), [ItemUsable](#), [Material](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### ItemBase(Dictionary)

```
protected ItemBase(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### ItemBase(ItemBase)

```
protected ItemBase(ItemBase other)
```

#### Parameters

other [ItemBase](#)

## ItemBase(Row)

`protected ItemBase(ItemSheet.Row data)`

### Parameters

`data` [ItemSheet.Row](#)

## ItemBase(SerializationInfo, StreamingContext)

`protected ItemBase(SerializationInfo info, StreamingContext _)`

### Parameters

`info` [SerializationInfo](#)

`_` [StreamingContext](#)

## Fields

### Codec

`protected static readonly Codec Codec`

### Field Value

Codec

## Properties

### ElementalType

```
public ElementalType ElementalType { get; set; }
```

Property Value

[ElementalType](#)

## Grade

```
public int Grade { get; }
```

Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## ItemSubType

```
public ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

## ItemType

```
public ItemType ItemType { get; }
```

Property Value

[ItemType](#)

## Methods

### Equals(ItemBase)

```
protected bool Equals(ItemBase other)
```

Parameters

other [ItemBase](#)

Returns

[bool](#)

### Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetObjectData(SerializationInfo, StreamingContext)

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Serialize()

```
public virtual IValue Serialize()
```

Returns

IValue

## ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

## Returns

[string](#)

A string that represents the current object.

# Class ItemFactory

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public static class ItemFactory
```

## Inheritance

[object](#) ← ItemFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### CreateCostume(Row, Guid)

```
public static Costume CreateCostume(CostumeItemSheet.Row row, Guid itemId)
```

#### Parameters

row [CostumeItemSheet.Row](#)

itemId [Guid](#)

#### Returns

[Costume](#)

### CreateItem(Row, IRandom)

```
public static ItemBase CreateItem(ItemSheet.Row row, IRandom random)
```

## Parameters

row [ItemSheet.Row](#)

random IRandom

## Returns

[ItemBase](#)

## CreateItemUsable(Row, Guid, long, int, bool)

```
public static ItemUsable CreateItemUsable(ItemSheet.Row itemRow, Guid id, long requiredBlockIndex, int level = 0, bool madeWithMimisbrunnrRecipe = false)
```

## Parameters

itemRow [ItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

level [int](#)

madeWithMimisbrunnrRecipe [bool](#)

## Returns

[ItemUsable](#)

## CreateItemUsableV2(Row, Guid, long, int, IRandom, Row, bool, bool)

```
public static ItemUsable CreateItemUsableV2(ItemSheet.Row itemRow, Guid id, long requiredBlockIndex, int level, IRandom random, EnhancementCostSheetV2.Row row, bool isGreatSuccess, bool madeWithMimisbrunnrRecipe = false)
```

## Parameters

itemRow [ItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

level [int](#)

random [IRandom](#)

row [EnhancementCostSheetV2.Row](#)

isGreatSuccess [bool](#)

madewithMimisbrunnrRecipe [bool](#)

Returns

[ItemUsable](#)

## CreateMaterial(MaterialItemSheet, int)

public static Material [CreateMaterial](#)(MaterialItemSheet sheet, [int](#) itemId)

Parameters

sheet [MaterialItemSheet](#)

itemId [int](#)

Returns

[Material](#)

## CreateMaterial(Row)

public static Material [CreateMaterial](#)(MaterialItemSheet.Row row)

Parameters

`row` [MaterialItemSheet.Row](#)

Returns

[Material](#)

## CreateTradableMaterial(Row)

```
public static TradableMaterial CreateTradableMaterial(MaterialItemSheet.Row row)
```

Parameters

`row` [MaterialItemSheet.Row](#)

Returns

[TradableMaterial](#)

## Deserialize(Dictionary)

```
public static ItemBase Deserialize(Dictionary serialized)
```

Parameters

`serialized` Dictionary

Returns

[ItemBase](#)

## SelectIconId(int, bool, Row, int, CustomEquipmentCraftIconSheet, IRandom)

```
public static (int, bool) SelectIconId(int iconId, bool isRandom,  
EquipmentItemSheet.Row equipmentRow, int relationship, CustomEquipmentCraftIconSheet
```

```
iconSheet, IRandom random)
```

## Parameters

iconId [int](#)

isRandom [bool](#)

equipmentRow [EquipmentItemSheet.Row](#)

relationship [int](#)

iconSheet [CustomEquipmentCraftIconSheet](#)

random IRandom

## Returns

([int](#), [bool](#))

## SelectOption(ItemSubType, CustomEquipmentCraftOptionSheet, IRandom)

```
public static CustomEquipmentCraftOptionSheet.Row SelectOption(ItemSubType  
itemSubType, CustomEquipmentCraftOptionSheet optionSheet, IRandom random)
```

## Parameters

itemSubType [ItemSubType](#)

optionSheet [CustomEquipmentCraftOptionSheet](#)

random IRandom

## Returns

[CustomEquipmentCraftOptionSheet.Row](#)

# Enum ItemSubType

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public enum ItemSubType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

ApStone = 16

Armor = 7

Aura = 19

Belt = 8

[Obsolete("ItemSubType.Chest has never been used outside the MaterialItemSheet. And we won't use it in the future until we have a specific reason.")] Chest = 17

Circle = 22

EarCostume = 3

EquipmentMaterial = 11

EyeCostume = 4

Food = 0

FoodMaterial = 12

FullCostume = 1

Grimoire = 20

HairCostume = 2

Hourglass = 15

MonsterPart = 13

Necklace = 9

NormalMaterial = 14

Ring = 10

Scroll = 21

TailCostume = 5

Title = 18

Weapon = 6

# Class ItemSubTypeComparer

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public class ItemSubTypeComparer : IEqualityComparer<ItemSubType>
```

## Inheritance

[object](#) ← ItemSubTypeComparer

## Implements

[IEqualityComparer](#)<[ItemSubType](#)>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Instance

```
public static readonly ItemSubTypeComparer Instance
```

### Field Value

[ItemSubTypeComparer](#)

## Methods

### Equals(ItemSubType, ItemSubType)

Determines whether the specified objects are equal.

```
public bool Equals(ItemSubType x, ItemSubType y)
```

## Parameters

x [ItemSubType](#)

The first object of type [T](#) to compare.

y [ItemSubType](#)

The second object of type [T](#) to compare.

## Returns

[bool](#)

[true](#) if the specified objects are equal; otherwise, [false](#).

## GetHashCode(ItemSubType)

Returns a hash code for the specified object.

```
public int GetHashCode(ItemSubType obj)
```

## Parameters

obj [ItemSubType](#)

The [object](#) for which a hash code is to be returned.

## Returns

[int](#)

A hash code for the specified object.

## Exceptions

[ArgumentNullException](#)

The type of [obj](#) is a reference type and [obj](#) is [null](#).

# Enum ItemType

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public enum ItemType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Consumable = 0

Costume = 1

Equipment = 2

Material = 3

# Class ItemTypeComparer

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public class ItemTypeComparer : IEqualityComparer<ItemType>
```

## Inheritance

[object](#) ← ItemTypeComparer

## Implements

[IEqualityComparer](#)<ItemType>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Instance

```
public static readonly ItemTypeComparer Instance
```

### Field Value

[ItemTypeComparer](#)

## Methods

### Equals(ItemType, ItemType)

Determines whether the specified objects are equal.

```
public bool Equals(ItemType x, ItemType y)
```

## Parameters

x [ItemType](#)

The first object of type [T](#) to compare.

y [ItemType](#)

The second object of type [T](#) to compare.

## Returns

[bool](#)

[true](#) if the specified objects are equal; otherwise, [false](#).

## GetHashCode([ItemType](#))

Returns a hash code for the specified object.

```
public int GetHashCode(ItemType obj)
```

## Parameters

obj [ItemType](#)

The [object](#) for which a hash code is to be returned.

## Returns

[int](#)

A hash code for the specified object.

## Exceptions

[ArgumentNullException](#)

The type of [obj](#) is a reference type and [obj](#) is [null](#).

# Class ItemUsable

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ItemUsable : ItemBase, INonFungibleItem, IItem, IState
```

## Inheritance

[object](#) ← [ItemBase](#) ← ItemUsable

## Implements

[INonFungibleItem](#), [IItem](#), [IState](#)

## Derived

[Consumable](#), [Equipment](#)

## Inherited Members

[ItemBase.Codec](#) , [ItemBase.Id](#) , [ItemBase.Grade](#) , [ItemBase.ItemType](#) ,  
[ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [ItemBase.ToString\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ItemUsable(Dictionary)

```
protected ItemUsable(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### ItemUsable(Row, Guid, long)

```
protected ItemUsable(ItemSheet.Row data, Guid id, long requiredBlockIndex)
```

## Parameters

data [ItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

## ItemUsable(SerializationInfo, StreamingContext)

```
protected ItemUsable(SerializationInfo info, StreamingContext _)
```

## Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

## Properties

### BuffSkills

```
public List<BuffSkill> BuffSkills { get; }
```

#### Property Value

[List](#)<[BuffSkill](#)>

### ItemId

```
public Guid ItemId { get; }
```

#### Property Value

[Guid](#)

## NonFungibleId

```
public Guid NonFungibleId { get; }
```

Property Value

[Guid](#)

## RequiredBlockIndex

```
public long RequiredBlockIndex { get; set; }
```

Property Value

[long](#)

## Skills

```
public List<Skill> Skills { get; }
```

Property Value

[List](#)<[Skill](#)>

## StatsMap

```
public StatsMap StatsMap { get; }
```

Property Value

[StatsMap](#)

# Methods

## Equals(ItemUsable)

```
protected bool Equals(ItemUsable other)
```

### Parameters

other [ItemUsable](#)

### Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

### Parameters

obj [object](#)

The object to compare with the current object.

### Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetOptionCount()

```
public int GetOptionCount()
```

Returns

[int](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Update(long)

```
public void Update(long blockIndex)
```

Parameters

blockIndex [long](#)

# Enum LockType

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
public enum LockType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Order = 0

# Class Material

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Material : ItemBase, ISerializable, IFungibleItem, IItem, IState
```

## Inheritance

[object](#) ← [ItemBase](#) ← Material

## Implements

[ISerializable](#) , [IFungibleItem](#) , [IItem](#) , [IState](#)

## Derived

[TradableMaterial](#)

## Inherited Members

[ItemBase.Codec](#) , [ItemBase.Id](#) , [ItemBase.Grade](#) , [ItemBase.ItemType](#) ,  
[ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### Material(Dictionary)

```
public Material(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Material(Material)

```
public Material(Material other)
```

Parameters

other [Material](#)

## Material(Row)

```
public Material(MaterialItemSheet.Row data)
```

Parameters

data [MaterialItemSheet.Row](#)

## Material(SerializationInfo, StreamingContext)

```
protected Material(SerializationInfo info, StreamingContext _)
```

Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

## Properties

### FungibleId

```
public HashDigest<SHA256> FungibleId { get; }
```

Property Value

HashDigest<[SHA256](#)>

# ItemId

```
public HashDigest<SHA256> ItemId { get; }
```

Property Value

HashDigest<[SHA256](#)>

## Methods

### Equals(Material)

```
protected bool Equals(Material other)
```

Parameters

other [Material](#)

Returns

[bool](#)

### Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

Returns

[string](#)

A string that represents the current object.

# Class Necklace

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Necklace : Equipment, INonFungibleItem, IEquippableItem, ITradableItem,
IItem, IState
```

## Inheritance

[object](#) ↵ [ItemBase](#) ↵ [ItemUsable](#) ↵ [Equipment](#) ↵ Necklace

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#) , [Equipment.level](#) , [Equipment.Exp](#) ,  
[Equipment.optionCountFromCombination](#) , [Equipment.IconId](#) , [Equipment.ByCustomCraft](#) ,  
[Equipment.CraftWithRandom](#) , [Equipment.HasRandomOnlyIcon](#) , [Equipment.Stat](#) ,  
[Equipment.SetId](#) , [Equipment.SpineResourcePath](#) ,  
[Equipment.MadeWithMimisbrunnrRecipe](#) , [Equipment.UniqueStatType](#) ,  
[Equipment.Equipped](#) , [Equipment.GetIncrementAmountOfEnhancement\(\)](#) ,  
[Equipment.Serialize\(\)](#) , [Equipment.Equip\(\)](#) , [Equipment.Unequip\(\)](#) , [Equipment.LevelUpV1\(\)](#) ,  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#) ,  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#) ,  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#) ,  
[Equipment.GetOptions\(\)](#) , [Equipment.Equals\(Equipment\)](#) , [Equipment.Equals\(object\)](#) ,  
[Equipment.GetHashCode\(\)](#) , [ItemUsable.ItemId](#) , [ItemUsable.NonFungibleId](#) ,  
[ItemUsable.StatsMap](#) , [ItemUsable.Skills](#) , [ItemUsable.BuffSkills](#) ,  
[ItemUsable.RequiredBlockIndex](#) , [ItemUsable.Equals\(ItemUsable\)](#) ,  
[ItemUsable.GetOptionCount\(\)](#) , [ItemUsable.Update\(long\)](#) , [ItemBase.Codec](#) , [ItemBase.Id](#) ,  
[ItemBase.Grade](#) , [ItemBase.ItemType](#) , [ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [ItemBase.ToString\(\)](#) , [object.Equals\(object, object\)](#) ↵ ,  
[object.GetType\(\)](#) ↵ , [object.MemberwiseClone\(\)](#) ↵ , [object.ReferenceEquals\(object, object\)](#) ↵

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#) ,  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

## Constructors

### Necklace(Dictionary)

```
public Necklace(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Necklace(Row, Guid, long, bool)

```
public Necklace(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool madeWithMimisbrunnrRecipe = false)
```

#### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

### Necklace(SerializationInfo, StreamingContext)

```
protected Necklace(SerializationInfo info, StreamingContext _)
```

#### Parameters

info [SerializationInfo](#)

  | [StreamingContext](#)

## Properties

### TradableId

`public Guid TradableId { get; }`

Property Value

[Guid](#)

# Struct OrderLock

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public struct OrderLock : ILock
```

## Implements

[ILock](#)

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### OrderLock(List)

```
public OrderLock(List serialized)
```

#### Parameters

serialized List

### OrderLock(Guid)

```
public OrderLock(Guid orderId)
```

#### Parameters

orderId [Guid](#)

# Fields

## OrderId

```
public readonly Guid OrderId
```

### Field Value

[Guid](#)

# Properties

## Type

```
public LockType Type { get; }
```

### Property Value

[LockType](#)

# Methods

## Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Class Ring

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Ring : Equipment, INonFungibleItem, IEquippableItem, ITradableItem,
IItem, IState
```

## Inheritance

[object](#) ↵ [ItemBase](#) ↵ [ItemUsable](#) ↵ [Equipment](#) ↵ Ring

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#) , [Equipment.level](#) , [Equipment.Exp](#) ,  
[Equipment.optionCountFromCombination](#) , [Equipment.IconId](#) , [Equipment.ByCustomCraft](#) ,  
[Equipment.CraftWithRandom](#) , [Equipment.HasRandomOnlyIcon](#) , [Equipment.Stat](#) ,  
[Equipment.SetId](#) , [Equipment.SpineResourcePath](#) ,  
[Equipment.MadeWithMimisbrunnrRecipe](#) , [Equipment.UniqueStatType](#) ,  
[Equipment.Equipped](#) , [Equipment.GetIncrementAmountOfEnhancement\(\)](#) ,  
[Equipment.Serialize\(\)](#) , [Equipment.Equip\(\)](#) , [Equipment.Unequip\(\)](#) , [Equipment.LevelUpV1\(\)](#) ,  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#) ,  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#) ,  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#) ,  
[Equipment.GetOptions\(\)](#) , [Equipment.Equals\(Equipment\)](#) , [Equipment.Equals\(object\)](#) ,  
[Equipment.GetHashCode\(\)](#) , [ItemUsable.ItemId](#) , [ItemUsable.NonFungibleId](#) ,  
[ItemUsable.StatsMap](#) , [ItemUsable.Skills](#) , [ItemUsable.BuffSkills](#) ,  
[ItemUsable.RequiredBlockIndex](#) , [ItemUsable.Equals\(ItemUsable\)](#) ,  
[ItemUsable.GetOptionCount\(\)](#) , [ItemUsable.Update\(long\)](#) , [ItemBase.Codec](#) , [ItemBase.Id](#) ,  
[ItemBase.Grade](#) , [ItemBase.ItemType](#) , [ItemBase.ItemSubType](#) , [ItemBase.ElementalType](#) ,  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ItemBase.Equals\(ItemBase\)](#) , [ItemBase.ToString\(\)](#) , [object.Equals\(object, object\)](#) ↵ ,  
[object.GetType\(\)](#) ↵ , [object.MemberwiseClone\(\)](#) ↵ , [object.ReferenceEquals\(object, object\)](#) ↵

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#) ,  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

## Constructors

### Ring(Dictionary)

```
public Ring(Dictionary<string, object> serialized)
```

#### Parameters

serialized Dictionary

### Ring(Row, Guid, long, bool)

```
public Ring(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool madeWithMimisbrunnrRecipe = false)
```

#### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

### Ring(SerializationInfo, StreamingContext)

```
protected Ring(SerializationInfo info, StreamingContext _)
```

#### Parameters

info [SerializationInfo](#)

  | [StreamingContext](#)

## Properties

### TradableId

`public Guid TradableId { get; }`

Property Value

[Guid](#)

# Class ShopItem

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ShopItem
```

## Inheritance

[object](#) ← ShopItem

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ShopItem(Dictionary)

```
public ShopItem(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### ShopItem(Address, Address, Guid, FungibleAssetValue, ITradableItem)

```
public ShopItem(Address sellerAgentAddress, Address sellerAvatarAddress, Guid
productId, FungibleAssetValue price, ITradableItem tradableItem)
```

#### Parameters

sellerAgentAddress Address

`sellerAvatarAddress` Address

`productId` [Guid](#)

`price` FungibleAssetValue

`tradableItem` [ITradableItem](#)

## ShopItem(Address, Address, Guid, FungibleAssetValue, long, ITradableItem, int)

```
public ShopItem(Address sellerAgentAddress, Address sellerAvatarAddress, Guid  
productId, FungibleAssetValue price, long expiredBlockIndex, ITradableItem  
tradableItem, int tradableItemCount = 1)
```

### Parameters

`sellerAgentAddress` Address

`sellerAvatarAddress` Address

`productId` [Guid](#)

`price` FungibleAssetValue

`expiredBlockIndex` [long](#)

`tradableItem` [ITradableItem](#)

`tradableItemCount` [int](#)

## ShopItem(SerializationInfo, StreamingContext)

```
protected ShopItem(SerializationInfo info, StreamingContext _)
```

### Parameters

`info` [SerializationInfo](#)

`_` [StreamingContext](#)

## Fields

### Codec

```
protected static readonly Codec Codec
```

#### Field Value

Codec

## Costume

```
public readonly Costume Costume
```

#### Field Value

[Costume](#)

## ExpiredBlockIndexKey

```
public const string ExpiredBlockIndexKey = "ebi"
```

#### Field Value

[string](#)

## ItemUsable

```
public readonly ItemUsable ItemUsable
```

#### Field Value

## ItemUsable

### Price

```
public readonly FungibleAssetValue Price
```

#### Field Value

FungibleAssetValue

### ProductId

```
public readonly Guid ProductId
```

#### Field Value

[Guid](#)

### SellerAgentAddress

```
public readonly Address SellerAgentAddress
```

#### Field Value

Address

### SellerAvatarAddress

```
public readonly Address SellerAvatarAddress
```

#### Field Value

Address

# TradableFungibleItem

```
public readonly ITradableFungibleItem TradableFungibleItem
```

Field Value

[ITradableFungibleItem](#)

## TradableFungibleItemCount

```
public readonly int TradableFungibleItemCount
```

Field Value

[int](#)

## Properties

### ExpiredBlockIndex

```
public long ExpiredBlockIndex { get; }
```

Property Value

[long](#)

## Methods

### Equals(ShopItem)

```
protected bool Equals(ShopItem other)
```

Parameters

other [ShopItem](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetObjectData(SerializationInfo, StreamingContext)

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Serialize()

```
public IValue Serialize()
```

## Returns

IValue

## SerializeBackup1()

```
public IValue SerializeBackup1()
```

## Returns

IValue

# Class TradableMaterial

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TradableMaterial : Material, ISerializable, ITradableFungibleItem,
ITradableItem, IFungibleItem, IItem, IState, ICloneable
```

## Inheritance

[object](#) ← [ItemBase](#) ← [Material](#) ← [TradableMaterial](#)

## Implements

[ISerializable](#), [ITradableFungibleItem](#), [ITradableItem](#), [IFungibleItem](#), [IItem](#), [IState](#),  
[ICloneable](#)

## Inherited Members

[Material.ItemId](#), [Material.FungibleId](#), [Material.Equals\(Material\)](#), [ItemBase.Codec](#),  
[ItemBase.Id](#), [ItemBase.Grade](#), [ItemBase.ItemType](#), [ItemBase.ItemSubType](#),  
[ItemBase.ElementalType](#), [ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#),  
[ItemBase.Equals\(ItemBase\)](#), [object.Equals\(object, object\)](#), [object.GetType\(\)](#),  
[object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Constructors

### TradableMaterial(Dictionary)

```
public TradableMaterial(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### TradableMaterial(Row)

```
public TradableMaterial(MaterialItemSheet.Row data)
```

## Parameters

data [MaterialItemSheet.Row](#)

## TradableMaterial(SerializationInfo, StreamingContext)

`protected TradableMaterial(SerializationInfo info, StreamingContext _)`

## Parameters

info [SerializationInfo](#)

\_ [StreamingContext](#)

## Properties

### RequiredBlockIndex

`public long RequiredBlockIndex { get; set; }`

#### Property Value

[long](#)

### TradableId

`public Guid TradableId { get; }`

#### Property Value

[Guid](#)

## Methods

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## DeriveTradableId(HashDigest<SHA256>)

```
public static Guid DeriveTradableId(HashDigest<SHA256> fungibleId)
```

Parameters

**fungibleId** HashDigest<[SHA256](#)>

Returns

[Guid](#)

## Equals(TradableMaterial)

```
protected bool Equals(TradableMaterial other)
```

Parameters

**other** [TradableMaterial](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

Returns

[string](#)

A string that represents the current object.

# Class Weapon

Namespace: [Nekoyume.Model.Item](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Weapon : Equipment, INonFungibleItem, IEquippableItem, ITradableItem,
IItem, IState
```

## Inheritance

[object](#) ↵ [ItemBase](#) ↵ [ItemUsable](#) ↵ [Equipment](#) ↵ Weapon

## Implements

[INonFungibleItem](#), [IEquippableItem](#), [ITradableItem](#), [IItem](#), [IState](#)

## Inherited Members

[Equipment.equipped](#), [Equipment.level](#), [Equipment.Exp](#),  
[Equipment.optionCountFromCombination](#), [Equipment.IconId](#), [Equipment.ByCustomCraft](#),  
[Equipment.CraftWithRandom](#), [Equipment.HasRandomOnlyIcon](#), [Equipment.Stat](#),  
[Equipment.SetId](#), [Equipment.SpineResourcePath](#),  
[Equipment.MadeWithMimisbrunnrRecipe](#), [Equipment.UniqueStatType](#),  
[Equipment.Equipped](#), [Equipment.GetIncrementAmountOfEnhancement\(\)](#),  
[Equipment.Serialize\(\)](#), [Equipment.Equip\(\)](#), [Equipment.Unequip\(\)](#), [Equipment.LevelUpV1\(\)](#),  
[Equipment.LevelUp\(IRandom, EnhancementCostSheetV2.Row, bool\)](#),  
[Equipment.SetLevel\(IRandom, int, EnhancementCostSheetV3\)](#),  
[Equipment.GetRealExp\(EquipmentItemSheet, EnhancementCostSheetV3\)](#),  
[Equipment.GetOptions\(\)](#), [Equipment.Equals\(Equipment\)](#), [Equipment.Equals\(object\)](#),  
[Equipment.GetHashCode\(\)](#), [ItemUsable.ItemId](#), [ItemUsable.NonFungibleId](#),  
[ItemUsable.StatsMap](#), [ItemUsable.Skills](#), [ItemUsable.BuffSkills](#),  
[ItemUsable.RequiredBlockIndex](#), [ItemUsable.Equals\(ItemUsable\)](#),  
[ItemUsable.GetOptionCount\(\)](#), [ItemUsable.Update\(long\)](#), [ItemBase.Codec](#), [ItemBase.Id](#),  
[ItemBase.Grade](#), [ItemBase.ItemType](#), [ItemBase.ItemSubType](#), [ItemBase.ElementalType](#),  
[ItemBase.GetObjectData\(SerializationInfo, StreamingContext\)](#),  
[ItemBase.Equals\(ItemBase\)](#), [ItemBase.ToString\(\)](#), [object.Equals\(object, object\) ↵](#),  
[object.GetType\(\) ↵](#), [object.MemberwiseClone\(\) ↵](#), [object.ReferenceEquals\(object, object\) ↵](#)

## Extension Methods

[EquipmentExtensions.GetRequirementLevel\(Equipment, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#) ,  
[EquipmentExtensions.IsMadeWithMimisbrunnrRecipe\(Equipment, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet\)](#)

## Constructors

### Weapon(Dictionary)

```
public Weapon(Dictionary<string, object> serialized)
```

#### Parameters

serialized Dictionary

### Weapon(Row, Guid, long, bool)

```
public Weapon(EquipmentItemSheet.Row data, Guid id, long requiredBlockIndex, bool madeWithMimisbrunnrRecipe = false)
```

#### Parameters

data [EquipmentItemSheet.Row](#)

id [Guid](#)

requiredBlockIndex [long](#)

madeWithMimisbrunnrRecipe [bool](#)

### Weapon(SerializationInfo, StreamingContext)

```
protected Weapon(SerializationInfo info, StreamingContext _)
```

#### Parameters

info [SerializationInfo](#)

  | [StreamingContext](#)

## Properties

### TradableId

`public Guid TradableId { get; }`

Property Value

[Guid](#)

# Namespace Nekoyume.Model.Mail

## Classes

[AdventureBossRaffleWinnerMail](#)

[AttachmentMail](#)

[BuyerMail](#)

[CancelOrderMail](#)

[ClaimItemsMail](#)

[CombinationMail](#)

[CustomCraftMail](#)

[DailyRewardMail](#)

[GrindingMail](#)

[ItemEnhanceMail](#)

[Mail](#)

[MailBox](#)

[MaterialCraftMail](#)

[MonsterCollectionMail](#)

[OrderBuyerMail](#)

[OrderExpirationMail](#)

[OrderSellerMail](#)

[ProductBuyerMail](#)

[ProductCancelMail](#)

[ProductSellerMail](#)

[SellCancelMail](#)

[SellerMail](#)

[UnloadFromMyGaragesRecipientMail](#)

## Interfaces

[IMail](#)

## Enums

[MailType](#)

# Class AdventureBossRaffleWinnerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public class AdventureBossRaffleWinnerMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← AdventureBossRaffleWinnerMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.MailType](#) , [Mail.requiredBlockIndex](#) ,  
[Mail.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AdventureBossRaffleWinnerMail(Dictionary)

```
public AdventureBossRaffleWinnerMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### AdventureBossRaffleWinnerMail(long, Guid, long, long, FungibleAssetValue)

```
public AdventureBossRaffleWinnerMail(long blockIndex, Guid id, long  
requiredBlockIndex, long season, FungibleAssetValue reward)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

season [long](#)

reward FungibleAssetValue

## Fields

### Reward

`public FungibleAssetValue Reward`

### Field Value

FungibleAssetValue

### Season

`public readonly long Season`

### Field Value

[long](#)

## Properties

### TypeId

`protected override string TypeId { get; }`

Property Value

[string](#) ↗

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class AttachmentMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class AttachmentMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← AttachmentMail

## Implements

[IState](#)

## Derived

[BuyerMail](#), [CombinationMail](#), [DailyRewardMail](#), [ItemEnhanceMail](#), [MonsterCollectionMail](#),  
[SellCancelMail](#), [SellerMail](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.MailType](#) , [Mail.requiredBlockIndex](#) ,  
[Mail.Read\(IMail\)](#) , [Mail.TypeId](#) , [Mail.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AttachmentMail(Dictionary)

```
public AttachmentMail(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

### AttachmentMail(AttachmentActionResult, long, Guid, long)

```
protected AttachmentMail(AttachmentActionResult attachmentActionResult, long  
blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [AttachmentActionResult](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Fields

### attachment

```
public AttachmentActionResult attachment
```

## Field Value

[AttachmentActionResult](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

# Class BuyerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuyerMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← BuyerMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### BuyerMail(Dictionary)

```
public BuyerMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### BuyerMail(AttachmentActionResult, long, Guid, long)

```
public BuyerMail(AttachmentActionResult attachmentActionResult, long blockIndex,
Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [AttachmentActionResult](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

#### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

#### Property Value

[string](#)

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

## Parameters

[mail](#) [IMail](#)

# Class CancelOrderMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CancelOrderMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← CancelOrderMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CancelOrderMail(Dictionary)

```
public CancelOrderMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### CancelOrderMail(long, Guid, long, Guid)

```
public CancelOrderMail(long blockIndex, Guid id, long requiredBlockIndex,
Guid orderId)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

orderId [Guid](#)

## Fields

### OrderId

```
public readonly Guid OrderId
```

### Field Value

[Guid](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

#### Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

#### Returns

IValue

# Class ClaimItemsMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public class ClaimItemsMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← ClaimItemsMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ClaimItemsMail(Dictionary)

```
public ClaimItemsMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ClaimItemsMail(long, Guid, long, List<FungibleAssetValue>, List<(int id, int count)>, string)

```
public ClaimItemsMail(long blockIndex, Guid id, long requiredBlockIndex,  
List<FungibleAssetValue> fungibleAssetValues, List<(int id, int count)> items,
```

```
    string memo)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

fungibleAssetValues [List](#)<FungibleAssetValue>

items [List](#)<(int id, int count)>

memo [string](#)

## Fields

### FungibleAssetValues

```
public List<FungibleAssetValue> FungibleAssetValues
```

#### Field Value

[List](#)<FungibleAssetValue>

### Items

```
public List<(int id, int count)> Items
```

#### Field Value

[List](#)<(int id, int count)>

### Memo

```
public string Memo
```

Field Value

[string](#) ↗

## Properties

### MailType

```
public override MailType MailType { get; }
```

Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

mail [IMail](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class CombinationMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← CombinationMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CombinationMail(Dictionary)

```
public CombinationMail(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### CombinationMail(ResultModel, long, Guid, long)

```
public CombinationMail(CombinationConsumable5.ResultModel attachmentActionResult,
long blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [CombinationConsumable5.ResultModel](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

`public override MailType MailType { get; }`

#### Property Value

[MailType](#)

### TypeId

`protected override string TypeId { get; }`

#### Property Value

[string](#)

## Methods

### Read(IMail)

`public override void Read(IMail mail)`

## Parameters

[mail](#) [IMail](#)

# Class CustomCraftMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public class CustomCraftMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← CustomCraftMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CustomCraftMail(Dictionary)

```
public CustomCraftMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### CustomCraftMail(long, Guid, long, Equipment)

```
public CustomCraftMail(long blockIndex, Guid id, long requiredBlockIndex,  
Equipment equipment)
```

#### Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

equipment [Equipment](#)

## Fields

### Equipment

`public Equipment Equipment`

Field Value

[Equipment](#)

## Properties

### MailType

`public override MailType MailType { get; }`

Property Value

[MailType](#)

## TypeId

`protected override string TypeId { get; }`

Property Value

[string](#)

# Methods

## Read(IMail)

```
public override void Read(IMail mail)
```

### Parameters

mail [IMail](#)

## Serialize()

```
public override IValue Serialize()
```

### Returns

IValue

# Class DailyRewardMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DailyRewardMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← DailyRewardMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### DailyRewardMail(Dictionary)

```
public DailyRewardMail(Dictionary serialized)
```

#### Parameters

`serialized` Dictionary

### DailyRewardMail(AttachmentActionResult, long, Guid, long)

```
public DailyRewardMail(AttachmentActionResult attachmentActionResult, long
```

```
blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [AttachmentActionResult](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

#### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

#### Property Value

[string](#)

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

## Parameters

[mail](#) [IMail](#)

# Class GrindingMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public class GrindingMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← GrindingMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GrindingMail(Dictionary)

```
public GrindingMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### GrindingMail(long, Guid, long, int, FungibleAssetValue, int)

```
public GrindingMail(long blockIndex, Guid id, long requiredBlockIndex, int itemCount, FungibleAssetValue asset, int rewardMaterialCount)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

itemCount [int](#)

asset FungibleAssetValue

rewardMaterialCount [int](#)

## Fields

### Asset

[public](#) FungibleAssetValue Asset

#### Field Value

FungibleAssetValue

### ItemCount

[public readonly int](#) ItemCount

#### Field Value

[int](#)

### RewardMaterialCount

[public readonly int](#) RewardMaterialCount

Field Value

[int](#) ↗

## Properties

### MailType

```
public override MailType MailType { get; }
```

Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

[mail](#) [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Interface IMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public interface IMail
```

## Methods

### Read(AdventureBossRaffleWinnerMail)

```
void Read(AdventureBossRaffleWinnerMail adventureBossRaffleWinnerMail)
```

#### Parameters

adventureBossRaffleWinnerMail [AdventureBossRaffleWinnerMail](#)

### Read(BuyerMail)

```
void Read(BuyerMail buyerMail)
```

#### Parameters

buyerMail [BuyerMail](#)

### Read(CancelOrderMail)

```
void Read(CancelOrderMail cancelOrderMail)
```

#### Parameters

cancelOrderMail [CancelOrderMail](#)

## Read(ClaimItemsMail)

```
void Read(ClaimItemsMail claimItemsMail)
```

### Parameters

claimItemsMail [ClaimItemsMail](#)

## Read(CombinationMail)

```
void Read(CombinationMail mail)
```

### Parameters

mail [CombinationMail](#)

## Read(CustomCraftMail)

```
void Read(CustomCraftMail customCraftMail)
```

### Parameters

customCraftMail [CustomCraftMail](#)

## Read(DailyRewardMail)

```
void Read(DailyRewardMail dailyRewardMail)
```

### Parameters

dailyRewardMail [DailyRewardMail](#)

## Read(GrindingMail)

```
void Read(GrindingMail grindingMail)
```

Parameters

grindingMail [GrindingMail](#)

## Read(ItemEnhanceMail)

```
void Read(ItemEnhanceMail itemEnhanceMail)
```

Parameters

itemEnhanceMail [ItemEnhanceMail](#)

## Read(MaterialCraftMail)

```
void Read(MaterialCraftMail materialCraftMail)
```

Parameters

materialCraftMail [MaterialCraftMail](#)

## Read(MonsterCollectionMail)

```
void Read(MonsterCollectionMail monsterCollectionMail)
```

Parameters

monsterCollectionMail [MonsterCollectionMail](#)

## Read(OrderBuyerMail)

```
void Read(OrderBuyerMail orderBuyerMail)
```

Parameters

orderBuyerMail [OrderBuyerMail](#)

## Read(OrderExpirationMail)

```
void Read(OrderExpirationMail orderExpirationMail)
```

Parameters

orderExpirationMail [OrderExpirationMail](#)

## Read(OrderSellerMail)

```
void Read(OrderSellerMail orderSellerMail)
```

Parameters

orderSellerMail [OrderSellerMail](#)

## Read(ProductBuyerMail)

```
void Read(ProductBuyerMail productBuyerMail)
```

Parameters

productBuyerMail [ProductBuyerMail](#)

## Read(ProductCancelMail)

```
void Read(ProductCancelMail productCancelMail)
```

Parameters

productCancelMail [ProductCancelMail](#)

## Read(ProductSellerMail)

```
void Read(ProductSellerMail productSellerMail)
```

Parameters

productSellerMail [ProductSellerMail](#)

## Read(SellCancelMail)

```
void Read(SellCancelMail mail)
```

Parameters

mail [SellCancelMail](#)

## Read(SellerMail)

```
void Read(SellerMail sellerMail)
```

Parameters

sellerMail [SellerMail](#)

## Read(UnloadFromMyGaragesRecipientMail)

```
void Read(UnloadFromMyGaragesRecipientMail unloadFromMyGaragesRecipientMail)
```

## Parameters

unloadFromMyGaragesRecipientMail [UnloadFromMyGaragesRecipientMail](#)

# Class ItemEnhanceMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhanceMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← ItemEnhanceMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemEnhanceMail(Dictionary)

```
public ItemEnhanceMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ItemEnhanceMail(AttachmentActionResult, long, Guid, long)

```
public ItemEnhanceMail(AttachmentActionResult attachmentActionResult, long
```

```
blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [AttachmentActionResult](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

#### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

#### Property Value

[string](#)

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

## Parameters

[mail](#) [IMail](#)

# Class Mail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Mail : IState
```

## Inheritance

[object](#) ← Mail

## Implements

[IState](#)

## Derived

[AdventureBossRaffleWinnerMail](#), [AttachmentMail](#), [CancelOrderMail](#), [ClaimItemsMail](#),  
[CustomCraftMail](#), [GrindingMail](#), [MaterialCraftMail](#), [OrderBuyerMail](#), [OrderExpirationMail](#),  
[OrderSellerMail](#), [ProductBuyerMail](#), [ProductCancelMail](#), [ProductSellerMail](#),  
[UnloadFromMyGaragesRecipientMail](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### Mail(Dictionary)

```
protected Mail(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Mail(long, Guid, long)

```
protected Mail(long blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Fields

### New

```
public bool New
```

### Field Value

[bool](#)

### blockIndex

```
public long blockIndex
```

### Field Value

[long](#)

### id

```
public Guid id
```

### Field Value

[Guid](#)

## requiredBlockIndex

```
public long requiredBlockIndex
```

Field Value

[long](#) ↗

## Properties

### MailType

```
public virtual MailType MailType { get; }
```

Property Value

[MailType](#)

### TypeId

```
protected abstract string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Deserialize(Dictionary)

```
public static Mail Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[Mail](#)

## Read(IMail)

```
public abstract void Read(IMail mail)
```

Parameters

**mail** [IMail](#)

## Serialize()

```
public virtual IValue Serialize()
```

Returns

IValue

# Class MailBox

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MailBox : IEnumerable<Mail>, IEnumerable, IState
```

## Inheritance

[object](#) ← MailBox

## Implements

[IEnumerable](#)<[Mail](#)>, [IEnumerable](#), [IState](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### MailBox()

```
public MailBox()
```

### MailBox(List)

```
public MailBox(List serialized)
```

## Parameters

serialized List

## Properties

# Count

```
public int Count { get; }
```

Property Value

[int](#)

## this[int]

```
public Mail this[int idx] { get; }
```

Parameters

idx [int](#)

Property Value

[Mail](#)

# Methods

## Add(Mail)

```
public void Add(Mail mail)
```

Parameters

mail [Mail](#)

## CleanUp()

```
public void CleanUp()
```

## CleanUpTemp(long)

```
[Obsolete("No longer in use.")]  
public void CleanUpTemp(long blockIndex)
```

### Parameters

blockIndex [long](#)

## CleanUpV1()

```
[Obsolete("Use CleanUp")]  
public void CleanUpV1()
```

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<Mail> GetEnumerator()
```

### Returns

[IEnumerator](#)<Mail>

An enumerator that can be used to iterate through the collection.

## Remove(Mail)

```
public void Remove(Mail mail)
```

### Parameters

mail [Mail](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Enum MailType

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public enum MailType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Auction = 2

CustomCraft = 6

Grinding = 4

Summon = 5

System = 3

Workshop = 1

# Class MaterialCraftMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MaterialCraftMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← MaterialCraftMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### MaterialCraftMail(Dictionary)

```
public MaterialCraftMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### MaterialCraftMail(long, Guid, long, int, int)

```
public MaterialCraftMail(long blockIndex, Guid id, long requiredBlockIndex, int
itemCount, int itemId)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

itemCount [int](#)

itemId [int](#)

## Fields

### ItemCount

`public int ItemCount`

#### Field Value

[int](#)

### ItemId

`public int ItemId`

#### Field Value

[int](#)

## Properties

### MailType

`public override MailType MailType { get; }`

Property Value

[MailType](#)

TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

Methods

Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

mail [IMail](#)

Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class MonsterCollectionMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← MonsterCollectionMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### MonsterCollectionMail(Dictionary)

```
public MonsterCollectionMail(Dictionary serialized)
```

#### Parameters

`serialized` Dictionary

### MonsterCollectionMail(AttachmentActionResult, long, Guid, long)

```
public MonsterCollectionMail(AttachmentActionResult attachmentActionResult, long
```

```
blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [AttachmentActionResult](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

#### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

#### Property Value

[string](#)

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

## Parameters

[mail](#) [IMail](#)

# Class OrderBuyerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderBuyerMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← OrderBuyerMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### OrderBuyerMail(Dictionary)

```
public OrderBuyerMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### OrderBuyerMail(long, Guid, long, Guid)

```
public OrderBuyerMail(long blockIndex, Guid id, long requiredBlockIndex,
Guid orderId)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

orderId [Guid](#)

## Fields

### OrderId

```
public readonly Guid OrderId
```

### Field Value

[Guid](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

#### Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

#### Returns

IValue

# Class OrderExpirationMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderExpirationMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← OrderExpirationMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### OrderExpirationMail(Dictionary)

```
public OrderExpirationMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### OrderExpirationMail(long, Guid, long, Guid)

```
public OrderExpirationMail(long blockIndex, Guid id, long requiredBlockIndex,
Guid orderId)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

orderId [Guid](#)

## Fields

### OrderId

```
public readonly Guid OrderId
```

### Field Value

[Guid](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

[string](#)

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

#### Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

#### Returns

IValue

# Class OrderSellerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class OrderSellerMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← OrderSellerMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### OrderSellerMail(Dictionary)

```
public OrderSellerMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### OrderSellerMail(long, Guid, long, Guid)

```
public OrderSellerMail(long blockIndex, Guid id, long requiredBlockIndex,
Guid orderId)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

orderId [Guid](#)

## Fields

### OrderId

```
public readonly Guid OrderId
```

### Field Value

[Guid](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

### Property Value

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

#### Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

#### Returns

IValue

# Class ProductBuyerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ProductBuyerMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← ProductBuyerMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ProductBuyerMail(Dictionary)

```
public ProductBuyerMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ProductBuyerMail(long, Guid, long, Guid, Product)

```
public ProductBuyerMail(long blockIndex, Guid id, long requiredBlockIndex, Guid
productId, Product product)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

productId [Guid](#)

product [Product](#)

## Fields

### Product

`public readonly Product Product`

#### Field Value

[Product](#)

### ProductId

`public readonly Guid ProductID`

#### Field Value

[Guid](#)

### ProductKey

`public const string ProductKey = "p"`

#### Field Value

[string](#) ↗

## Properties

### MailType

```
public override MailType MailType { get; }
```

Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ProductCancelMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ProductCancelMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← ProductCancelMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ProductCancelMail(Dictionary)

```
public ProductCancelMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ProductCancelMail(long, Guid, long, Guid, Product)

```
public ProductCancelMail(long blockIndex, Guid id, long requiredBlockIndex, Guid
productId, Product product)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

productId [Guid](#)

product [Product](#)

## Fields

### Product

`public readonly Product Product`

#### Field Value

[Product](#)

### ProductId

`public readonly Guid ProductID`

#### Field Value

[Guid](#)

### ProductKey

`public const string ProductKey = "p"`

#### Field Value

[string](#) ↗

## Properties

### MailType

```
public override MailType MailType { get; }
```

Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ProductSellerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ProductSellerMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← ProductSellerMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ProductSellerMail(Dictionary)

```
public ProductSellerMail(Dictionary serialized)
```

#### Parameters

`serialized` Dictionary

### ProductSellerMail(long, Guid, long, Guid, Product)

```
public ProductSellerMail(long blockIndex, Guid id, long requiredBlockIndex, Guid
productId, Product product)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

productId [Guid](#)

product [Product](#)

## Fields

### Product

`public readonly Product Product`

#### Field Value

[Product](#)

### ProductId

`public readonly Guid ProductID`

#### Field Value

[Guid](#)

### ProductKey

`public const string ProductKey = "p"`

#### Field Value

[string](#) ↗

## Properties

### MailType

```
public override MailType MailType { get; }
```

Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#) ↗

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

Parameters

mail [IMail](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class SellCancelMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SellCancelMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← SellCancelMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### SellCancelMail(Dictionary)

```
public SellCancelMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### SellCancelMail(Result, long, Guid, long)

```
public SellCancelMail(SellCancellation.Result attachmentActionResult, long
blockIndex, Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [SellCancellation.Result](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

```
public override MailType MailType { get; }
```

#### Property Value

[MailType](#)

### TypeId

```
protected override string TypeId { get; }
```

#### Property Value

[string](#)

## Methods

### Read(IMail)

```
public override void Read(IMail mail)
```

## Parameters

[mail](#) [IMail](#)

# Class SellerMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SellerMail : AttachmentMail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← [AttachmentMail](#) ← SellerMail

## Implements

[IState](#)

## Inherited Members

[AttachmentMail.attachment](#) , [AttachmentMail.Serialize\(\)](#) , [Mail.id](#) , [Mail.New](#) ,  
[Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### SellerMail(Dictionary)

```
public SellerMail(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### SellerMail(AttachmentActionResult, long, Guid, long)

```
public SellerMail(AttachmentActionResult attachmentActionResult, long blockIndex,
Guid id, long requiredBlockIndex)
```

## Parameters

attachmentActionResult [AttachmentActionResult](#)

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

## Properties

### MailType

`public override MailType MailType { get; }`

### Property Value

[MailType](#)

### TypeId

`protected override string TypeId { get; }`

### Property Value

[string](#)

## Methods

### Read(IMail)

`public override void Read(IMail mail)`

## Parameters

[mail](#) [IMail](#)

# Class UnloadFromMyGaragesRecipientMail

Namespace: [Nekoyume.Model.Mail](#)

Assembly: Lib9c.dll

```
public class UnloadFromMyGaragesRecipientMail : Mail, IState
```

## Inheritance

[object](#) ← [Mail](#) ← UnloadFromMyGaragesRecipientMail

## Implements

[IState](#)

## Inherited Members

[Mail.id](#) , [Mail.New](#) , [Mail.blockIndex](#) , [Mail.requiredBlockIndex](#) , [Mail.Deserialize\(Dictionary\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### UnloadFromMyGaragesRecipientMail(Dictionary)

```
public UnloadFromMyGaragesRecipientMail(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

UnloadFromMyGaragesRecipientMail(long, Guid, long,  
IEnumerable<(Address balanceAddr,  
FungibleAssetValue value)>?,  
IEnumerable<(HashDigest<SHA256> fungibleId, int  
count)>?, string?)

```
public UnloadFromMyGaragesRecipientMail(long blockIndex, Guid id, long requiredBlockIndex, IEnumerable<(Address balanceAddr, FungibleAssetValue value)>? fungibleAssetValue, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>? fungibleIdAndCount, string? memo)
```

## Parameters

blockIndex [long](#)

id [Guid](#)

requiredBlockIndex [long](#)

fungibleAssetValue [IEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

fungibleIdAndCount [IEnumerable](#)<(HashDigest<[SHA256](#)> [fungibleId](#), [int](#) [count](#))>

memo [string](#)

## Fields

### FungibleAssetValues

```
public readonly IOrderedEnumerable<(Address balanceAddr, FungibleAssetValue value)>? FungibleAssetValues
```

#### Field Value

[IOrderedEnumerable](#)<(Address [balanceAddr](#), FungibleAssetValue [value](#))>

### FungibleIdAndCounts

```
public readonly IOrderedEnumerable<(HashDigest<SHA256> fungibleId, int count)>? FungibleIdAndCounts
```

#### Field Value

[IOrderedEnumerable](#)<(HashDigest<[SHA256](#)> fungibleId, int count)>

## Memo

public readonly string? Memo

### Field Value

[string](#)

## Properties

### MailType

public override MailType MailType { get; }

### Property Value

[MailType](#)

### Typeld

protected override string TypeId { get; }

### Property Value

[string](#)

## Methods

### Read(IMail)

public override void Read(IMail mail)

## Parameters

mail [IMail](#)

## Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

# Namespace Nekoyume.Model.Market

## Classes

[FavProduct](#)

[ItemProduct](#)

[Product](#)

[ProductFactory](#)

[ProductReceipt](#)

[ProductsState](#)

## Enums

[ProductType](#)

# Class FavProduct

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public class FavProduct : Product
```

## Inheritance

[object](#) ← [Product](#) ← FavProduct

## Inherited Members

[Product.DeriveAddress\(Guid\)](#) , [Product.ProductId](#) , [Product.Type](#) , [Product.Price](#) ,  
[Product.RegisteredBlockIndex](#) , [Product.SellerAvatarAddress](#) , [Product.SellerAgentAddress](#) ,  
[Product.Validate\(IProductInfo\)](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### FavProduct()

```
public FavProduct()
```

### FavProduct(List)

```
public FavProduct(List serialized)
```

## Parameters

serialized List

## Fields

### Asset

```
public FungibleAssetValue Asset
```

## Field Value

FungibleAssetValue

## Methods

### Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

# Class ItemProduct

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public class ItemProduct : Product
```

## Inheritance

[object](#) ← [Product](#) ← ItemProduct

## Inherited Members

[Product.DeriveAddress\(Guid\)](#) , [Product.ProductId](#) , [Product.Type](#) , [Product.Price](#) ,  
[Product.RegisteredBlockIndex](#) , [Product.SellerAvatarAddress](#) , [Product.SellerAgentAddress](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemProduct()

```
public ItemProduct()
```

### ItemProduct(List)

```
public ItemProduct(List serialized)
```

## Parameters

serialized List

## Fields

### ItemCount

```
public int ItemCount
```

Field Value

[int](#)

## TradableItem

```
public ITradableItem TradableItem
```

Field Value

[ITradableItem](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

### Validate(IProductInfo)

```
public override void Validate(IProductInfo productInfo)
```

Parameters

productInfo [IProductInfo](#)

# Class Product

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public class Product
```

## Inheritance

[object](#) ← Product

## Derived

[FavProduct](#), [ItemProduct](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

Product()

```
protected Product()
```

Product(List)

```
protected Product(List serialized)
```

## Parameters

serialized List

## Fields

## Price

```
public FungibleAssetValue Price
```

### Field Value

FungibleAssetValue

## ProductId

```
public Guid ProductId
```

### Field Value

[Guid](#)

## RegisteredBlockIndex

```
public long RegisteredBlockIndex
```

### Field Value

[long](#)

## SellerAgentAddress

```
public Address SellerAgentAddress
```

### Field Value

Address

## SellerAvatarAddress

```
public Address SellerAvatarAddress
```

Field Value

Address

Type

```
public ProductType Type
```

Field Value

[ProductType](#)

Methods

DeriveAddress(Guid)

```
public static Address DeriveAddress(Guid productId)
```

Parameters

productId [Guid](#)

Returns

Address

Serialize()

```
public virtual IValue Serialize()
```

Returns

## Validate(IProductInfo)

```
public virtual void Validate(IProductInfo productInfo)
```

### Parameters

productInfo [IProductInfo](#)

# Class ProductFactory

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public static class ProductFactory
```

## Inheritance

[object](#) ← ProductFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### DeserializeProduct(List)

```
public static Product DeserializeProduct(List serialized)
```

#### Parameters

serialized List

#### Returns

[Product](#)

### DeserializeProductInfo(List)

```
public static IProductInfo DeserializeProductInfo(List serialized)
```

#### Parameters

**serialized** List

Returns

[IProductInfo](#)

## DeserializeRegisterInfo(List)

```
public static IRegisterInfo DeserializeRegisterInfo(List serialized)
```

Parameters

**serialized** List

Returns

[IRegisterInfo](#)

# Class ProductReceipt

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public class ProductReceipt
```

## Inheritance

[object](#) ← ProductReceipt

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ProductReceipt(List)

```
public ProductReceipt(List serialized)
```

#### Parameters

serialized List

### ProductReceipt(Guid, Address, Address, FungibleAssetValue, long)

```
public ProductReceipt(Guid productId, Address sellerAvatarAddress, Address  
buyerAvatarAddress, FungibleAssetValue price, long purchasedBlockIndex)
```

#### Parameters

productId [Guid](#)

`sellerAvatarAddress` Address

`buyerAvatarAddress` Address

`price` FungibleAssetValue

`purchasedBlockIndex` [long](#)

## Fields

### BuyerAvatarAddress

`public readonly` Address BuyerAvatarAddress

Field Value

Address

### Price

`public` FungibleAssetValue Price

Field Value

FungibleAssetValue

### ProductId

`public readonly` Guid ProductId

Field Value

[Guid](#)

### PurchasedBlockIndex

```
public long PurchasedBlockIndex
```

Field Value

[long](#) ↗

## SellerAvatarAddress

```
public readonly Address SellerAvatarAddress
```

Field Value

Address

## Methods

### DeriveAddress(Guid)

```
public static Address DeriveAddress(Guid productId)
```

Parameters

productId [Guid](#) ↗

Returns

Address

### Serialize()

```
public IValue Serialize()
```

Returns



# Enum ProductType

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public enum ProductType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Fungible = 0

FungibleAssetValue = 1

NonFungible = 2

# Class ProductsState

Namespace: [Nekoyume.Model.Market](#)

Assembly: Lib9c.dll

```
public class ProductsState
```

## Inheritance

[object](#) ← ProductsState

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ProductsState()

```
public ProductsState()
```

### ProductsState(List)

```
public ProductsState(List serialized)
```

## Parameters

serialized List

## Fields

### ProductIds

```
public List<Guid> ProductIds
```

## Field Value

[List](#)<[Guid](#)>

## Methods

### DeriveAddress(Address)

```
public static Address DeriveAddress(Address avatarAddress)
```

#### Parameters

avatarAddress Address

#### Returns

Address

### Serialize()

```
public IValue Serialize()
```

#### Returns

IValue

# Namespace Nekoyume.Model.Pet

## Enums

[PetOptionType](#)

# Enum PetOptionType

Namespace: [Nekoyume.Model.Pet](#)

Assembly: Lib9c.dll

```
public enum PetOptionType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

AdditionalOptionRate = 2

AdditionalOptionRateByFixedValue = 3

DiscountMaterialCostCrystal = 5

IncreaseBlockPerHourglass = 4

ReduceRequiredBlock = 0

ReduceRequiredBlockByFixedValue = 1

# Namespace Nekoyume.Model.Quest

## Classes

[CollectQuest](#)

[CombinationEquipmentQuest](#)

[CombinationQuest](#)

[GeneralQuest](#)

[GoldQuest](#)

[ItemEnhancementQuest](#)

[ItemGradeQuest](#)

[ItemTypeCollectQuest](#)

[MonsterQuest](#)

[Quest](#)

[QuestList](#)

[QuestReward](#)

[TradeQuest](#)

[UpdateListQuestsCountException](#)

[UpdateListVersionException](#)

[WorldQuest](#)

## Enums

[QuestEventType](#)

[QuestType](#)

# Class CollectQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CollectQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← CollectQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CollectQuest(Dictionary)

```
public CollectQuest(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### CollectQuest(List)

```
public CollectQuest(List serialized)
```

Parameters

**serialized** List

## CollectQuest(Row, QuestReward)

```
public CollectQuest(CollectQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [CollectQuestSheet.Row](#)

**reward** [QuestReward](#)

## Fields

### ItemId

```
public readonly int ItemId
```

Field Value

[int](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

# TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

Check()

```
public override void Check()
```

GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

Serialize()

```
public override IValue Serialize()
```

Returns

IValue

SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(CollectionMap)

```
public void Update(CollectionMap itemMap)
```

Parameters

itemMap [CollectionMap](#)

# Class CombinationEquipmentQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
public class CombinationEquipmentQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← CombinationEquipmentQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CombinationEquipmentQuest(Dictionary)

```
public CombinationEquipmentQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### CombinationEquipmentQuest(List)

```
public CombinationEquipmentQuest(List serialized)
```

Parameters

**serialized** List

## CombinationEquipmentQuest(Row, QuestReward, int)

```
public CombinationEquipmentQuest(QuestSheet.Row data, QuestReward reward,  
int stageId)
```

Parameters

**data** [QuestSheet.Row](#)

**reward** [QuestReward](#)

**stageId** [int](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

### Recipeld

```
public int RecipeId { get; }
```

Property Value

[int](#)

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(int)

```
public void Update(int recipeId)
```

Parameters

recipeId [int](#)

# Class CombinationQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← CombinationQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CombinationQuest(Dictionary)

```
public CombinationQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### CombinationQuest(List)

```
public CombinationQuest(List serialized)
```

Parameters

**serialized** List

## CombinationQuest(Row, QuestReward)

```
public CombinationQuest(CombinationQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [CombinationQuestSheet.Row](#)

**reward** [QuestReward](#)

## Properties

### ItemSubType

```
public ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

### ItemType

```
public ItemType ItemType { get; }
```

Property Value

[ItemType](#)

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

## TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(List<ItemBase>)

```
public void Update(List<ItemBase> items)
```

Parameters

items [List](#)<ItemBase>

# Class GeneralQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GeneralQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← GeneralQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GeneralQuest(Dictionary)

```
public GeneralQuest(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### GeneralQuest(List)

```
public GeneralQuest(List serialized)
```

Parameters

**serialized** List

## GeneralQuest(Row, QuestReward)

```
public GeneralQuest(GeneralQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [GeneralQuestSheet.Row](#)

**reward** [QuestReward](#)

## Fields

### Event

```
public readonly QuestEventType Event
```

Field Value

[QuestEventType](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

# TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

Check()

```
public override void Check()
```

GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

Serialize()

```
public override IValue Serialize()
```

Returns

IValue

SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(CollectionMap)

```
public void Update(CollectionMap eventMap)
```

Parameters

eventMap [CollectionMap](#)

# Class GoldQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GoldQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← GoldQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GoldQuest(Dictionary)

```
public GoldQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### GoldQuest(List)

```
public GoldQuest(List serialized)
```

Parameters

**serialized** List

## GoldQuest(Row, QuestReward)

```
public GoldQuest(GoldQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [GoldQuestSheet.Row](#)

**reward** [QuestReward](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

### Type

```
public TradeType Type { get; }
```

Property Value

[TradeType](#)

### Typeld

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

### SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(FungibleAssetValue)

```
public void Update(FungibleAssetValue gold)
```

Parameters

**gold** FungibleAssetValue

# Class ItemEnhancementQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancementQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← ItemEnhancementQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.GoalFormat](#) , [Quest.Deserialize\(Dictionary\)](#) ,  
[Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ItemEnhancementQuest(Dictionary)

```
public ItemEnhancementQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ItemEnhancementQuest(List)

```
public ItemEnhancementQuest(List serialized)
```

Parameters

**serialized** List

## ItemEnhancementQuest(Row, QuestReward)

```
public ItemEnhancementQuest(ItemEnhancementQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [ItemEnhancementQuestSheet.Row](#)

**reward** [QuestReward](#)

## Properties

### Count

```
public int Count { get; }
```

Property Value

[int](#)

### Grade

```
public int Grade { get; }
```

Property Value

[int](#)

### Progress

```
public override float Progress { get; }
```

Property Value

[float](#)

## QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

## TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(Equipment)

```
public void Update(Equipment equipment)
```

Parameters

equipment [Equipment](#)

# Class ItemGradeQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemGradeQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← ItemGradeQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemGradeQuest(Dictionary)

```
public ItemGradeQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ItemGradeQuest(List)

```
public ItemGradeQuest(List serialized)
```

Parameters

**serialized** List

## ItemGradeQuest(Row, QuestReward)

```
public ItemGradeQuest(ItemGradeQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [ItemGradeQuestSheet.Row](#)

**reward** [QuestReward](#)

## Fields

### Grade

```
public readonly int Grade
```

Field Value

[int](#)

### ItemIds

```
public readonly List<int> ItemIds
```

Field Value

[List](#)<[int](#)>

## Properties

## QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

## TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(ItemUsable)

```
public void Update(ItemUsable itemUsable)
```

Parameters

itemUsable [ItemUsable](#)

# Class ItemTypeCollectQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemTypeCollectQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← [ItemTypeCollectQuest](#)

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemTypeCollectQuest(Dictionary)

```
public ItemTypeCollectQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ItemTypeCollectQuest(List)

```
public ItemTypeCollectQuest(List serialized)
```

Parameters

**serialized** List

## ItemTypeCollectQuest(Row, QuestReward)

```
public ItemTypeCollectQuest(ItemTypeCollectQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [ItemTypeCollectQuestSheet.Row](#)

**reward** [QuestReward](#)

## Fields

### ItemIds

```
public readonly List<int> ItemIds
```

Field Value

[List](#)<[int](#)>

### ItemType

```
public readonly ItemType ItemType
```

Field Value

[ItemType](#)

## Properties

## QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

## TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(ItemBase)

```
public void Update(ItemBase item)
```

Parameters

item [ItemBase](#)

# Class MonsterQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← MonsterQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### MonsterQuest(Dictionary)

```
public MonsterQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### MonsterQuest(List)

```
public MonsterQuest(List serialized)
```

Parameters

**serialized** List

## MonsterQuest(Row, QuestReward)

```
public MonsterQuest(MonsterQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [MonsterQuestSheet.Row](#)

**reward** [QuestReward](#)

## Fields

### MonsterId

```
public readonly int MonsterId
```

Field Value

[int](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

# TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

Check()

```
public override void Check()
```

GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

Serialize()

```
public override IValue Serialize()
```

Returns

IValue

SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

## Update(CollectionMap)

```
public void Update(CollectionMap monsterMap)
```

Parameters

monsterMap [CollectionMap](#)

# Class Quest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Quest : IState
```

## Inheritance

[object](#) ← Quest

## Implements

[IState](#)

## Derived

[CollectQuest](#), [CombinationEquipmentQuest](#), [CombinationQuest](#), [GeneralQuest](#), [GoldQuest](#),  
[ItemEnhancementQuest](#), [ItemGradeQuest](#), [ItemTypeCollectQuest](#), [MonsterQuest](#),  
[TradeQuest](#), [WorldQuest](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### Quest(Dictionary)

```
protected Quest(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Quest(List)

```
protected Quest(List serialized)
```

Parameters

serialized List

## Quest(Row, QuestReward)

```
protected Quest(QuestSheet.Row data, QuestReward reward)
```

Parameters

data [QuestSheet.Row](#)

reward [QuestReward](#)

## Fields

### GoalFormat

```
public const string GoalFormat = "({0}/{1})"
```

Field Value

[string](#)

### \_current

```
protected int _current
```

Field Value

[int](#)

## isReceivable

```
[NonSerialized]  
public bool isReceivable
```

### Field Value

[bool](#) ↗

## Properties

### Complete

```
public bool Complete { get; protected set; }
```

### Property Value

[bool](#) ↗

## Goal

```
public int Goal { get; }
```

### Property Value

[int](#) ↗

## Id

```
public int Id { get; }
```

### Property Value

[int](#) ↗

## IsPaidInAction

bool یعنی چه کاری می‌کند؟

```
public bool IsPaidInAction { get; set; }
```

Property Value

[bool](#)

## Progress

```
public virtual float Progress { get; }
```

Property Value

[float](#)

## QuestType

```
public abstract QuestType QuestType { get; }
```

Property Value

[QuestType](#)

## Reward

```
public QuestReward Reward { get; }
```

Property Value

[QuestReward](#)

# TypeId

```
protected abstract string TypeId { get; }
```

Property Value

[string](#)

## Methods

Check()

```
public abstract void Check()
```

Deserialize(Dictionary)

```
public static Quest Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[Quest](#)

Deserialize(IValue)

```
public static Quest Deserialize(IValue arg)
```

Parameters

**arg** IValue

Returns

[Quest](#)

## DeserializeList(List)

```
public static Quest DeserializeList(List serialized)
```

Parameters

**serialized** List

Returns

[Quest](#)

## GetProgressText()

```
public abstract string GetProgressText()
```

Returns

[string](#)

## Serialize()

```
public virtual IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public virtual IValue SerializeList()
```

Returns

IValue

# Enum QuestEventType

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
public enum QuestEventType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Complete = 4

Consumable = 6

Create = 0

Die = 3

Enhancement = 1

Equipment = 5

Level = 2

# Class QuestList

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestList : IEnumerable<Quest>, IEnumerable, IState, ICloneable
```

## Inheritance

[object](#) ← QuestList

## Implements

[IEnumerable](#)<[Quest](#)>, [IEnumerable](#), [IState](#), [ICloneable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### QuestList(Dictionary)

```
public QuestList(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### QuestList(List)

```
public QuestList(List serialized)
```

#### Parameters

serialized List

# QuestList(QuestSheet, QuestRewardSheet, QuestItemRewardSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheet)

```
public QuestList(QuestSheet questSheet, QuestRewardSheet questRewardSheet, QuestItemRewardSheet questItemRewardSheet, EquipmentItemRecipeSheet equipmentItemRecipeSheet, EquipmentItemSubRecipeSheet equipmentItemSubRecipeSheet)
```

## Parameters

questSheet [QuestSheet](#)

questRewardSheet [QuestRewardSheet](#)

questItemRewardSheet [QuestItemRewardSheet](#)

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

equipmentItemSubRecipeSheet [EquipmentItemSubRecipeSheet](#)

## Fields

### CompletedQuestIdsKey

```
public const string CompletedQuestIdsKey = "c"
```

## Field Value

[string](#)

### ListVersionKey

```
public const string ListVersionKey = "v"
```

## Field Value

[string](#)

## QuestsKey

```
public const string QuestsKey = "q"
```

### Field Value

[string](#)

## completedQuestIds

```
public List<int> completedQuestIds
```

### Field Value

[List](#)<[int](#)>

## Properties

### ListVersion

```
public int ListVersion { get; }
```

### Property Value

[int](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<Quest> GetEnumerator()
```

Returns

[IEnumerator](#)<[Quest](#)>

An enumerator that can be used to iterate through the collection.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## SerializeDictionary()

```
[Obsolete("Dictionary type is obsolete.")]  
public IValue SerializeDictionary()
```

Returns

IValue

## SerializeList()

```
public IValue SerializeList()
```

Returns

IValue

## UpdateCollectQuest(CollectionMap)

```
public void UpdateCollectQuest(CollectionMap itemMap)
```

Parameters

itemMap [CollectionMap](#)

## UpdateCombinationEquipmentQuest(int)

```
public void UpdateCombinationEquipmentQuest(int recipeId)
```

Parameters

recipeId [int](#)

## UpdateCombinationQuest(ItemUsable)

```
public void UpdateCombinationQuest(ItemUsable itemUsable)
```

Parameters

itemUsable [ItemUsable](#)

## UpdateCompletedQuest(CollectionMap)

```
public CollectionMap UpdateCompletedQuest(CollectionMap eventMap)
```

### Parameters

eventMap [CollectionMap](#)

### Returns

[CollectionMap](#)

## UpdateGeneralQuest(IEnumerable<QuestEventType>, CollectionMap)

```
public CollectionMap UpdateGeneralQuest(IEnumerable<QuestEventType> types, CollectionMap eventMap)
```

### Parameters

types [IEnumerable](#)<[QuestEventType](#)>

eventMap [CollectionMap](#)

### Returns

[CollectionMap](#)

## UpdateItemEnhancementQuest(Equipment)

```
public void UpdateItemEnhancementQuest(Equipment equipment)
```

### Parameters

equipment [Equipment](#)

## UpdateItemGradeQuest(ItemUsable)

```
public void UpdateItemGradeQuest(ItemUsable itemUsable)
```

### Parameters

itemUsable [ItemUsable](#)

## UpdateItemTypeCollectQuest(IEnumerable<ItemBase>)

```
public void UpdateItemTypeCollectQuest(IEnumerable<ItemBase> items)
```

### Parameters

items [IEnumerable](#)<ItemBase>

## UpdateList(QuestSheet, QuestRewardSheet, QuestItemRewardSheet, EquipmentItemRecipeSheet, ICollection<int>)

```
public void UpdateList(QuestSheet questSheet, QuestRewardSheet questRewardSheet,  
QuestItemRewardSheet questItemRewardSheet, EquipmentItemRecipeSheet  
equipmentItemRecipeSheet, ICollection<int> newIds)
```

### Parameters

questSheet [QuestSheet](#)

questRewardSheet [QuestRewardSheet](#)

questItemRewardSheet [QuestItemRewardSheet](#)

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

newIds [ICollection](#)<int>

## UpdateListV1(int, QuestSheet, QuestRewardSheet, QuestItemRewardSheet, EquipmentItemRecipeSheet)

```
[Obsolete("Use UpdateList())")]
public void UpdateListV1(int listVersion, QuestSheet questSheet, QuestRewardSheet
questRewardSheet, QuestItemRewardSheet questItemRewardSheet,
EquipmentItemRecipeSheet equipmentItemRecipeSheet)
```

### Parameters

listVersion [int](#)

questSheet [QuestSheet](#)

questRewardSheet [QuestRewardSheet](#)

questItemRewardSheet [QuestItemRewardSheet](#)

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

### Exceptions

[UpdateListVersionException](#)

[UpdateListQuestsCountException](#)

[Exception](#)

## UpdateMonsterQuest(CollectionMap)

```
public void UpdateMonsterQuest(CollectionMap monsterMap)
```

### Parameters

monsterMap [CollectionMap](#)

## UpdateStageQuest(CollectionMap)

```
public void UpdateStageQuest(CollectionMap stageMap)
```

Parameters

stageMap [CollectionMap](#)

## UpdateTradeQuest(TradeType, FungibleAssetValue)

```
public void UpdateTradeQuest(TradeType type, FungibleAssetValue price)
```

Parameters

type [TradeType](#)

price FungibleAssetValue

# Class QuestReward

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestReward : IState
```

## Inheritance

[object](#) ← QuestReward

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### QuestReward(Dictionary)

```
public QuestReward(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### QuestReward(Dictionary<int, int>)

```
public QuestReward(Dictionary<int, int> map)
```

#### Parameters

map [Dictionary](#)<[int](#), [int](#)>

# Fields

## ItemMap

```
public readonly IEnumerable<Tuple<int, int>> ItemMap
```

### Field Value

[IEnumerable<Tuple<int, int>>](#)

# Methods

## Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Enum QuestType

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
public enum QuestType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Adventure = 0

Craft = 2

Exchange = 3

Obtain = 1

# Class TradeQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TradeQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← TradeQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Deserialize\(Dictionary\)](#) , [Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### TradeQuest(Dictionary)

```
public TradeQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### TradeQuest(List)

```
public TradeQuest(List serialized)
```

Parameters

**serialized** List

## TradeQuest(TradeQuestSheet.Row data, QuestReward reward)

```
public TradeQuest(TradeQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [TradeQuestSheet.Row](#)

**reward** [QuestReward](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

### Type

```
public TradeType Type { get; }
```

Property Value

[TradeType](#)

### Typeld

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

### Check()

```
public override void Check()
```

### GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

### SerializeList()

```
public override IValue SerializeList()
```

Returns

IValue

# Class UpdateListQuestsCountException

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class UpdateListQuestsCountException : ArgumentException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [ArgumentException](#) ←  
UpdateListQuestsCountException

## Implements

[ISerializable](#)

## Inherited Members

[ArgumentException.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ArgumentException.Message](#) , [ArgumentException.ParamName](#) ,  
[Exception.GetBaseException\(\)](#) , [Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) ,  
[Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) , [Exception.InnerException](#) ,  
[Exception.Source](#) , [Exception.StackTrace](#) , [Exception.TargetSite](#) ,  
[Exception.SerializeObjectState](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### UpdateListQuestsCountException()

```
public UpdateListQuestsCountException()
```

### UpdateListQuestsCountException(int, int)

```
public UpdateListQuestsCountException(int expected, int actual)
```

Parameters

expected [int](#)

actual [int](#)

## UpdateListQuestsCountException(SerializationInfo, StreamingContext)

```
protected UpdateListQuestsCountException(SerializationInfo info,  
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## UpdateListQuestsCountException(string)

```
public UpdateListQuestsCountException(string s)
```

Parameters

s [string](#)

# Class UpdateListVersionException

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class UpdateListVersionException : ArgumentOutOfRangeException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [ArgumentException](#) ←  
[ArgumentOutOfRangeException](#) ← [UpdateListVersionException](#)

## Implements

[ISerializable](#)

## Inherited Members

[ArgumentOutOfRangeException.GetObjectData\(SerializationInfo, StreamingContext\)](#) ,  
[ArgumentOutOfRangeException.ActualValue](#) , [ArgumentOutOfRangeException.Message](#) ,  
[ArgumentException.ParamName](#) , [Exception.GetBaseException\(\)](#) ,  
[Exception.GetType\(\)](#) , [Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) ,  
[Exception.HResult](#) , [Exception.InnerException](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### UpdateListVersionException()

```
public UpdateListVersionException()
```

### UpdateListVersionException(int, int)

```
public UpdateListVersionException(int expected, int actual)
```

Parameters

expected [int](#)

actual [int](#)

## UpdateListVersionException(SerializationInfo, StreamingContext)

```
protected UpdateListVersionException(SerializationInfo info,  
StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## UpdateListVersionException(string)

```
public UpdateListVersionException(string s)
```

Parameters

s [string](#)

# Class WorldQuest

Namespace: [Nekoyume.Model.Quest](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldQuest : Quest, IState
```

## Inheritance

[object](#) ← [Quest](#) ← WorldQuest

## Implements

[IState](#)

## Inherited Members

[Quest.isReceivable](#) , [Quest.\\_current](#) , [Quest.Complete](#) , [Quest.Goal](#) , [Quest.Id](#) ,  
[Quest.Reward](#) , [Quest.IsPaidInAction](#) , [Quest.Progress](#) , [Quest.GoalFormat](#) ,  
[Quest.Serialize\(\)](#) , [Quest.SerializeList\(\)](#) , [Quest.Deserialize\(Dictionary\)](#) ,  
[Quest.DeserializeList\(List\)](#) , [Quest.Deserialize\(IValue\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### WorldQuest(Dictionary)

```
public WorldQuest(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### WorldQuest(List)

```
public WorldQuest(List serialized)
```

Parameters

**serialized** List

## WorldQuest(Row, QuestReward)

```
public WorldQuest(WorldQuestSheet.Row data, QuestReward reward)
```

Parameters

**data** [WorldQuestSheet.Row](#)

**reward** [QuestReward](#)

## Properties

### QuestType

```
public override QuestType QuestType { get; }
```

Property Value

[QuestType](#)

### TypeId

```
protected override string TypeId { get; }
```

Property Value

[string](#)

## Methods

## Check()

```
public override void Check()
```

## GetProgressText()

```
public override string GetProgressText()
```

Returns

[string](#)

## Update(CollectionMap)

```
public void Update(CollectionMap stageMap)
```

Parameters

stageMap [CollectionMap](#)

# Namespace Nekoyume.Model.Rune Classes

[DuplicatedRunIdException](#)

[DuplicatedRuneSlotIndexException](#)

[IsEquippableRuneException](#)

[IsUsableSlotException](#)

[MismatchRuneSlotTypeException](#)

[RuneCostDataNotFoundException](#)

[RuneCostNotFoundException](#)

[RuneInfoIsEmptyException](#)

[RuneListNotFoundException](#)

[RuneNotFoundException](#)

[RuneSlot](#)

[RuneStateNotFoundException](#)

[SlotIsAlreadyUnlockedException](#)

[SlotIsLockedException](#)

[SlotNotFoundException](#)

[SlotRuneTypeException](#)

[TryCountIsZeroException](#)

# Class DuplicatedRunIdException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicatedRunIdException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← DuplicatedRunIdException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### DuplicatedRunIdException(SerializationInfo, StreamingContext)

```
protected DuplicatedRunIdException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicatedRunIdException(string)

```
public DuplicatedRunIdException(string message)
```

### Parameters

message [string](#)

# Class DuplicatedRuneSlotIndexException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicatedRuneSlotIndexException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← DuplicatedRuneSlotIndexException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### DuplicatedRuneSlotIndexException(SerializationInfo, StreamingContext)

```
protected DuplicatedRuneSlotIndexException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicatedRuneSlotIndexException(string)

```
public DuplicatedRuneSlotIndexException(string message)
```

### Parameters

message [string](#)

# Class IsEquippableRuneException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class IsEquippableRuneException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← IsEquippableRuneException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### IsEquippableRuneException(SerializationInfo, StreamingContext)

```
protected IsEquippableRuneException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## IsEquippableRuneException(string)

```
public IsEquippableRuneException(string message)
```

### Parameters

message [string](#)

# Class IsUsableSlotException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class IsUsableSlotException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← IsUsableSlotException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### IsUsableSlotException(SerializationInfo, StreamingContext)

```
protected IsUsableSlotException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## IsUsableSlotException(string)

```
public IsUsableSlotException(string message)
```

### Parameters

message [string](#)

# Class MismatchRuneSlotTypeException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MismatchRuneSlotTypeException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← MismatchRuneSlotTypeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### MismatchRuneSlotTypeException(SerializationInfo, StreamingContext)

```
protected MismatchRuneSlotTypeException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## MismatchRuneSlotTypeException(string)

```
public MismatchRuneSlotTypeException(string message)
```

### Parameters

message [string](#)

# Class RuneCostDataNotFoundException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneCostDataNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RuneCostDataNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneCostDataNotFoundException(SerializationInfo, StreamingContext)

```
protected RuneCostDataNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RuneCostDataNotFoundException(string)

```
public RuneCostDataNotFoundException(string message)
```

### Parameters

message [string](#)

# Class RuneCostNotFoundException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneCostNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RuneCostNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneCostNotFoundException(SerializationInfo, StreamingContext)

```
protected RuneCostNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RuneCostNotFoundException(string)

```
public RuneCostNotFoundException(string message)
```

### Parameters

message [string](#)

# Class RuneInfosIsEmptyException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneInfosIsEmptyException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RuneInfosIsEmptyException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneInfosIsEmptyException(SerializationInfo, StreamingContext)

```
protected RuneInfosIsEmptyException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RuneInfosIsEmptyException(string)

```
public RuneInfosIsEmptyException(string message)
```

### Parameters

message [string](#)

# Class RuneListNotFoundException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneListNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RuneListNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneListNotFoundException(SerializationInfo, StreamingContext)

```
protected RuneListNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RuneListNotFoundException(string)

```
public RuneListNotFoundException(string message)
```

### Parameters

message [string](#)

# Class RuneNotFoundException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RuneNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneNotFoundException(SerializationInfo, StreamingContext)

```
protected RuneNotFoundException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RuneNotFoundException(string)

```
public RuneNotFoundException(string message)
```

### Parameters

message [string](#)

# Class RuneSlot

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
public class RuneSlot : IState
```

## Inheritance

[object](#) ← RuneSlot

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RuneSlot(List)

```
public RuneSlot(List serialized)
```

#### Parameters

serialized List

### RuneSlot(int, RuneSlotType, RuneType, bool)

```
public RuneSlot(int index, RuneSlotType runeSlotType, RuneType runeType,  
bool isLock)
```

#### Parameters

index [int](#)

runeSlotType [RuneSlotType](#)

runeType [RuneType](#)

isLock [bool](#) ↗

## Properties

### Index

`public int Index { get; }`

Property Value

[int](#) ↗

### IsLock

`public bool IsLock { get; }`

Property Value

[bool](#) ↗

### Runeld

`public int? RuneId { get; }`

Property Value

[int](#) ↗?

### RuneSlotType

```
public RuneSlotType RuneSlotType { get; }
```

Property Value

[RuneSlotType](#)

## RuneType

```
public RuneType RuneType { get; }
```

Property Value

[RuneType](#)

## Methods

### Equip(int)

```
public void Equip(int runeId)
```

Parameters

runeId [int](#)

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

### Unequip()

```
public void Unequip()
```

## Unlock()

```
public void Unlock()
```

# Class RuneStateNotFoundException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneStateNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← RuneStateNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneStateNotFoundException(SerializationInfo, StreamingContext)

```
protected RuneStateNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## RuneStateNotFoundException(string)

```
public RuneStateNotFoundException(string message)
```

### Parameters

message [string](#)

# Class SlotIsAlreadyUnlockedException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SlotIsAlreadyUnlockedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SlotIsAlreadyUnlockedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SlotIsAlreadyUnlockedException(SerializationInfo, StreamingContext)

```
protected SlotIsAlreadyUnlockedException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SlotIsAlreadyUnlockedException(string)

```
public SlotIsAlreadyUnlockedException(string message)
```

### Parameters

message [string](#)

# Class SlotIsLockedException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SlotIsLockedException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SlotIsLockedException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SlotIsLockedException(SerializationInfo, StreamingContext)

```
protected SlotIsLockedException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SlotIsLockedException(string)

```
public SlotIsLockedException(string message)
```

### Parameters

message [string](#)

# Class SlotNotFoundException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SlotNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SlotNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SlotNotFoundException(SerializationInfo, StreamingContext)

```
protected SlotNotFoundException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SlotNotFoundException(string)

```
public SlotNotFoundException(string message)
```

### Parameters

message [string](#)

# Class SlotRuneTypeException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SlotRuneTypeException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SlotRuneTypeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SlotRuneTypeException(SerializationInfo, StreamingContext)

```
protected SlotRuneTypeException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SlotRuneTypeException(string)

```
public SlotRuneTypeException(string message)
```

### Parameters

message [string](#)

# Class TryCountIsZeroException

Namespace: [Nekoyume.Model.Rune](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TryCountIsZeroException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← TryCountIsZeroException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### TryCountIsZeroException(SerializationInfo, StreamingContext)

```
public TryCountIsZeroException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## TryCountIsZeroException(string)

```
public TryCountIsZeroException(string msg)
```

### Parameters

msg [string](#)

# Namespace Nekoyume.Model.Skill

## Classes

[AreaAttack](#)

[AttackSkill](#)

[BlowAttack](#)

[BuffRemovalAttack](#)

[BuffSkill](#)

[DoubleAttack](#)

[HealSkill](#)

[NormalAttack](#)

[ShatterStrike](#)

[Skill](#)

[SkillFactory](#)

[SkillTargetTypeExtension](#)

[UnexpectedOperationException](#)

## Structs

[SkillCustomField](#)

## Interfaces

[ISkill](#)

## Enums

[ActionBuffType](#)

[SkillCategory](#)

[SkillTargetType](#)

[SkillType](#)

# Enum ActionBuffType

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public enum ActionBuffType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Bleed = 0

Dispel = 4

Focus = 3

IceShield = 5

Stun = 1

Vampiric = 2

# Class AreaAttack

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AreaAttack : AttackSkill, IState, ISkill
```

## Inheritance

[object](#) ↳ [Skill](#) ↳ [AttackSkill](#) ↳ [AreaAttack](#)

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[AttackSkill.ProcessDamage\(CharacterBase, int, bool, bool\)](#), [Skill.SkillRow](#), [Skill.Power](#),  
[Skill.Chance](#), [Skill.StatPowerRatio](#), [Skill.ReferencedStatType](#), [Skill.CustomField](#),  
[Skill.Equals\(Skill\)](#), [Skill.Equals\(object\)](#), [Skill.GetHashCode\(\)](#),  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#), [Skill.Update\(int, long, int\)](#),  
[Skill.IsBuff\(\)](#), [Skill.IsDebuff\(\)](#), [Skill.Serialize\(\)](#), [object.Equals\(object, object\)](#) ↳,  
[object.GetType\(\)](#) ↳, [object.MemberwiseClone\(\)](#) ↳, [object.ReferenceEquals\(object, object\)](#) ↳,  
[object.ToString\(\)](#) ↳

## Constructors

### AreaAttack(Row, long, int, int, StatType)

```
public AreaAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↳

chance [int](#) ↳

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class AttackSkill

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class AttackSkill : Skill, IState, ISkill
```

## Inheritance

[object](#) ↳ [Skill](#) ↳ AttackSkill

## Implements

[IState](#), [ISkill](#)

## Derived

[AreaAttack](#), [BlowAttack](#), [BuffRemovalAttack](#), [DoubleAttack](#), [NormalAttack](#), [ShatterStrike](#)

## Inherited Members

[Skill.SkillRow](#), [Skill.Power](#), [Skill.Chance](#), [Skill.StatPowerRatio](#), [Skill.ReferencedStatType](#),  
[Skill.CustomField](#), [Skill.Use\(CharacterBase, int, IEnumerable<Buff>, bool\)](#),  
[Skill.Equals\(Skill\)](#), [Skill.Equals\(object\)](#), [Skill.GetHashCode\(\)](#),  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#), [Skill.Update\(int, long, int\)](#),  
[Skill.IsBuff\(\)](#), [Skill.IsDebuff\(\)](#), [Skill.Serialize\(\)](#), [object.Equals\(object, object\)](#) ↳,  
[object.GetType\(\)](#) ↳, [object.MemberwiseClone\(\)](#) ↳, [object.ReferenceEquals\(object, object\)](#) ↳,  
[object.ToString\(\)](#) ↳

## Constructors

### AttackSkill(Row, long, int, int, StatType)

```
protected AttackSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↳

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### ProcessDamage(CharacterBase, int, bool, bool)

todo: 重新实现此方法以处理模拟波次。 todo: 完成此方法后，将此方法移到其他类中。 此方法已移除。

```
protected IEnumerable<Skill.SkillInfo> ProcessDamage(CharacterBase caster, int
simulatorWaveTurn, bool isNormalAttack = false, bool copyCharacter = true)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

isNormalAttack [bool](#)

copyCharacter [bool](#)

#### Returns

[IEnumerable](#)<[Skill.SkillInfo](#)>

# Class BlowAttack

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BlowAttack : AttackSkill, IState, ISkill
```

## Inheritance

[object](#) ↳ [Skill](#) ↳ [AttackSkill](#) ↳ [BlowAttack](#)

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[AttackSkill.ProcessDamage\(CharacterBase, int, bool, bool\)](#), [Skill.SkillRow](#), [Skill.Power](#),  
[Skill.Chance](#), [Skill.StatPowerRatio](#), [Skill.ReferencedStatType](#), [Skill.CustomField](#),  
[Skill.Equals\(Skill\)](#), [Skill.Equals\(object\)](#), [Skill.GetHashCode\(\)](#),  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#), [Skill.Update\(int, long, int\)](#),  
[Skill.IsBuff\(\)](#), [Skill.IsDebuff\(\)](#), [Skill.Serialize\(\)](#), [object.Equals\(object, object\)](#) ↳,  
[object.GetType\(\)](#) ↳, [object.MemberwiseClone\(\)](#) ↳, [object.ReferenceEquals\(object, object\)](#) ↳,  
[object.ToString\(\)](#) ↳

## Constructors

### BlowAttack(Row, long, int, int, StatType)

```
public BlowAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↳

chance [int](#) ↳

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class BuffRemovalAttack

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuffRemovalAttack : AttackSkill, IState, ISkill
```

## Inheritance

[object](#) ↗ ← [Skill](#) ← [AttackSkill](#) ← BuffRemovalAttack

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[AttackSkill.ProcessDamage\(CharacterBase, int, bool, bool\)](#), [Skill.SkillRow](#), [Skill.Power](#),  
[Skill.Chance](#), [Skill.StatPowerRatio](#), [Skill.ReferencedStatType](#), [Skill.CustomField](#),  
[Skill.Equals\(Skill\)](#), [Skill.Equals\(object\)](#), [Skill.GetHashCode\(\)](#),  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#), [Skill.Update\(int, long, int\)](#),  
[Skill.IsBuff\(\)](#), [Skill.IsDebuff\(\)](#), [Skill.Serialize\(\)](#), [object.Equals\(object, object\)](#) ↗,  
[object.GetType\(\)](#) ↗, [object.MemberwiseClone\(\)](#) ↗, [object.ReferenceEquals\(object, object\)](#) ↗,  
[object.ToString\(\)](#) ↗

## Constructors

### BuffRemovalAttack(Row, long, int, int, StatType)

```
public BuffRemovalAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↗

chance [int](#) ↗

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class BuffSkill

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuffSkill : Skill, IState, ISkill
```

## Inheritance

[object](#) ← [Skill](#) ← BuffSkill

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[Skill.SkillRow](#) , [Skill.Power](#) , [Skill.Chance](#) , [Skill.StatPowerRatio](#) , [Skill.ReferencedStatType](#) ,  
[Skill.CustomField](#) , [Skill.Equals\(Skill\)](#) , [Skill.Equals\(object\)](#) , [Skill.GetHashCode\(\)](#) ,  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#) , [Skill.Update\(int, long, int\)](#) ,  
[Skill.IsBuff\(\)](#) , [Skill.IsDebuff\(\)](#) , [Skill.Serialize\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### BuffSkill(Row, long, int, int, StatType)

```
public BuffSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class DoubleAttack

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DoubleAttack : AttackSkill, IState, ISkill
```

## Inheritance

[object](#) ↳ [Skill](#) ↳ [AttackSkill](#) ↳ DoubleAttack

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[AttackSkill.ProcessDamage\(CharacterBase, int, bool, bool\)](#), [Skill.SkillRow](#), [Skill.Power](#),  
[Skill.Chance](#), [Skill.StatPowerRatio](#), [Skill.ReferencedStatType](#), [Skill.CustomField](#),  
[Skill.Equals\(Skill\)](#), [Skill.Equals\(object\)](#), [Skill.GetHashCode\(\)](#),  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#), [Skill.Update\(int, long, int\)](#),  
[Skill.IsBuff\(\)](#), [Skill.IsDebuff\(\)](#), [Skill.Serialize\(\)](#), [object.Equals\(object, object\)](#) ↳,  
[object.GetType\(\)](#) ↳, [object.MemberwiseClone\(\)](#) ↳, [object.ReferenceEquals\(object, object\)](#) ↳,  
[object.ToString\(\)](#) ↳

## Constructors

### DoubleAttack(Row, long, int, int, StatType)

```
public DoubleAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↳

chance [int](#) ↳

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class HealSkill

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class HealSkill : Skill, IState, ISkill
```

## Inheritance

[object](#) ← [Skill](#) ← HealSkill

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[Skill.SkillRow](#) , [Skill.Power](#) , [Skill.Chance](#) , [Skill.StatPowerRatio](#) , [Skill.ReferencedStatType](#) ,  
[Skill.CustomField](#) , [Skill.Equals\(Skill\)](#) , [Skill.Equals\(object\)](#) , [Skill.GetHashCode\(\)](#) ,  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#) , [Skill.Update\(int, long, int\)](#) ,  
[Skill.IsBuff\(\)](#) , [Skill.IsDebuff\(\)](#) , [Skill.Serialize\(\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### HealSkill(Row, long, int, int, StatType)

```
public HealSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### ProcessHeal(CharacterBase, int, bool)

```
protected IEnumerable<Skill.SkillInfo> ProcessHeal(CharacterBase caster, int
simulatorWaveTurn, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

copyCharacter [bool](#)

#### Returns

[IEnumerable](#)<[Skill.SkillInfo](#)>

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<[Buff](#)>

copyCharacter [bool](#)

#### Returns

[Skill](#)



# Interface ISkill

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public interface ISkill
```

## Properties

### Chance

```
int Chance { get; }
```

#### Property Value

[int](#)

### CustomField

```
SkillCustomField? CustomField { get; }
```

#### Property Value

[SkillCustomField?](#)

### Power

Determines damage of [AttackSkill](#). Determines effect of [BuffSkill](#).

```
long Power { get; }
```

#### Property Value

[long](#) ↴

## ReferencedStatType

StatType ReferencedStatType { [get](#); }

Property Value

[StatType](#)

## SkillRow

SkillSheet.Row SkillRow { [get](#); }

Property Value

[SkillSheet.Row](#)

## StatPowerRatio

[int](#) StatPowerRatio { [get](#); }

Property Value

[int](#) ↴

## Methods

### IsBuff()

[bool](#) [IsBuff](#)()

Returns

[bool](#)

## IsDebuff()

`bool IsDebuff()`

Returns

[bool](#)

## Update(int, long, int)

`void Update(int chance, long power, int statPowerRatio)`

Parameters

chance [int](#)

power [long](#)

statPowerRatio [int](#)

# Class NormalAttack

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class NormalAttack : AttackSkill, IState, ISkill
```

## Inheritance

[object](#) ↳ [Skill](#) ↳ [AttackSkill](#) ↳ NormalAttack

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[AttackSkill.ProcessDamage\(CharacterBase, int, bool, bool\)](#), [Skill.SkillRow](#), [Skill.Power](#),  
[Skill.Chance](#), [Skill.StatPowerRatio](#), [Skill.ReferencedStatType](#), [Skill.CustomField](#),  
[Skill.Equals\(Skill\)](#), [Skill.Equals\(object\)](#), [Skill.GetHashCode\(\)](#),  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#), [Skill.Update\(int, long, int\)](#),  
[Skill.IsBuff\(\)](#), [Skill.IsDebuff\(\)](#), [Skill.Serialize\(\)](#), [object.Equals\(object, object\)](#) ↳,  
[object.GetType\(\)](#) ↳, [object.MemberwiseClone\(\)](#) ↳, [object.ReferenceEquals\(object, object\)](#) ↳,  
[object.ToString\(\)](#) ↳

## Constructors

### NormalAttack(Row, long, int, int, StatType)

```
public NormalAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↳

chance [int](#) ↳

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class ShatterStrike

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ShatterStrike : AttackSkill, IState, ISkill
```

## Inheritance

[object](#) ↳ [Skill](#) ↳ [AttackSkill](#) ↳ ShatterStrike

## Implements

[IState](#), [ISkill](#)

## Inherited Members

[AttackSkill.ProcessDamage\(CharacterBase, int, bool, bool\)](#) , [Skill.SkillRow](#) , [Skill.Power](#) ,  
[Skill.Chance](#) , [Skill.StatPowerRatio](#) , [Skill.ReferencedStatType](#) , [Skill.CustomField](#) ,  
[Skill.Equals\(Skill\)](#) , [Skill.Equals\(object\)](#) , [Skill.GetHashCode\(\)](#) ,  
[Skill.ProcessBuff\(CharacterBase, int, IEnumerable<Buff>, bool\)](#) , [Skill.Update\(int, long, int\)](#) ,  
[Skill.IsBuff\(\)](#) , [Skill.IsDebuff\(\)](#) , [Skill.Serialize\(\)](#) , [object.Equals\(object, object\)](#) ↳ ,  
[object.GetType\(\)](#) ↳ , [object.MemberwiseClone\(\)](#) ↳ , [object.ReferenceEquals\(object, object\)](#) ↳ ,  
[object.ToString\(\)](#) ↳

## Constructors

### ShatterStrike(Row, long, int, int, StatType)

```
public ShatterStrike(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#) ↳

chance [int](#) ↳

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

### Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public override Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

#### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

#### Returns

[Skill](#)

# Class Skill

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Skill : IState, ISkill
```

## Inheritance

[object](#) ← Skill

## Implements

[IState](#), [ISkill](#)

## Derived

[AttackSkill](#), [BuffSkill](#), [HealSkill](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## Skill(Row, long, int, int, StatType)

```
protected Skill(SkillSheet.Row skillRow, long power, int chance, int statPowerRatio,  
StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Properties

### Chance

```
public int Chance { get; }
```

#### Property Value

[int](#)

### CustomField

```
public SkillCustomField? CustomField { get; set; }
```

#### Property Value

[SkillCustomField?](#)

### Power

Determines damage of [AttackSkill](#). Determines effect of [BuffSkill](#).

```
public long Power { get; }
```

#### Property Value

[long](#)

### ReferencedStatType

```
public StatType ReferencedStatType { get; }
```

Property Value

[StatType](#)

## SkillRow

```
public SkillSheet.Row SkillRow { get; }
```

Property Value

[SkillSheet.Row](#)

## StatPowerRatio

```
public int StatPowerRatio { get; }
```

Property Value

[int](#)

## Methods

### Equals(Skill)

```
protected bool Equals(Skill other)
```

Parameters

`other` [Skill](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## IsBuff()

```
public bool IsBuff()
```

Returns

[bool](#)

## IsDebuff()

```
public bool IsDebuff()
```

Returns

[bool](#)

## ProcessBuff(CharacterBase, int, IEnumerable<Buff>, bool)

```
protected IEnumerable<Skill.SkillInfo> ProcessBuff(CharacterBase caster, int
simulatorWaveTurn, IEnumerable<Buff> buffs, bool copyCharacter)
```

Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<Buff>

copyCharacter [bool](#)

Returns

[IEnumerable](#)<Skill.SkillInfo>

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Update(int, long, int)

```
public void Update(int chance, long power, int statPowerRatio)
```

### Parameters

chance [int](#)

power [long](#)

statPowerRatio [int](#)

## Use(CharacterBase, int, IEnumerable<Buff>, bool)

```
public abstract Skill Use(CharacterBase caster, int simulatorWaveTurn,  
IEnumerable<Buff> buffs, bool copyCharacter)
```

### Parameters

caster [CharacterBase](#)

simulatorWaveTurn [int](#)

buffs [IEnumerable](#)<[Buff](#)>

copyCharacter [bool](#)

### Returns

[Skill](#)

# Enum SkillCategory

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public enum SkillCategory
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

AreaAttack = 3

AttackBuff = 8

BlowAttack = 1

Buff = 15

BuffRemovalAttack = 4

CriticalBuff = 10

CriticalDamageBuff = 14

DamageReductionBuff = 13

Debuff = 16

DefenseBuff = 9

Dispel = 19

DoubleAttack = 2

Focus = 18

HPPuff = 7

Heal = 6

HitBuff = 11

NormalAttack = 0

ShatterStrike = 5

SpeedBuff = 12

TickDamage = 17

# Struct SkillCustomField

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public struct SkillCustomField
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Properties

### BuffDuration

```
public int BuffDuration { readonly get; set; }
```

#### Property Value

[int](#)

### BuffValue

```
public long BuffValue { readonly get; set; }
```

#### Property Value

[long](#)

# Class SkillFactory

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public static class SkillFactory
```

## Inheritance

[object](#) ← SkillFactory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Deserialize(Dictionary)

```
public static Skill Deserialize(Dictionary serialized)
```

#### Parameters

serialized Dictionary

#### Returns

[Skill](#)

### Get(Row, long, int, int, StatType)

```
public static Skill Get(SkillSheet.Row skillRow, long power, int chance, int  
statPowerRatio, StatType referencedStatType)
```

#### Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

Returns

[Skill](#)

## GetForArena(Row, long, int, int, StatType)

```
public static ArenaSkill GetForArena(SkillSheet.Row skillRow, long power, int chance, int statPowerRatio, StatType referencedStatType)
```

Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

Returns

[ArenaSkill](#)

## GetV1(Row, long, int)

```
[Obsolete("Use Get() instead.")]  
public static Skill GetV1(SkillSheet.Row skillRow, long power, int chance)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

## Returns

[Skill](#)

**SelectSkill(ItemSubType, CustomEquipmentCraftRecipeSkillSheet, EquipmentItemOptionSheet, SkillSheet, IRandom)**

```
public static Skill SelectSkill(ItemSubType itemSubType,  
CustomEquipmentCraftRecipeSkillSheet recipeSkillSheet, EquipmentItemOptionSheet  
itemOptionSheet, SkillSheet skillSheet, IRandom random)
```

## Parameters

itemSubType [ItemSubType](#)

recipeSkillSheet [CustomEquipmentCraftRecipeSkillSheet](#)

itemOptionSheet [EquipmentItemOptionSheet](#)

skillSheet [SkillSheet](#)

random [IRandom](#)

## Returns

[Skill](#)

# Enum SkillTargetType

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public enum SkillTargetType
```

## Extension Methods

[SkillTargetTypeExtension.GetTarget\(SkillTargetType, CharacterBase\)](#) ,  
[StateExtensions.Serialize\(Enum\)](#).

## Fields

Ally = 3

Enemies = 1

Enemy = 0

Self = 2

# Class SkillTargetTypeExtension

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public static class SkillTargetTypeExtension
```

## Inheritance

[object](#) ← SkillTargetTypeExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetTarget(SkillTargetType, CharacterBase)

```
public static IEnumerable<CharacterBase> GetTarget(this SkillTargetType value,  
CharacterBase caster)
```

#### Parameters

value [SkillTargetType](#)

caster [CharacterBase](#)

#### Returns

[IEnumerable](#)<[CharacterBase](#)>

# Enum SkillType

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public enum SkillType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Attack = 0

Buff = 2

Debuff = 3

Heal = 1

# Class UnexpectedOperationException

Namespace: [Nekoyume.Model.Skill](#)

Assembly: Lib9c.dll

```
public class UnexpectedOperationException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← UnexpectedOperationException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### UnexpectedOperationException(string)

```
public UnexpectedOperationException(string message)
```

## Parameters

message [string](#)

# Namespace Nekoyume.Model.Skill.Arena

## Classes

[ArenaAreaAttack](#)

[ArenaAttackSkill](#)

[ArenaBlowAttack](#)

[ArenaBuffRemovalAttack](#)

[ArenaBuffSkill](#)

[ArenaDoubleAttack](#)

[ArenaHealSkill](#)

[ArenaNormalAttack](#)

[ArenaShatterStrike](#)

[ArenaSkill](#)

# Class ArenaAreaAttack

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaAreaAttack : ArenaAttackSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← [ArenaAttackSkill](#) ← [ArenaAreaAttack](#)

## Implements

[ISkill](#)

## Inherited Members

[ArenaAttackSkill.ProcessDamage\(ArenaCharacter, ArenaCharacter, int, bool\)](#) ,  
[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaAreaAttack(Row, long, int, int, StatType)

```
public ArenaAreaAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaAttackSkill

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ArenaAttackSkill : ArenaSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← ArenaAttackSkill

## Implements

[ISkill](#)

## Derived

[ArenaAreaAttack](#), [ArenaBlowAttack](#), [ArenaBuffRemovalAttack](#), [ArenaDoubleAttack](#),  
[ArenaNormalAttack](#), [ArenaShatterStrike](#)

## Inherited Members

[ArenaSkill.SkillRow](#), [ArenaSkill.Power](#), [ArenaSkill.Chance](#), [ArenaSkill.StatPowerRatio](#),  
[ArenaSkill.ReferencedStatType](#), [ArenaSkill.CustomField](#),  
[ArenaSkill.Use\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#),  
[ArenaSkill.Equals\(Skill\)](#), [ArenaSkill.Equals\(object\)](#), [ArenaSkill.GetHashCode\(\)](#),  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#),  
[ArenaSkill.Update\(int, long, int\)](#), [ArenaSkill.IsBuff\(\)](#), [ArenaSkill.IsDebuff\(\)](#),  
[object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

### ArenaAttackSkill(Row, long, int, int, StatType)

```
protected ArenaAttackSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

ProcessDamage(ArenaCharacter, ArenaCharacter, int, bool)

```
protected IEnumerable<ArenaSkill.ArenaSkillInfo> ProcessDamage(ArenaCharacter  
caster, ArenaCharacter target, int simulatorWaveTurn, bool isNormalAttack = false)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

simulatorWaveTurn [int](#)

isNormalAttack [bool](#)

### Returns

[IEnumerable](#)<[ArenaSkill.ArenaSkillInfo](#)>

# Class ArenaBlowAttack

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaBlowAttack : ArenaAttackSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← [ArenaAttackSkill](#) ← [ArenaBlowAttack](#)

## Implements

[ISkill](#)

## Inherited Members

[ArenaAttackSkill.ProcessDamage\(ArenaCharacter, ArenaCharacter, int, bool\)](#) ,  
[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaBlowAttack(Row, long, int, int, StatType)

```
public ArenaBlowAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaBuffRemovalAttack

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaBuffRemovalAttack : ArenaAttackSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← [ArenaAttackSkill](#) ← ArenaBuffRemovalAttack

## Implements

[ISkill](#)

## Inherited Members

[ArenaAttackSkill.ProcessDamage\(ArenaCharacter, ArenaCharacter, int, bool\)](#) ,  
[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaBuffRemovalAttack(Row, long, int, int, StatType)

```
public ArenaBuffRemovalAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaBuffSkill

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaBuffSkill : ArenaSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← ArenaBuffSkill

## Implements

[ISkill](#)

## Inherited Members

[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaBuffSkill(Row, long, int, int, StatType)

```
public ArenaBuffSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaDoubleAttack

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaDoubleAttack : ArenaAttackSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← [ArenaAttackSkill](#) ← ArenaDoubleAttack

## Implements

[ISkill](#)

## Inherited Members

[ArenaAttackSkill.ProcessDamage\(ArenaCharacter, ArenaCharacter, int, bool\)](#) ,  
[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaDoubleAttack(Row, long, int, int, StatType)

```
public ArenaDoubleAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaHealSkill

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaHealSkill : ArenaSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← ArenaHealSkill

## Implements

[ISkill](#)

## Inherited Members

[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaHealSkill(Row, long, int, int, StatType)

```
public ArenaHealSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaNormalAttack

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaNormalAttack : ArenaAttackSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← [ArenaAttackSkill](#) ← [ArenaNormalAttack](#)

## Implements

[ISkill](#)

## Inherited Members

[ArenaAttackSkill.ProcessDamage\(ArenaCharacter, ArenaCharacter, int, bool\)](#) ,  
[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaNormalAttack(Row, long, int, int, StatType)

```
public ArenaNormalAttack(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaShatterStrike

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaShatterStrike : ArenaAttackSkill, ISkill
```

## Inheritance

[object](#) ← [ArenaSkill](#) ← [ArenaAttackSkill](#) ← ArenaShatterStrike

## Implements

[ISkill](#)

## Inherited Members

[ArenaAttackSkill.ProcessDamage\(ArenaCharacter, ArenaCharacter, int, bool\)](#) ,  
[ArenaSkill.SkillRow](#) , [ArenaSkill.Power](#) , [ArenaSkill.Chance](#) , [ArenaSkill.StatPowerRatio](#) ,  
[ArenaSkill.ReferencedStatType](#) , [ArenaSkill.CustomField](#) , [ArenaSkill.Equals\(Skill\)](#) ,  
[ArenaSkill.Equals\(object\)](#) , [ArenaSkill.GetHashCode\(\)](#) ,  
[ArenaSkill.ProcessBuff\(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>\)](#) ,  
[ArenaSkill.Update\(int, long, int\)](#) , [ArenaSkill.IsBuff\(\)](#) , [ArenaSkill.IsDebuff\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaShatterStrike(Row, long, int, int, StatType)

```
public ArenaShatterStrike(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Methods

Use(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
public override ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

### Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

### Returns

[ArenaSkill](#)

# Class ArenaSkill

Namespace: [Nekoyume.Model.Skill.Arena](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ArenaSkill : ISkill
```

## Inheritance

[object](#) ← ArenaSkill

## Implements

[ISkill](#)

## Derived

[ArenaAttackSkill](#), [ArenaBuffSkill](#), [ArenaHealSkill](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ArenaSkill(Row, long, int, int, StatType)

```
protected ArenaSkill(SkillSheet.Row skillRow, long power, int chance, int
statPowerRatio, StatType referencedStatType)
```

## Parameters

skillRow [SkillSheet.Row](#)

power [long](#)

chance [int](#)

statPowerRatio [int](#)

referencedStatType [StatType](#)

## Properties

### Chance

```
public int Chance { get; }
```

#### Property Value

[int](#)

### CustomField

```
public SkillCustomField? CustomField { get; set; }
```

#### Property Value

[SkillCustomField?](#)

### Power

Determines damage of [AttackSkill](#). Determines effect of [BuffSkill](#).

```
public long Power { get; }
```

#### Property Value

[long](#)

### ReferencedStatType

```
public StatType ReferencedStatType { get; }
```

Property Value

[StatType](#)

## SkillRow

```
public SkillSheet.Row SkillRow { get; }
```

Property Value

[SkillSheet.Row](#)

## StatPowerRatio

```
public int StatPowerRatio { get; }
```

Property Value

[int](#)

## Methods

### Equals(Skill)

```
protected bool Equals(Skill other)
```

Parameters

`other` [Skill](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## IsBuff()

```
public bool IsBuff()
```

Returns

[bool](#)

## IsDebuff()

```
public bool IsDebuff()
```

Returns

[bool](#)

## ProcessBuff(ArenaCharacter, ArenaCharacter, int, IEnumerable<Buff>)

```
protected IEnumerable<ArenaSkill.ArenaSkillInfo> ProcessBuff(ArenaCharacter caster,  
ArenaCharacter target, int turn, IEnumerable<Buff> buffs)
```

Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

Returns

[IEnumerable](#)<ArenaSkill.ArenaSkillInfo>

## Update(int, long, int)

```
public void Update(int chance, long power, int statPowerRatio)
```

Parameters

chance [int](#)

power [long](#)

statPowerRatio [int](#)

Use(ArenaCharacter, ArenaCharacter, int,  
IEnumerable<Buff>)

```
public abstract ArenaSkill Use(ArenaCharacter caster, ArenaCharacter target, int  
turn, IEnumerable<Buff> buffs)
```

Parameters

caster [ArenaCharacter](#)

target [ArenaCharacter](#)

turn [int](#)

buffs [IEnumerable](#)<Buff>

Returns

[ArenaSkill](#)

# Namespace Nekoyume.Model.Stake

## Classes

[Contract](#)

[StakeStateUtils](#)

## Structs

[StakeState](#)

# Class Contract

Namespace: [Nekoyume.Model.Stake](#)

Assembly: Lib9c.dll

```
public class Contract
```

## Inheritance

[object](#) ← Contract

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Contract(IValue)

```
public Contract(IValue serialized)
```

#### Parameters

serialized IValue

### Contract(StakePolicySheet)

```
public Contract(StakePolicySheet stakePolicySheet)
```

#### Parameters

stakePolicySheet [StakePolicySheet](#)

### Contract(string, string, long, long)

```
public Contract(string stakeRegularFixedRewardSheetTableName, string  
stakeRegularRewardSheetTableName, long rewardInterval, long lockupInterval)
```

## Parameters

stakeRegularFixedRewardSheetTableName [string](#)

stakeRegularRewardSheetTableName [string](#)

rewardInterval [long](#)

lockupInterval [long](#)

## Fields

### StakeRegularFixedRewardSheetPrefix

```
public const string StakeRegularFixedRewardSheetPrefix =  
"StakeRegularFixedRewardSheet_"
```

#### Field Value

[string](#)

### StakeRegularRewardSheetPrefix

```
public const string StakeRegularRewardSheetPrefix = "StakeRegularRewardSheet_"
```

#### Field Value

[string](#)

### StateTypeName

```
public const string StateTypeName = "stake_contract"
```

Field Value

[string](#) ↗

## StateTypeVersion

```
public const long StateTypeVersion = 1
```

Field Value

[long](#) ↗

## Properties

### LockupInterval

```
[Obsolete("Not used because of guild system")]
public long LockupInterval { get; }
```

Property Value

[long](#) ↗

### RewardInterval

```
public long RewardInterval { get; }
```

Property Value

[long](#) ↗

### StakeRegularFixedRewardSheetTableName

```
public string StakeRegularFixedRewardSheetTableName { get; }
```

Property Value

[string](#)

## StakeRegularRewardSheetTableName

```
public string StakeRegularRewardSheetTableName { get; }
```

Property Value

[string](#)

## Methods

### Equals(Contract)

```
protected bool Equals(Contract other)
```

Parameters

other [Contract](#)

Returns

[bool](#)

### Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public List Serialize()
```

Returns

List

# Struct StakeState

Namespace: [Nekoyume.Model.Stake](#)

Assembly: Lib9c.dll

```
public readonly struct StakeState : IState
```

## Implements

[IState](#)

## Inherited Members

[ValueType.ToString\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### StakeState(IValue)

```
public StakeState(IValue serialized)
```

#### Parameters

serialized IValue

### StakeState(Contract, long, long, int)

```
public StakeState(Contract contract, long startedBlockIndex, long receivedBlockIndex  
= 0, int stateVersion = 3)
```

#### Parameters

contract [Contract](#)

startedBlockIndex [long](#)

receivedBlockIndex [long](#)

stateVersion [int](#)

## StakeState(LegacyStakeState, Contract)

public StakeState(LegacyStakeState legacyStakeState, Contract contract)

### Parameters

legacyStakeState [LegacyStakeState](#)

contract [Contract](#)

### Fields

#### Contract

public readonly Contract Contract

### Field Value

[Contract](#)

#### LatestStateTypeVersion

public const int LatestStateTypeVersion = 3

### Field Value

[int](#)

#### ReceivedBlockIndex

public readonly long ReceivedBlockIndex

Field Value

[long](#) ↗

## StartedBlockIndex

```
public readonly long StartedBlockIndex
```

Field Value

[long](#) ↗

## StateTypeName

```
public const string StateTypeName = "stake_state"
```

Field Value

[string](#) ↗

## StateVersion

```
public readonly int StateVersion
```

Field Value

[int](#) ↗

## Properties

### CancellableBlockIndex

```
public long CancellableBlockIndex { get; }
```

Property Value

[long](#) ↗

## ClaimableBlockIndex

```
public long ClaimableBlockIndex { get; }
```

Property Value

[long](#) ↗

## ClaimedBlockIndex

```
public long ClaimedBlockIndex { get; }
```

Property Value

[long](#) ↗

## Methods

### DeriveAddress(Address)

```
public static Address DeriveAddress(Address address)
```

Parameters

**address** Address

Returns

Address

## Equals(StakeState)

```
public bool Equals(StakeState other)
```

Parameters

other [StakeState](#)

Returns

[bool](#)

## Equals(object)

Indicates whether this instance and a specified object are equal.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current instance.

Returns

[bool](#)

[true](#) if [obj](#) and this instance are the same type and represent the same value; otherwise, [false](#).

## GetHashCode()

Returns the hash code for this instance.

```
public override int GetHashCode()
```

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Operators

### operator ==(StakeState, StakeState)

```
public static bool operator ==(StakeState left, StakeState right)
```

Parameters

left [StakeState](#)

right [StakeState](#)

Returns

[bool](#)

### operator !=(StakeState, StakeState)

```
public static bool operator !=(StakeState left, StakeState right)
```

Parameters

`left` [StakeState](#)

`right` [StakeState](#)

Returns

[bool](#) ↗

# Class StakeStateUtils

Namespace: [Nekoyume.Model.Stake](#)

Assembly: Lib9c.dll

```
public static class StakeStateUtils
```

## Inheritance

[object](#) ← StakeStateUtils

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### MigrateV1ToV2(IValue, GameConfigState)

```
public static StakeState? MigrateV1ToV2(IValue serialized,  
GameConfigState gameConfigState)
```

#### Parameters

serialized IValue

gameConfigState [GameConfigState](#)

#### Returns

[StakeState?](#)

### TryMigrateV1ToV2(IValue, GameConfigState, out StakeState)

```
public static bool TryMigrateV1ToV2(IValue serialized, GameConfigState gameConfigState, out StakeState stakeState)
```

## Parameters

serialized [IValue](#)

gameConfigState [GameConfigState](#)

stakeState [StakeState](#)

## Returns

[bool](#) ↗

## TryMigrateV1ToV2(IWorldState, Address, out StakeState)

```
public static bool TryMigrateV1ToV2(IWorldState state, Address stakeStateAddr, out StakeState stakeState)
```

## Parameters

state [IWorldState](#)

stakeStateAddr [Address](#)

stakeState [StakeState](#)

## Returns

[bool](#) ↗

## TryMigrateV2ToV3(IActionContext, IWorld, Address, StakeState, out (IWorld world, StakeState newStakeState)?)

```
public static bool TryMigrateV2ToV3(IActionContext context, IWorld world, Address stakeStateAddr, StakeState stakeState, out (IWorld world, StakeState newStakeState)?)
```

```
newStakeState)? result)
```

## Parameters

`context` `IActionContext`

`world` `IWorld`

`stakeStateAddr` `Address`

`stakeState` [StakeState](#)

`result` (`IWorld` [world](#), [StakeState newStakeState](#))?

## Returns

[bool](#)

# Namespace Nekoyume.Model.Stat

## Classes

### [CharacterStats](#)

Stat is built with \_baseStats based on level, \_equipmentStats based on equipments, \_consumableStats based on consumables, \_buffStats based on buffs and \_optionalStats for runes, etc... Stat of character is built with total of these stats.

### [DecimalStat](#)

### [StatMap](#)

### [StatModifier](#)

### [StatTypeComparer](#)

### [StatTypeExtension](#)

### [Stats](#)

### [StatsMap](#)

## Interfaces

### [IBaseAndAdditionalStats](#)

### [IStats](#)

## Enums

### [StatModifier.OperationType](#)

### [StatType](#)

# Class CharacterStats

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

Stat is built with \_baseStats based on level, \_equipmentStats based on equipments, \_consumableStats based on consumables, \_buffStats based on buffs and \_optionalStats for runes, etc... Stat of character is built with total of these stats.

```
[Serializable]
public class CharacterStats : Stats, IStats, IBaseAndAdditionalStats, ICloneable
```

## Inheritance

[object](#) ↗ ← [Stats](#) ← CharacterStats

## Implements

[IStats](#), [IBaseAndAdditionalStats](#), [ICloneable](#) ↗

## Inherited Members

[Stats.\\_statMap](#), [Stats.HP](#), [Stats.ATK](#), [Stats.DEF](#), [Stats.CRI](#), [Stats.HIT](#), [Stats.SPD](#),  
[Stats.DRV](#), [Stats.DRR](#), [Stats.CDMG](#), [Stats.ArmorPenetration](#), [Stats.Thorn](#),  
[Stats.LegacyDecimalStatTypes](#), [Stats.Reset\(\)](#), [Stats.Set\(StatMap, params Stats\[\]\)](#),  
[Stats.Set\(StatsMap\)](#), [Stats.Modify\(IEnumerable<StatModifier>\)](#),  
[Stats.Set\(IEnumerable<StatModifier>, params Stats\[\]\)](#), [Stats.GetStatAsLong\(StatType\)](#),  
[Stats.GetStat\(StatType\)](#), [Stats.SetStatForTest\(StatType, decimal\)](#), [Stats.GetStats\(bool\)](#),  
[object.Equals\(object\)](#) ↗, [object.Equals\(object, object\)](#) ↗, [object.GetHashCode\(\)](#) ↗,  
[object.GetType\(\)](#) ↗, [object.MemberwiseClone\(\)](#) ↗, [object.ReferenceEquals\(object, object\)](#) ↗,  
[object.ToString\(\)](#) ↗

## Constructors

### CharacterStats(CharacterStats)

```
public CharacterStats(CharacterStats value)
```

## Parameters

value [CharacterStats](#)

## CharacterStats(Row, int, IReadOnlyList<StatModifier>)

```
public CharacterStats(CharacterSheet.Row row, int level, IReadOnlyList<StatModifier>
initialStatModifiers = null)
```

### Parameters

row [CharacterSheet.Row](#)

level [int](#)

initialStatModifiers [IReadOnlyList](#)<[StatModifier](#)>

## CharacterStats(WaveStatData)

```
public CharacterStats(WorldBossCharacterSheet.WaveStatData stat)
```

### Parameters

stat [WorldBossCharacterSheet.WaveStatData](#)

## Fields

### StatWithoutBuffs

```
public readonly StatMap StatWithoutBuffs
```

### Field Value

[StatMap](#)

## Properties

## AdditionalATK

```
public long AdditionalATK { get; }
```

Property Value

[long](#) ↗

## AdditionalArmorPenetration

```
public long AdditionalArmorPenetration { get; }
```

Property Value

[long](#) ↗

## AdditionalCDMG

```
public long AdditionalCDMG { get; }
```

Property Value

[long](#) ↗

## AdditionalCRI

```
public long AdditionalCRI { get; }
```

Property Value

[long](#) ↗

## AdditionalDEF

```
public long AdditionalDEF { get; }
```

Property Value

[long](#) ↗

## AdditionalDRR

```
public long AdditionalDRR { get; }
```

Property Value

[long](#) ↗

## AdditionalDRV

```
public long AdditionalDRV { get; }
```

Property Value

[long](#) ↗

## AdditionalHIT

```
public long AdditionalHIT { get; }
```

Property Value

[long](#) ↗

## AdditionalHP

```
public long AdditionalHP { get; }
```

Property Value

[long](#) ↗

## AdditionalSPD

```
public long AdditionalSPD { get; }
```

Property Value

[long](#) ↗

## AdditionalThorn

```
public long AdditionalThorn { get; }
```

Property Value

[long](#) ↗

## BaseATK

```
public long BaseATK { get; }
```

Property Value

[long](#) ↗

## BaseArmorPenetration

```
public long BaseArmorPenetration { get; }
```

Property Value

[long](#) ↗

## BaseCDMG

```
public long BaseCDMG { get; }
```

Property Value

[long](#) ↗

## BaseCRI

```
public long BaseCRI { get; }
```

Property Value

[long](#) ↗

## BaseDEF

```
public long BaseDEF { get; }
```

Property Value

[long](#) ↗

## BaseDRR

```
public long BaseDRR { get; }
```

Property Value

[long](#) ↗

## BaseDRV

```
public long BaseDRV { get; }
```

Property Value

[long](#) ↗

## BaseHIT

```
public long BaseHIT { get; }
```

Property Value

[long](#) ↗

## BaseHP

```
public long BaseHP { get; }
```

Property Value

[long](#) ↗

## BaseSPD

```
public long BaseSPD { get; }
```

Property Value

[long](#) ↗

## BaseStats

```
public IStats BaseStats { get; }
```

Property Value

[IStats](#)

## BaseThorn

```
public long BaseThorn { get; }
```

Property Value

[long](#) ↗

## BuffStats

```
public IStats BuffStats { get; }
```

Property Value

[IStats](#)

## CollectionStats

```
public IStats CollectionStats { get; }
```

Property Value

[IStats](#)

## ConsumableStats

```
public IStats ConsumableStats { get; }
```

Property Value

[IStats](#)

## CostumeStats

```
public IStats CostumeStats { get; }
```

Property Value

[IStats](#)

## EquipmentStats

```
public IStats EquipmentStats { get; }
```

Property Value

[IStats](#)

## HpIncreasingModifier

```
public long HpIncreasingModifier { set; }
```

Property Value

[long](#) ↗

## IsArenaCharacter

```
public bool IsArenaCharacter { set; }
```

Property Value

[bool](#) ↗

## Level

```
public int Level { get; }
```

Property Value

[int](#) ↗

## RuneStats

```
public IStats RuneStats { get; }
```

Property Value

[IStats](#)

## Methods

AddBuff(StatBuff, DeBuffLimitSheet, bool)

```
public void AddBuff(StatBuff buff, DeBuffLimitSheet deBuffLimitSheet, bool updateImmediate = true)
```

## Parameters

buff [StatBuff](#)

deBuffLimitSheet [DeBuffLimitSheet](#)

updateImmediate [bool](#)

## AddCostume(IEnumerable<StatModifier>)

```
public void AddCostume(IEnumerable<StatModifier> statModifiers)
```

## Parameters

statModifiers [IEnumerable](#)<[StatModifier](#)>

## AddRune(IEnumerable<StatModifier>)

```
public void AddRune(IEnumerable<StatModifier> statModifiers)
```

## Parameters

statModifiers [IEnumerable](#)<[StatModifier](#)>

## AddRuneStat(RuneOptionInfo, int)

```
public void AddRuneStat(RuneOptionSheet.Row.RuneOptionInfo optionInfo,  
int runeLevelBonus)
```

## Parameters

optionInfo [RuneOptionSheet](#).[Row](#).[RuneOptionInfo](#)

runeLevelBonus [int](#)

## Clone()

Creates a new object that is a copy of the current instance.

```
public override object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

ConfigureStats(IReadOnlyCollection<Equipment>,  
IReadOnlyCollection<Costume>,  
IReadOnlyCollection<RuneOptionInfo>,  
CostumeStatSheet, List<StatModifier>, int)

```
public void ConfigureStats(IReadOnlyCollection<Equipment> equipments,  
IReadOnlyCollection<Costume> costumes,  
IReadOnlyCollection<RuneOptionSheet.Row.RuneOptionInfo> runeOptions,  
CostumeStatSheet costumeStatSheet, List<StatModifier> collectionStatModifiers,  
int runeLevelBonus)
```

Parameters

equipments [IReadOnlyCollection](#)<[Equipment](#)>

costumes [IReadOnlyCollection](#)<[Costume](#)>

runeOptions [IReadOnlyCollection](#)<[RuneOptionSheet.Row.RuneOptionInfo](#)>

costumeStatSheet [CostumeStatSheet](#)

collectionStatModifiers [List](#)<[StatModifier](#)>

runeLevelBonus [int](#)

## GetAdditionalStats(bool)

```
public IEnumerable<(StatType statType, long additionalValue)>
GetAdditionalStats(bool ignoreZero = false)
```

### Parameters

ignoreZero [bool](#)

### Returns

[IEnumerable](#)<(StatType statType, long baseValue)>

## GetBaseAndAdditionalStats(bool)

```
public IEnumerable<(StatType statType, long baseValue, long additionalValue)>
GetBaseAndAdditionalStats(bool ignoreZero = false)
```

### Parameters

ignoreZero [bool](#)

### Returns

[IEnumerable](#)<(StatType statType, long baseValue, long additionalValue)>

## GetBaseStats(bool)

```
public IEnumerable<(StatType statType, long baseValue)> GetBaseStats(bool ignoreZero
= false)
```

### Parameters

ignoreZero [bool](#)

### Returns

[IEnumerable](#)<(StatType statType, long baseValue)>

## IncreaseHpForArena()

Increases the HP of the character for the arena.

```
public void IncreaseHpForArena()
```

## RemoveBuff(StatBuff, bool)

```
public void RemoveBuff(StatBuff buff, bool updateImmediate = true)
```

Parameters

buff [StatBuff](#)

updateImmediate [bool](#)

## SetBuffs(IEnumerable<StatBuff>, DeBuffLimitSheet, bool)

Set stats based on buffs.

```
public CharacterStats SetBuffs(IEnumerable<StatBuff> value, DeBuffLimitSheet deBuffLimitSheet, bool updateImmediate = true)
```

Parameters

value [IEnumerable](#)<[StatBuff](#)>

deBuffLimitSheet [DeBuffLimitSheet](#)

updateImmediate [bool](#)

Returns

[CharacterStats](#)

## SetCollections(IEnumerable<StatModifier>, bool)

```
public CharacterStats SetCollections(IEnumerable<StatModifier> statModifiers, bool updateImmediate = true)
```

### Parameters

statModifiers [IEnumerable<StatModifier>](#)

updateImmediate [bool](#)

### Returns

[CharacterStats](#)

## SetConsumables(IEnumerable<Consumable>, bool)

Set stats based on consumables. Also recalculates stats from buffs.

```
public CharacterStats SetConsumables(IEnumerable<Consumable> consumables, bool updateImmediate = true)
```

### Parameters

consumables [IEnumerable<Consumable>](#)

updateImmediate [bool](#)

### Returns

[CharacterStats](#)

## SetCostume(IEnumerable<StatModifier>)

```
public void SetCostume(IEnumerable<StatModifier> statModifiers)
```

## Parameters

statModifiers [IEnumerable](#)<[StatModifier](#)>

## SetCostumeStat(IReadOnlyCollection<Costume>, CostumeStatSheet)

```
public void SetCostumeStat(IReadOnlyCollection<Costume> costumes,  
CostumeStatSheet costumeStatSheet)
```

## Parameters

costumes [IReadOnlyCollection](#)<[Costume](#)>

costumeStatSheet [CostumeStatSheet](#)

## SetEquipments(IEnumerable<Equipment>, EquipmentItemSetEffectSheet, bool)

Set stats based on equipments. Also recalculates stats from consumables and buffs.

```
public CharacterStats SetEquipments(IEnumerable<Equipment> equipments,  
EquipmentItemSetEffectSheet sheet, bool updateImmediate = true)
```

## Parameters

equipments [IEnumerable](#)<[Equipment](#)>

sheet [EquipmentItemSetEffectSheet](#)

updateImmediate [bool](#)

## Returns

[CharacterStats](#)

## SetRunes(IEnumerable<StatModifier>, bool)

Set stats based on runes.

```
public CharacterStats SetRunes(IEnumerable<StatModifier> value, bool updateImmediate = true)
```

Parameters

`value` [IEnumerable](#)<`StatModifier`>

`updateImmediate` [bool](#)

Returns

[CharacterStats](#)

## SetStats(int, bool)

Set base stats based on character level.

```
public CharacterStats SetStats(int level, bool updateImmediate = true)
```

Parameters

`level` [int](#)

`updateImmediate` [bool](#)

Returns

[CharacterStats](#)

# Class DecimalStat

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DecimalStat : ICloneable, IState
```

## Inheritance

[object](#) ← DecimalStat

## Implements

[ICloneable](#), [IState](#)

## Inherited Members

[object.Equals\(object, object\)](#), [object.GetType\(\)](#), [object.MemberwiseClone\(\)](#),  
[object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Extension Methods

[StateExtensions.SerializeForLegacyEquipmentStat\(DecimalStat\)](#)

## Constructors

### DecimalStat(Dictionary)

```
public DecimalStat(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### DecimalStat(DecimalStat)

```
protected DecimalStat(DecimalStat value)
```

#### Parameters

value [DecimalStat](#)

## DecimalStat(StatType, decimal, decimal)

```
public DecimalStat(StatType type, decimal value = 0, decimal additionalValue = 0)
```

### Parameters

type [StatType](#)

value [decimal](#)

additionalValue [decimal](#)

### Fields

#### StatType

```
public StatType StatType
```

#### Field Value

[StatType](#)

### Properties

#### AdditionalValue

```
public decimal AdditionalValue { get; }
```

#### Property Value

[decimal](#)

## AdditionalValueAsLong

```
public long AdditionalValueAsLong { get; }
```

Property Value

[long](#) ↗

## BaseValue

```
public decimal BaseValue { get; }
```

Property Value

[decimal](#) ↗

## BaseValueAsLong

```
public long BaseValueAsLong { get; }
```

Property Value

[long](#) ↗

## HasAdditionalValue

```
public bool HasAdditionalValue { get; }
```

Property Value

[bool](#) ↗

## HasAdditionalValueAsLong

```
public bool HasAdditionalValueAsLong { get; }
```

Property Value

[bool](#) ↗

## HasBaseValue

```
public bool HasBaseValue { get; }
```

Property Value

[bool](#) ↗

## HasBaseValueAsLong

```
public bool HasBaseValueAsLong { get; }
```

Property Value

[bool](#) ↗

## HasTotalValueAsLong

```
public bool HasTotalValueAsLong { get; }
```

Property Value

[bool](#) ↗

## TotalValue

```
public decimal TotalValue { get; }
```

Property Value

[decimal](#)

## TotalValueAsLong

```
[Obsolete("For legacy equipments. (Before world 7 patch)")]
public long TotalValueAsLong { get; }
```

Property Value

[long](#)

## Methods

### AddAdditionalValue(decimal)

```
public void AddAdditionalValue(decimal value)
```

Parameters

[value decimal](#)

### AddBaseValue(decimal)

```
public void AddBaseValue(decimal value)
```

Parameters

[value decimal](#)

## Clone()

Creates a new object that is a copy of the current instance.

```
public virtual object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## Deserialize(Dictionary)

```
public virtual void Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

## Equals(DecimalStat)

```
protected bool Equals(DecimalStat other)
```

Parameters

**other** [DecimalStat](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

## Parameters

**obj** [object](#)

The object to compare with the current object.

## Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

## Returns

[int](#)

A hash code for the current object.

## Reset()

```
public virtual void Reset()
```

## Serialize()

```
public virtual IValue Serialize()
```

## Returns

IValue

## SerializeWithoutAdditional()

```
public IValue SerializeWithoutAdditional()
```

Returns

IValue

## SetAdditionalValue(decimal)

```
public void SetAdditionalValue(decimal value)
```

Parameters

value [decimal](#)

## SetBaseValue(decimal)

```
public void SetBaseValue(decimal value)
```

Parameters

value [decimal](#)

# Interface IBaseAndAdditionalStats

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
public interface IBaseAndAdditionalStats
```

## Properties

### AdditionalATK

```
long AdditionalATK { get; }
```

Property Value

[long](#) ↗

### AdditionalArmorPenetration

```
long AdditionalArmorPenetration { get; }
```

Property Value

[long](#) ↗

### AdditionalCDMG

```
long AdditionalCDMG { get; }
```

Property Value

[long](#) ↗

## AdditionalCRI

```
long AdditionalCRI { get; }
```

Property Value

[long](#) ↗

## AdditionalDEF

```
long AdditionalDEF { get; }
```

Property Value

[long](#) ↗

## AdditionalDRR

```
long AdditionalDRR { get; }
```

Property Value

[long](#) ↗

## AdditionalDRV

```
long AdditionalDRV { get; }
```

Property Value

[long](#) ↗

## AdditionalHIT

```
long AdditionalHIT { get; }
```

Property Value

[long](#) ↗

## AdditionalHP

```
long AdditionalHP { get; }
```

Property Value

[long](#) ↗

## AdditionalSPD

```
long AdditionalSPD { get; }
```

Property Value

[long](#) ↗

## AdditionalThorn

```
long AdditionalThorn { get; }
```

Property Value

[long](#) ↗

## BaseATK

```
long BaseATK { get; }
```

Property Value

[long](#) ↗

## BaseArmorPenetration

```
long BaseArmorPenetration { get; }
```

Property Value

[long](#) ↗

## BaseCDMG

```
long BaseCDMG { get; }
```

Property Value

[long](#) ↗

## BaseCRI

```
long BaseCRI { get; }
```

Property Value

[long](#) ↗

## BaseDEF

```
long BaseDEF { get; }
```

Property Value

[long](#) ↗

## BaseDRR

```
long BaseDRR { get; }
```

Property Value

[long](#) ↗

## BaseDRV

```
long BaseDRV { get; }
```

Property Value

[long](#) ↗

## BaseHIT

```
long BaseHIT { get; }
```

Property Value

[long](#) ↗

## BaseHP

```
long BaseHP { get; }
```

Property Value

[long](#) ↗

## BaseSPD

```
long BaseSPD { get; }
```

Property Value

[long](#) ↗

## BaseThorn

```
long BaseThorn { get; }
```

Property Value

[long](#) ↗

## Methods

### GetAdditionalStats(bool)

```
IEnumerable<(StatType statType, long additionalValue)> GetAdditionalStats(bool ignoreZero = false)
```

Parameters

`ignoreZero` [bool](#) ↗

Returns

[IEnumerable](#)<(StatType statType, long baseValue)>

## GetBaseAndAdditionalStats(bool)

IEnumerable<(StatType statType, long baseValue, long additionalValue)>  
GetBaseAndAdditionalStats(bool ignoreZero = false)

### Parameters

ignoreZero [bool](#)

### Returns

[IEnumerable](#)<(StatType statType, long baseValue, long additionalValue)>

## GetBaseStats(bool)

IEnumerable<(StatType statType, long baseValue)> GetBaseStats(bool ignoreZero = false)

### Parameters

ignoreZero [bool](#)

### Returns

[IEnumerable](#)<(StatType statType, long baseValue)>

# Interface IStats

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
public interface IStats
```

## Properties

### ATK

```
long ATK { get; }
```

Property Value

[long](#) ↗

### ArmorPenetration

```
long ArmorPenetration { get; }
```

Property Value

[long](#) ↗

### CDMG

```
long CDMG { get; }
```

Property Value

[long](#) ↗

## CRI

```
long CRI { get; }
```

Property Value

[long](#) ↗

## DEF

```
long DEF { get; }
```

Property Value

[long](#) ↗

## DRR

```
long DRR { get; }
```

Property Value

[long](#) ↗

## DRV

```
long DRV { get; }
```

Property Value

[long](#) ↗

## HIT

```
long HIT { get; }
```

Property Value

[long](#) ↗

HP

```
long HP { get; }
```

Property Value

[long](#) ↗

SPD

```
long SPD { get; }
```

Property Value

[long](#) ↗

Thorn

```
long Thorn { get; }
```

Property Value

[long](#) ↗

Methods

GetStats(bool)

```
IEnumerable<(StatType statType, long value)> GetStats(bool ignoreZero = false)
```

## Parameters

ignoreZero [bool](#)

## Returns

[IEnumerable](#)<(StatType statType, long baseValue)>

# Class StatMap

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatMap : IStats, IBaseAndAdditionalStats, IState
```

## Inheritance

[object](#) ← StatMap

## Implements

[IStats](#), [IBaseAndAdditionalStats](#), [IState](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### StatMap()

```
public StatMap()
```

### StatMap(StatMap)

```
public StatMap(StatMap statMap)
```

## Parameters

statMap [StatMap](#)

## Properties

## ATK

```
public long ATK { get; }
```

Property Value

[long](#) ↗

## AdditionalATK

```
public long AdditionalATK { get; }
```

Property Value

[long](#) ↗

## AdditionalArmorPenetration

```
public long AdditionalArmorPenetration { get; }
```

Property Value

[long](#) ↗

## AdditionalCDMG

```
public long AdditionalCDMG { get; }
```

Property Value

[long](#) ↗

## AdditionalCRI

```
public long AdditionalCRI { get; }
```

Property Value

[long](#) ↗

## AdditionalDEF

```
public long AdditionalDEF { get; }
```

Property Value

[long](#) ↗

## AdditionalDRR

```
public long AdditionalDRR { get; }
```

Property Value

[long](#) ↗

## AdditionalDRV

```
public long AdditionalDRV { get; }
```

Property Value

[long](#) ↗

## AdditionalHIT

```
public long AdditionalHIT { get; }
```

Property Value

[long](#) ↗

## AdditionalHP

```
public long AdditionalHP { get; }
```

Property Value

[long](#) ↗

## AdditionalSPD

```
public long AdditionalSPD { get; }
```

Property Value

[long](#) ↗

## AdditionalThorn

```
public long AdditionalThorn { get; }
```

Property Value

[long](#) ↗

## ArmorPenetration

```
public long ArmorPenetration { get; }
```

Property Value

[long](#) ↗

## BaseATK

```
public long BaseATK { get; }
```

Property Value

[long](#) ↗

## BaseArmorPenetration

```
public long BaseArmorPenetration { get; }
```

Property Value

[long](#) ↗

## BaseCDMG

```
public long BaseCDMG { get; }
```

Property Value

[long](#) ↗

## BaseCRI

```
public long BaseCRI { get; }
```

Property Value

[long](#) ↗

## BaseDEF

```
public long BaseDEF { get; }
```

Property Value

[long](#) ↗

## BaseDRR

```
public long BaseDRR { get; }
```

Property Value

[long](#) ↗

## BaseDRV

```
public long BaseDRV { get; }
```

Property Value

[long](#) ↗

## BaseHIT

```
public long BaseHIT { get; }
```

Property Value

[long](#) ↗

## BaseHP

```
public long BaseHP { get; }
```

Property Value

[long](#) ↗

## BaseSPD

```
public long BaseSPD { get; }
```

Property Value

[long](#) ↗

## BaseThorn

```
public long BaseThorn { get; }
```

Property Value

[long](#) ↗

## CDMG

```
public long CDMG { get; }
```

Property Value

[long](#) ↗

CRI

```
public long CRI { get; }
```

Property Value

[long](#) ↗

DEF

```
public long DEF { get; }
```

Property Value

[long](#) ↗

DRR

```
public long DRR { get; }
```

Property Value

[long](#) ↗

DRV

```
public long DRV { get; }
```

Property Value

[long](#) ↗

## HIT

```
public long HIT { get; }
```

Property Value

[long](#) ↗

## HP

```
public long HP { get; }
```

Property Value

[long](#) ↗

## this[StatType]

```
public DecimalStat this[StatType type] { get; }
```

Parameters

type [StatType](#)

Property Value

[DecimalStat](#)

## SPD

```
public long SPD { get; }
```

Property Value

[long](#) ↗

## Thorn

```
public long Thorn { get; }
```

Property Value

[long](#) ↗

## Methods

### Deserialize(Dictionary)

```
public void Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

### GetAdditionalStat(StatType)

```
public long GetAdditionalStat(StatType statType)
```

Parameters

**statType** [StatType](#)

Returns

[long](#)

## GetAdditionalStats(bool)

```
public IEnumerable<(StatType statType, long additionalValue)>
GetAdditionalStats(bool ignoreZero = false)
```

Parameters

ignoreZero [bool](#)

Returns

[IEnumerable](#)<(StatType statType, long baseValue)>

## GetBaseAndAdditionalStats(bool)

```
public IEnumerable<(StatType statType, long baseValue, long additionalValue)>
GetBaseAndAdditionalStats(bool ignoreZero = false)
```

Parameters

ignoreZero [bool](#)

Returns

[IEnumerable](#)<(StatType statType, long baseValue, long additionalValue)>

## GetBaseStat(StatType)

```
public long GetBaseStat(StatType statType)
```

Parameters

`statType` [StatType](#)

Returns

[long](#)

## GetBaseStats(bool)

```
public IEnumerable<(StatType statType, long baseValue)> GetBaseStats(bool ignoreZero  
= false)
```

Parameters

`ignoreZero` [bool](#)

Returns

[IEnumerable](#)<(StatType [statType](#), long [baseValue](#))>

## GetDecimalStats(bool)

```
public IEnumerable<DecimalStat> GetDecimalStats(bool ignoreZero)
```

Parameters

`ignoreZero` [bool](#)

Returns

[IEnumerable](#)<DecimalStat>

## GetStat(StatType)

```
public decimal GetStat(StatType statType)
```

Parameters

statType [StatType](#)

Returns

[decimal](#)

## GetStatAsLong(StatType)

```
public long GetStatAsLong(StatType statType)
```

Parameters

statType [StatType](#)

Returns

[long](#)

## GetStats(bool)

```
public IEnumerable<(StatType statType, long value)> GetStats(bool ignoreZero = false)
```

Parameters

ignoreZero [bool](#)

Returns

[IEnumerable](#)<(StatType [statType](#), [long](#) [baseValue](#))>

## Reset()

```
public void Reset()
```

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class StatModifier

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatModifier
```

## Inheritance

[object](#) ← StatModifier

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### StatModifier(DecimalStat)

```
public StatModifier(DecimalStat decimalStat)
```

#### Parameters

decimalStat [DecimalStat](#)

### StatModifier(StatType, OperationType, long)

```
public StatModifier(StatType statType, StatModifier.OperationType operation,
long value)
```

#### Parameters

statType [StatType](#)

operation [StatModifier.OperationType](#)

value long ↗

# Properties

# Operation

```
public StatModifier.OperationType Operation { get; }
```

## Property Value

## StatModifier.OperationType

# StatType

```
public StatType StatType { get; }
```

## Property Value

## StatType

## Value

```
public long value { get; }
```

## Property Value

long ↗

## Methods

## GetModifiedAll(decimal)

```
public decimal GetModifiedAll(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

## GetModifiedAll(long)

value[] မှု value[] မြတ်စွမ်း များ ပေါ် မြတ်စွမ်း.

```
public long GetModifiedAll(long value)
```

Parameters

value [long](#)

Returns

[long](#)

## GetModifiedValue(decimal)

value[] မြတ်စွမ်း များ ပေါ် မြတ်စွမ်း.

```
public decimal GetModifiedValue(decimal value)
```

Parameters

value [decimal](#)

Returns

[decimal](#)

# Exceptions

## ArgumentOutOfRangeException

# GetModifiedValue(int)

value□ □□□□ □□□□ □□□ □□□□.

```
public long GetModifiedValue(int value)
```

## Parameters

value int ↗

## Returns

long ↗

# Exceptions

## ArgumentOutOfRangeException

## Modify(DecimalStat)

value █ █ █ █ █.

```
public void Modify(DecimalStat value)
```

### Parameters

## value DecimalStat

## Exceptions

## ArgumentOutOfRangeException ↗

## SetForTest(int)

```
public void SetForTest(int value)
```

### Parameters

value [int](#)

## ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

### Returns

[string](#)

A string that represents the current object.

# Enum StatModifier.OperationType

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
public enum StatModifier.OperationType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Add = 0

Percentage = 1

# Enum StatType

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
public enum StatType
```

## Extension Methods

[StatTypeExtension.Serialize\(StatType\)](#) , [StateExtensions.Serialize\(Enum\)](#)

## Fields

ATK = 2

Attack Power

ArmorPenetration = 10

Armor penetration

CDMG = 9

Crit Damage (Permyriad)

CRI = 4

Critical Chance

DEF = 3

Defence

DRR = 8

Damage Reduction Rate. Multiplied on original damage. (Permyriad)

DRV = 7

Damage Reduction Value. Subtracted from original damage.

HIT = 5

Hit Chance

**HP = 1**

Health Point

**NONE = 0**

Default, It's same with null.

**SPD = 6**

Speed

**Thorn = 11**

Apply {Thorn} damage to the attacker when an attack is attempted.

# Class StatTypeComparer

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatTypeComparer : IEqualityComparer<StatType>
```

## Inheritance

[object](#) ← StatTypeComparer

## Implements

[IEqualityComparer](#)<[StatType](#)>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Instance

```
public static readonly StatTypeComparer Instance
```

### Field Value

[StatTypeComparer](#)

## Methods

### Equals(StatType, StatType)

Determines whether the specified objects are equal.

```
public bool Equals(StatType x, StatType y)
```

## Parameters

x [StatType](#)

The first object of type T to compare.

y [StatType](#)

The second object of type T to compare.

## Returns

[bool](#)

[true](#) if the specified objects are equal; otherwise, [false](#).

## GetHashCode(StatType)

Returns a hash code for the specified object.

```
public int GetHashCode(StatType obj)
```

## Parameters

obj [StatType](#)

The [object](#) for which a hash code is to be returned.

## Returns

[int](#)

A hash code for the specified object.

## Exceptions

[ArgumentNullException](#)

The type of obj is a reference type and obj is [null](#).

# Class StatTypeExtension

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
public static class StatTypeExtension
```

## Inheritance

[object](#) ← StatTypeExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Deserialize(Binary)

```
public static StatType Deserialize(Binary serialized)
```

#### Parameters

serialized Binary

#### Returns

[StatType](#)

### Serialize(StatType)

```
public static IKey Serialize(this StatType statType)
```

#### Parameters

statType [StatType](#)

Returns

IKey

# Class Stats

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
[Serializable]
public class Stats : IStats, ICloneable
```

## Inheritance

[object](#) ← Stats

## Implements

[IStats](#), [ICloneable](#)

## Derived

[CharacterStats](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

# Constructors

## Stats()

```
public Stats()
```

## Stats(Stats)

```
public Stats(Stats value)
```

## Parameters

value [Stats](#)

## Fields

### LegacyDecimalStatTypes

```
protected readonly HashSet<StatType> LegacyDecimalStatTypes
```

#### Field Value

[HashSet](#)<[StatType](#)>

#### \_statMap

```
protected readonly StatMap _statMap
```

#### Field Value

[StatMap](#)

## Properties

### ATK

```
public long ATK { get; }
```

#### Property Value

[long](#)

### ArmorPenetration

```
public long ArmorPenetration { get; }
```

#### Property Value

[long](#) ↗

## CDMG

```
public long CDMG { get; }
```

Property Value

[long](#) ↗

## CRI

```
public long CRI { get; }
```

Property Value

[long](#) ↗

## DEF

```
public long DEF { get; }
```

Property Value

[long](#) ↗

## DRR

```
public long DRR { get; }
```

Property Value

[long](#) ↗

## DRV

```
public long DRV { get; }
```

Property Value

[long](#) ↗

## HIT

```
public long HIT { get; }
```

Property Value

[long](#) ↗

## HP

```
public long HP { get; }
```

Property Value

[long](#) ↗

## SPD

```
public long SPD { get; }
```

Property Value

[long](#) ↗

## Thorn

```
public long Thorn { get; }
```

Property Value

[long](#) ↗

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public virtual object Clone()
```

Returns

[object](#) ↗

A new object that is a copy of this instance.

### GetStat(StatType)

```
public decimal GetStat(StatType statType)
```

Parameters

statType [StatType](#)

Returns

[decimal](#) ↗

### GetStatAsLong(StatType)

```
public long GetStatAsLong(StatType statType)
```

Parameters

`statType` [StatType](#)

Returns

[long](#)

## GetStats(bool)

```
public IEnumerable<(StatType statType, long value)> GetStats(bool ignoreZero = false)
```

Parameters

`ignoreZero` [bool](#)

Returns

[IEnumerable](#)<(StatType [statType](#), long [baseValue](#))>

## Modify(IEnumerable<StatModifier>)

```
public void Modify(IEnumerable<StatModifier> statModifiers)
```

Parameters

`statModifiers` [IEnumerable](#)<[StatModifier](#)>

## Reset()

```
public void Reset()
```

## Set(StatMap, params Stats[])

```
public void Set(StatMap statMap, params Stats[] statsArray)
```

Parameters

statMap [StatMap](#)

statsArray [Stats\[\]](#)

## Set(StatsMap)

```
public void Set(StatsMap value)
```

Parameters

value [StatsMap](#)

## Set(IEnumerable<StatModifier>, params Stats[])

```
public void Set(IEnumerable<StatModifier> statModifiers, params Stats[] baseStats)
```

Parameters

statModifiers [IEnumerable](#)<[StatModifier](#)>

baseStats [Stats\[\]](#)

## SetStatForTest(StatType, decimal)

Use this only for testing.

```
public void SetStatForTest(StatType statType, decimal value)
```

Parameters

statType [StatType](#)

value [decimal](#)

# Class StatsMap

Namespace: [Nekoyume.Model.Stat](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatsMap : IStats, IBaseAndAdditionalStats, IState
```

## Inheritance

[object](#) ← StatsMap

## Implements

[IStats](#), [IBaseAndAdditionalStats](#), [IState](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Properties

## ATK

```
public long ATK { get; }
```

Property Value

[long](#)

## AdditionalATK

```
public long AdditionalATK { get; }
```

Property Value

[long](#)

## AdditionalArmorPenetration

```
public long AdditionalArmorPenetration { get; }
```

Property Value

[long](#) ↗

## AdditionalCDMG

```
public long AdditionalCDMG { get; }
```

Property Value

[long](#) ↗

## AdditionalCRI

```
public long AdditionalCRI { get; }
```

Property Value

[long](#) ↗

## AdditionalDEF

```
public long AdditionalDEF { get; }
```

Property Value

[long](#) ↗

## AdditionalDRR

```
public long AdditionalDR { get; }
```

Property Value

[long](#) ↗

## AdditionalDRV

```
public long AdditionalDRV { get; }
```

Property Value

[long](#) ↗

## AdditionalHIT

```
public long AdditionalHIT { get; }
```

Property Value

[long](#) ↗

## AdditionalHP

```
public long AdditionalHP { get; }
```

Property Value

[long](#) ↗

## AdditionalSPD

```
public long AdditionalSPD { get; }
```

Property Value

[long](#) ↗

## AdditionalThorn

```
public long AdditionalThorn { get; }
```

Property Value

[long](#) ↗

## ArmorPenetration

```
public long ArmorPenetration { get; }
```

Property Value

[long](#) ↗

## BaseATK

```
public long BaseATK { get; }
```

Property Value

[long](#) ↗

## BaseArmorPenetration

```
public long BaseArmorPenetration { get; }
```

Property Value

[long](#) ↗

## BaseCDMG

```
public long BaseCDMG { get; }
```

Property Value

[long](#) ↗

## BaseCRI

```
public long BaseCRI { get; }
```

Property Value

[long](#) ↗

## BaseDEF

```
public long BaseDEF { get; }
```

Property Value

[long](#) ↗

## BaseDRR

```
public long BaseDRR { get; }
```

Property Value

[long](#) ↗

## BaseDRV

```
public long BaseDRV { get; }
```

Property Value

[long](#) ↗

## BaseHIT

```
public long BaseHIT { get; }
```

Property Value

[long](#) ↗

## BaseHP

```
public long BaseHP { get; }
```

Property Value

[long](#) ↗

## BaseSPD

```
public long BaseSPD { get; }
```

Property Value

[long](#) ↗

## BaseThorn

```
public long BaseThorn { get; }
```

Property Value

[long](#) ↗

## CDMG

```
public long CDMG { get; }
```

Property Value

[long](#) ↗

## CRI

```
public long CRI { get; }
```

Property Value

[long](#) ↗

## DEF

```
public long DEF { get; }
```

Property Value

[long](#) ↗

## DRR

```
public long DRR { get; }
```

Property Value

[long](#) ↗

## DRV

```
public long DRV { get; }
```

Property Value

[long](#) ↗

## HIT

```
public long HIT { get; }
```

Property Value

[long](#) ↗

## HP

```
public long HP { get; }
```

Property Value

[long](#) ↗

## SPD

```
public long SPD { get; }
```

Property Value

[long](#) ↗

## Thorn

```
public long Thorn { get; }
```

Property Value

[long](#) ↗

## Methods

### AddStatAdditionalValue(StatModifier)

```
public void AddStatAdditionalValue(StatModifier statModifier)
```

Parameters

statModifier [StatModifier](#)

### AddStatAdditionalValue(StatType, decimal)

```
public void AddStatAdditionalValue(StatType key, decimal additionalValue)
```

Parameters

key [StatType](#)

additionalValue [decimal](#)

## AddStatValue(StatType, decimal)

```
public void AddStatValue(StatType key, decimal value)
```

Parameters

key [StatType](#)

value [decimal](#)

## Deserialize(Dictionary)

```
public void Deserialize(Dictionary serialized)
```

Parameters

serialized Dictionary

## Equals(StatsMap)

```
protected bool Equals(StatsMap other)
```

Parameters

other [StatsMap](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

[obj](#) [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetAdditionalStat(StatType)

```
public long GetAdditionalStat(StatType statType)
```

Parameters

[statType](#) [StatType](#)

Returns

[long](#)

## GetAdditionalStats()

```
public IEnumerable<DecimalStat> GetAdditionalStats()
```

Returns

[IEnumerable](#)<[DecimalStat](#)>

## GetAdditionalStats(bool)

```
public IEnumerable<(StatType statType, long additionalValue)>
GetAdditionalStats(bool ignoreZero = false)
```

Parameters

ignoreZero [bool](#)

Returns

[IEnumerable](#)<([StatType](#) [statType](#), [long](#) [baseValue](#))>

## GetBaseAndAdditionalStats(bool)

```
public IEnumerable<(StatType statType, long baseValue, long additionalValue)>
GetBaseAndAdditionalStats(bool ignoreZero = false)
```

Parameters

ignoreZero [bool](#)

Returns

[IEnumerable](#)<([StatType](#) [statType](#), [long](#) [baseValue](#), [long](#) [additionalValue](#))>

## GetBaseStat(StatType)

```
public long GetBaseStat(StatType statType)
```

Parameters

`statType` [StatType](#)

Returns

[long](#)

## GetBaseStats(bool)

```
public IEnumerable<(StatType statType, long baseValue)> GetBaseStats(bool ignoreZero = false)
```

Parameters

`ignoreZero` [bool](#)

Returns

[IEnumerable](#)<(StatType [statType](#), long [baseValue](#))>

## GetDecimalStats(bool)

```
public IEnumerable<DecimalStat> GetDecimalStats(bool ignoreZero)
```

Parameters

`ignoreZero` [bool](#)

Returns

[IEnumerable](#)<DecimalStat>

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetStat(StatType)

```
public long GetStat(StatType statType)
```

Parameters

statType [StatType](#)

Returns

[long](#)

## GetStats(bool)

```
public IEnumerable<(StatType statType, long value)> GetStats(bool ignoreZero = false)
```

Parameters

ignoreZero [bool](#)

Returns

[IEnumerable](#)<(StatType [statType](#), long [baseValue](#))>

## Serialize()

```
public IValue Serialize()
```

Returns

## SetStatAdditionalValue(StatType, decimal)

```
public void SetStatAdditionalValue(StatType key, decimal additionalValue)
```

### Parameters

key [StatType](#)

additionalValue [decimal](#)

# Namespace Nekoyume.Model.State

## Classes

[ActivatedAccountsState](#)

[AdminState](#)

[AgentState](#)

Agentの状態。

[AllCombinationSlotState](#)

This is new version of combination slot State. This state stores all combinationSlot states of an avatar. AllCombinationSlotState has all combinationSlotStates as dictionary and has methods to get/set/update each combinationSlotState. Use this with [CombinationSlotStateModule](#).

[AllRuneState](#)

This is new version of rune state. This state stores all rune states of an avatar.

AllRuneState has all RuneStates as dictionary and has methods to get/set/update each RuneState. Use this with [RuneStateModule](#).

[ArenaAvatarState](#)

Introduced at <https://github.com/planetarium/lib9c/pull/1156>

[ArenalInfo](#)

[ArenalInfo.Record](#)

[AuthorizedMinersState](#)

[AvatarState](#)

Agentの状態。Avatarの状態。

[CollectionState](#)

Represents the state of a collection.

[CombinationSlotState](#)

[CreditsState](#)

[CrystalCostState](#)

[CrystalRandomSkillState](#)

[DuplicateRedeemException](#)

[FailedToUnregisterInShopStateException](#)

[GameConfigState](#)

[GoldBalanceState](#)

[GoldCurrencyState](#)

[HammerPointState](#)

[InvalidRedeemCodeException](#)

[ItemSlotState](#)

[LazyState<TState, TEncoding>](#)

[LegacyStakeState](#)

[LegacyStakeState.StakeAchievements](#)

[MarketState](#)

[MonsterCollectionResult](#)

[MonsterCollectionState](#)

[MonsterCollectionState0](#)

[PendingActivationState](#)

[PetState](#)

[RaiderState](#)

[RankingInfo](#)

[RankingMapState](#)

[RankingState](#)

[RankingState0](#)

[RankingState1](#)

[RedeemCodeState](#)

[RedeemCodeState.Reward](#)

[RuneSlotState](#)

[RuneState](#)

[ShardedShopState](#)

[ShardedShopStateV2](#)

[ShopState](#)

This is a model class of shop state.

[ShopStateAlreadyContainsException](#)

[StakeRewardCalculator](#)

[State](#)

[StateExtensions](#)

[WeeklyArenaState](#)

[WorldBossKillRewardRecord](#)

[WorldBossState](#)

## Interfaces

[IState](#)

## Delegates

[StateExtensions.IValueTryParseDelegate<T>](#)

# Class ActivatedAccountsState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActivatedAccountsState : State, IState, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← ActivatedAccountsState

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ActivatedAccountsState()

```
public ActivatedAccountsState()
```

### ActivatedAccountsState(Dictionary)

```
public ActivatedAccountsState(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## ActivatedAccountsState(IImmutableSet<Address>)

```
public ActivatedAccountsState(IImmutableSet<Address> accounts)
```

### Parameters

accounts [IImmutableSet](#)<Address>

## ActivatedAccountsState(SerializationInfo, StreamingContext)

```
protected ActivatedAccountsState(SerializationInfo info, StreamingContext context)
```

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

### Address

```
public static readonly Address Address
```

### Field Value

Address

## Properties

### Accounts

```
public IImmutableSet<Address> Accounts { get; }
```

## Property Value

[IImmutableSet](#)<Address>

## Methods

### AddAccount(Address)

`public ActivatedAccountsState AddAccount(Address account)`

#### Parameters

`account Address`

#### Returns

[ActivatedAccountsState](#)

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

`public void GetObjectData(SerializationInfo info, StreamingContext context)`

#### Parameters

`info SerializationInfo`

The [SerializationInfo](#) to populate with data.

`context StreamingContext`

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Remove(Address)

```
public void Remove(Address account)
```

### Parameters

account Address

## Serialize()

```
public override IValue Serialize()
```

### Returns

IValue

# Class AdminState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdminState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← AdminState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### AdminState(Dictionary)

```
public AdminState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### AdminState(Address, long)

```
public AdminState(Address adminAddress, long validUntil)
```

#### Parameters

`adminAddress` Address

`validUntil` long ↗

## Fields

### Address

`public static readonly Address Address`

Field Value

Address

## Properties

### AdminAddress

`public Address AdminAddress { get; }`

Property Value

Address

## ValidUntil

`public long ValidUntil { get; }`

Property Value

long ↗

## Methods

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class AgentState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

Agentの状態を表す。

```
[Serializable]
public class AgentState : State, IState, ICloneable
```

## Inheritance

[object](#) ← [State](#) ← AgentState

## Implements

[IState](#), [ICloneable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### AgentState(Dictionary)

```
public AgentState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### AgentState(List)

```
public AgentState(List serialized)
```

Parameters

**serialized** List

## AgentState(Address)

```
public AgentState(Address address)
```

Parameters

**address** Address

## Fields

### CurrentVersion

```
public const int CurrentVersion = 1
```

Field Value

[int](#)

### avatarAddresses

```
public readonly Dictionary<int, Address> avatarAddresses
```

Field Value

[Dictionary](#) <[int](#), Address>

## Properties

### MonsterCollectionRound

```
public int MonsterCollectionRound { get; }
```

Property Value

[int](#)

## Version

```
public int Version { get; }
```

Property Value

[int](#)

## Methods

### Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

### IncreaseCollectionRound()

```
public void IncreaseCollectionRound()
```

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public IValue SerializeList()
```

Returns

IValue

# Class AllCombinationSlotState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

This is new version of combination slot State. This state stores all combinationSlot states of an avatar. AllCombinationSlotState has all combinationSlotStates as dictionary and has methods to get/set/update each combinationSlotState. Use this with [CombinationSlotState Module](#).

```
public class AllCombinationSlotState : IState, IEnumerable<CombinationSlotState>,  
IEnumerable
```

## Inheritance

[object](#) ← AllCombinationSlotState

## Implements

[IState](#), [IEnumerable](#)<[CombinationSlotState](#)>, [IEnumerable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### AllCombinationSlotState()

```
public AllCombinationSlotState()
```

### AllCombinationSlotState(List)

```
public AllCombinationSlotState(List serialized)
```

## Parameters

**serialized** List

## Properties

### CombinationSlots

```
public Dictionary<int, CombinationSlotState> CombinationSlots { get; }
```

Property Value

[Dictionary](#)<[int](#), [CombinationSlotState](#)>

## Methods

### AddSlot(Address, int)

```
public void AddSlot(Address address, int index = 0)
```

Parameters

address Address

index [int](#)

### AddSlot(CombinationSlotState)

```
public void AddSlot(CombinationSlotState combinationSlotState)
```

Parameters

combinationSlotState [CombinationSlotState](#)

### GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumerator<CombinationSlotState> GetEnumerator()
```

Returns

[IEnumerator](#)<CombinationSlotState>

An enumerator that can be used to iterate through the collection.

## GetSlot(int)

```
public CombinationSlotState GetSlot(int slotStateIndex)
```

Parameters

slotStateIndex [int](#)

Returns

[CombinationSlotState](#)

## MigrationLegacySlotState(IWorldState, Address)

AllCombinationSlotState یه، چهار یکمین ۴۰۰ یکمین ۱۰۰۰.

```
public static AllCombinationSlotState MigrationLegacySlotState(IWorldState
worldState, Address avatarAddress)
```

Parameters

worldState IWorldState

Slot یه، چهار یکمین world state

avatarAddress Address

Migration یه، چهار یکمین

Returns

## [AllCombinationSlotState](#)

Migration[] AllCombinationSlotState

## MigrationLegacySlotState(Func<int, CombinationSlotState?>, Address)

□□ AllCombinationSlotState[] □□□, □□ □□ □□□□ □ 4□□ □□□ □□□□□.

```
public static AllCombinationSlotState MigrationLegacySlotState(Func<int, CombinationSlotState?> stateFactory, Address avatarAddress)
```

Parameters

stateFactory [Func<int, CombinationSlotState>](#)

CombinationSlotState[] □□□ □□

avatarAddress Address

Migration[] □□□ □□□

Returns

## [AllCombinationSlotState](#)

Migration[] AllCombinationSlotState

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## SetSlot(CombinationSlotState)

```
public void SetSlot(CombinationSlotState combinationSlotState)
```

### Parameters

combinationSlotState [CombinationSlotState](#)

## TryGetSlot(int, out CombinationSlotState?)

```
public bool TryGetSlot(int slotStateIndex, out CombinationSlotState? combinationSlotState)
```

### Parameters

slotStateIndex [int](#)

combinationSlotState [CombinationSlotState](#)

### Returns

[bool](#)

## UnlockSlot(Address, int)

```
public void UnlockSlot(Address avatarAddress, int index)
```

### Parameters

avatarAddress [Address](#)

index [int](#)

# Class AllRuneState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

This is new version of rune state. This state stores all rune states of an avatar. AllRuneState has all RuneStates as dictionary and has methods to get/set/update each RuneState. Use this with [RuneStateModule](#).

```
public class AllRuneState : IState
```

## Inheritance

[object](#) ← AllRuneState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### AllRuneState()

```
public AllRuneState()
```

### AllRuneState(List)

```
public AllRuneState(List serialized)
```

## Parameters

serialized List

## AllRuneState(int, int)

```
public AllRuneState(int runeId, int level = 0)
```

### Parameters

runeId [int](#)

level [int](#)

## Properties

### Runes

```
public Dictionary<int, RuneState> Runes { get; }
```

### Property Value

[Dictionary](#)<[int](#), [RuneState](#)>

## Methods

### AddRuneState(RuneState)

```
public void AddRuneState(RuneState runeState)
```

### Parameters

runeState [RuneState](#)

### AddRuneState(int, int)

```
public void AddRuneState(int runeId, int level = 0)
```

Parameters

runeId [int](#)

level [int](#)

## GetRuneState(int)

```
public RuneState GetRuneState(int runeId)
```

Parameters

runeId [int](#)

Returns

[RuneState](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## SetRuneState(RuneState)

```
public void SetRuneState(RuneState runeState)
```

Parameters

runeState [RuneState](#)

## SetRuneState(int, int)

```
public void SetRuneState(int runeId, int level)
```

### Parameters

runeId [int](#)

level [int](#)

## TryGetRuneState(int, out RuneState)

```
public bool TryGetRuneState(int runeId, out RuneState runeState)
```

### Parameters

runeId [int](#)

runeState [RuneState](#)

### Returns

[bool](#)

# Class ArenaAvatarState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

Introduced at <https://github.com/planetarium/lib9c/pull/1156>

```
public class ArenaAvatarState : IState
```

## Inheritance

[object](#) ← ArenaAvatarState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaAvatarState(List)

```
public ArenaAvatarState(List serialized)
```

#### Parameters

serialized List

### ArenaAvatarState(AvatarState)

```
public ArenaAvatarState(AvatarState avatarState)
```

#### Parameters

avatarState [AvatarState](#)

## Fields

### Address

`public Address Address`

#### Field Value

Address

## LastBattleBlockIndex

`public long LastBattleBlockIndex`

#### Field Value

[long](#)

## Properties

### Costumes

`public List<Guid> Costumes { get; }`

#### Property Value

[List](#)<[Guid](#)>

## Equipments

`public List<Guid> Equipments { get; }`

## Property Value

[List](#) <[Guid](#)>

## Methods

### DeriveAddress(Address)

`public static Address DeriveAddress(Address avatarAddress)`

#### Parameters

`avatarAddress` Address

#### Returns

Address

### Serialize()

`public IValue Serialize()`

#### Returns

IValue

### UpdateCostumes(List<Guid>)

`public void UpdateCostumes(List<Guid> costumes)`

#### Parameters

`costumes` [List](#) <[Guid](#)>

## UpdateEquipment(List<Guid>)

```
public void UpdateEquipment(List<Guid> equipments)
```

### Parameters

equipments [List](#)<[Guid](#)>

# Class ArenaInfo

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class ArenaInfo : IState
```

## Inheritance

[object](#) ← ArenaInfo

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ArenaInfo(Dictionary)

```
public ArenaInfo(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### ArenaInfo(ArenaInfo)

```
public ArenaInfo(ArenaInfo prevInfo)
```

#### Parameters

prevInfo [ArenaInfo](#)

## ArenaInfo(AvatarState, CharacterSheet, CostumeStatSheet, bool)

```
public ArenaInfo(AvatarState avatarState, CharacterSheet characterSheet,  
CostumeStatSheet costumeStatSheet, bool active)
```

### Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

costumeStatSheet [CostumeStatSheet](#)

active [bool](#)

## ArenaInfo(AvatarState, CharacterSheet, bool)

```
public ArenaInfo(AvatarState avatarState, CharacterSheet characterSheet,  
bool active)
```

### Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

active [bool](#)

## Fields

### AgentAddress

```
public readonly Address AgentAddress
```

### Field Value

Address

## ArenaRecord

```
public readonly ArenaInfo.Record ArenaRecord
```

Field Value

[ArenaInfo.Record](#)

## ArmorId

```
[Obsolete("Not used anymore since v100070")]
public int ArmorId
```

Field Value

[int](#)

## AvatarAddress

```
public readonly Address AvatarAddress
```

Field Value

Address

## AvatarName

```
public readonly string AvatarName
```

Field Value

[string](#)

## CombatPoint

```
[Obsolete("Not used anymore since v100070")]
public int CombatPoint
```

### Field Value

[int](#) ↗

## Level

```
[Obsolete("Not used anymore since v100070")]
public int Level
```

### Field Value

[int](#) ↗

## Receive

```
[Obsolete("Not used anymore since v100070")]
public bool Receive
```

### Field Value

[bool](#) ↗

## Properties

### Active

```
public bool Active { get; }
```

### Property Value

[bool](#)

## DailyChallengeCount

`public int DailyChallengeCount { get; }`

Property Value

[int](#)

## Score

`public int Score { get; }`

Property Value

[int](#)

## Methods

### Activate()

`public void Activate()`

### Clone()

`public ArenaInfo Clone()`

Returns

[ArenaInfo](#)

## GetRewardCount()

```
public int GetRewardCount()
```

Returns

[int](#)

## IsActive(Dictionary)

```
public static bool IsActive(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[bool](#)

## ResetCount()

```
public void ResetCount()
```

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Update(ArenaInfo, Result, Func<int, int, Result, (int challengerScore, int defenderScore)>)

```
public int Update(ArenaInfo enemyInfo, BattleLog.Result result, Func<int, int, BattleLog.Result, (int challengerScore, int defenderScore)> scoreGetter)
```

### Parameters

enemyInfo [ArenaInfo](#)

result [BattleLog.Result](#)

scoreGetter [Func<int, int, BattleLog.Result, \(int challengerScore, int defenderScore\)>](#)

### Returns

[int](#)

## UpdateV1(AvatarState, CharacterSheet)

```
[Obsolete("Use Update()")]
public void UpdateV1(AvatarState state, CharacterSheet characterSheet)
```

### Parameters

state [AvatarState](#)

characterSheet [CharacterSheet](#)

## UpdateV2(AvatarState, CharacterSheet, CostumeStatSheet)

```
[Obsolete("Use Update()")]
public void UpdateV2(AvatarState state, CharacterSheet characterSheet,
CostumeStatSheet costumeStatSheet)
```

Parameters

state [AvatarState](#)

characterSheet [CharacterSheet](#)

costumeStatSheet [CostumeStatSheet](#)

## UpdateV3(AvatarState, ArenaInfo, Result)

```
[Obsolete("Use Update()")]
```

```
public int UpdateV3(AvatarState avatarState, ArenaInfo enemyInfo,  
BattleLog.Result result)
```

Parameters

avatarState [AvatarState](#)

enemyInfo [ArenaInfo](#)

result [BattleLog.Result](#)

Returns

[int](#)

## UpdateV4(ArenaInfo, Result)

```
[Obsolete("Use Update()")]
```

```
public int UpdateV4(ArenaInfo enemyInfo, BattleLog.Result result)
```

Parameters

enemyInfo [ArenaInfo](#)

result [BattleLog.Result](#)

Returns

[int](#)

## UpdateV5(ArenaInfo, Result)

```
[Obsolete("Use Update()")]
public int UpdateV5(ArenaInfo enemyInfo, BattleLog.Result result)
```

### Parameters

enemyInfo [ArenaInfo](#)

result [BattleLog.Result](#)

### Returns

[int](#)

# Class ArenaInfo.Record

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class ArenaInfo.Record : IState
```

## Inheritance

[object](#) ← ArenaInfo.Record

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### Record()

```
public Record()
```

### Record(Dictionary)

```
public Record(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Fields

# Draw

```
public int Draw
```

Field Value

[int ↗](#)

# Lose

```
public int Lose
```

Field Value

[int ↗](#)

# Win

```
public int Win
```

Field Value

[int ↗](#)

# Methods

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class AuthorizedMinersState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class AuthorizedMinersState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← AuthorizedMinersState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### AuthorizedMinersState(Dictionary)

```
public AuthorizedMinersState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### AuthorizedMinersState(IEnumerable<Address>, long, long)

```
public AuthorizedMinersState(IEnumerable<Address> miners, long interval,  
long validUntil)
```

## Parameters

miners [IEnumerable](#)<Address>

interval [long](#)

validUntil [long](#)

## Fields

### Address

`public static readonly Address Address`

### Field Value

Address

## Properties

### Interval

`public long Interval { get; }`

### Property Value

[long](#)

### Miners

`public ImmutableHashSet<Address> Miners { get; }`

### Property Value

[ImmutableHashSet](#)<Address>

# ValidUntil

```
public long ValidUntil { get; }
```

Property Value

[long](#) ↗

## Methods

Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class AvatarState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

Agentの状態を Avatar の状態に.

```
[Serializable]
public class AvatarState : State, IState, ICloneable
```

## Inheritance

[object](#) ← [State](#) ← AvatarState

## Implements

[IState](#), [ICloneable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Extension Methods

[AvatarStateExtensions.GetLevelAndExp\(AvatarState, CharacterLevelSheet, int, int\)](#) ,  
[AvatarStateExtensions.GetLevelAndExpV1\(AvatarState, CharacterLevelSheet, int, int\)](#) ,  
[AvatarStateExtensions.UpdateExp\(AvatarState, int, long\)](#) ,  
[AvatarStateExtensions.UpdateInventory\(AvatarState, List<ItemBase>\)](#) ,  
[AvatarStateExtensions.UpdateMonsterMap\(AvatarState, StageWaveSheet, int\)](#) ,  
[AvatarStateExtensions.ValidEquipmentAndCostume\(AvatarState, IEnumerable<Guid>, List<Guid>, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet, long, string\)](#) ,  
[AvatarStateExtensions.ValidEquipmentAndCostumeV2\(AvatarState, IEnumerable<Guid>, List<Guid>, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet, long, string, GameConfigState\)](#).

## Constructors

## AvatarState(Dictionary)

```
public AvatarState(Dictionary serialized)
```

Parameters

**serialized** Dictionary

## AvatarState(List)

```
public AvatarState(List serialized)
```

Parameters

**serialized** List

## AvatarState(Address, Address, long, QuestList, WorldInformation, Address, string)

```
public AvatarState(Address address, Address agentAddress, long blockIndex, QuestList questList, WorldInformation worldInformation, Address rankingMapAddress, string name = null)
```

Parameters

**address** Address

**agentAddress** Address

**blockIndex** [long](#)

**questList** [QuestList](#)

**worldInformation** [WorldInformation](#)

**rankingMapAddress** Address

**name** [string](#)

## AvatarState(AvatarState)

```
public AvatarState(AvatarState avatarState)
```

### Parameters

avatarState [AvatarState](#)

## Fields

### CombinationSlotCapacity

```
public const int CombinationSlotCapacity = 8
```

### Field Value

[int](#)

### CurrentVersion

```
public const int CurrentVersion = 2
```

### Field Value

[int](#)

### DefaultCombinationSlotCount

```
public const int DefaultCombinationSlotCount = 4
```

### Field Value

[int](#)

## RankingMapAddress

```
[Obsolete("don't use this field.")]  
public readonly Address RankingMapAddress
```

Field Value

Address

## actionPoint

```
public int actionPoint
```

Field Value

[int](#)

## agentAddress

```
public Address agentAddress
```

Field Value

Address

## blockIndex

```
public long blockIndex
```

Field Value

[long](#)

## characterId

```
public int characterId
```

### Field Value

[int](#)

## combinationSlotAddresses

```
[Obsolete("don't use this field, use AllCombinationSlotState instead.")]  
public List<Address> combinationSlotAddresses
```

### Field Value

[List](#)<Address>

## dailyRewardReceivedIndex

```
public long dailyRewardReceivedIndex
```

### Field Value

[long](#)

## ear

```
public int ear
```

### Field Value

[int](#)

## eventMap

`public CollectionMap eventMap`

Field Value

[CollectionMap](#)

## exp

`public long exp`

Field Value

[long](#)

## hair

`public int hair`

Field Value

[int](#)

## inventory

`public Inventory inventory`

Field Value

[Inventory](#)

## itemMap

```
public CollectionMap itemMap
```

Field Value

[CollectionMap](#)

**lens**

```
public int lens
```

Field Value

[int](#)

**level**

```
public int level
```

Field Value

[int](#)

**mailBox**

```
public MailBox mailBox
```

Field Value

[MailBox](#)

**monsterMap**

```
public CollectionMap monsterMap
```

Field Value

[CollectionMap](#)

name

```
public string name
```

Field Value

[string](#) ↗

questList

```
public QuestList questList
```

Field Value

[QuestList](#)

stageMap

```
public CollectionMap stageMap
```

Field Value

[CollectionMap](#)

tail

```
public int tail
```

Field Value

[int](#)

## updatedAt

```
public long updatedAt
```

Field Value

[long](#)

## worldInformation

```
public WorldInformation worldInformation
```

Field Value

[WorldInformation](#)

## Properties

### NameWithHash

```
public string NameWithHash { get; }
```

Property Value

[string](#)

### Nonce

```
public int Nonce { get; }
```

Property Value

[int](#)

## Version

```
public int Version { get; }
```

Property Value

[int](#)

## Methods

### Apply(Player, long)

```
public void Apply(Player player, long index)
```

Parameters

player [Player](#)

index [long](#)

### Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## Create(Address, Address, long, AvatarSheets, Address, string)

```
public static AvatarState Create(Address address, Address agentAddress, long  
blockIndex, AvatarSheets avatarSheets, Address rankingMapAddress, string name  
= null)
```

### Parameters

address Address

agentAddress Address

blockIndex [long](#)

avatarSheets [AvatarSheets](#)

rankingMapAddress Address

name [string](#)

### Returns

[AvatarState](#)

## CreateAvatarAddress()

```
public static Address CreateAvatarAddress()
```

### Returns

Address

## Customize(int, int, int, int)

```
public void Customize(int hair, int lens, int ear, int tail)
```

## Parameters

hair [int](#)

lens [int](#)

ear [int](#)

tail [int](#)

## EquipCostumes(HashSet<int>)

```
public void EquipCostumes(HashSet<int> costumeIds)
```

## Parameters

costumeIds [HashSet](#)<[int](#)>

## EquipEquipments(List<Guid>)

```
public void EquipEquipments(List<Guid> equipmentIds)
```

## Parameters

equipmentIds [List](#)<[Guid](#)>

## EquipItems(IEnumerable<Guid>)

```
public void EquipItems(IEnumerable<Guid> itemIds)
```

## Parameters

itemIds [IEnumerable](#)<[Guid](#)>

## GetArmorId()

```
public int GetArmorId()
```

Returns

[int](#)

## GetNonFungibleItems<T>(List<Guid>)

```
public List<T> GetNonFungibleItems<T>(List<Guid> itemIds)
```

Parameters

itemIds [List](#)<[Guid](#)>

Returns

[List](#)<T>

Type Parameters

T

## GetPortraitId()

```
public int GetPortraitId()
```

Returns

[int](#)

## GetRandomSeed()

```
public int GetRandomSeed()
```

Returns

[int](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeList()

```
public IValue SerializeList()
```

Returns

IValue

## Update(IStageSimulator)

```
public void Update(IStageSimulator stageSimulator)
```

Parameters

stageSimulator [IStageSimulator](#)

## Update(Mail)

```
public void Update(Mail mail)
```

Parameters

mail [Mail](#)

## Update2(Mail)

```
[Obsolete("Use Update")]
public void Update2(Mail mail)
```

Parameters

mail [Mail](#)

## Update3(Mail)

```
[Obsolete("Use Update")]
public void Update3(Mail mail)
```

Parameters

mail [Mail](#)

## UpdateFromAddCostume(Costume, bool)

```
public void UpdateFromAddCostume(Costume costume, bool canceled)
```

Parameters

costume [Costume](#)

canceled [bool](#) ↴

## UpdateFromAddItem(ItemBase, int, bool)

```
public void UpdateFromAddItem(ItemBase itemUsable, int count, bool canceled)
```

### Parameters

itemUsable [ItemBase](#)

count [int](#)

canceled [bool](#)

## UpdateFromAddItem(ItemUsable, bool)

```
public void UpdateFromAddItem(ItemUsable itemUsable, bool canceled)
```

### Parameters

itemUsable [ItemUsable](#)

canceled [bool](#)

## UpdateFromAddItem2(ItemBase, int, bool)

[Obsolete("Use UpdateFromAddItem")]

```
public void UpdateFromAddItem2(ItemBase itemUsable, int count, bool canceled)
```

### Parameters

itemUsable [ItemBase](#)

count [int](#)

canceled [bool](#)

## UpdateFromAddItem2(ItemUsable, bool)

```
[Obsolete("Use UpdateFromAddItem")]
public void UpdateFromAddItem2(ItemUsable itemUsable, bool canceled)
```

## Parameters

itemUsable [ItemUsable](#)

canceled [bool](#)

## UpdateFromCombination(ItemUsable)

```
public void UpdateFromCombination(ItemUsable itemUsable)
```

## Parameters

itemUsable [ItemUsable](#)

## UpdateFromCombination2(ItemUsable)

```
public void UpdateFromCombination2(ItemUsable itemUsable)
```

## Parameters

itemUsable [ItemUsable](#)

## UpdateFromItemEnhancement(Equipment)

```
public void UpdateFromItemEnhancement(Equipment equipment)
```

## Parameters

equipment [Equipment](#)

## UpdateFromItemEnhancement2(Equipment)

```
public void UpdateFromItemEnhancement2(Equipment equipment)
```

### Parameters

equipment [Equipment](#)

## UpdateFromQuestReward(Quest, MaterialItemSheet)

```
public void UpdateFromQuestReward(Quest quest, MaterialItemSheet materialItemSheet)
```

### Parameters

quest [Quest](#)

materialItemSheet [MaterialItemSheet](#)

## UpdateFromQuestReward2(Quest, MaterialItemSheet)

[Obsolete("Use UpdateFromQuestReward")]

```
public void UpdateFromQuestReward2(Quest quest, MaterialItemSheet materialItemSheet)
```

### Parameters

quest [Quest](#)

materialItemSheet [MaterialItemSheet](#)

## UpdateFromRapidCombination(ResultModel, long)

```
public void UpdateFromRapidCombination(CombinationConsumable5.ResultModel result,  
long requiredIndex)
```

### Parameters

result [CombinationConsumable5.ResultModel](#)

requiredIndex [long](#)

## UpdateFromRapidCombinationV2(ResultModel, long)

```
public void UpdateFromRapidCombinationV2(RapidCombination5.ResultModel result,  
long requiredIndex)
```

### Parameters

result [RapidCombination5.ResultModel](#)

requiredIndex [long](#)

## UpdateGeneralQuest(IEnumerable<QuestEventType>)

```
public void UpdateGeneralQuest(IEnumerable<QuestEventType> types)
```

### Parameters

types [IEnumerable](#)<QuestEventType>

## UpdateQuestRewards(MaterialItemSheet)

```
public void UpdateQuestRewards(MaterialItemSheet materialItemSheet)
```

### Parameters

materialItemSheet [MaterialItemSheet](#)

## UpdateQuestRewards2(MaterialItemSheet)

```
[Obsolete("Use UpdateQuestRewards")]
```

```
public void UpdateQuestRewards2(MaterialItemSheet materialItemSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

## UpdateTemp(Mail, long)

```
[Obsolete("No longer in use.")]  
public void UpdateTemp(Mail mail, long currentBlockIndex)
```

## Parameters

mail [Mail](#)

currentBlockIndex [long](#)

## ValidateConsumable(List<Guid>, long)

```
public List<int> ValidateConsumable(List<Guid> consumableIds,  
long currentBlockIndex)
```

## Parameters

consumableIds [List](#)<[Guid](#)>

currentBlockIndex [long](#)

## Returns

[List](#)<[int](#)>

## ValidateConsumableV2(List<Guid>, long, GameConfigState)

```
public List<int> ValidateConsumableV2(List<Guid> consumableIds, long currentBlockIndex, GameConfigState gameConfigState)
```

## Parameters

consumableIds [List](#)<[Guid](#)>

currentBlockIndex [long](#)

gameConfigState [GameConfigState](#)

## Returns

[List](#)<[int](#)>

## ValidateCostume(HashSet<int>)

```
public void ValidateCostume(HashSet<int> costumeIds)
```

## Parameters

costumeIds [HashSet](#)<[int](#)>

## ValidateCostume(IEnumerable<Guid>)

```
public List<int> ValidateCostume(IEnumerable<Guid> costumeIds)
```

## Parameters

costumeIds [IEnumerable](#)<[Guid](#)>

## Returns

[List](#)<[int](#)>

## ValidateCostumeV2(IEnumerable<Guid>, GameConfigState)

```
public List<Costume> ValidateCostumeV2(IEnumerable<Guid> costumeIds, GameConfigState gameConfigState)
```

### Parameters

costumeIds [IEnumerable<Guid>](#)

gameConfigState [GameConfigState](#)

### Returns

[List<Costume>](#)

## ValidateEquipments(List<Guid>, long)

```
public void ValidateEquipments(List<Guid> equipmentIds, long blockIndex)
```

### Parameters

equipmentIds [List<Guid>](#)

blockIndex [long](#)

## ValidateEquipmentsV2(List<Guid>, long)

```
public List<Equipment> ValidateEquipmentsV2(List<Guid> equipmentIds, long blockIndex)
```

### Parameters

equipmentIds [List<Guid>](#)

blockIndex [long](#)

Returns

[List](#)<[Equipment](#)>

## ValidateEquipmentsV3(List<Guid>, long, GameConfigState)

```
public List<Equipment> ValidateEquipmentsV3(List<Guid> equipmentIds, long  
blockIndex, GameConfigState gameConfigState)
```

Parameters

equipmentIds [List](#)<[Guid](#)>

blockIndex [long](#)

gameConfigState [GameConfigState](#)

Returns

[List](#)<[Equipment](#)>

## ValidateItemRequirement(List<int>, List<Equipment>, ItemRequirementSheet, EquipmentItemRecipeSheet, EquipmentItemSubRecipeSheetV2, EquipmentItemOptionSheet, string)

```
public void ValidateItemRequirement(List<int> itemIds, List<Equipment> equipments,  
ItemRequirementSheet requirementSheet, EquipmentItemRecipeSheet recipeSheet,  
EquipmentItemSubRecipeSheetV2 subRecipeSheet, EquipmentItemOptionSheet  
itemOptionSheet, string addressesHex)
```

Parameters

itemIds [List](#)<[int](#)>

equipments [List](#)<[Equipment](#)>

requirementSheet [ItemRequirementSheet](#)

recipeSheet [EquipmentItemRecipeSheet](#)

subRecipeSheet [EquipmentItemSubRecipeSheetV2](#)

itemOptionSheet [EquipmentItemOptionSheet](#)

addressesHex [string](#) ↴

# Class CollectionState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

Represents the state of a collection.

```
public class CollectionState : IBencodable
```

## Inheritance

[object](#) ← CollectionState

## Implements

IBencodable

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CollectionState()

```
public CollectionState()
```

### CollectionState(IValue)

```
public CollectionState(IValue bencoded)
```

## Parameters

bencoded IValue

# CollectionState(List)

```
public CollectionState(List serialized)
```

## Parameters

serialized List

## Fields

### Ids

```
public SortedSet<int> Ids
```

## Field Value

[SortedSet](#) <[int](#)>

## Properties

### Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

## Property Value

IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that

additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implemnetation may decide to only allow the canonical representation to be decoded.

## Methods

### GetModifiers(CollectionSheet)

```
public List<StatModifier> GetModifiers(CollectionSheet collectionSheet)
```

#### Parameters

collectionSheet [CollectionSheet](#)

#### Returns

[List](#) <[StatModifier](#)>

# Class CombinationSlotState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class CombinationSlotState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← CombinationSlotState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Extension Methods

[CombinationSlotStateExtensions.TryGetMail\(CombinationSlotState, long, long, out CombinationMail, out ItemEnhanceMail\)](#) ,  
[CombinationSlotStateExtensions.TryGetResultId\(CombinationSlotState, out Guid\)](#) ,  
[CombinationSlotStateExtensions.ValidateFromAction\(CombinationSlotState, long, AvatarState, int, string, string\)](#).

## Constructors

### CombinationSlotState(Dictionary)

```
public CombinationSlotState(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## CombinationSlotState(Address, int)

```
public CombinationSlotState(Address address, int index = 0)
```

### Parameters

address Address

index [int](#)

### Fields

#### DeriveFormat

```
public const string DeriveFormat = "combination-slot-{0}"
```

### Field Value

[string](#)

### Properties

#### Index

It is a CombinationSlot index. start from 0.

```
public int Index { get; set; }
```

### Property Value

[int](#)

#### IsUnlocked

```
public bool IsUnlocked { get; }
```

Property Value

[bool](#) ↗

PetId

`public int? PetId { get; }`

Property Value

[int](#) ↗?

RequiredBlockIndex

`public long RequiredBlockIndex { get; }`

Property Value

[long](#) ↗

Result

`public AttachmentActionResult Result { get; }`

Property Value

[AttachmentActionResult](#)

WorkCompleteBlockIndex

Serialize key is "unlockBlockIndex".

`public long WorkCompleteBlockIndex { get; }`

Property Value

[long](#) ↗

## WorkStartBlockIndex

Serialize key is "startBlockIndex".

```
public long WorkStartBlockIndex { get; }
```

Property Value

[long](#) ↗

## Methods

### DeriveAddress(Address, int)

Is only used for migration legacy state.

```
public static Address DeriveAddress(Address address, int slotIndex)
```

Parameters

address Address

slotIndex [int](#) ↗

Returns

Address

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Unlock()

```
public bool Unlock()
```

Returns

[bool](#)

## Update(AttachmentActionResult, long, long, int?)

```
public void Update(AttachmentActionResult result, long blockIndex, long  
workCompleteBlockIndex, int? petId = null)
```

Parameters

result [AttachmentActionResult](#)

blockIndex [long](#)

workCompleteBlockIndex [long](#)

petId [int](#)?

## Update(long)

```
public void Update(long blockIndex)
```

Parameters

blockIndex [long](#)

## Update(long, Material, int)

```
public void Update(long blockIndex, Material material, int count)
```

### Parameters

blockIndex [long](#)

material [Material](#)

count [int](#)

## UpdateV2(long, Material, int)

```
public void UpdateV2(long blockIndex, Material material, int count)
```

### Parameters

blockIndex [long](#)

material [Material](#)

count [int](#)

## Validate(AvatarState, long)

```
[Obsolete("Use ValidateV2")]
public bool Validate(AvatarState avatarState, long blockIndex)
```

### Parameters

avatarState [AvatarState](#)

blockIndex [long](#)

### Returns

[bool](#)

## ValidateSlotIndex(int)

```
public static bool ValidateSlotIndex(int index)
```

### Parameters

index [int](#)

### Returns

[bool](#)

## ValidateV2(long)

```
public bool ValidateV2(long blockIndex)
```

### Parameters

blockIndex [long](#)

### Returns

[bool](#)

# Class CreditsState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CreditsState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← CreditsState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CreditsState(Dictionary)

```
public CreditsState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### CreditsState(IEnumerable<string>)

```
public CreditsState(IEnumerable<string> names)
```

#### Parameters

names [IEnumerable](#)<[string](#)>

## Fields

### Address

`public static readonly Address Address`

### Field Value

Address

## Properties

### Names

`public ImmutableList<string> Names { get; }`

### Property Value

[ImmutableList](#)<[string](#)>

## Methods

### Serialize()

`public override IValue Serialize()`

### Returns

IValue

# Class CrystalCostState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CrystalCostState : IState
```

## Inheritance

[object](#) ← CrystalCostState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CrystalCostState(Address, List)

```
public CrystalCostState(Address address, List serialized)
```

#### Parameters

address Address

serialized List

### CrystalCostState(Address, FungibleAssetValue)

```
public CrystalCostState(Address address, FungibleAssetValue crystal)
```

#### Parameters

`address` Address

`crystal` FungibleAssetValue

## Fields

### Address

`public` Address Address

#### Field Value

Address

### CRYSTAL

`public` FungibleAssetValue CRYSTAL

#### Field Value

FungibleAssetValue

### Count

`public int` Count

#### Field Value

[int](#)

### DailyIntervalIndex

`public const long` DailyIntervalIndex = 7200

Field Value

[long](#) ↗

## Methods

### Serialize()

`public IValue Serialize()`

Returns

IValue

# Class CrystalRandomSkillState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class CrystalRandomSkillState : IState
```

## Inheritance

[object](#) ← CrystalRandomSkillState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### CrystalRandomSkillState(Address, List)

```
public CrystalRandomSkillState(Address address, List serialized)
```

#### Parameters

address Address

serialized List

### CrystalRandomSkillState(Address, int)

```
public CrystalRandomSkillState(Address address, int stageId)
```

#### Parameters

address Address

stageId [int](#)

## Properties

### Address

```
public Address Address { get; }
```

Property Value

Address

### SkillIds

```
public List<int> SkillIds { get; }
```

Property Value

[List](#)<[int](#)>

### StageId

```
public int StageId { get; }
```

Property Value

[int](#)

### StarCount

```
public int StarCount { get; }
```

Property Value

[int](#)

## Methods

### GetHighestRankSkill(CrystalRandomBuffSheet)

```
public int GetHighestRankSkill(CrystalRandomBuffSheet crystalRandomBuffSheet)
```

Parameters

crystalRandomBuffSheet [CrystalRandomBuffSheet](#)

Returns

[int](#)

### GetSkill(int, CrystalRandomBuffSheet, SkillSheet)

```
public static Skill GetSkill(int skillId, CrystalRandomBuffSheet  
crystalRandomBuffSheet, SkillSheet skillSheet)
```

Parameters

skillId [int](#)

crystalRandomBuffSheet [CrystalRandomBuffSheet](#)

skillSheet [SkillSheet](#)

Returns

[Skill](#)

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Update(List<int>)

```
public void Update(List<int> skillIds)
```

Parameters

skillIds [List](#)<[int](#)>

## Update(int, CrystalStageBuffGachaSheet)

```
public void Update(int gotStarCount, CrystalStageBuffGachaSheet sheet)
```

Parameters

gotStarCount [int](#)

sheet [CrystalStageBuffGachaSheet](#)

# Class DuplicateRedeemException

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicateRedeemException : InvalidOperationException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [InvalidOperationException](#) ←  
DuplicateRedeemException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

DuplicateRedeemException(SerializationInfo,  
StreamingContext)

```
public DuplicateRedeemException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicateRedeemException(string)

```
public DuplicateRedeemException(string s)
```

### Parameters

s [string](#)

# Class FailedToUnregisterInShopStateException

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class FailedToUnregisterInShopStateException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← FailedToUnregisterInShopStateException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### FailedToUnregisterInShopStateException(SerializationInfo, StreamingContext)

```
protected FailedToUnregisterInShopStateException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## FailedToUnregisterInShopStateException(string)

```
public FailedToUnregisterInShopStateException(string message)
```

### Parameters

message [string](#)

# Class GameConfigState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GameConfigState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← GameConfigState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GameConfigState()

```
public GameConfigState()
```

### GameConfigState(Dictionary)

```
public GameConfigState(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## GameConfigState(string)

```
public GameConfigState(string csv)
```

### Parameters

csv [string](#)

## Fields

### Address

```
public static readonly Address Address
```

### Field Value

Address

## Properties

### ActionPointMax

```
public int ActionPointMax { get; }
```

### Property Value

[int](#)

### AdventureBossActiveInterval

```
public long AdventureBossActiveInterval { get; }
```

### Property Value

[long](#) ↴

## AdventureBossClaimInterval

```
public long AdventureBossClaimInterval { get; }
```

Property Value

[long](#) ↴

## AdventureBossInactiveInterval

```
public long AdventureBossInactiveInterval { get; }
```

Property Value

[long](#) ↴

## AdventureBossMinBounty

```
public BigInteger AdventureBossMinBounty { get; }
```

Property Value

[BigInteger](#) ↴

## AdventureBossNcgApRatio

```
public decimal AdventureBossNcgApRatio { get; }
```

Property Value

[decimal](#) ↴

## AdventureBossNcgRuneRatio

```
public decimal AdventureBossNcgRuneRatio { get; }
```

Property Value

[decimal](#) ↗

## AdventureBossWantedRequiredStakingLevel

```
public int AdventureBossWantedRequiredStakingLevel { get; }
```

Property Value

[int](#) ↗

## BattleArenaInterval

```
public int BattleArenaInterval { get; }
```

Property Value

[int](#) ↗

## CustomEquipmentCraftIconCostMultiplier

```
public long CustomEquipmentCraftIconCostMultiplier { get; }
```

Property Value

[long](#) ↗

## DailyArenaInterval

```
public int DailyArenaInterval { get; }
```

Property Value

[int](#)

## DailyRewardInterval

```
public int DailyRewardInterval { get; }
```

Property Value

[int](#)

## DailyRuneRewardAmount

```
public int DailyRuneRewardAmount { get; }
```

Property Value

[int](#)

## DailyWorldBossInterval

```
public int DailyWorldBossInterval { get; }
```

Property Value

[int](#)

## HourglassPerBlock

```
public int HourglassPerBlock { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_ConsumableSlot1

```
public int RequireCharacterLevel_ConsumableSlot1 { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_ConsumableSlot2

```
public int RequireCharacterLevel_ConsumableSlot2 { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_ConsumableSlot3

```
public int RequireCharacterLevel_ConsumableSlot3 { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_ConsumableSlot4

```
public int RequireCharacterLevel_ConsumableSlot4 { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_ConsumableSlot5

```
public int RequireCharacterLevel_ConsumableSlot5 { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EarCostumeSlot

```
public int RequireCharacterLevel_EarCostumeSlot { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EquipmentSlotArmor

```
public int RequireCharacterLevel_EquipmentSlotArmor { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EquipmentSlotAura

```
public int RequireCharacterLevel_EquipmentSlotAura { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EquipmentSlotBelt

```
public int RequireCharacterLevel_EquipmentSlotBelt { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EquipmentSlotGrimoire

```
public int RequireCharacterLevel_EquipmentSlotGrimoire { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EquipmentSlotNecklace

```
public int RequireCharacterLevel_EquipmentSlotNecklace { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_EquipmentSlotRing1

```
public int RequireCharacterLevel_EquipmentSlotRing1 { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_EquipmentSlotRing2

```
public int RequireCharacterLevel_EquipmentSlotRing2 { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_EquipmentSlotWeapon

```
public int RequireCharacterLevel_EquipmentSlotWeapon { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_EyeCostumeSlot

```
public int RequireCharacterLevel_EyeCostumeSlot { get; }
```

Property Value

[int ↗](#)

## RequireCharacterLevel\_FullCostumeSlot

```
public int RequireCharacterLevel_FullCostumeSlot { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_HairCostumeSlot

```
public int RequireCharacterLevel_HairCostumeSlot { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_TailCostumeSlot

```
public int RequireCharacterLevel_TailCostumeSlot { get; }
```

Property Value

[int](#)

## RequireCharacterLevel\_TitleSlot

```
public int RequireCharacterLevel_TitleSlot { get; }
```

Property Value

[int](#)

## RuneSkillSlotCrystalUnlockCost

```
public int RuneSkillSlotCrystalUnlockCost { get; }
```

Property Value

[int](#)

## RuneSkillSlotUnlockCost

```
public int RuneSkillSlotUnlockCost { get; }
```

Property Value

[int](#)

## RuneStatSlotCrystalUnlockCost

```
public int RuneStatSlotCrystalUnlockCost { get; }
```

Property Value

[int](#)

## RuneStatSlotUnlockCost

```
public int RuneStatSlotUnlockCost { get; }
```

Property Value

[int](#)

## ShatterStrikeMaxDamage

```
public long ShatterStrikeMaxDamage { get; }
```

Property Value

[long](#) ↗

## StakeRegularFixedRewardSheet\_V2\_StartBlockIndex

```
public long StakeRegularFixedRewardSheet_V2_StartBlockIndex { get; }
```

Property Value

[long](#) ↗

## StakeRegularRewardSheet\_V2\_StartBlockIndex

```
public long StakeRegularRewardSheet_V2_StartBlockIndex { get; }
```

Property Value

[long](#) ↗

## StakeRegularRewardSheet\_V3\_StartBlockIndex

```
public long StakeRegularRewardSheet_V3_StartBlockIndex { get; }
```

Property Value

[long](#) ↗

## StakeRegularRewardSheet\_V4\_StartBlockIndex

```
public long StakeRegularRewardSheet_V4_StartBlockIndex { get; }
```

Property Value

[long](#) ↗

## StakeRegularRewardSheet\_V5\_StartBlockIndex

```
public long StakeRegularRewardSheet_V5_StartBlockIndex { get; }
```

Property Value

[long](#) ↗

## WeeklyArenaInterval

```
public int WeeklyArenaInterval { get; }
```

Property Value

[int](#) ↗

## WorldBossRequiredInterval

```
public int WorldBossRequiredInterval { get; }
```

Property Value

[int](#) ↗

## Methods

**Serialize()**

```
public override IValue Serialize()
```

Returns

IValue

## Set(GameConfigSheet)

```
public void Set(GameConfigSheet sheet)
```

Parameters

sheet [GameConfigSheet](#)

## Update(Row)

```
public void Update(GameConfigSheet.Row row)
```

Parameters

row [GameConfigSheet.Row](#)

# Class GoldBalanceState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GoldBalanceState : State, IState, ICloneable
```

## Inheritance

[object](#) ← [State](#) ← GoldBalanceState

## Implements

[IState](#), [ICloneable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GoldBalanceState(Dictionary)

```
public GoldBalanceState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### GoldBalanceState(IValue)

```
public GoldBalanceState(IValue serialized)
```

#### Parameters

**serialized** **IValue**

## GoldBalanceState(Address, FungibleAssetValue)

```
public GoldBalanceState(Address address, FungibleAssetValue gold)
```

### Parameters

**address** Address

**gold** FungibleAssetValue

### Fields

#### Gold

```
public readonly FungibleAssetValue Gold
```

### Field Value

FungibleAssetValue

### Methods

#### Add(FungibleAssetValue)

```
public GoldBalanceState Add(FungibleAssetValue adder)
```

### Parameters

**adder** FungibleAssetValue

### Returns

[GoldBalanceState](#)

## Clone()

Creates a new object that is a copy of the current instance.

```
public object Clone()
```

Returns

[object](#)

A new object that is a copy of this instance.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class GoldCurrencyState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GoldCurrencyState : State, IState, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← GoldCurrencyState

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### GoldCurrencyState(Dictionary)

```
public GoldCurrencyState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### GoldCurrencyState(Currency)

```
public GoldCurrencyState(Currency currency)
```

#### Parameters

`currency` Currency

## GoldCurrencyState(Currency, long)

`public GoldCurrencyState(Currency currency, long initialSupply)`

### Parameters

`currency` Currency

`initialSupply` [long](#)

## GoldCurrencyState(SerializationInfo, StreamingContext)

`protected GoldCurrencyState(SerializationInfo info, StreamingContext context)`

### Parameters

`info` [SerializationInfo](#)

`context` [StreamingContext](#)

## Fields

### Address

`public static readonly Address Address`

### Field Value

Address

## DEFAULT\_INITIAL\_SUPPLY

```
public const long DEFAULT_INITIAL_SUPPLY = 10000000000
```

Field Value

[long](#) ↗

## Properties

### Currency

```
public Currency Currency { get; }
```

Property Value

Currency

### InitialSupply

```
public long InitialSupply { get; }
```

Property Value

[long](#) ↗

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) ↗ with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

Parameters

`info` [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

`context` [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

# Class HammerPointState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class HammerPointState : IState
```

## Inheritance

[object](#) ← HammerPointState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### HammerPointState(Address, List)

```
public HammerPointState(Address address, List serialized)
```

#### Parameters

address Address

serialized List

### HammerPointState(Address, int)

```
public HammerPointState(Address address, int recipeId)
```

#### Parameters

address Address

recipeId [int](#)

## Properties

### Address

```
public Address Address { get; }
```

Property Value

Address

### HammerPoint

```
public int HammerPoint { get; }
```

Property Value

[int](#)

### RecipId

```
public int RecipeId { get; }
```

Property Value

[int](#)

## Methods

AddHammerPoint(int, CrystalHammerPointSheet)

```
public void AddHammerPoint(int point, CrystalHammerPointSheet sheet = null)
```

## Parameters

point [int](#)

sheet [CrystalHammerPointSheet](#)

## ResetHammerPoint()

```
public void ResetHammerPoint()
```

## Serialize()

```
public IValue Serialize()
```

## Returns

IValue

# Interface IState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public interface IState
```

## Methods

### Serialize()

```
IValue Serialize()
```

Returns

IValue

# Class InvalidRedeemCodeException

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class InvalidRedeemCodeException : KeyNotFoundException, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← [SystemException](#) ← [KeyNotFoundException](#) ←  
InvalidRedeemCodeException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### InvalidRedeemCodeException()

```
public InvalidRedeemCodeException()
```

### InvalidRedeemCodeException(SerializationInfo, StreamingContext)

```
public InvalidRedeemCodeException(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## InvalidRedeemCodeException(string)

public [InvalidRedeemCodeException](#)(string s)

Parameters

s [string](#)

# Class ItemSlotState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class ItemSlotState : IState
```

## Inheritance

[object](#) ← ItemSlotState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ItemSlotState(List)

```
public ItemSlotState(List serialized)
```

#### Parameters

serialized List

### ItemSlotState(BattleType)

```
public ItemSlotState(BattleType battleType)
```

#### Parameters

battleType [BattleType](#)

# Properties

## BattleType

```
public BattleType BattleType { get; }
```

### Property Value

[BattleType](#)

# Costumes

```
public List<Guid> Costumes { get; }
```

### Property Value

[List<Guid>](#)

# Equipments

```
public List<Guid> Equipments { get; }
```

### Property Value

[List<Guid>](#)

# Methods

## DeriveAddress(Address, BattleType)

```
public static Address DeriveAddress(Address avatarAddress, BattleType battleType)
```

### Parameters

avatarAddress Address

battleType [BattleType](#)

Returns

Address

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## UpdateCostumes(List<Guid>)

```
public void UpdateCostumes(List<Guid> costumes)
```

Parameters

costumes [List](#)<[Guid](#)>

## UpdateEquipment(List<Guid>)

```
public void UpdateEquipment(List<Guid> equipments)
```

Parameters

equipments [List](#)<[Guid](#)>

# Class LazyState<TState, TEncoding>

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public sealed class LazyState<TState, TEncoding> : IState, ISerializable where
TState : IState where TEncoding : IValue
```

## Type Parameters

TState

TEncoding

## Inheritance

[object](#) ← LazyState<TState, TEncoding>

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.ReferenceEquals\(object, object\)](#), [object.ToString\(\)](#)

## Constructors

### LazyState(TState)

```
public LazyState(TState loadedValue)
```

## Parameters

loadedValue TState

### LazyState(TEncoding, Func<TEncoding, TState>)

```
public LazyState(TEncoding serialized, Func<TEncoding, TState> loader)
```

## Parameters

serialized TEncoding

loader [Func](#)<TEncoding, TState>

## Properties

### State

```
public TState State { get; }
```

## Property Value

TState

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

context [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

## [SecurityException](#)

The caller does not have the required permission.

## GetStateOrSerializedEncoding(out TState, out TEncoding)

```
public bool GetStateOrSerializedEncoding(out TState loadedState, out  
TEncoding serialized)
```

### Parameters

`loadedState` TState

`serialized` TEncoding

### Returns

`bool`

## LoadState(LazyState<TState, TEncoding>)

```
public static TState LoadState(LazyState<TState, TEncoding> lazyState)
```

### Parameters

`lazyState` [LazyState](#)<TState, TEncoding>

### Returns

TState

## LoadStatePair<T>(KeyValuePair<T, LazyState<TState, TEncoding>>)

```
public static KeyValuePair<T, TState> LoadStatePair<T>(KeyValuePair<T,  
LazyState<TState, TEncoding>> lazyPair)
```

## Parameters

lazyPair [KeyValuePair](#)<T, [LazyState](#)<TState, TEncoding>>

## Returns

[KeyValuePair](#)<T, TState>

## Type Parameters

T

## Serialize()

```
public IValue Serialize()
```

## Returns

IValue

# Class LegacyStakeState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class LegacyStakeState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← LegacyStakeState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### LegacyStakeState(Dictionary)

```
public LegacyStakeState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### LegacyStakeState(Address, long)

```
public LegacyStakeState(Address address, long startedBlockIndex)
```

#### Parameters

**address** Address

startedBlockIndex [long](#)

## LegacyStakeState(Address, long, long, long, StakeAchievements)

```
public LegacyStakeState(Address address, long startedBlockIndex, long receivedBlockIndex, long cancellableBlockIndex, LegacyStakeState.StakeAchievements achievements)
```

### Parameters

address Address

startedBlockIndex [long](#)

receivedBlockIndex [long](#)

cancellableBlockIndex [long](#)

achievements [LegacyStakeState.StakeAchievements](#)

### Fields

#### CurrencyAsRewardStartIndex

```
public const long CurrencyAsRewardStartIndex = 6910000
```

#### Field Value

[long](#)

#### LockupInterval

```
public const long LockupInterval = 201600
```

#### Field Value

[long](#) ↴

## RewardInterval

```
public const long RewardInterval = 50400
```

Field Value

[long](#) ↴

## StakeRewardSheetV2Index

```
public const long StakeRewardSheetV2Index = 6700000
```

Field Value

[long](#) ↴

## StakeRewardSheetV3Index

```
public const long StakeRewardSheetV3Index = 7650000
```

Field Value

[long](#) ↴

## Properties

### Achievements

```
public LegacyStakeState.StakeAchievements Achievements { get; }
```

Property Value

## CancellableBlockIndex

```
public long CancellableBlockIndex { get; }
```

Property Value

[long](#) ↗

## ReceivedBlockIndex

```
public long ReceivedBlockIndex { get; }
```

Property Value

[long](#) ↗

## StartedBlockIndex

```
public long StartedBlockIndex { get; }
```

Property Value

[long](#) ↗

## Methods

### CalculateAccumulatedCurrencyCrystalRewards(long, out int, out int, out int)

```
public int CalculateAccumulatedCurrencyCrystalRewards(long blockIndex, out int v1Step, out int v2Step, out int v3Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

v3Step [int](#)

Returns

[int](#)

CalculateAccumulatedCurrencyRewards(long, out int,  
out int, out int)

```
public int CalculateAccumulatedCurrencyRewards(long blockIndex, out int v1Step, out  
int v2Step, out int v3Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

v3Step [int](#)

Returns

[int](#)

CalculateAccumulatedCurrencyRewardsV1(long, out int,  
out int)

```
[Obsolete("Use CalculateAccumulatedCurrencyRewards() instead.")]  
public int CalculateAccumulatedCurrencyRewardsV1(long blockIndex, out int v1Step,
```

```
    out int v2Step)
```

## Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

## Returns

[int](#)

## CalculateAccumulatedCurrencyRewardsV2(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedCurrencyRewards() instead.")]
```

```
public int CalculateAccumulatedCurrencyRewardsV2(long blockIndex, out int v1Step,  
out int v2Step)
```

## Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

## Returns

[int](#)

## CalculateAccumulatedItemRewards(long, out int, out int, out int)

```
public int CalculateAccumulatedItemRewards(long blockIndex, out int v1Step, out int  
v2Step, out int v3Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

v3Step [int](#)

Returns

[int](#)

## CalculateAccumulatedItemRewardsV1(long)

```
[Obsolete("Use CalculateAccumulatedItemRewards() instead.")]
public int CalculateAccumulatedItemRewardsV1(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[int](#)

## CalculateAccumulatedItemRewardsV1(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedItemRewards() instead.")]
public int CalculateAccumulatedItemRewardsV1(long blockIndex, out int v1Step, out
int v2Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

Returns

[int](#)

## CalculateAccumulatedItemRewardsV2(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedItemRewards() instead.")]
public int CalculateAccumulatedItemRewardsV2(long blockIndex, out int v1Step, out
int v2Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

Returns

[int](#)

## CalculateAccumulatedItemRewardsV3(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedItemRewards() instead.")]
public int CalculateAccumulatedItemRewardsV3(long blockIndex, out int v1Step, out
int v2Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

Returns

[int](#)

## CalculateAccumulatedRuneRewards(long)

```
public int CalculateAccumulatedRuneRewards(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[int](#)

## CalculateAccumulatedRuneRewards(long, out int, out int, out int)

```
public int CalculateAccumulatedRuneRewards(long blockIndex, out int v1Step, out int v2Step, out int v3Step)
```

Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

v3Step [int](#)

Returns

[int](#)

## CalculateAccumulatedRuneRewardsV1(long)

```
[Obsolete("Use CalculateAccumulatedRuneRewards() instead.")]
public int CalculateAccumulatedRuneRewardsV1(long blockIndex)
```

### Parameters

blockIndex [long](#)

### Returns

[int](#)

## CalculateAccumulatedRuneRewardsV1(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedRuneRewards() instead.")]
public int CalculateAccumulatedRuneRewardsV1(long blockIndex, out int v1Step, out
int v2Step)
```

### Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

### Returns

[int](#)

## CalculateAccumulatedRuneRewardsV2(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedRuneRewards() instead.")]
public int CalculateAccumulatedRuneRewardsV2(long blockIndex, out int v1Step, out
```

```
int v2Step)
```

## Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

## Returns

[int](#)

## CalculateAccumulatedRuneRewardsV3(long, out int, out int)

```
[Obsolete("Use CalculateAccumulatedRuneRewards() instead.")]
```

```
public int CalculateAccumulatedRuneRewardsV3(long blockIndex, out int v1Step, out int v2Step)
```

## Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

## Returns

[int](#)

## Claim(long)

```
public void Claim(long blockIndex)
```

## Parameters

blockIndex [long](#)

## DeriveAddress(Address)

```
public static Address DeriveAddress(Address agentAddress)
```

Parameters

agentAddress [Address](#)

Returns

[Address](#)

## GetClaimableBlockIndex(long)

```
public long GetClaimableBlockIndex(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[long](#)

## GetRewardStep(long, long?)

Calculate accumulated rewards step.

```
public int GetRewardStep(long currentBlockIndex, long? rewardStartBlockIndex)
```

Parameters

currentBlockIndex [long](#)

The block index of the current block.

**rewardStartBlockIndex** [long](#)?

The block index of the reward start block. If not null, the return value is calculated differently. StakeStateTest.GetRewardStep()

Returns

[int](#)

The accumulated rewards step.

## IsCancellable(long)

`public bool IsCancellable(long blockIndex)`

Parameters

**blockIndex** [long](#)

Returns

[bool](#)

## IsClaimable(long)

`public bool IsClaimable(long blockIndex)`

Parameters

**blockIndex** [long](#)

Returns

[bool](#)

## IsClaimable(long, out int, out int)

```
public bool IsClaimable(long blockIndex, out int v1Step, out int v2Step)
```

### Parameters

blockIndex [long](#)

v1Step [int](#)

v2Step [int](#)

### Returns

[bool](#)

## Serialize()

```
public override IValue Serialize()
```

### Returns

IValue

## SerializeV2()

```
public IValue SerializeV2()
```

### Returns

IValue

# Class LegacyStakeState.StakeAchievements

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class LegacyStakeState.StakeAchievements
```

## Inheritance

[object](#) ← LegacyStakeState.StakeAchievements

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### StakeAchievements(Dictionary)

```
public StakeAchievements(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### StakeAchievements(Dictionary<int, int>)

```
public StakeAchievements(Dictionary<int, int> achievements = null)
```

#### Parameters

achievements [Dictionary](#)<[int](#), [int](#)>

# Methods

## Achieve(int, int)

```
public void Achieve(int level, int step)
```

### Parameters

level [int](#)

step [int](#)

## Check(int, int)

```
public bool Check(int level, int step)
```

### Parameters

level [int](#)

step [int](#)

### Returns

[bool](#)

## Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Class MarketState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class MarketState
```

## Inheritance

[object](#) ← MarketState

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### MarketState()

```
public MarketState()
```

### MarketState(IValue)

```
public MarketState(IValue rawList)
```

## Parameters

rawList IValue

## Fields

### AvatarAddresses

```
public List<Address> AvatarAddresses
```

Field Value

[List](#)<Address>

Methods

Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class MonsterCollectionResult

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionResult : AttachmentActionResult, IState
```

## Inheritance

[object](#) ← [AttachmentActionResult](#) ← MonsterCollectionResult

## Implements

[IState](#)

## Inherited Members

[AttachmentActionResult.itemUsable](#) , [AttachmentActionResult.costume](#) ,  
[AttachmentActionResult.tradeableFungibleItem](#) ,  
[AttachmentActionResult.tradeableFungibleItemCount](#) ,  
[AttachmentActionResult.SerializeBackup1\(\)](#) ,  
[AttachmentActionResult.Deserialize\(Dictionary\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MonsterCollectionResult(Dictionary)

```
public MonsterCollectionResult(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### MonsterCollectionResult(Guid, Address, List<RewardInfo>)

```
public MonsterCollectionResult(Guid guid, Address address,  
List<MonsterCollectionRewardSheet.RewardInfo> rewardInfos)
```

## Parameters

guid [Guid](#)

address Address

rewardInfos [List](#)<[MonsterCollectionRewardSheet.RewardInfo](#)>

## Fields

### avatarAddress

```
public Address avatarAddress
```

#### Field Value

Address

### id

```
public Guid id
```

#### Field Value

[Guid](#)

### rewards

```
public List<MonsterCollectionRewardSheet.RewardInfo> rewards
```

#### Field Value

[List](#) <[MonsterCollectionRewardSheet.RewardInfo](#)>

## Properties

### TypeId

`protected override string TypeId { get; }`

### Property Value

[string](#)

## Methods

### Serialize()

`public override IValue Serialize()`

### Returns

[IValue](#)

# Class MonsterCollectionState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← MonsterCollectionState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MonsterCollectionState(Dictionary)

```
public MonsterCollectionState(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### MonsterCollectionState(Address, int, long)

```
public MonsterCollectionState(Address address, int level, long blockIndex)
```

#### Parameters

address Address

level [int](#)

blockIndex [long](#)

## MonsterCollectionState(Address, int, long, MonsterCollectionRewardSheet)

```
public MonsterCollectionState(Address address, int level, long blockIndex,  
MonsterCollectionRewardSheet monsterCollectionRewardSheet)
```

### Parameters

address Address

level [int](#)

blockIndex [long](#)

monsterCollectionRewardSheet [MonsterCollectionRewardSheet](#)

### Fields

#### DeriveFormat

```
public const string DeriveFormat = "monster-collection-{0}"
```

#### Field Value

[string](#)

#### LockUpInterval

```
public const long LockUpInterval = 201600
```

#### Field Value

[long](#) ↴

## RewardInterval

```
public const long RewardInterval = 50400
```

Field Value

[long](#) ↴

## Properties

### ExpiredBlockIndex

```
public long ExpiredBlockIndex { get; }
```

Property Value

[long](#) ↴

### Level

```
public int Level { get; }
```

Property Value

[int](#) ↴

### ReceivedBlockIndex

```
public long ReceivedBlockIndex { get; }
```

Property Value

[long](#) ↴

## RewardLevel

`public long RewardLevel { get; }`

Property Value

[long](#) ↴

## StartedBlockIndex

`public long StartedBlockIndex { get; }`

Property Value

[long](#) ↴

## Methods

### CalculateRewards(MonsterCollectionRewardSheet, long)

`public List<MonsterCollectionRewardSheet.RewardInfo> calculateRewards(MonsterCollectionRewardSheet sheet, long blockIndex)`

Parameters

`sheet` [MonsterCollectionRewardSheet](#)

`blockIndex` [long](#) ↴

Returns

[List](#) ↴<[MonsterCollectionRewardSheet.RewardInfo](#)>

## CalculateStep(long)

```
public int CalculateStep(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[int](#)

## Claim(long)

```
public void Claim(long blockIndex)
```

Parameters

blockIndex [long](#)

## DeriveAddress(Address, int)

```
public static Address DeriveAddress(Address baseAddress, int round)
```

Parameters

baseAddress Address

round [int](#)

Returns

Address

## Equals(MonsterCollectionState)

```
protected bool Equals(MonsterCollectionState other)
```

Parameters

other [MonsterCollectionState](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## IsLocked(long)

```
public bool IsLocked(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[bool](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class MonsterCollectionState0

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Obsolete("This class is reserved for backwards compatibility with  
MonsterCollection0 only.", false)]  
[Serializable]  
public class MonsterCollectionState0 : State, IState
```

## Inheritance

[object](#) ← [State](#) ← MonsterCollectionState0

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### MonsterCollectionState0(Dictionary)

```
public MonsterCollectionState0(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### MonsterCollectionState0(Address, int, long)

```
public MonsterCollectionState0(Address address, int level, long blockIndex)
```

#### Parameters

address Address

level [int](#)

blockIndex [long](#)

## MonsterCollectionState0(Address, int, long, MonsterCollectionRewardSheet)

```
public MonsterCollectionState0(Address address, int level, long blockIndex,  
MonsterCollectionRewardSheet monsterCollectionRewardSheet)
```

### Parameters

address Address

level [int](#)

blockIndex [long](#)

monsterCollectionRewardSheet [MonsterCollectionRewardSheet](#)

### Fields

#### DeriveFormat

```
public const string DeriveFormat = "monster-collection-{0}"
```

#### Field Value

[string](#)

#### ExpirationIndex

```
public const long ExpirationIndex = 201600
```

Field Value

[long](#) ↗

## LockUpInterval

```
public const long LockUpInterval = 201600
```

Field Value

[long](#) ↗

## RewardCapacity

```
public const int RewardCapacity = 4
```

Field Value

[int](#) ↗

## RewardInterval

```
public const long RewardInterval = 50400
```

Field Value

[long](#) ↗

## Properties

### End

```
public bool End { get; }
```

Property Value

[bool](#) ↗

## ExpiredBlockIndex

```
public long ExpiredBlockIndex { get; }
```

Property Value

[long](#) ↗

## Level

```
public int Level { get; }
```

Property Value

[int](#) ↗

## ReceivedBlockIndex

```
public long ReceivedBlockIndex { get; }
```

Property Value

[long](#) ↗

## RewardLevel

```
public long RewardLevel { get; }
```

Property Value

[long](#)

## RewardLevelMap

```
public Dictionary<long, List<MonsterCollectionRewardSheet.RewardInfo>>
RewardLevelMap { get; }
```

Property Value

[Dictionary](#)<[long](#), [List](#)<[MonsterCollectionRewardSheet.RewardInfo](#)>>

## RewardMap

```
public Dictionary<long, MonsterCollectionResult> RewardMap { get; }
```

Property Value

[Dictionary](#)<[long](#), [MonsterCollectionResult](#)>

## StartedBlockIndex

```
public long StartedBlockIndex { get; }
```

Property Value

[long](#)

## Methods

### CanReceive(long)

```
public bool CanReceive(long blockIndex)
```

Parameters

**blockIndex** [long](#)

Returns

[bool](#)

## DeriveAddress(Address, int)

```
public static Address DeriveAddress(Address baseAddress, int collectRound)
```

Parameters

**baseAddress** Address

**collectRound** [int](#)

Returns

Address

## Equals(MonsterCollectionState0)

```
protected bool Equals(MonsterCollectionState0 other)
```

Parameters

**other** [MonsterCollectionState0](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

**obj** [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## GetRewardLevel(long)

```
public long GetRewardLevel(long blockIndex)
```

Parameters

**blockIndex** [long](#)

Returns

[long](#)

## IsLock(long)

```
public bool IsLock(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[bool](#)

## Receive(long)

```
public void Receive(long blockIndex)
```

Parameters

blockIndex [long](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## SerializeV2()

```
public IValue SerializeV2()
```

Returns

IValue

## Update(int, long, MonsterCollectionRewardSheet)

```
public void Update(int level, long rewardLevel, MonsterCollectionRewardSheet  
monsterCollectionRewardSheet)
```

### Parameters

level [int](#)

rewardLevel [long](#)

monsterCollectionRewardSheet [MonsterCollectionRewardSheet](#)

## UpdateRewardMap(long, MonsterCollectionResult, long)

```
public void UpdateRewardMap(long rewardLevel, MonsterCollectionResult avatarAddress,  
long blockIndex)
```

### Parameters

rewardLevel [long](#)

avatarAddress [MonsterCollectionResult](#)

blockIndex [long](#)

# Class PendingActivationState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PendingActivationState : State, IState, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← PendingActivationState

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### PendingActivationState(Dictionary)

```
public PendingActivationState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### PendingActivationState(byte[], PublicKey)

```
public PendingActivationState(byte[] nonce, PublicKey publicKey)
```

#### Parameters

nonce [byte\[\]](#)

publicKey PublicKey

## PendingActivationState(SerializationInfo, StreamingContext)

**protected PendingActivationState**(SerializationInfo info, StreamingContext context)

### Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### Nonce

**public byte[] Nonce { get; }**

### Property Value

[byte\[\]](#)

### PublicKey

**public PublicKey PublicKey { get; }**

### Property Value

PublicKey

## Methods

# GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

**info** [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

**context** [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

# Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

# Verify(byte[])

```
public bool Verify(byte[] signature)
```

## Parameters

**signature** [byte](#)[]

## Returns

[bool](#) ↗

# Class PetState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class PetState : IState
```

## Inheritance

[object](#) ← PetState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### PetState(List)

```
public PetState(List serialized)
```

#### Parameters

serialized List

### PetState(int)

```
public PetState(int petId)
```

#### Parameters

petId [int](#)

# Properties

## Level

```
public int Level { get; }
```

### Property Value

[int](#)

## PetId

```
public int PetId { get; }
```

### Property Value

[int](#)

## UnlockedBlockIndex

```
public long UnlockedBlockIndex { get; }
```

### Property Value

[long](#)

## Methods

### DeriveAddress(Address, int)

```
public static Address DeriveAddress(Address avatarAddress, int petId)
```

### Parameters

avatarAddress Address

petId [int](#)

Returns

Address

## LevelUp()

```
public void LevelUp()
```

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Update(long)

```
public void Update(long unlockedIndex)
```

Parameters

unlockedIndex [long](#)

## Validate(long)

```
public bool Validate(long blockIndex)
```

Parameters

`blockIndex` [long ↗](#)

Returns

[bool ↗](#)

# Class RaiderState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RaiderState : IState
```

## Inheritance

[object](#) ← RaiderState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RaiderState()

```
public RaiderState()
```

### RaiderState(List)

```
public RaiderState(List rawState)
```

## Parameters

rawState List

## Fields

## AvatarAddress

```
public Address AvatarAddress
```

Field Value

Address

## AvatarName

```
public string AvatarName
```

Field Value

[string](#)

## ClaimedBlockIndex

```
public long ClaimedBlockIndex
```

Field Value

[long](#)

## Cp

```
public int Cp
```

Field Value

[int](#)

## HighScore

```
public long HighScore
```

Field Value

[long](#)

## IconId

```
public int IconId
```

Field Value

[int](#)

## LatestBossLevel

```
public int LatestBossLevel
```

Field Value

[int](#)

## LatestRewardRank

```
public int LatestRewardRank
```

Field Value

[int](#)

## Level

```
public int Level
```

Field Value

[int](#)

## PurchaseCount

```
public int PurchaseCount
```

Field Value

[int](#)

## RefillBlockIndex

```
public long RefillBlockIndex
```

Field Value

[long](#)

## RemainChallengeCount

```
public int RemainChallengeCount
```

Field Value

[int](#)

## TotalChallengeCount

```
public int TotalChallengeCount
```

Field Value

[int](#)

## TotalScore

```
public long TotalScore
```

Field Value

[long](#)

## UpdatedBlockIndex

```
public long UpdatedBlockIndex
```

Field Value

[long](#)

## Methods

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

### Update(AvatarState, int, long, bool, long)

```
public void Update(AvatarState avatarState, int cp, long score, bool payNcg,  
long blockIndex)
```

## Parameters

avatarState [AvatarState](#)

cp [int](#)

score [long](#)

payNcg [bool](#)

blockIndex [long](#)

# Class RankingInfo

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class RankingInfo : IState
```

## Inheritance

[object](#) ← RankingInfo

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RankingInfo(Dictionary)

```
public RankingInfo(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### RankingInfo(AvatarState)

```
public RankingInfo(AvatarState avatarState)
```

#### Parameters

avatarState [AvatarState](#)

# Fields

## AgentAddress

```
public readonly Address AgentAddress
```

### Field Value

Address

## ArmorId

```
public readonly int ArmorId
```

### Field Value

[int](#)

## AvatarAddress

```
public readonly Address AvatarAddress
```

### Field Value

Address

## AvatarName

```
public readonly string AvatarName
```

### Field Value

[string](#)

## Exp

```
public readonly long Exp
```

Field Value

[long](#) ↗

## Level

```
public readonly int Level
```

Field Value

[int](#) ↗

## StageClearedBlockIndex

```
public readonly long StageClearedBlockIndex
```

Field Value

[long](#) ↗

## UpdatedAt

```
public readonly long UpdatedAt
```

Field Value

[long](#) ↗

## Methods

## Equals(RankingInfo)

```
protected bool Equals(RankingInfo other)
```

Parameters

other [RankingInfo](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int ↗](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class RankingMapState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class RankingMapState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← RankingMapState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RankingMapState(Dictionary)

```
public RankingMapState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### RankingMapState(Address)

```
public RankingMapState(Address address)
```

#### Parameters

**address** Address

# Fields

## Capacity

```
public const int Capacity = 500
```

## Field Value

[int](#)

# Methods

## GetRankingInfos(long?)

```
public List<RankingInfo> GetRankingInfos(long? blockOffset)
```

## Parameters

blockOffset [long](#)?

## Returns

[List](#)<[RankingInfo](#)>

## Serialize()

```
public override IValue Serialize()
```

## Returns

IValue

## Update(AvatarState)

```
public void Update(AvatarState state)
```

## Parameters

state [AvatarState](#)

# Class RankingState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RankingState : State, IState, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← RankingState

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RankingState()

```
public RankingState()
```

### RankingState(Dictionary)

```
public RankingState(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

# RankingState(SerializationInfo, StreamingContext)

```
public RankingState(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

### Address

```
public static readonly Address Address
```

#### Field Value

Address

### RankingMapCapacity

```
public const int RankingMapCapacity = 150
```

#### Field Value

[int](#)

## Properties

### RankingMap

```
public Dictionary<Address, ImmutableHashSet<Address>> RankingMap { get; }
```

## Property Value

[Dictionary](#)<Address, [ImmutableHashSet](#)<Address>>

## Methods

### Derive(int)

```
public static Address Derive(int index)
```

#### Parameters

index [int](#)

#### Returns

Address

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

#### Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

context [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

#### Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## UpdateRankingMap(Address)

```
public Address UpdateRankingMap(Address avatarAddress)
```

Parameters

avatarAddress Address

Returns

Address

# Class RankingState0

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RankingState0 : State, IState, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← RankingState0

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RankingState0()

```
public RankingState0()
```

### RankingState0(Dictionary)

```
public RankingState0(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

# RankingState0(SerializationInfo, StreamingContext)

```
public RankingState0(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

### Address

```
public static readonly Address Address
```

#### Field Value

Address

### RankingMapCapacity

```
public const int RankingMapCapacity = 100
```

#### Field Value

[int](#)

## Properties

### RankingMap

```
public Dictionary<Address, ImmutableHashSet<Address>> RankingMap { get; }
```

## Property Value

[Dictionary](#)<Address, [ImmutableHashSet](#)<Address>>

## Methods

### Derive(int)

```
public static Address Derive(int index)
```

#### Parameters

index [int](#)

#### Returns

Address

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

#### Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

context [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

#### Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## UpdateRankingMap(Address)

```
public Address UpdateRankingMap(Address avatarAddress)
```

Parameters

avatarAddress Address

Returns

Address

# Class RankingState1

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RankingState1 : State, IState, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← RankingState1

## Implements

[IState](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RankingState1()

```
public RankingState1()
```

### RankingState1(Dictionary)

```
public RankingState1(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

# RankingState1(SerializationInfo, StreamingContext)

```
public RankingState1(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

### Address

```
public static readonly Address Address
```

#### Field Value

Address

### RankingMapCapacity

```
public const int RankingMapCapacity = 150
```

#### Field Value

[int](#)

## Properties

### RankingMap

```
public Dictionary<Address, ImmutableHashSet<Address>> RankingMap { get; }
```

## Property Value

[Dictionary](#)<Address, [ImmutableHashSet](#)<Address>>

## Methods

### Derive(int)

```
public static Address Derive(int index)
```

#### Parameters

index [int](#)

#### Returns

Address

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

#### Parameters

info [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

context [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

#### Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## UpdateRankingMap(Address)

```
public Address UpdateRankingMap(Address avatarAddress)
```

Parameters

avatarAddress Address

Returns

Address

# Class RedeemCodeState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RedeemCodeState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← RedeemCodeState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RedeemCodeState(Dictionary)

```
public RedeemCodeState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### RedeemCodeState(RedeemCodeListSheet)

```
public RedeemCodeState(RedeemCodeListSheet sheet)
```

#### Parameters

sheet [RedeemCodeListSheet](#)

## RedeemCodeState(Dictionary<PublicKey, Reward>)

`public RedeemCodeState(Dictionary<PublicKey, RedeemCodeState.Reward> rewardMap)`

### Parameters

`rewardMap` [Dictionary](#)<PublicKey, [RedeemCodeState.Reward](#)>

### Fields

#### Address

`public static readonly Address Address`

#### Field Value

Address

### Properties

#### Map

`public IReadOnlyDictionary<PublicKey, RedeemCodeState.Reward> Map { get; }`

#### Property Value

[IReadOnlyDictionary](#)<PublicKey, [RedeemCodeState.Reward](#)>

### Methods

#### Redeem(string, Address)

```
public int Redeem(string code, Address userAddress)
```

Parameters

code [string](#)

userAddress [Address](#)

Returns

[int](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

[IValue](#)

## Update(RedeemCodeListSheet)

```
public void Update(RedeemCodeListSheet sheet)
```

Parameters

sheet [RedeemCodeListSheet](#)

# Class RedeemCodeState.Reward

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class RedeemCodeState.Reward
```

## Inheritance

[object](#) ← RedeemCodeState.Reward

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### Reward(Dictionary)

```
public Reward(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### Reward(int)

```
public Reward(int rewardId)
```

#### Parameters

rewardId [int](#)

## Fields

## RewardId

```
public readonly int RewardId
```

### Field Value

[int](#)

## UserAddress

```
public Address? UserAddress
```

### Field Value

Address?

## Methods

### Serialize()

```
public IValue Serialize()
```

### Returns

IValue

# Class RuneSlotState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class RuneSlotState : IState
```

## Inheritance

[object](#) ← RuneSlotState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RuneSlotState(List)

```
public RuneSlotState(List serialized)
```

#### Parameters

serialized [List](#)

### RuneSlotState(BattleType)

```
public RuneSlotState(BattleType battleType)
```

#### Parameters

battleType [BattleType](#)

# Properties

## BattleType

```
public BattleType BattleType { get; }
```

### Property Value

[BattleType](#)

# Methods

## DeriveAddress(Address, BattleType)

```
public static Address DeriveAddress(Address avatarAddress, BattleType battleType)
```

### Parameters

avatarAddress Address

battleType [BattleType](#)

### Returns

Address

## GetEquippedRuneSlotInfos()

```
public List<RuneSlotInfo> GetEquippedRuneSlotInfos()
```

### Returns

[List](#)<[RuneSlotInfo](#)>

## GetRuneSlot()

```
public List<RuneSlot> GetRuneSlot()
```

Returns

[List](#)<[RuneSlot](#)>

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Unlock(int)

```
public void Unlock(int index)
```

Parameters

index [int](#)

## UpdateSlot(List<RuneSlotInfo>, RuneListSheet)

```
public void UpdateSlot(List<RuneSlotInfo> runeInfos, RuneListSheet runeListSheet)
```

Parameters

runeInfos [List](#)<[RuneSlotInfo](#)>

runeListSheet [RuneListSheet](#)

## UpdateSlotV2(List<RuneSlotInfo>, RuneListSheet)

```
[Obsolete("UpdateSlotV2")]
public void UpdateSlotV2(List<RuneSlotInfo> runeInfos, RuneListSheet runeListSheet)
```

## Parameters

runeInfos [List](#)<[RuneSlotInfo](#)>

runeListSheet [RuneListSheet](#)

# Class RuneState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class RuneState : IState
```

## Inheritance

[object](#) ← RuneState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RuneState(List)

```
public RuneState(List serialized)
```

#### Parameters

serialized List

### RuneState(int)

```
public RuneState(int runeId)
```

#### Parameters

runeId [int](#)

## RuneState(int, int)

```
public RuneState(int runeId, int level)
```

### Parameters

runeId [int](#)

level [int](#)

## Properties

### Level

```
public int Level { get; }
```

### Property Value

[int](#)

### RunelId

```
public int RuneId { get; }
```

### Property Value

[int](#)

## Methods

### DeriveAddress(Address, int)

```
public static Address DeriveAddress(Address avatarAddress, int runeId)
```

Parameters

avatarAddress Address

runeId [int](#)

Returns

Address

## LevelUp(int)

```
public void LevelUp(int level = 1)
```

Parameters

level [int](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class ShardedShopState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class ShardedShopState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← ShardedShopState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ShardedShopState(Dictionary)

```
public ShardedShopState(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### ShardedShopState(Address)

```
public ShardedShopState(Address address)
```

#### Parameters

address Address

# Fields

## AddressKeys

```
public static readonly IReadOnlyList<string> AddressKeys
```

### Field Value

[IReadOnlyList](#)<[string](#)>

# Products

```
public readonly Dictionary<Guid, ShopItem> Products
```

### Field Value

[Dictionary](#)<[Guid](#), [ShopItem](#)>

# Methods

## DeriveAddress(ItemSubType, Guid)

```
public static Address DeriveAddress(ItemSubType itemSubType, Guid productId)
```

### Parameters

itemSubType [ItemSubType](#)

productId [Guid](#)

### Returns

Address

## DeriveAddress(ItemSubType, string)

```
public static Address DeriveAddress(ItemSubType itemSubType, string nonce)
```

### Parameters

itemSubType [ItemSubType](#)

nonce [string](#)

### Returns

Address

## Register(ShopItem)

```
public void Register(ShopItem shopItem)
```

### Parameters

shopItem [ShopItem](#)

## Serialize()

```
public override IValue Serialize()
```

### Returns

IValue

# Class ShardedShopStateV2

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ShardedShopStateV2 : State, IState
```

## Inheritance

[object](#) ← [State](#) ← ShardedShopStateV2

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ShardedShopStateV2(Dictionary)

```
public ShardedShopStateV2(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### ShardedShopStateV2(Address)

```
public ShardedShopStateV2(Address address)
```

#### Parameters

address Address

## Properties

### OrderDigestList

```
public IReadOnlyList<OrderDigest> OrderDigestList { get; }
```

Property Value

[IReadOnlyList](#)<[OrderDigest](#)>

## Methods

### Add(OrderDigest, long)

```
public void Add(OrderDigest orderDigest, long blockIndex)
```

Parameters

orderDigest [OrderDigest](#)

blockIndex [long](#)

### DeriveAddress(ItemSubType, Guid)

```
public static Address DeriveAddress(ItemSubType itemSubType, Guid orderId)
```

Parameters

itemSubType [ItemSubType](#)

orderId [Guid](#)

Returns

Address

## DeriveAddress(ItemSubType, string)

```
public static Address DeriveAddress(ItemSubType itemSubType, string nonce)
```

Parameters

itemSubType [ItemSubType](#)

nonce [string](#)

Returns

Address

## Remove(Order, long)

```
public void Remove(Order order, long blockIndex)
```

Parameters

order [Order](#)

blockIndex [long](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

# Class ShopState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

This is a model class of shop state.

```
[Serializable]
public class ShopState : State, IState
```

## Inheritance

[object](#) ← [State](#) ← ShopState

## Implements

[IState](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ShopState()

```
public ShopState()
```

### ShopState(Dictionary)

```
public ShopState(Dictionary serialized)
```

## Parameters

serialized Dictionary

# Fields

## Address

```
public static readonly Address Address
```

## Field Value

Address

# Properties

## Products

```
public IReadOnlyDictionary<Guid, ShopItem> Products { get; }
```

## Property Value

[IReadOnlyDictionary](#)<[Guid](#), [ShopItem](#)>

# Methods

## Register(ShopItem)

```
public void Register(ShopItem shopItem)
```

## Parameters

shopItem [ShopItem](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## TryGet(Address, Guid, out ShopItem)

```
public bool TryGet(Address sellerAgentAddress, Guid productId, out  
ShopItem shopItem)
```

Parameters

sellerAgentAddress Address

productId [Guid](#)

shopItem [ShopItem](#)

Returns

[bool](#)

## TryUnregister(Guid, out ShopItem)

```
public bool TryUnregister(Guid productId, out ShopItem unregisteredItem)
```

Parameters

productId [Guid](#)

unregisteredItem [ShopItem](#)

Returns

[bool](#)

## Unregister(ShopItem)

```
public void Unregister(ShopItem shopItem)
```

Parameters

shopItem [ShopItem](#)

## Unregister(Guid)

```
public void Unregister(Guid productId)
```

Parameters

productId [Guid](#)

# Class ShopStateAlreadyContainsException

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ShopStateAlreadyContainsException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← ShopStateAlreadyContainsException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### ShopStateAlreadyContainsException(SerializationInfo, StreamingContext)

```
protected ShopStateAlreadyContainsException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## ShopStateAlreadyContainsException(string)

```
public ShopStateAlreadyContainsException(string message)
```

### Parameters

message [string](#)

# Class StakeRewardCalculator

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public static class StakeRewardCalculator
```

## Inheritance

[object](#) ← StakeRewardCalculator

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

**CalculateFixedRewards(int, IRandom,  
StakeRegularFixedRewardSheet, ItemSheet, int)**

```
public static Dictionary<ItemBase, int> CalculateFixedRewards(int stakeLevel,  
IRandom random, StakeRegularFixedRewardSheet stakeRegularFixedRewardSheet, ItemSheet  
itemSheet, int rewardSteps)
```

### Parameters

stakeLevel [int](#)

random [IRandom](#)

stakeRegularFixedRewardSheet [StakeRegularFixedRewardSheet](#)

itemSheet [ItemSheet](#)

rewardSteps [int](#)

### Returns

[Dictionary](#)<ItemBase, int>

## CalculateRewards(Currency, FungibleAssetValue, int, int, StakeRegularRewardSheet, ItemSheet, IRandom)

```
public static (Dictionary<ItemBase, int> itemResult, List<FungibleAssetValue>
favResult) CalculateRewards(Currency ncg, FungibleAssetValue stakedNcg, int
stakingLevel, int rewardSteps, StakeRegularRewardSheet stakeRegularRewardSheet,
ItemSheet itemSheet, IRandom random)
```

### Parameters

ncg Currency

stakedNcg FungibleAssetValue

stakingLevel int

rewardSteps int

stakeRegularRewardSheet [StakeRegularRewardSheet](#)

itemSheet [ItemSheet](#)

random IRandom

### Returns

([Dictionary](#)<ItemBase, int> itemResult, [List](#)<FungibleAssetValue> favResult)

# Class State

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class State : IState
```

## Inheritance

[object](#) ← State

## Implements

[IState](#)

## Derived

[ActivatedAccountsState](#), [AdminState](#), [AgentState](#), [AuthorizedMinersState](#), [AvatarState](#),  
[CombinationSlotState](#), [CreditsState](#), [GameConfigState](#), [GoldBalanceState](#),  
[GoldCurrencyState](#), [LegacyStakeState](#), [MonsterCollectionState](#), [MonsterCollectionState0](#),  
[PendingActivationState](#), [RankingMapState](#), [RankingState](#), [RankingState0](#), [RankingState1](#),  
[RedeemCodeState](#), [ShardedShopState](#), [ShardedShopStateV2](#), [ShopState](#), [WeeklyArenaState](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

# Constructors

## State(IValue)

```
protected State(IValue iValue)
```

## Parameters

iValue IValue

# State(Address)

`protected State(Address address)`

## Parameters

`address` Address

## Fields

`address`

`public Address address`

## Field Value

Address

## Methods

`Serialize()`

`public abstract IValue Serialize()`

## Returns

IValue

`SerializeBase()`

`protected IValue SerializeBase()`

## Returns

IValue

## SerializeListBase()

```
protected IValue SerializeListBase()
```

Returns

IValue

## SerializeV2Base()

```
protected IValue SerializeV2Base()
```

Returns

IValue

# Class StateExtensions

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public static class StateExtensions
```

## Inheritance

[object](#) ← StateExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Deserialize<T>(Func<IValue, T>, IValue)

```
public static T? Deserialize<T>(Func<IValue, T> deserializer, IValue serialized)  
where T : struct
```

#### Parameters

deserializer [Func](#)<IValue, T>

serialized IValue

#### Returns

T?

#### Type Parameters

T

## GetAddress(Dictionary, string, Address)

```
public static Address GetAddress(this Dictionary serialized, string key, Address  
defaultValue = default)
```

### Parameters

serialized Dictionary

key [string](#)

defaultValue Address

### Returns

Address

## GetBoolean(Dictionary, string, bool)

```
public static bool GetBoolean(this Dictionary serialized, string key, bool  
defaultValue = false)
```

### Parameters

serialized Dictionary

key [string](#)

defaultValue [bool](#)

### Returns

[bool](#)

## GetDateTimeOffset(Dictionary, string, DateTimeOffset)

```
public static DateTimeOffset GetDateTimeOffset(this Dictionary serialized, string  
key, DateTimeOffset defaultValue = default)
```

## Parameters

**serialized** Dictionary

key [string](#)

defaultValue [DateTimeOffset](#)

## Returns

[DateTimeOffset](#)

## GetDecimal(Dictionary, string, decimal)

```
public static decimal GetDecimal(this Dictionary serialized, string key, decimal defaultValue = 0)
```

## Parameters

**serialized** Dictionary

key [string](#)

defaultValue [decimal](#)

## Returns

[decimal](#)

## GetDecimalStat(Dictionary, string, DecimalStat)

```
public static DecimalStat GetDecimalStat(this Dictionary serialized, string key, DecimalStat defaultValue = null)
```

## Parameters

**serialized** Dictionary

key [string](#)

`defaultValue` [DecimalStat](#)

Returns

[DecimalStat](#)

## GetGuid(Dictionary, string, Guid)

```
public static Guid GetGuid(this Dictionary serialized, string key, Guid defaultValue = default)
```

Parameters

`serialized` Dictionary

`key` [string](#)

`defaultValue` [Guid](#)

Returns

[Guid](#)

## GetInteger(Dictionary, string, int)

```
public static int GetInteger(this Dictionary serialized, string key, int defaultValue = 0)
```

Parameters

`serialized` Dictionary

`key` [string](#)

`defaultValue` [int](#)

Returns

[int](#)

## GetLong(Dictionary, string, long)

```
public static long GetLong(this Dictionary serialized, string key, long defaultValue = 0)
```

### Parameters

**serialized** Dictionary

**key** [string](#)

**defaultValue** [long](#)

### Returns

[long](#)

## GetStat(Dictionary)

```
public static (StatType statType, decimal baseValue, decimal additionalValue)
GetStat(this Dictionary serialized)
```

### Parameters

**serialized** Dictionary

### Returns

([StatType](#) [statType](#), [decimal](#) [baseValue](#), [decimal](#) [additionalValue](#))

## GetString(Dictionary, string, string)

```
public static string GetString(this Dictionary serialized, string key, string defaultValue = "")
```

## Parameters

**serialized** Dictionary

**key** [string](#)

**defaultValue** [string](#)

## Returns

[string](#)

## GetValue<T>(Dictionary, string, T, IValueTryParseDelegate<T>)

```
public static T GetValue<T>(this Dictionary serialized, string key, T defaultValue,  
StateExtensions.IValueTryParseDelegate<T> tryParser)
```

## Parameters

**serialized** Dictionary

**key** [string](#)

**defaultValue** T

**tryParser** [StateExtensions.IValueTryParseDelegate](#)<T>

## Returns

T

## Type Parameters

T

## Serialize(HashDigest<SHA256>)

```
public static IValue Serialize(this HashDigest<SHA256> hashDigest)
```

Parameters

hashDigest HashDigest<[SHA256](#)>

Returns

IValue

## Serialize(Address)

```
public static IValue Serialize(this Address address)
```

Parameters

address Address

Returns

IValue

## Serialize(PublicKey)

```
public static IValue Serialize(this PublicKey key)
```

Parameters

key PublicKey

Returns

IValue

## Serialize(FungibleAssetValue)

```
public static IValue Serialize(this FungibleAssetValue value)
```

Parameters

**value** FungibleAssetValue

Returns

IValue

## Serialize(bool)

```
public static IValue Serialize(this bool boolean)
```

Parameters

**boolean** [bool](#)

Returns

IValue

## Serialize(Dictionary<Material, int>)

```
public static IValue Serialize(this Dictionary<Material, int> value)
```

Parameters

**value** [Dictionary](#)<[Material](#), [int](#)>

Returns

IValue

## Serialize(IEnumerable<(Address, Address, IValue)>)

```
public static IValue Serialize(this IEnumerable<(Address, Address, IValue)> value)
```

Parameters

value [IEnumerable](#)<(Address, Address, IValue)>

Returns

IValue

## Serialize(DateTimeOffset)

```
public static IValue Serialize(this DateTimeOffset dateTIme)
```

Parameters

dateTIme [DateTimeOffset](#)

Returns

IValue

## Serialize(decimal)

```
public static IValue Serialize(this decimal number)
```

Parameters

number [decimal](#)

Returns

IValue

## Serialize(Enum)

```
public static IValue Serialize(this Enum type)
```

Parameters

type [Enum](#)

Returns

IValue

## Serialize(Guid)

```
public static IValue Serialize(this Guid number)
```

Parameters

number [Guid](#)

Returns

IValue

## Serialize(int)

```
public static IValue Serialize(this int number)
```

Parameters

number [int](#)

Returns

IValue

## Serialize(long)

```
public static IValue Serialize(this long number)
```

Parameters

number `long`

Returns

IValue

## Serialize(Address?)

```
public static IValue Serialize(this Address? address)
```

Parameters

address Address?

Returns

IValue

## Serialize(FungibleAssetValue?)

```
public static IValue Serialize(this FungibleAssetValue? value)
```

Parameters

value FungibleAssetValue?

Returns

IValue

## Serialize(bool?)

```
public static IValue Serialize(this bool? boolean)
```

Parameters

boolean bool?

Returns

IValue

## Serialize(DateTimeOffset?)

```
public static IValue Serialize(this DateTimeOffset? dateTime)
```

Parameters

dateTime DateTimeOffset?

Returns

IValue

## Serialize(decimal?)

```
public static IValue Serialize(this decimal? number)
```

Parameters

number decimal?

Returns

IValue

## Serialize(Guid?)

```
public static IValue Serialize(this Guid? number)
```

Parameters

number [Guid](#)?

Returns

IValue

## Serialize(int?)

```
public static IValue Serialize(this int? number)
```

Parameters

number [int](#)?

Returns

IValue

## Serialize(long?)

```
public static IValue Serialize(this long? number)
```

Parameters

number [long](#)?

Returns

IValue

## Serialize(BigInteger?)

```
public static IValue Serialize(this BigInteger? number)
```

Parameters

number [BigInteger](#)?

Returns

IValue

## Serialize(BigInteger)

```
public static IValue Serialize(this BigInteger number)
```

Parameters

number [BigInteger](#)

Returns

IValue

## Serialize(string)

```
public static IValue Serialize(this string text)
```

Parameters

text [string](#)

Returns

IValue

## SerializeForLegacyEquipmentStat(DecimalStat)

```
public static IValue SerializeForLegacyEquipmentStat(this DecimalStat decimalStat)
```

Parameters

decimalStat [DecimalStat](#)

Returns

IValue

## Serialize<T>(IEnumerable<T>)

```
public static IValue Serialize<T>(this IEnumerable<T> values) where T : IValue
```

Parameters

values [IEnumerable](#)<T>

Returns

IValue

Type Parameters

T

## Serialize<T>(Func<T, IValue>, T?)

```
public static IValue Serialize<T>(Func<T, IValue> serializer, T? value) where T : struct
```

Parameters

serializer [Func](#)<T, IValue>

value T?

Returns

IValue

Type Parameters

T

## ToAddress(IValue)

```
public static Address ToAddress(this IValue serialized)
```

Parameters

serialized IValue

Returns

Address

## ToArray<T>(IValue, Func<IValue, T>)

```
public static T[] ToArray<T>(this IValue serialized, Func<IValue, T> deserializer)
```

Parameters

serialized IValue

deserializer [Func](#)<IValue, T>

Returns

T[]

Type Parameters

T

## ToBigInteger(IValue)

```
public static BigInteger ToBigInteger(this IValue serialized)
```

Parameters

serialized IValue

Returns

[BigInteger](#)

## ToBoolean(IValue)

```
public static bool ToBoolean(this IValue serialized)
```

Parameters

serialized IValue

Returns

[bool](#)

## ToDateTimeOffset(IValue)

```
public static DateTimeOffset ToDateTimeOffset(this IValue serialized)
```

Parameters

serialized IValue

Returns

[DateTimeOffset](#)

## ToDecimal(IValue)

```
public static decimal ToDecimal(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[decimal](#)

## ToDecimalStat(Dictionary)

```
public static DecimalStat ToDecimalStat(this Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[DecimalStat](#)

## ToDecimalStat(IValue)

```
public static DecimalStat ToDecimalStat(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[DecimalStat](#)

## ToDictionary\_Material\_int(IValue)

```
public static Dictionary<Material, int> ToDictionary_Material_int(this  
IValue serialized)
```

Parameters

serialized IValue

Returns

[Dictionary](#)<[Material](#), [int](#)>

## ToDotnetString(IValue)

```
public static string ToDotnetString(this IValue serialized)
```

Parameters

serialized IValue

Returns

[string](#)

## ToEnum<T>(IValue)

```
public static T ToEnum<T>(this IValue serialized) where T : struct
```

Parameters

serialized IValue

Returns

T

Type Parameters

T

## ToEnumerable<T>(IValue, Func<IValue, T>)

```
public static IEnumerable<T> ToEnumerable<T>(this IValue serialized, Func<IValue,
T> deserializer)
```

### Parameters

serialized IValue

deserializer [Func](#)<IValue, T>

### Returns

[IEnumerable](#)<T>

### Type Parameters

T

## ToFungibleAssetValue(IValue)

```
public static FungibleAssetValue ToFungibleAssetValue(this IValue serialized)
```

### Parameters

serialized IValue

### Returns

FungibleAssetValue

## ToGuid(IValue)

```
public static Guid ToGuid(this IValue serialized)
```

Parameters

**serialized** **IValue**

Returns

[Guid](#)

## ToHashSet<T>(IValue, Func<IValue, T>)

```
public static HashSet<T> ToHashSet<T>(this IValue serialized, Func<IValue, T> deserializer)
```

Parameters

**serialized** **IValue**

**deserializer** [Func](#)<IValue, T>

Returns

[HashSet](#)<T>

Type Parameters

**T**

## ToImmutableHashSet<T>(IValue, Func<IValue, T>)

```
public static ImmutableHashSet<T> ToImmutableHashSet<T>(this IValue serialized, Func<IValue, T> deserializer)
```

Parameters

**serialized** **IValue**

**deserializer** [Func](#)<IValue, T>

Returns

[ImmutableHashSet](#)<T>

Type Parameters

T

## ToInteger(IValue)

```
public static int ToInteger(this IValue serialized)
```

Parameters

serialized IValue

Returns

[int](#)

## ToItemId(IValue)

```
public static HashDigest<SHA256> ToItemId(this IValue serialized)
```

Parameters

serialized IValue

Returns

HashDigest<[SHA256](#)>

## ToList<T>(IValue, Func<IValue, T>)

```
public static List<T> ToList<T>(this IValue serialized, Func<IValue, T> deserializer)
```

Parameters

**serialized** **IValue**

**deserializer** [Func](#)<IValue, T>

Returns

[List](#)<T>

Type Parameters

T

## ToLock(IValue)

```
public static ILock ToLock(this IValue serialized)
```

Parameters

**serialized** **IValue**

Returns

[ILock](#)

## ToLong(IValue)

```
public static long ToLong(this IValue serialized)
```

Parameters

**serialized** **IValue**

Returns

[long](#)

## ToNullableAddress(IValue)

```
public static Address? ToNullableAddress(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

Address?

## ToNullableBigInteger(IValue)

```
public static BigInteger? ToNullableBigInteger(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[BigInteger](#)?

## ToNullableBoolean(IValue)

```
public static bool? ToNullableBoolean(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[bool](#)?

## ToNullableDateTimeOffset(IValue)

```
public static DateTimeOffset? ToNullableDateTimeOffset(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[DateTimeOffset](#)?

## ToNullableDecimal(IValue)

```
public static decimal? ToNullableDecimal(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[decimal](#)?

## ToNullableFungibleAssetValue(IValue)

```
public static FungibleAssetValue? ToNullableFungibleAssetValue(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

FungibleAssetValue?

## ToNullableGuid(IValue)

```
public static Guid? ToNullableGuid(this IValue serialized)
```

Parameters

serialized IValue

Returns

[Guid](#)?

## ToNullableInteger(IValue)

```
public static int? ToNullableInteger(this IValue serialized)
```

Parameters

serialized IValue

Returns

[int](#)?

## ToNullableLong(IValue)

```
public static long? ToNullableLong(this IValue serialized)
```

Parameters

serialized IValue

Returns

[long](#)?

## ToPublicKey(IValue)

```
public static PublicKey ToPublicKey(this IValue serialized)
```

Parameters

serialized IValue

Returns

PublicKey

## ToPurchaseInfo(IValue)

```
public static PurchaseInfo0 ToPurchaseInfo(this IValue serialized)
```

Parameters

serialized IValue

Returns

[PurchaseInfo0](#)

## ToPurchaseInfoLegacy(IValue)

```
public static BuyMultiple.PurchaseInfo ToPurchaseInfoLegacy(this IValue serialized)
```

Parameters

serialized IValue

Returns

[BuyMultiple.PurchaseInfo](#)

## ToPurchaseResult(IValue)

```
public static Buy7.PurchaseResult ToPurchaseResult(this IValue serialized)
```

Parameters

serialized IValue

Returns

[Buy7.PurchaseResult](#)

## ToPurchaseResultLegacy(IValue)

```
public static BuyMultiple.PurchaseResult ToPurchaseResultLegacy(this IValue serialized)
```

Parameters

serialized IValue

Returns

[BuyMultiple.PurchaseResult](#)

## ToSellerResult(IValue)

```
public static Buy7.SellerResult ToSellerResult(this IValue serialized)
```

Parameters

serialized IValue

Returns

[Buy7.SellerResult](#)

## ToStateList(IValue)

```
public static List<(Address, Address, IValue)> ToStateList(this IValue serialized)
```

Parameters

**serialized** IValue

Returns

[List](#)<(Address, Address, IValue)>

# Delegate StateExtensions.IValueTryParseDelegate< T>

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public delegate bool StateExtensions.IValueTryParseDelegate<T>(IValue input, out  
T output)
```

## Parameters

**input** IValue

**output** T

## Returns

[bool](#) ↗

## Type Parameters

T

# Class WeeklyArenaState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WeeklyArenaState : State, IState, IDictionary<Address, ArenaInfo>,
ICollection<KeyValuePair<Address, ArenaInfo>>, IEnumerable<KeyValuePair<Address, ArenaInfo>>, IEnumerable, ISerializable
```

## Inheritance

[object](#) ← [State](#) ← WeeklyArenaState

## Implements

[IState](#), [IDictionary](#)<Address, [ArenaInfo](#)>,  
[ICollection](#)<[KeyValuePair](#)<Address, [ArenaInfo](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<Address, [ArenaInfo](#)>>, [IEnumerable](#), [ISerializable](#)

## Inherited Members

[State.address](#) , [State.SerializeBase\(\)](#) , [State.SerializeV2Base\(\)](#) , [State.SerializeListBase\(\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### WeeklyArenaState(Dictionary)

```
public WeeklyArenaState(Dictionary serialized)
```

#### Parameters

**serialized** Dictionary

### WeeklyArenaState(IValue)

```
public WeeklyArenaState(IValue iValue)
```

Parameters

iValue IValue

## WeeklyArenaState(Address)

```
public WeeklyArenaState(Address address)
```

Parameters

address Address

## WeeklyArenaState(int)

```
public WeeklyArenaState(int index)
```

Parameters

index [int](#)

## WeeklyArenaState(SerializationInfo, StreamingContext)

```
protected WeeklyArenaState(SerializationInfo info, StreamingContext context)
```

Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Fields

# Ended

```
public bool Ended
```

## Field Value

[bool](#)

# ResetIndex

```
public long ResetIndex
```

## Field Value

[long](#)

# Properties

## Count

Gets the number of elements contained in the [ICollection<T>](#).

```
public int Count { get; }
```

## Property Value

[int](#)

The number of elements contained in the [ICollection<T>](#).

## IsReadOnly

Gets a value indicating whether the [ICollection<T>](#) is read-only.

```
public bool IsReadOnly { get; }
```

## Property Value

### [bool](#)

[true](#) if the [ICollection<T>](#) is read-only; otherwise, [false](#).

## this[Address]

Gets or sets the element with the specified key.

```
public ArenaInfo this[Address key] { get; set; }
```

## Parameters

### **key** Address

The key of the element to get or set.

## Property Value

### [ArenaInfo](#)

The element with the specified key.

## Exceptions

### [ArgumentNullException](#)

**key** is [null](#).

### [KeyNotFoundException](#)

The property is retrieved and **key** is not found.

### [NotSupportedException](#)

The property is set and the [IDictionary<TKey, TValue>](#) is read-only.

## Keys

Gets an [ICollection<T>](#) containing the keys of the [IDictionary<TKey, TValue>](#).

```
public ICollection<Address> Keys { get; }
```

Property Value

#### [ICollection](#)<Address>

An [ICollection](#)<T> containing the keys of the object that implements [IDictionary](#)< TKey, TValue >.

Map

```
public IReadOnlyDictionary<Address, LazyState<ArenaInfo, Dictionary>> Map { get; }
```

Property Value

#### [IReadOnlyDictionary](#)<Address, [LazyState](#)<ArenaInfo, Dictionary>>

OrderedArenaInfos

```
public List<ArenaInfo> OrderedArenaInfos { get; }
```

Property Value

#### [List](#)<ArenaInfo>

Values

Gets an [ICollection](#)<T> containing the values in the [IDictionary](#)< TKey, TValue >.

```
public ICollection<ArenaInfo> Values { get; }
```

Property Value

#### [ICollection](#)<ArenaInfo>

An [ICollection<T>](#) containing the values in the object that implements [IDictionary< TKey, TValue >](#).

## Methods

### Add(Address, ArenaInfo)

Adds an element with the provided key and value to the [IDictionary< TKey, TValue >](#).

```
public void Add(Address key, ArenaInfo value)
```

#### Parameters

**key** Address

The object to use as the key of the element to add.

**value** [ArenaInfo](#)

The object to use as the value of the element to add.

#### Exceptions

[ArgumentNullException](#)

key is [null](#).

[ArgumentException](#)

An element with the same key already exists in the [IDictionary< TKey, TValue >](#).

[NotSupportedException](#)

The [IDictionary< TKey, TValue >](#) is read-only.

### Add(KeyValuePair<Address, ArenaInfo>)

Adds an item to the [ICollection<T>](#).

```
public void Add(KeyValuePair<Address, ArenaInfo> item)
```

## Parameters

**item** [KeyValuePair](#)<Address, ArenaInfo>

The object to add to the [ICollection<T>](#).

## Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Clear()

Removes all items from the [ICollection<T>](#).

```
public void Clear()
```

## Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Contains(KeyValuePair<Address, ArenaInfo>)

Determines whether the [ICollection<T>](#) contains a specific value.

```
public bool Contains(KeyValuePair<Address, ArenaInfo> item)
```

## Parameters

**item** [KeyValuePair](#)<Address, ArenaInfo>

The object to locate in the [ICollection<T>](#).

## Returns

[bool](#)

[true](#) if `item` is found in the [ICollection<T>](#); otherwise, [false](#).

## ContainsKey(Address)

Determines whether the [IDictionary< TKey, TValue >](#) contains an element with the specified key.

```
public bool ContainsKey(Address key)
```

### Parameters

`key` Address

The key to locate in the [IDictionary< TKey, TValue >](#).

### Returns

[bool](#)

[true](#) if the [IDictionary< TKey, TValue >](#) contains an element with the key; otherwise, [false](#).

### Exceptions

[ArgumentNullException](#)

`key` is [null](#).

## CopyTo(KeyValuePair<Address, ArenaInfo>[], int)

Copies the elements of the [ICollection<T>](#) to an [Array](#), starting at a particular [Array](#) index.

```
public void CopyTo(KeyValuePair<Address, ArenaInfo>[] array, int arrayIndex)
```

### Parameters

`array` [KeyValuePair](#)<Address, [ArenaInfo](#)>[]

The one-dimensional [Array](#) that is the destination of the elements copied from [ICollection<T>](#). The [Array](#) must have zero-based indexing.

#### arrayIndex [int](#)

The zero-based index in [array](#) at which copying begins.

### Exceptions

#### [ArgumentNullException](#)

[array](#) is [null](#).

#### [ArgumentOutOfRangeException](#)

[arrayIndex](#) is less than 0.

#### [ArgumentException](#)

The number of elements in the source [ICollection<T>](#) is greater than the available space from [arrayIndex](#) to the end of the destination [array](#).

## DeriveAddress(int)

```
public static Address DeriveAddress(int index)
```

### Parameters

#### index [int](#)

### Returns

Address

## End()

```
public void End()
```

## GetArenaInfo(Address)

```
public ArenaInfo GetArenaInfo(Address avatarAddress)
```

Parameters

avatarAddress Address

Returns

[ArenaInfo](#)

## GetArenaInfos(Address, int, int)

Get arena rank information.

```
public List<(int rank, ArenaInfo arenaInfo)> GetArenaInfos(Address avatarAddress,  
int upperRange = 10, int lowerRange = 10)
```

Parameters

avatarAddress Address

The base value of the range that want to get.

upperRange [int](#)

The upper range than base value in the ranges that want to get.

lowerRange [int](#)

The lower range than base value in the ranges that want to get.

Returns

[List](#)<(int rank, ArenaInfo arenaInfo)>

A list of tuples that contains [int](#) and [ArenaInfo](#).

## GetArenaInfos(int, int?)

Get arena rank information.

```
public List<(int rank, ArenaInfo arenaInfo)> GetArenaInfos(int firstRank = 1, int?  
count = null)
```

Parameters

`firstRank` [int](#)

The first rank in the range that want to get.

`count` [int](#)?

The count of the range that want to get.

Returns

[List](#)<(int rank, ArenaInfo arenaInfo)>

A list of tuples that contains `int` and `ArenaInfo`.

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumarator<KeyValuePair<Address, ArenaInfo>> GetEnumerator()
```

Returns

[IEnumarator](#)<KeyValuePair<Address, ArenaInfo>>

An enumerator that can be used to iterate through the collection.

## GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

**info** [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

**context** [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Remove(Address)

Removes the element with the specified key from the [IDictionary<TKey, TValue>](#).

```
public bool Remove(Address key)
```

## Parameters

**key** Address

The key of the element to remove.

## Returns

[bool](#)

[true](#) if the element is successfully removed; otherwise, [false](#). This method also returns [false](#) if **key** was not found in the original [IDictionary<TKey, TValue>](#).

## Exceptions

[ArgumentNullException](#)

`key` is [null](#).

## [NotSupportedException](#)

The [IDictionary<TKey, TValue>](#) is read-only.

# Remove(KeyValuePair<Address, ArenaInfo>)

Removes the first occurrence of a specific object from the [ICollection<T>](#).

```
public bool Remove(KeyValuePair<Address, ArenaInfo> item)
```

## Parameters

`item` [KeyValuePair](#)<Address, ArenaInfo>

The object to remove from the [ICollection<T>](#).

## Returns

[bool](#)

[true](#) if `item` was successfully removed from the [ICollection<T>](#); otherwise, [false](#). This method also returns [false](#) if `item` is not found in the original [ICollection<T>](#).

## Exceptions

### [NotSupportedException](#)

The [ICollection<T>](#) is read-only.

# ResetCount(long)

```
public void ResetCount(long ctxBlockIndex)
```

## Parameters

`ctxBlockIndex` [long](#)

## Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Set(AvatarState, CharacterSheet)

```
public void Set(AvatarState avatarState, CharacterSheet characterSheet)
```

Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

## SetReceive(Address)

```
public void SetReceive(Address avatarAddress)
```

Parameters

avatarAddress Address

## SetV2(AvatarState, CharacterSheet, CostumeStatSheet)

```
public void SetV2(AvatarState avatarState, CharacterSheet characterSheet,  
CostumeStatSheet costumeStatSheet)
```

Parameters

avatarState [AvatarState](#)

characterSheet [CharacterSheet](#)

costumeStatSheet [CostumeStatSheet](#)

## TryGetValue(Address, out ArenaInfo)

Gets the value associated with the specified key.

```
public bool TryGetValue(Address key, out ArenaInfo value)
```

### Parameters

**key** Address

The key whose value to get.

**value** [ArenaInfo](#)

When this method returns, the value associated with the specified key, if the key is found; otherwise, the default value for the type of the **value** parameter. This parameter is passed uninitialized.

### Returns

[bool](#)

[true](#) if the object that implements [IDictionary<TKey, TValue>](#) contains an element with the specified key; otherwise, [false](#).

### Exceptions

[ArgumentNullException](#)

**key** is [null](#).

## Update(ArenaInfo)

```
public void Update(ArenaInfo info)
```

Parameters

info [ArenaInfo](#)

## Update(WeeklyArenaState, long)

```
public void Update(WeeklyArenaState prevState, long index)
```

Parameters

prevState [WeeklyArenaState](#)

index [long](#) ↴

# Class WorldBossKillRewardRecord

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
public class WorldBossKillRewardRecord : IDictionary<int, bool>,
ICollection<KeyValuePair<int, bool>>, IEnumerable<KeyValuePair<int, bool>>,
IEnumerable, IState
```

## Inheritance

[object](#) ← WorldBossKillRewardRecord

## Implements

[IDictionary](#)<[int](#), [bool](#)>, [ICollection](#)<[KeyValuePair](#)<[int](#), [bool](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [bool](#)>>, [IEnumerable](#), [IState](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### WorldBossKillRewardRecord()

```
public WorldBossKillRewardRecord()
```

### WorldBossKillRewardRecord(List)

```
public WorldBossKillRewardRecord(List serialized)
```

## Parameters

serialized List

# Properties

## Count

Gets the number of elements contained in the [ICollection<T>](#).

```
public int Count { get; }
```

### Property Value

[int](#)

The number of elements contained in the [ICollection<T>](#).

## IsReadOnly

Gets a value indicating whether the [ICollection<T>](#) is read-only.

```
public bool IsReadOnly { get; }
```

### Property Value

[bool](#)

[true](#) if the [ICollection<T>](#) is read-only; otherwise, [false](#).

## this[int]

Gets or sets the element with the specified key.

```
public bool this[int key] { get; set; }
```

### Parameters

**key** [int](#)

The key of the element to get or set.

## Property Value

### [bool](#)

The element with the specified key.

## Exceptions

### [ArgumentNullException](#)

`key` is [null](#).

### [KeyNotFoundException](#)

The property is retrieved and `key` is not found.

### [NotSupportedException](#)

The property is set and the [IDictionary<TKey, TValue>](#) is read-only.

## Keys

Gets an [ICollection<T>](#) containing the keys of the [IDictionary<TKey, TValue>](#).

```
public ICollection<int> Keys { get; }
```

## Property Value

### [ICollection<int>](#)

An [ICollection<T>](#) containing the keys of the object that implements [IDictionary<TKey, TValue>](#).

## Values

Gets an [ICollection<T>](#) containing the values in the [IDictionary<TKey, TValue>](#).

```
public ICollection<bool> Values { get; }
```

## Property Value

## [ICollection<bool>](#)

An [ICollection<T>](#) containing the values in the object that implements [IDictionary< TKey, TValue >](#).

## Methods

### Add(KeyValuePair<int, bool>)

Adds an item to the [ICollection<T>](#).

```
public void Add(KeyValuePair<int, bool> item)
```

#### Parameters

**item** [KeyValuePair<int, bool>](#)

The object to add to the [ICollection<T>](#).

#### Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

### Add(int, bool)

Adds an element with the provided key and value to the [IDictionary< TKey, TValue >](#).

```
public void Add(int key, bool value)
```

#### Parameters

**key** [int](#)

The object to use as the key of the element to add.

**value** [bool](#)

The object to use as the value of the element to add.

## Exceptions

### [ArgumentNullException](#)

key is [null](#).

### [ArgumentException](#)

An element with the same key already exists in the [IDictionary<TKey, TValue>](#).

### [NotSupportedException](#)

The [IDictionary<TKey, TValue>](#) is read-only.

## Clear()

Removes all items from the [ICollection<T>](#).

```
public void Clear()
```

## Exceptions

### [NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Contains(KeyValuePair<int, bool>)

Determines whether the [ICollection<T>](#) contains a specific value.

```
public bool Contains(KeyValuePair<int, bool> item)
```

## Parameters

### item [KeyValuePair<int, bool>](#)

The object to locate in the [ICollection<T>](#).

## Returns

[bool](#)

[true](#) if `item` is found in the [ICollection<T>](#); otherwise, [false](#).

## ContainsKey(int)

Determines whether the [IDictionary< TKey, TValue >](#) contains an element with the specified key.

```
public bool ContainsKey(int key)
```

### Parameters

`key` [int](#)

The key to locate in the [IDictionary< TKey, TValue >](#).

### Returns

[bool](#)

[true](#) if the [IDictionary< TKey, TValue >](#) contains an element with the key; otherwise, [false](#).

### Exceptions

[ArgumentNullException](#)

`key` is [null](#).

## CopyTo(KeyValuePair<int, bool>[], int)

Copies the elements of the [ICollection<T>](#) to an [Array](#), starting at a particular [Array](#) index.

```
public void CopyTo(KeyValuePair<int, bool>[] array, int arrayIndex)
```

### Parameters

`array` [KeyValuePair<int, bool>\[\]](#)

The one-dimensional [Array](#) that is the destination of the elements copied from [ICollection<T>](#). The [Array](#) must have zero-based indexing.

`arrayIndex` [int](#)

The zero-based index in `array` at which copying begins.

## Exceptions

[ArgumentNullException](#)

`array` is [null](#).

[ArgumentOutOfRangeException](#)

`arrayIndex` is less than 0.

[ArgumentException](#)

The number of elements in the source [ICollection<T>](#) is greater than the available space from `arrayIndex` to the end of the destination `array`.

## GetEnumerator()

Returns an enumerator that iterates through the collection.

```
public IEnumrator<KeyValuePair<int, bool>> GetEnumerator()
```

## Returns

[IEnumrator<KeyValuePair<int, bool>>](#)

An enumerator that can be used to iterate through the collection.

## IsClaimable(int)

```
public bool IsClaimable(int level)
```

## Parameters

`level` [int](#)

## Returns

[bool](#)

## Remove(KeyValuePair<int, bool>)

Removes the first occurrence of a specific object from the [ICollection<T>](#).

```
public bool Remove(KeyValuePair<int, bool> item)
```

## Parameters

`item` [KeyValuePair](#)<[int](#), [bool](#)>

The object to remove from the [ICollection<T>](#).

## Returns

[bool](#)

[true](#) if `item` was successfully removed from the [ICollection<T>](#); otherwise, [false](#). This method also returns [false](#) if `item` is not found in the original [ICollection<T>](#).

## Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Remove(int)

Removes the element with the specified key from the [IDictionary<TKey, TValue>](#).

```
public bool Remove(int key)
```

## Parameters

`key` [int](#)

The key of the element to remove.

## Returns

[bool](#)

[true](#) if the element is successfully removed; otherwise, [false](#). This method also returns [false](#) if `key` was not found in the original [IDictionary< TKey, TValue >](#).

## Exceptions

[ArgumentNullException](#)

`key` is [null](#).

[NotSupportedException](#)

The [IDictionary< TKey, TValue >](#) is read-only.

## Serialize()

`public IValue Serialize()`

## Returns

`IValue`

## TryGetValue(int, out bool)

Gets the value associated with the specified key.

`public bool TryGetValue(int key, out bool value)`

## Parameters

**key** [int](#)

The key whose value to get.

**value** [bool](#)

When this method returns, the value associated with the specified key, if the key is found; otherwise, the default value for the type of the **value** parameter. This parameter is passed uninitialized.

Returns

[bool](#)

[true](#) if the object that implements [IDictionary<TKey, TValue>](#) contains an element with the specified key; otherwise, [false](#).

Exceptions

[ArgumentNullException](#)

**key** is [null](#).

# Class WorldBossState

Namespace: [Nekoyume.Model.State](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldBossState : IState
```

## Inheritance

[object](#) ← WorldBossState

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### WorldBossState(List)

```
public WorldBossState(List serialized)
```

#### Parameters

serialized List

### WorldBossState(Row, Row)

```
public WorldBossState(WorldBossListSheet.Row row, WorldBossGlobalHpSheet.Row hpRow)
```

#### Parameters

row [WorldBossListSheet.Row](#)

hpRow [WorldBossGlobalHpSheet.Row](#)

## Fields

### CurrentHp

`public BigInteger CurrentHp`

Field Value

[BigInteger](#)

### EndedBlockIndex

`public long EndedBlockIndex`

Field Value

[long](#)

### Id

`public int Id`

Field Value

[int](#)

### Level

`public int Level`

Field Value

[int](#) ↴

## StartedBlockIndex

`public long StartedBlockIndex`

Field Value

[long](#) ↴

## Methods

### Serialize()

`public IValue Serialize()`

Returns

IValue

# Namespace Nekoyume.Module

## Classes

[ActionPointModule](#)

[AdventureBossModule](#)

[AgentModule](#)

[ArenaModule](#)

[AvatarModule](#)

[CollectionModule](#)

Provides utility methods for working with collection states in the world state.

[CombinationSlotStateModule](#)

CombinationSlotStateModule is the module to use

CombinationSlotState/AllCombinationSlotState with account.

[DailyRewardModule](#)

[LegacyModule](#)

[RelationshipModule](#)

[RuneStateModule](#)

RuneStateModule is the module to use RuneState/AllRuneState with account.

[StateResolver](#)

# Class ActionPointModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class ActionPointModule
```

## Inheritance

[object](#) ← ActionPointModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetActionPoint(IWorldState, Address)

```
public static long GetActionPoint(this IWorldState worldState,  
Address avatarAddress)
```

#### Parameters

**worldState** IWorldState

**avatarAddress** Address

#### Returns

[long](#)

### SetActionPoint(IWorld, Address, long)

```
public static IWorld SetActionPoint(this IWorld world, Address avatarAddress,  
long actionPoint)
```

## Parameters

`world` IWorld

`avatarAddress` Address

`actionPoint` [long](#)

## Returns

IWorld

## TryGetActionPoint(IWorldState, Address, out long)

```
public static bool TryGetActionPoint(this IWorldState worldState, Address  
avatarAddress, out long actionPoint)
```

## Parameters

`worldState` IWorldState

`avatarAddress` Address

`actionPoint` [long](#)

## Returns

[bool](#)

# Class AdventureBossModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class AdventureBossModule
```

## Inheritance

[object](#) ← AdventureBossModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### LatestSeasonAddress

```
public static readonly Address LatestSeasonAddress
```

Field Value

Address

## Methods

### GetBountyBoard(IWorldState, long)

```
public static BountyBoard GetBountyBoard(this IWorldState worldState, long season)
```

Parameters

worldState IWorldState

season [long](#)

Returns

[BountyBoard](#)

## GetExploreBoard(IWorldState, long)

```
public static ExploreBoard GetExploreBoard(this IWorldState worldState, long season)
```

Parameters

worldState IWorldState

season [long](#)

Returns

[ExploreBoard](#)

## GetExplorer(IWorldState, long, Address)

```
public static Explorer GetExplorer(this IWorldState worldState, long season,  
Address avatarAddress)
```

Parameters

worldState IWorldState

season [long](#)

avatarAddress Address

Returns

[Explorer](#)

## GetExplorerList(IWorldState, long)

```
public static ExplorerList GetExplorerList(this IWorldState worldState, long season)
```

### Parameters

`worldState` IWorldState

`season` [long](#)

### Returns

[ExplorerList](#)

## GetLatestAdventureBossSeason(IWorldState)

Get brief adventure boss season info of latest season. This only has brief, readonly data, so we must use [SeasonInfo](#) to handle season itself.

```
public static SeasonInfo GetLatestAdventureBossSeason(this IWorldState worldState)
```

### Parameters

`worldState` IWorldState

### Returns

[SeasonInfo](#)

LatestSeason state. If no season is started, return a state with default value(0)

## GetSeasonInfo(IWorldState, long)

```
public static SeasonInfo GetSeasonInfo(this IWorldState worldState, long season)
```

### Parameters

`worldState` [IWorldState](#)

`season` [long](#)

Returns

[SeasonInfo](#)

## SetBountyBoard(IWorld, long, BountyBoard)

```
public static IWorld SetBountyBoard(this IWorld world, long season,  
BountyBoard bountyBoard)
```

Parameters

`world` [IWorld](#)

`season` [long](#)

`bountyBoard` [BountyBoard](#)

Returns

[IWorld](#)

## SetExploreBoard(IWorld, long, ExploreBoard)

```
public static IWorld SetExploreBoard(this IWorld world, long season,  
ExploreBoard exploreBoard)
```

Parameters

`world` [IWorld](#)

`season` [long](#)

`exploreBoard` [ExploreBoard](#)

Returns

IWorld

## SetExplorer(IWorld, long, Explorer)

```
public static IWorld SetExplorer(this IWorld world, long season, Explorer explorer)
```

### Parameters

**world** IWorld

**season** [long](#)

**explorer** [Explorer](#)

### Returns

IWorld

## SetExplorerList(IWorld, long, ExplorerList)

```
public static IWorld SetExplorerList(this IWorld world, long season,  
ExplorerList explorerList)
```

### Parameters

**world** IWorld

**season** [long](#)

**explorerList** [ExplorerList](#)

### Returns

IWorld

## SetLatestAdventureBossSeason(IWorld, SeasonInfo)

```
public static IWorld SetLatestAdventureBossSeason(this IWorld world,  
SeasonInfo latestSeasonInfo)
```

## Parameters

**world** IWorld

**latestSeasonInfo** [SeasonInfo](#)

## Returns

IWorld

## SetSeasonInfo(IWorld, SeasonInfo)

```
public static IWorld SetSeasonInfo(this IWorld world, SeasonInfo seasonInfo)
```

## Parameters

**world** IWorld

**seasonInfo** [SeasonInfo](#)

## Returns

IWorld

## TryGetExploreBoard(IWorldState, long, out ExploreBoard)

```
public static bool TryGetExploreBoard(this IWorldState worldState, long season, out  
ExploreBoard exploreBoard)
```

## Parameters

**worldState** IWorldState

season [long](#)

exploreBoard [ExploreBoard](#)

Returns

[bool](#)

## TryGetExplorer(IWorldState, long, Address, out Explorer)

```
public static bool TryGetExplorer(this IWorldState worldState, long season, Address  
avatarAddress, out Explorer explorer)
```

Parameters

worldState [IWorldState](#)

season [long](#)

avatarAddress [Address](#)

explorer [Explorer](#)

Returns

[bool](#)

# Class AgentModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class AgentModule
```

## Inheritance

[object](#) ← AgentModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetAgentState(IWorldState, Address)

```
public static AgentState? GetAgentState(this IWorldState worldState,  
Address address)
```

#### Parameters

**worldState** IWorldState

**address** Address

#### Returns

[AgentState](#)

### SetAgentState(IWorld, Address, AgentState)

```
public static IWorld SetAgentState(this IWorld world, Address agent,  
AgentState state)
```

## Parameters

**world** IWorld

**agent** Address

**state** [AgentState](#)

## Returns

IWorld

# Class ArenaModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class ArenaModule
```

## Inheritance

[object](#) ← ArenaModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetArenaParticipant(IWorldState, int, int, Address)

```
public static ArenaParticipant GetArenaParticipant(this IWorldState worldState, int  
championshipId, int round, Address avatarAddress)
```

#### Parameters

worldState IWorldState

championshipId [int](#)

round [int](#)

avatarAddress Address

#### Returns

[ArenaParticipant](#)

## GetArenaParticipantState(IWorldState, Address, Address)

```
public static IValue GetArenaParticipantState(this IWorldState worldState, Address accountAddress, Address stateAddress)
```

### Parameters

`worldState` IWorldState

`accountAddress` Address

`stateAddress` Address

### Returns

IValue

## GetArenaParticipantState(IWorldState, int, int, Address)

```
public static IValue GetArenaParticipantState(this IWorldState worldState, int championshipId, int round, Address avatarAddress)
```

### Parameters

`worldState` IWorldState

`championshipId` [int](#)

`round` [int](#)

`avatarAddress` Address

### Returns

IValue

## SetArenaParticipant(IWorld, Address, Address, IValue)

```
public static IWorld SetArenaParticipant(this IWorld world, Address accountAddress,  
Address stateAddress, IValue arenaParticipantState)
```

## Parameters

**world** IWorld

**accountAddress** Address

**stateAddress** Address

**arenaParticipantState** IValue

## Returns

IWorld

## SetArenaParticipant(IWorld, int, int, Address, IValue)

```
public static IWorld SetArenaParticipant(this IWorld world, int championshipId, int  
round, Address avatarAddress, IValue arenaParticipantState)
```

## Parameters

**world** IWorld

**championshipId** [int](#)

**round** [int](#)

**avatarAddress** Address

**arenaParticipantState** IValue

## Returns

IWorld

## SetArenaParticipant(IWorld, int, int, Address, ArenaParticipant)

```
public static IWorld SetArenaParticipant(this IWorld world, int championshipId, int round, Address avatarAddress, ArenaParticipant arenaParticipant)
```

### Parameters

`world` IWorld

`championshipId` [int](#)

`round` [int](#)

`avatarAddress` Address

`arenaParticipant` [ArenaParticipant](#)

### Returns

IWorld

# Class AvatarModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class AvatarModule
```

## Inheritance

[object](#) ← AvatarModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetAvatarState(IWorldState, Address, bool, bool, bool)

```
public static AvatarState GetAvatarState(this IWorldState worldState, Address  
address, bool getInventory = true, bool getWorldInformation = true, bool  
getQuestList = true)
```

#### Parameters

**worldState** IWorldState

**address** Address

**getInventory** [bool](#)

**getWorldInformation** [bool](#)

**getQuestList** [bool](#)

#### Returns

[AvatarState](#)

## GetEnemyAvatarState(IWorldState, Address)

```
public static AvatarState GetEnemyAvatarState(this IWorldState worldState,  
Address avatarAddress)
```

### Parameters

`worldState` IWorldState

`avatarAddress` Address

### Returns

[AvatarState](#)

## SetAvatarState(IWorld, Address, AvatarState, bool, bool, bool, bool)

```
public static IWorld SetAvatarState(this IWorld world, Address avatarAddress,  
AvatarState state, bool setAvatar = true, bool setInventory = true, bool  
setWorldInformation = true, bool setQuestList = true)
```

### Parameters

`world` IWorld

`avatarAddress` Address

`state` [AvatarState](#)

`setAvatar` [bool](#)

`setInventory` [bool](#)

`setWorldInformation` [bool](#)

`setQuestList` [bool](#)

### Returns

## TryGetAvatarState(IWorldState, Address, Address, out AvatarState)

```
public static bool TryGetAvatarState(this IWorldState worldState, Address  
agentAddress, Address avatarAddress, out AvatarState avatarState)
```

### Parameters

**worldState** IWorldState

**agentAddress** Address

**avatarAddress** Address

**avatarState** [AvatarState](#)

### Returns

[bool](#) ↗

# Class CollectionModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

Provides utility methods for working with collection states in the world state.

```
public static class CollectionModule
```

## Inheritance

[object](#) ← CollectionModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetCollectionState(IWorldState, Address)

Get the CollectionState for a given address from the world state.

```
public static CollectionState GetCollectionState(this IWorldState worldState,  
Address address)
```

#### Parameters

**worldState** IWorldState

The world state object.

**address** Address

The address to retrieve the CollectionState.

#### Returns

[CollectionState](#)

The CollectionState corresponding to the address.

## Exceptions

### [FailedLoadStateException](#)

Thrown when the Collection state for the given address is not found in the world state.

### [InvalidOperationException](#)

Thrown when the serialized Collection state is not in the correct format.

## GetCollectionStates(IWorldState, IReadOnlyList<Address>)

Retrieves the collection states for the given addresses from the world state.

```
public static Dictionary<Address, CollectionState> GetCollectionStates(this  
IWorldState worldState, IReadOnlyList<Address> addresses)
```

## Parameters

### `worldState` IWorldState

The world state used to retrieve the collection states.

### `addresses` [IReadOnlyList](#)<Address>

The list of addresses to retrieve the collection states for.

## Returns

### [Dictionary](#)<Address, CollectionState>

A dictionary of Address and CollectionState pairs representing the collection states for the given addresses, or an empty dictionary for addresses that do not have a collection state.

## SetCollectionState(IWorld, Address, CollectionState)

Sets the state of a collection in the world.

```
public static IWorld SetCollectionState(this IWorld world, Address collection,
CollectionState state)
```

## Parameters

**world** IWorld

**collection** Address

The address of the collection.

**state** [CollectionState](#)

The state of the collection.

## Returns

IWorld

The updated world with the updated collection state.

## TryGetCollectionState(IWorldState, Address, out CollectionState)

Tries to get the collection state for a specific address from the given world state.

```
public static bool TryGetCollectionState(this IWorldState worldState, Address
address, out CollectionState collectionState)
```

## Parameters

**worldState** IWorldState

The world state from which to get the collection state.

**address** Address

The address for which to retrieve the collection state.

**collectionState** [CollectionState](#)

The resulting collection state, if found.

Returns

[bool](#)

True if the collection state is found, otherwise false.

# Class CombinationSlotStateModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

CombinationSlotStateModule is the module to use CombinationSlotState/AllCombinationSlotState with account.

```
public static class CombinationSlotStateModule
```

## Inheritance

[object](#) ← CombinationSlotStateModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetAllCombinationSlotState(IWorldState, Address)

```
public static AllCombinationSlotState GetAllCombinationSlotState(this IWorldState  
worldState, Address avatarAddress)
```

#### Parameters

**worldState** IWorldState

**avatarAddress** Address

#### Returns

[AllCombinationSlotState](#)

## SetCombinationSlotState(IWorld, Address, AllCombinationSlotState)

```
public static IWorld SetCombinationSlotState(this IWorld world, Address  
avatarAddress, AllCombinationSlotState combinationSlotState)
```

### Parameters

`world` IWorld

`avatarAddress` Address

`combinationSlotState` [AllCombinationSlotState](#)

### Returns

IWorld

# Class DailyRewardModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class DailyRewardModule
```

## Inheritance

[object](#) ← DailyRewardModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetDailyRewardReceivedBlockIndex(IWorldState, Address)

```
public static long GetDailyRewardReceivedBlockIndex(this IWorldState worldState,  
Address avatarAddress)
```

#### Parameters

[worldState](#) IWorldState

[avatarAddress](#) Address

#### Returns

[long](#)

### SetDailyRewardReceivedBlockIndex(IWorld, Address, long)

```
public static IWorld SetDailyRewardReceivedBlockIndex(this IWorld world, Address  
avatarAddress, long blockIndex)
```

## Parameters

**world** IWorld

**avatarAddress** Address

**blockIndex** [long](#)

## Returns

IWorld

## TryGetDailyRewardReceivedBlockIndex(IWorldState, Address, out long)

```
public static bool TryGetDailyRewardReceivedBlockIndex(this IWorldState worldState,  
Address avatarAddress, out long receivedBlockIndex)
```

## Parameters

**worldState** IWorldState

**avatarAddress** Address

**receivedBlockIndex** [long](#)

## Returns

[bool](#)

# Class LegacyModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class LegacyModule
```

## Inheritance

[object](#) ← LegacyModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetArenaAvatarState(IWorldState, Address, AvatarState)

```
public static ArenaAvatarState GetArenaAvatarState(this IWorldState worldState,  
Address arenaAvatarStateAddress, AvatarState avatarState)
```

#### Parameters

`worldState` IWorldState

`arenaAvatarStateAddress` Address

`avatarState` [AvatarState](#)

#### Returns

[ArenaAvatarState](#)

## GetArenaInfo(IWorldState, Address, AvatarState, CharacterSheet, CostumeStatSheet)

```
public static (Address arenaInfoAddress, ArenaInfo arenaInfo, bool isNewArenaInfo)
GetArenaInfo(this IWorldState worldState, Address weeklyArenaAddress, AvatarState
avatarState, CharacterSheet characterSheet, CostumeStatSheet costumeStatSheet)
```

### Parameters

`worldState` IWorldState

`weeklyArenaAddress` Address

`avatarState` [AvatarState](#)

`characterSheet` [CharacterSheet](#)

`costumeStatSheet` [CostumeStatSheet](#)

### Returns

(Address [arenaInfoAddress](#), ArenaInfo [arenaInfo](#), bool [isNewArenaInfo](#))

## GetArenaParticipants(IWorldState, Address, int, int)

```
public static ArenaParticipants GetArenaParticipants(this IWorldState worldState,
Address arenaParticipantsAddress, int id, int round)
```

### Parameters

`worldState` IWorldState

`arenaParticipantsAddress` Address

`id` [int](#)

`round` [int](#)

### Returns

## [ArenaParticipants](#)

### GetAvatarSheets(IWorldState)

```
public static AvatarSheets GetAvatarSheets(this IWorldState worldState)
```

#### Parameters

`worldState` IWorldState

#### Returns

[AvatarSheets](#)

### GetCombinationSlotStateLegacy(IWorldState, Address, int)

```
[Obsolete("Use AllCombinationSlotState.GetRuneState() instead.")]
public static CombinationSlotState GetCombinationSlotStateLegacy(this IWorldState
worldState, Address avatarAddress, int index)
```

#### Parameters

`worldState` IWorldState

`avatarAddress` Address

`index` [int](#)

#### Returns

[CombinationSlotState](#)

### GetCouponWallet(IWorldState, Address)

```
public static IImmutableDictionary<Guid, Coupon> GetCouponWallet(this IWorldState
```

```
worldState, Address agentAddress)
```

## Parameters

**worldState** IWorldState

**agentAddress** Address

## Returns

[IImmutableDictionary<Guid, Coupon>](#)

## GetCrystalCostState(IWorldState, Address)

```
public static CrystalCostState GetCrystalCostState(this IWorldState worldState,  
Address address)
```

## Parameters

**worldState** IWorldState

**address** Address

## Returns

[CrystalCostState](#)

## GetCrystalCostStates(IWorldState, long, long)

```
public static (CrystalCostState DailyCostState, CrystalCostState WeeklyCostState,  
CrystalCostState PrevWeeklyCostState, CrystalCostState BeforePrevWeeklyCostState)  
GetCrystalCostStates(this IWorldState worldState, long blockIndex, long interval)
```

## Parameters

**worldState** IWorldState

**blockIndex** [long](#)

interval [long](#)

Returns

([CrystalCostState DailyCostState](#), [CrystalCostState WeeklyCostState](#), [CrystalCostState PrevWeeklyCostState](#), [CrystalCostState BeforePrevWeeklyCostState](#))

## GetGameConfigState(IWorldState)

```
public static GameConfigState GetGameConfigState(this IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

[GameConfigState](#)

## GetGoldBalanceState(IWorldState, Address, Currency)

```
public static GoldBalanceState GetGoldBalanceState(this IWorldState worldState,  
Address address, Currency currency)
```

Parameters

**worldState** IWorldState

**address** Address

**currency** Currency

Returns

[GoldBalanceState](#)

## GetGoldCurrency(IWorldState)

```
public static Currency GetGoldCurrency(this IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

Currency

## GetGoldDistribution(IWorldState)

```
public static IEnumerable<GoldDistribution> GetGoldDistribution(this IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

[IEnumerable](#)<[GoldDistribution](#)>

## GetItemSheet(IWorldState)

```
public static ItemSheet GetItemSheet(this IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

[ItemSheet](#)

## GetLegacyState(IWorldState, Address)

```
public static IValue GetLegacyState(this IWorldState worldState, Address address)
```

### Parameters

**worldState** IWorldState

**address** Address

### Returns

IValue

## GetLegacyStates(IWorldState, IReadOnlyList<Address>)

```
public static IReadOnlyList<IValue?> GetLegacyStates(this IWorldState worldState,  
IReadOnlyList<Address> addresses)
```

### Parameters

**worldState** IWorldState

**addresses** [IReadOnlyList](#)<Address>

### Returns

[IReadOnlyList](#)<IValue>

## GetLegacyStatesAsDict(IWorldState, params Address[])

```
public static Dictionary<Address, IValue> GetLegacyStatesAsDict(this IWorldState  
worldState, params Address[] addresses)
```

### Parameters

**worldState** IWorldState

**addresses** Address[]

Returns

[Dictionary](#) <Address, IValue>

## GetQuestSheet(IWorldState)

```
public static QuestSheet GetQuestSheet(this IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

[QuestSheet](#)

## GetRaiderState(IWorldState, Address)

```
public static RaiderState GetRaiderState(this IWorldState worldState,  
Address raiderAddress)
```

Parameters

**worldState** IWorldState

**raiderAddress** Address

Returns

[RaiderState](#)

## GetRaiderState(IWorldState, Address, int)

```
public static RaiderState GetRaiderState(this IWorldState worldState, Address  
avatarAddress, int raidId)
```

## Parameters

`worldState` IWorldState

`avatarAddress` Address

`raidId` [int](#)

## Returns

[RaiderState](#)

## GetRankingSimulatorSheets(IWorldState)

```
public static RankingSimulatorSheets GetRankingSimulatorSheets(this  
IWorldState worldState)
```

## Parameters

`worldState` IWorldState

## Returns

[RankingSimulatorSheets](#)

## GetRankingSimulatorSheetsV1(IWorldState)

```
public static RankingSimulatorSheetsV1 GetRankingSimulatorSheetsV1(this  
IWorldState worldState)
```

## Parameters

`worldState` IWorldState

Returns

[RankingSimulatorSheetsV1](#)

## GetRankingState(IWorldState)

```
public static RankingState GetRankingState(this IWorldState worldState)
```

Parameters

`worldState` IWorldState

Returns

[RankingState](#)

## GetRankingState0(IWorldState)

```
public static RankingState0 GetRankingState0(this IWorldState worldState)
```

Parameters

`worldState` IWorldState

Returns

[RankingState0](#)

## GetRankingState1(IWorldState)

```
public static RankingState1 GetRankingState1(this IWorldState worldState)
```

Parameters

`worldState` IWorldState

Returns

[RankingState1](#)

## GetRedeemCodeState(IWorldState)

```
public static RedeemCodeState GetRedeemCodeState(this IWorldState worldState)
```

Parameters

`worldState` IWorldState

Returns

[RedeemCodeState](#)

## GetSheetCsv(IWorldState, Address)

```
public static string GetSheetCsv(this IWorldState worldState, Address address)
```

Parameters

`worldState` IWorldState

`address` Address

Returns

[string](#)

## GetSheetCsv<T>(IWorldState)

```
public static string GetSheetCsv<T>(this IWorldState worldState) where T :  
ISheet, new()
```

Parameters

`worldState` IWorldState

Returns

[string](#) ↗

Type Parameters

T

## GetSheet<T>(IWorldState)

```
public static T GetSheet<T>(this IWorldState worldState) where T : ISheet, new()
```

Parameters

`worldState` IWorldState

Returns

T

Type Parameters

T

## GetSheet<T>(IWorldState, Address)

```
public static T GetSheet<T>(this IWorldState worldState, Address sheetAddr) where T : ISheet, new()
```

Parameters

`worldState` IWorldState

`sheetAddr` Address

Returns

T

Type Parameters

T

GetSheets(IWorldState, bool, bool, bool, bool, bool, bool, bool, bool, bool, IEnumerable<Type>)

```
public static Dictionary<Type, (Address address, ISheet sheet)> GetSheets(this  
IWorldState worldState, bool containAvatarSheets = false, bool containItemSheet =  
false, bool containQuestSheet = false, bool containSimulatorSheets = false, bool  
containStageSimulatorSheets = false, bool containRankingSimulatorSheets = false,  
bool containArenaSimulatorSheets = false, bool containValidateItemRequirementSheets  
= false, bool containRaidSimulatorSheets = false, IEnumerable<Type> sheetTypes  
= null)
```

Parameters

worldState IWorldState

containAvatarSheets [bool](#)

containItemSheet [bool](#)

containQuestSheet [bool](#)

containSimulatorSheets [bool](#)

containStageSimulatorSheets [bool](#)

containRankingSimulatorSheets [bool](#)

containArenaSimulatorSheets [bool](#)

containValidateItemRequirementSheets [bool](#)

containRaidSimulatorSheets [bool](#)

sheetTypes [IEnumerable](#)<Type>

Returns

[Dictionary](#)<[Type](#), (Address [address](#), [ISheet sheet](#))>

## GetSheets(IWorldState, params Type[])

```
public static Dictionary<Type, (Address address, ISheet sheet)> GetSheets(this  
IWorldState worldState, params Type[] sheetTypes)
```

Parameters

**worldState** IWorldState

**sheetTypes** [Type](#)[]

Returns

[Dictionary](#)<[Type](#), (Address [address](#), [ISheet sheet](#))>

## GetSheets(IWorldState, params (Type sheetType, string sheetName)[])

```
public static Dictionary<Type, (Address address, ISheet sheet)> GetSheets(this  
IWorldState worldState, params (Type sheetType, string sheetName)[] sheetTuples)
```

Parameters

**worldState** IWorldState

**sheetTuples** ([Type](#) [sheetType](#), [string](#) [sheetName](#))[]

Returns

[Dictionary](#)<[Type](#), (Address [address](#), [ISheet sheet](#))>

## GetSheetsV1(IWorldState, bool, bool, bool, bool, bool, bool, bool, bool, bool, IEnumerable<Type>)

```
public static Dictionary<Type, (Address address, ISheet sheet)> GetSheetsV1(this  
IWorldState worldState, bool containAvatarSheets = false, bool containItemSheet =  
false, bool containQuestSheet = false, bool containSimulatorSheets = false, bool  
containStageSimulatorSheets = false, bool containRankingSimulatorSheets = false,  
bool containArenaSimulatorSheets = false, bool containValidateItemRequirementSheets  
= false, bool containRaidSimulatorSheets = false, IEnumerable<Type> sheetTypes  
= null)
```

### Parameters

worldState IWorldState

containAvatarSheets [bool](#)

containItemSheet [bool](#)

containQuestSheet [bool](#)

containSimulatorSheets [bool](#)

containStageSimulatorSheets [bool](#)

containRankingSimulatorSheets [bool](#)

containArenaSimulatorSheets [bool](#)

containValidateItemRequirementSheets [bool](#)

containRaidSimulatorSheets [bool](#)

sheetTypes [IEnumerable](#)<[Type](#)>

### Returns

[Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

## GetSheetsV100291(IWorldState, bool, bool, bool, bool, bool, bool, bool, bool, bool, IEnumerable<Type>)

```
public static Dictionary<Type, (Address address, ISheet sheet)>
GetSheetsV100291(this IWorldState worldState, bool containAvatarSheets = false, bool
containItemSheet = false, bool containQuestSheet = false, bool
containSimulatorSheets = false, bool containStageSimulatorSheets = false, bool
containRankingSimulatorSheets = false, bool containArenaSimulatorSheets = false,
bool containValidateItemRequirementSheets = false, IEnumerable<Type> sheetTypes
= null)
```

## Parameters

worldState IWorldState

containAvatarSheets [bool](#)

containItemSheet [bool](#)

containQuestSheet [bool](#)

containSimulatorSheets [bool](#)

containStageSimulatorSheets [bool](#)

containRankingSimulatorSheets [bool](#)

containArenaSimulatorSheets [bool](#)

containValidateItemRequirementSheets [bool](#)

sheetTypes [IEnumerable](#)<[Type](#)>

## Returns

[Dictionary](#)<[Type](#), (Address [address](#), [ISheet](#) [sheet](#))>

## GetShopState(IWorldState)

```
public static ShopState GetShopState(this IWorldState worldState)
```

## Parameters

worldState IWorldState

Returns

[ShopState](#)

## GetStageSimulatorSheets(IWorldState)

```
public static StageSimulatorSheets GetStageSimulatorSheets(this  
IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

[StageSimulatorSheets](#)

## GetStageSimulatorSheetsV1(IWorldState)

```
public static StageSimulatorSheetsV1 GetStageSimulatorSheetsV1(this  
IWorldState worldState)
```

Parameters

**worldState** IWorldState

Returns

[StageSimulatorSheetsV1](#)

## GetStakedAmount(IWorldState, Address)

```
public static FungibleAssetValue GetStakedAmount(this IWorldState worldState,  
Address agentAddr)
```

## Parameters

`worldState` IWorldState

`agentAddr` Address

## Returns

FungibleAssetValue

## GetWeeklyArenaState(IWorldState, Address)

```
public static WeeklyArenaState GetWeeklyArenaState(this IWorldState worldState,  
Address address)
```

## Parameters

`worldState` IWorldState

`address` Address

## Returns

[WeeklyArenaState](#)

## GetWeeklyArenaState(IWorldState, int)

```
public static WeeklyArenaState GetWeeklyArenaState(this IWorldState worldState,  
int index)
```

## Parameters

`worldState` IWorldState

`index` [int](#)

## Returns

[WeeklyArenaState](#)

## MarkBalanceChanged(IWorld, IActionContext, Currency, params Address[])

```
public static IWorld MarkBalanceChanged(this IWorld world, IActionContext context, Currency currency, params Address[] accounts)
```

### Parameters

`world` IWorld

`context` IActionContext

`currency` Currency

`accounts` Address[]

### Returns

IWorld

## Mead(IWorld, IActionContext, Address, BigInteger)

```
public static IWorld Mead(this IWorld world, IActionContext context, Address signer, BigInteger rawValue)
```

### Parameters

`world` IWorld

`context` IActionContext

`signer` Address

`rawValue` [BigInteger](#)

### Returns

IWorld

## RemoveLegacyState(IWorld, Address)

```
public static IWorld RemoveLegacyState(this IWorld world, Address address)
```

### Parameters

**world** IWorld

**address** Address

### Returns

IWorld

## SetCouponWallet(IWorld, Address, IImmutableDictionary<Guid, Coupon>)

```
public static IWorld SetCouponWallet(this IWorld world, Address agentAddress,  
IImmutableDictionary<Guid, Coupon> couponWallet)
```

### Parameters

**world** IWorld

**agentAddress** Address

**couponWallet** [IImmutableDictionary<Guid, Coupon>](#)

### Returns

IWorld

## SetLegacyState(IWorld, Address, IValue)

```
public static IWorld SetLegacyState(this IWorld world, Address address,
```

```
IValue state)
```

## Parameters

**world** IWorld

**address** Address

**state** IValue

## Returns

IWorld

**SetWorldBossKillReward(IWorld, IActionContext, Address, WorldBossKillRewardRecord, int, WorldBossState, RuneWeightSheet, WorldBossKillRewardSheet, RuneSheet, MaterialItemSheet, IRandom, Inventory, Address, Address)**

```
public static IWorld SetWorldBossKillReward(this IWorld world, IActionContext context, Address rewardInfoAddress, WorldBossKillRewardRecord rewardRecord, int rank, WorldBossState bossState, RuneWeightSheet runeWeightSheet, WorldBossKillRewardSheet worldBossKillRewardSheet, RuneSheet runeSheet, MaterialItemSheet materialItemSheet, IRandom random, Inventory inventory, Address avatarAddress, Address agentAddress)
```

## Parameters

**world** IWorld

**context** IActionContext

**rewardInfoAddress** Address

**rewardRecord** [WorldBossKillRewardRecord](#)

**rank** [int](#)

bossState [WorldBossState](#)

runeWeightSheet [RuneWeightSheet](#)

worldBossKillRewardSheet [WorldBossKillRewardSheet](#)

runeSheet [RuneSheet](#)

materialItemSheet [MaterialItemSheet](#)

random IRandom

inventory [Inventory](#)

avatarAddress Address

agentAddress Address

Returns

IWorld

## TryGetArenaAvatarState(IWorldState, Address, out ArenaAvatarState)

```
public static bool TryGetArenaAvatarState(this IWorldState worldState, Address arenaAvatarStateAddress, out ArenaAvatarState arenaAvatarState)
```

Parameters

worldState IWorldState

arenaAvatarStateAddress Address

arenaAvatarState [ArenaAvatarState](#)

Returns

[bool](#) ↗

## TryGetArenaInformation(IWorldState, Address, out ArenaInformation)

```
public static bool TryGetArenaInformation(this IWorldState worldState, Address arenaInformationAddress, out ArenaInformation arenaInformation)
```

### Parameters

`worldState` IWorldState

`arenaInformationAddress` Address

`arenaInformation` [ArenaInformation](#)

### Returns

[bool](#) ↗

## TryGetArenaParticipants(IWorldState, Address, out ArenaParticipants)

```
public static bool TryGetArenaParticipants(this IWorldState worldState, Address arenaParticipantsAddress, out ArenaParticipants arenaParticipants)
```

### Parameters

`worldState` IWorldState

`arenaParticipantsAddress` Address

`arenaParticipants` [ArenaParticipants](#)

### Returns

[bool](#) ↗

## TryGetArenaScore(IWorldState, Address, out ArenaScore)

```
public static bool TryGetArenaScore(this IWorldState worldState, Address arenaScoreAddress, out ArenaScore arenaScore)
```

### Parameters

**worldState** IWorldState

**arenaScoreAddress** Address

**arenaScore** [ArenaScore](#)

### Returns

[bool](#)

## TryGetGoldBalance(IWorldState, Address, Currency, out FungibleAssetValue)

```
public static bool TryGetGoldBalance(this IWorldState worldState, Address address, Currency currency, out FungibleAssetValue balance)
```

### Parameters

**worldState** IWorldState

**address** Address

**currency** Currency

**balance** FungibleAssetValue

### Returns

[bool](#)

## TryGetLegacyStakeState(IWorldState, Address, out LegacyStakeState)

```
public static bool TryGetLegacyStakeState(this IWorldState worldState, Address agentAddress, out LegacyStakeState legacyStakeState)
```

### Parameters

`worldState` IWorldState

`agentAddress` Address

`legacyStakeState` [LegacyStakeState](#)

### Returns

`bool` ↗

## TryGetLegacyState<T>(IWorldState, Address, out T)

```
public static bool TryGetLegacyState<T>(this IWorldState worldState, Address address, out T result) where T : IValue
```

### Parameters

`worldState` IWorldState

`address` Address

`result` T

### Returns

`bool` ↗

### Type Parameters

T

## TryGetSheet<T>(IWorldState, Address, out T)

```
public static bool TryGetSheet<T>(this IWorldState worldState, Address address, out  
T sheet) where T : ISheet, new()
```

### Parameters

`worldState` IWorldState

`address` Address

`sheet` T

### Returns

[bool](#)

### Type Parameters

T

## TryGetSheet<T>(IWorldState, out T)

```
public static bool TryGetSheet<T>(this IWorldState worldState, out T sheet) where T  
: ISheet, new()
```

### Parameters

`worldState` IWorldState

`sheet` T

### Returns

[bool](#)

### Type Parameters

T

## TryGetStakeState(IWorldState, Address, out StakeState)

```
public static bool TryGetStakeState(this IWorldState worldState, Address agentAddr,  
out StakeState stakeState)
```

### Parameters

**worldState** IWorldState

**agentAddr** Address

**stakeState** [StakeState](#)

### Returns

[bool](#)

## ValidateWorldId(IWorldState, Address, int)

```
public static void ValidateWorldId(this IWorldState worldState, Address  
avatarAddress, int worldId)
```

### Parameters

**worldState** IWorldState

**avatarAddress** Address

**worldId** [int](#)

# Class RelationshipModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class RelationshipModule
```

## Inheritance

[object](#) ← RelationshipModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetRelationship(IWorldState, Address)

```
public static int GetRelationship(this IWorldState state, Address avatarAddress)
```

#### Parameters

state IWorldState

avatarAddress Address

#### Returns

[int](#)

### SetRelationship(IWorld, Address, int)

```
public static IWorld SetRelationship(this IWorld world, Address avatarAddress,  
int relationship)
```

## Parameters

`world` `IWorld`

`avatarAddress` `Address`

`relationship` `int`

## Returns

`IWorld`

# Class RuneStateModule

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

RuneStateModule is the module to use RuneState/AllRuneState with account.

```
public static class RuneStateModule
```

## Inheritance

[object](#) ← RuneStateModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetRuneState(IWorldState, Address, out bool)

```
public static AllRuneState GetRuneState(this IworldState worldState, Address  
avatarAddress, out bool migrateRequired)
```

#### Parameters

**worldState** IWorldState

**avatarAddress** Address

**migrateRequired** [bool](#)

#### Returns

[AllRuneState](#)

### SetRuneState(IWorld, Address, AllRuneState)

```
public static IWorld SetRuneState(this IWorld world, Address avatarAddress,  
AllRuneState allRuneState)
```

## Parameters

**world** IWorld

**avatarAddress** Address

**allRuneState** [AllRuneState](#)

## Returns

IWorld

# Class StateResolver

Namespace: [Nekoyume.Module](#)

Assembly: Lib9c.dll

```
public static class StateResolver
```

## Inheritance

[object](#) ← StateResolver

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetResolvedState(IWorldState, Address, Address)

Gets the account of address `address` if it exists. If not, returns from the legacy account.

```
public static IValue? GetResolvedState(this IWorldState world, Address address,  
Address accountAddress)
```

#### Parameters

`world` IWorldState

An `Libplanet.Action.State.IWorld` instance of to get state from.

`address` Address

The `Libplanet.Crypto.Address` of the state to get.

`accountAddress` Address

The `Libplanet.Crypto.Address` of the account to get.

#### Returns

## IValue

An Libplanet.Action.State.IAccountState instance of address `address` if exists. If not, legacy account.

## GetResolvedState(IWorldState, Address, Address, Address)

Gets the account of address `address` if it exists. If not, returns from the legacy account with `legacyAddress`.

```
public static IValue? GetResolvedState(this IWorldState world, Address address,  
Address accountAddress, Address legacyAddress)
```

### Parameters

`world` IWorldState

An Libplanet.Action.State.IWorld instance of to get state from.

`address` Address

The Libplanet.Crypto.Address of the state to get.

`accountAddress` Address

The Libplanet.Crypto.Address of the account to get.

`legacyAddress` Address

The Libplanet.Crypto.Address of the state to get from the legacy account.

### Returns

## IValue

An Libplanet.Action.State.IAccountState instance of address `address` if exists. If not, legacy account.

# Namespace Nekoyume.Module. CombinationSlot

## Classes

[CombinationSlotNotFoundException](#)

[DuplicatedCombinationSlotIndexException](#)

# Class CombinationSlotNotFoundException

Namespace: [Nekoyume.Module.CombinationSlot](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationSlotNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← CombinationSlotNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### CombinationSlotNotFoundException(SerializationInfo, StreamingContext)

```
protected CombinationSlotNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## CombinationSlotNotFoundException(string)

```
public CombinationSlotNotFoundException(string message)
```

### Parameters

message [string](#)

# Class DuplicatedCombinationSlotIndexException

Namespace: [Nekoyume.Module.CombinationSlot](#)

Assembly: Lib9c.dll

```
[Serializable]
public class DuplicatedCombinationSlotIndexException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← DuplicatedCombinationSlotIndexException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### DuplicatedCombinationSlotIndexException(SerializationInfo, StreamingContext)

```
protected DuplicatedCombinationSlotIndexException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## DuplicatedCombinationSlotIndexException(string)

```
public DuplicatedCombinationSlotIndexException(string message)
```

### Parameters

message [string](#)

# Namespace Nekoyume.Module.Guild

## Classes

[GuildBanModule](#)

[GuildDelegateeModule](#)

[GuildMemberCounterModule](#)

[GuildModule](#)

[GuildParticipantModule](#)

# Class GuildBanModule

Namespace: [Nekoyume.Module.Guild](#)

Assembly: Lib9c.dll

```
public static class GuildBanModule
```

## Inheritance

[object](#) ← GuildBanModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Ban(GuildRepository, GuildAddress, AgentAddress, AgentAddress)

```
public static void Ban(this GuildRepository repository, GuildAddress guildAddress,  
AgentAddress signer, AgentAddress target)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

signer [AgentAddress](#)

target [AgentAddress](#)

### IsBanned(GuildRepository, GuildAddress, Address)

```
public static bool IsBanned(this GuildRepository repository, GuildAddress
```

```
guildAddress, Address agentAddress)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

agentAddress Address

Returns

[bool](#) ↗

## RemoveBanList(GuildRepository, GuildAddress)

```
public static void RemoveBanList(this GuildRepository repository,  
GuildAddress guildAddress)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

## Unban(GuildRepository, GuildAddress, AgentAddress, Address)

```
public static void Unban(this GuildRepository repository, GuildAddress guildAddress,  
AgentAddress signer, Address target)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

signer [AgentAddress](#)

**target** Address

# Class GuildDelegateeModule

Namespace: [Nekoyume.Module.Guild](#)

Assembly: Lib9c.dll

```
public static class GuildDelegateeModule
```

## Inheritance

[object](#) ← GuildDelegateeModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### CreateGuildDelegatee(GuildRepository, Address)

```
public static GuildDelegatee CreateGuildDelegatee(this GuildRepository repository,  
Address address)
```

#### Parameters

repository [GuildRepository](#)

address Address

#### Returns

[GuildDelegatee](#)

### TryGetGuildDelegatee(GuildRepository, Address, out GuildDelegatee?)

```
public static bool TryGetGuildDelegatee(this GuildRepository repository, Address  
address, out GuildDelegatee? validatorDelegateeForGuildParticipant)
```

## Parameters

repository [GuildRepository](#)

address Address

validatorDelegateeForGuildParticipant [GuildDelegatee](#)

## Returns

[bool](#) ↗

# Class GuildMemberCounterModule

Namespace: [Nekoyume.Module.Guild](#)

Assembly: Lib9c.dll

```
public static class GuildMemberCounterModule
```

## Inheritance

[object](#) ← GuildMemberCounterModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### DecreaseGuildMemberCount(GuildRepository, GuildAddress)

```
public static GuildRepository DecreaseGuildMemberCount(this GuildRepository  
repository, GuildAddress guildAddress)
```

#### Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

#### Returns

[GuildRepository](#)

### GetGuildMemberCount(GuildRepository, GuildAddress)

```
public static BigInteger GetGuildMemberCount(this GuildRepository repository,  
GuildAddress guildAddress)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

Returns

[BigInteger](#)

## IncreaseGuildMemberCount(GuildRepository, GuildAddress)

```
public static GuildRepository IncreaseGuildMemberCount(this GuildRepository  
repository, GuildAddress guildAddress)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

Returns

[GuildRepository](#)

# Class GuildModule

Namespace: [Nekoyume.Module.Guild](#)

Assembly: Lib9c.dll

```
public static class GuildModule
```

## Inheritance

[object](#) ← GuildModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetGuildRepository(IWorld, IActionContext)

```
public static GuildRepository GetGuildRepository(this IWorld world,  
IActionContext context)
```

#### Parameters

**world** IWorld

**context** IActionContext

#### Returns

[GuildRepository](#)

### MakeGuild(GuildRepository, GuildAddress, Address)

```
public static GuildRepository MakeGuild(this GuildRepository repository,  
GuildAddress guildAddress, Address validatorAddress)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

validatorAddress Address

Returns

[GuildRepository](#)

## RemoveGuild(GuildRepository)

```
public static GuildRepository RemoveGuild(this GuildRepository repository)
```

Parameters

repository [GuildRepository](#)

Returns

[GuildRepository](#)

## TryGetGuild(GuildRepository, GuildAddress, out Guild?)

```
public static bool TryGetGuild(this GuildRepository repository, GuildAddress
    guildAddress, out Guild? guild)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

guild [Guild](#)

Returns

bool ↴

# Class GuildParticipantModule

Namespace: [Nekoyume.Module.Guild](#)

Assembly: Lib9c.dll

```
public static class GuildParticipantModule
```

## Inheritance

[object](#) ← GuildParticipantModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

Delegate(GuildRepository, AgentAddress,  
FungibleAssetValue)

```
public static GuildRepository Delegate(this GuildRepository repository, AgentAddress  
guildParticipantAddress, FungibleAssetValue fav)
```

### Parameters

repository [GuildRepository](#)

guildParticipantAddress [AgentAddress](#)

fav FungibleAssetValue

### Returns

[GuildRepository](#)

GetJoinedGuild(GuildRepository, AgentAddress)

```
public static GuildAddress? GetJoinedGuild(this GuildRepository repository,  
AgentAddress agentAddress)
```

Parameters

repository [GuildRepository](#)

agentAddress [AgentAddress](#)

Returns

[GuildAddress?](#)

## JoinGuild(GuildRepository, GuildAddress, AgentAddress)

```
public static GuildRepository JoinGuild(this GuildRepository repository,  
GuildAddress guildAddress, AgentAddress target)
```

Parameters

repository [GuildRepository](#)

guildAddress [GuildAddress](#)

target [AgentAddress](#)

Returns

[GuildRepository](#)

## LeaveGuild(GuildRepository, AgentAddress)

```
public static GuildRepository LeaveGuild(this GuildRepository repository,  
AgentAddress agentAddress)
```

Parameters

repository [GuildRepository](#)

agentAddress [AgentAddress](#)

Returns

[GuildRepository](#)

## MoveGuild(GuildRepository, AgentAddress, GuildAddress)

```
public static GuildRepository MoveGuild(this GuildRepository repository,  
AgentAddress guildParticipantAddress, GuildAddress dstGuildAddress)
```

Parameters

repository [GuildRepository](#)

guildParticipantAddress [AgentAddress](#)

dstGuildAddress [GuildAddress](#)

Returns

[GuildRepository](#)

## Redelegate(GuildRepository, AgentAddress, GuildAddress)

```
public static GuildRepository Redefine(this GuildRepository repository,  
AgentAddress guildParticipantAddress, GuildAddress dstGuildAddress)
```

Parameters

repository [GuildRepository](#)

guildParticipantAddress [AgentAddress](#)

dstGuildAddress [GuildAddress](#)

Returns

[GuildRepository](#)

**TryGetGuildParticipant(GuildRepository, AgentAddress, out GuildParticipant?)**

```
public static bool TryGetGuildParticipant(this GuildRepository repository,  
AgentAddress agentAddress, out GuildParticipant? guildParticipant)
```

Parameters

repository [GuildRepository](#)

agentAddress [AgentAddress](#)

guildParticipant [GuildParticipant](#)

Returns

[bool](#) ↗

# Namespace Nekoyume.Module.Validator Delegation

## Classes

[AbstainHistoryModule](#)

[ValidatorDelegateeModule](#)

[ValidatorDelegatorModule](#)

[ValidatorListModule](#)

[ValidatorUnbondingModule](#)

# Class AbstainHistoryModule

Namespace: [Nekoyume.Module.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public static class AbstainHistoryModule
```

## Inheritance

[object](#) ← AbstainHistoryModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetAbstainHistory(ValidatorRepository)

```
public static AbstainHistory GetAbstainHistory(this ValidatorRepository repository)
```

#### Parameters

repository [ValidatorRepository](#)

#### Returns

[AbstainHistory](#)

### SetAbstainHistory(ValidatorRepository, AbstainHistory)

```
public static ValidatorRepository SetAbstainHistory(this ValidatorRepository  
repository, AbstainHistory abstainHistory)
```

#### Parameters

repository [ValidatorRepository](#)

abstainHistory [AbstainHistory](#)

Returns

[ValidatorRepository](#)

# Class ValidatorDelegateeModule

Namespace: [Nekoyume.Module.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public static class ValidatorDelegateeModule
```

## Inheritance

[object](#) ← ValidatorDelegateeModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

CreateValidatorDelegatee(ValidatorRepository,  
PublicKey, BigInteger)

```
public static ValidatorDelegatee CreateValidatorDelegatee(this ValidatorRepository  
repository, PublicKey publicKey, BigInteger commissionPercentage)
```

### Parameters

repository [ValidatorRepository](#)

publicKey PublicKey

commissionPercentage [BigInteger](#)

### Returns

[ValidatorDelegatee](#)

## TryGetValidatorDelegatee(ValidatorRepository, Address, out ValidatorDelegatee?)

```
public static bool TryGetValidatorDelegatee(this ValidatorRepository repository,  
Address address, out ValidatorDelegatee? validatorDelegatee)
```

### Parameters

repository [ValidatorRepository](#)

address Address

validatorDelegatee [ValidatorDelegatee](#)

### Returns

[bool](#) ↗

# Class ValidatorDelegatorModule

Namespace: [Nekoyume.Module.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public static class ValidatorDelegatorModule
```

## Inheritance

[object](#) ← ValidatorDelegatorModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

TryGetValidatorDelegator(ValidatorRepository, Address, out ValidatorDelegator?)

```
public static bool TryGetValidatorDelegator(this ValidatorRepository repository,  
Address address, out ValidatorDelegator? validatorDelegator)
```

### Parameters

repository [ValidatorRepository](#)

address Address

validatorDelegator [ValidatorDelegator](#)

### Returns

[bool](#)

# Class ValidatorListModule

Namespace: [Nekoyume.Module.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public static class ValidatorListModule
```

## Inheritance

[object](#) ← ValidatorListModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### GetProposerInfo(ValidatorRepository)

```
public static ProposerInfo GetProposerInfo(this ValidatorRepository repository)
```

#### Parameters

repository [ValidatorRepository](#)

#### Returns

[ProposerInfo](#)

### GetValidatorList(ValidatorRepository)

```
public static ValidatorList GetValidatorList(this ValidatorRepository repository)
```

#### Parameters

repository [ValidatorRepository](#)

Returns

[ValidatorList](#)

## SetProposerInfo(ValidatorRepository, ProposerInfo)

```
public static ValidatorRepository SetProposerInfo(this ValidatorRepository  
repository, ProposerInfo proposerInfo)
```

Parameters

repository [ValidatorRepository](#)

proposerInfo [ProposerInfo](#)

Returns

[ValidatorRepository](#)

## TryGetProposerInfo(ValidatorRepository, Address, out ProposerInfo?)

```
public static bool TryGetProposerInfo(this ValidatorRepository repository, Address  
address, out ProposerInfo? proposerInfo)
```

Parameters

repository [ValidatorRepository](#)

address Address

proposerInfo [ProposerInfo](#)

Returns

[bool](#) ↗

## TryGetValidatorList(ValidatorRepository, out ValidatorList?)

```
public static bool TryGetValidatorList(this ValidatorRepository repository, out ValidatorList? validatorList)
```

### Parameters

repository [ValidatorRepository](#)

validatorList [ValidatorList](#)

### Returns

bool ↗

# Class ValidatorUnbondingModule

Namespace: [Nekoyume.Module.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public static class ValidatorUnbondingModule
```

## Inheritance

[object](#) ← ValidatorUnbondingModule

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ReleaseUnbondings(ValidatorRepository)

```
public static ValidatorRepository ReleaseUnbondings(this ValidatorRepository  
repository)
```

#### Parameters

repository [ValidatorRepository](#)

#### Returns

[ValidatorRepository](#)

# Namespace Nekoyume.TableData

## Classes

[ActionBuffSheet](#)

[ActionBuffSheet.Row](#)

[ArenaSheet](#)

[ArenaSheet.RoundData](#)

[ArenaSheet.Row](#)

[ArenaSimulatorSheets](#)

[ArenaSimulatorSheetsV1](#)

[ArenaSimulatorSheetsV100291](#)

[AvatarSheets](#)

[BuffLinkSheet](#)

[BuffLinkSheet.Row](#)

[BuffSheet](#)

[BuffSheet.Row](#)

[CharacterLevelSheet](#)

[CharacterLevelSheet.Row](#)

[CharacterSheet](#)

[CharacterSheet.Row](#)

[CharacterSheetExtension](#)

[CollectQuestSheet](#)

[CollectQuestSheet.Row](#)

[CollectionSheet](#)

[CollectionSheet.RequiredMaterial](#)

[CollectionSheet.Row](#)

[CombinationEquipmentQuestSheet](#)

[CombinationEquipmentQuestSheet.Row](#)

[CombinationQuestSheet](#)

[CombinationQuestSheet.Row](#)

[ConsumableItemRecipeSheet](#)

[ConsumableItemRecipeSheet.Row](#)

[ConsumableItemSheet](#)

[ConsumableItemSheet.Row](#)

[CostumeItemSheet](#)

[CostumeItemSheet.Row](#)

[CostumeStatSheet](#)

[CostumeStatSheet.Row](#)

[CreateAvatarFavSheet](#)

[CreateAvatarFavSheet.Row](#)

[CreateAvatarItemSheet](#)

[CreateAvatarItemSheet.Row](#)

[DeBuffLimitSheet](#)

[DeBuffLimitSheet.Row](#)

[EnemySkillSheet](#)

[EnemySkillSheet.Row](#)

[EnhancementCostSheet](#)

[EnhancementCostSheet.Row](#)

[EnhancementCostSheetV2](#)

[EnhancementCostSheetV2.Row](#)

[EnhancementCostSheetV3](#)

[EnhancementCostSheetV3.Row](#)

[EquipmentItemOptionSheet](#)

[EquipmentItemOptionSheet.Row](#)

[EquipmentItemRecipeSheet](#)

[EquipmentItemRecipeSheet.Row](#)

[EquipmentItemSetEffectSheet](#)

[EquipmentItemSetEffectSheet.Row](#)

[EquipmentItemSheet](#)

[EquipmentItemSheet.Row](#)

[EquipmentItemSubRecipeSheet](#)

[EquipmentItemSubRecipeSheet.Row](#)

[EquipmentItemSubRecipeSheetV2](#)

[EquipmentItemSubRecipeSheetV2.Row](#)

[GameConfigSheet](#)

[GameConfigSheet.Row](#)

[GeneralQuestSheet](#)

[GeneralQuestSheet.Row](#)

[GoldQuestSheet](#)

[GoldQuestSheet.Row](#)

[ItemConfigForGradeSheet](#)

[ItemConfigForGradeSheet.Row](#)

[ItemEnhancementQuestSheet](#)

[ItemEnhancementQuestSheet.Row](#)

[ItemGradeQuestSheet](#)

[ItemGradeQuestSheet.Row](#)

[ItemRequirementSheet](#)

[ItemRequirementSheet.Row](#)

[ItemSheet](#)

[ItemSheet.Row](#)

[ItemSheet.RowBase](#)

[ItemTypeCollectQuestSheet](#)

[ItemTypeCollectQuestSheet.Row](#)

[MaterialItemSheet](#)

[MaterialItemSheet.Row](#)

[MimisbrunnrSheet](#)

[MimisbrunnrSheet.Row](#)

[MonsterCollectionRewardSheet](#)

[MonsterCollectionRewardSheet.RewardInfo](#)

[MonsterCollectionRewardSheet.Row](#)

[MonsterCollectionSheet](#)

[MonsterCollectionSheet.Row](#)

[MonsterQuestSheet](#)

[MonsterQuestSheet.Row](#)

[PetCostSheet](#)

[PetCostSheet.PetCostData](#)

[PetCostSheet.Row](#)

[PetSheet](#)

[PetSheet.Row](#)

[QuestItemRewardSheet](#)

[QuestItemRewardSheet.Row](#)

[QuestRewardSheet](#)

[QuestRewardSheet.Row](#)

[QuestSheet](#)

[QuestSheet.Row](#)

[RaidSimulatorSheets](#)

[RaidSimulatorSheetsV1](#)

[RankingSimulatorSheets](#)

[RankingSimulatorSheetsV1](#)

[RankingSimulatorSheetsV100291](#)

[RedeemCodeListSheet](#)

[RedeemCodeListSheet.Row](#)

[RedeemRewardSheet](#)

[RedeemRewardSheet.RewardInfo](#)

[RedeemRewardSheet.Row](#)

[RuneCostSheet](#)

[RuneCostSheet.Row](#)

[RuneCostSheet.RuneCostData](#)

[RuneOptionSheet](#)

[RuneOptionSheet.Row](#)

[RuneOptionSheet.Row.RuneOptionInfo](#)

[RuneSheet](#)

[RuneSheet.Row](#)

[RuneWeightSheet](#)

[RuneWeightSheet.Row](#)

[RuneWeightSheet.RuneInfo](#)

[SetEffectExtension](#)

[SheetRowColumnException](#)

[SheetRowNotFoundException](#)

[SheetRowValidateException](#)

[SheetRow<T>](#)

[Sheet< TKey, TValue >](#)

[SimulatorSheets](#)

[SimulatorSheetsV1](#)

[SimulatorSheetsV100291](#)

[SkillActionBuffSheet](#)

[SkillActionBuffSheet.Row](#)

[SkillBuffSheet](#)

[SkillBuffSheet.Row](#)

[SkillSheet](#)

[SkillSheet.Row](#)

[StageDialogSheet](#)

[StageDialogSheet.Row](#)

[StageSheet](#)

[StageSheet.RewardData](#)

[StageSheet.Row](#)

[StageSimulatorSheets](#)

[StageSimulatorSheetsV1](#)

[StageSimulatorSheetsV100291](#)

[StageWaveSheet](#)

[StageWaveSheet.MonsterData](#)

[StageWaveSheet.Row](#)

[StageWaveSheet.WaveData](#)

[StakeAchievementRewardSheet](#)

[StakeAchievementRewardSheet.RewardInfo](#)

[StakeAchievementRewardSheet.Row](#)

[StakeAchievementRewardSheet.Step](#)

[StakeActionPointCoefficientSheet](#)

[StakeActionPointCoefficientSheet.Row](#)

[StakeRegularFixedRewardSheet](#)

This sheet is used for setting the regular rewards for staking. The difference between this sheet and [StakeRegularRewardSheet](#) is that the [StakeRegularFixedRewardSheet.RewardInfo](#) of this sheet has a fixed count of reward.

[StakeRegularFixedRewardSheet.RewardInfo](#)

[StakeRegularFixedRewardSheet.Row](#)

[StakeRegularRewardSheet](#)

This sheet is used for setting the regular rewards for staking. The difference between this sheet and [StakeRegularFixedRewardSheet](#) is that the [StakeRegularRewardSheet.RewardInfo](#) of this sheet has a rate and a reward type. The count of reward is calculated by the rate and the amount of staked currency.

[StakeRegularRewardSheet.RewardInfo](#)

[StakeRegularRewardSheet.Row](#)

[StatBuffSheet](#)

[StatBuffSheet.Row](#)

[SweepRequiredCPSheet](#)

[SweepRequiredCPSheet.Row](#)

[TableExtensions](#)

[TradeQuestSheet](#)

[TradeQuestSheet.Row](#)

[UnlockCombinationSlotCostSheet](#)

[UnlockCombinationSlotCostSheet.Row](#)

[WeeklyArenaRewardSheet](#)

[WeeklyArenaRewardSheet.RewardData](#)

[WeeklyArenaRewardSheet.Row](#)

[WorldBossActionPatternSheet](#)

[WorldBossActionPatternSheet.ActionPatternData](#)

[WorldBossActionPatternSheet.Row](#)

[WorldBossBattleRewardSheet](#)

[WorldBossBattleRewardSheet.Row](#)

[WorldBossCharacterSheet](#)

[WorldBossCharacterSheet.Row](#)

[WorldBossCharacterSheet.WaveStatData](#)

[WorldBossGlobalHpSheet](#)

[WorldBossGlobalHpSheet.Row](#)

[WorldBossKillRewardSheet](#)

[WorldBossKillRewardSheet.Row](#)

[WorldBossListSheet](#)

[WorldBossListSheet.Row](#)

[WorldBossRankRewardSheet](#)

[WorldBossRankRewardSheet.Row](#)

[WorldBossRankingRewardSheet](#)

[WorldBossRankingRewardSheet.Row](#)

[WorldBossStatSheetExtension](#)

[WorldQuestSheet](#)

[WorldQuestSheet.Row](#)

[WorldSheet](#)

[WorldSheet.Row](#)

[WorldUnlockSheet](#)

[WorldUnlockSheet.Row](#)

## Structs

[EquipmentItemSubRecipeSheet.MaterialInfo](#)

[EquipmentItemSubRecipeSheet.OptionInfo](#)

[EquipmentItemSubRecipeSheetV2.OptionInfo](#)

[WorldBossRankingRewardSheet.Row.RuneInfo](#)

## Interfaces

[ISheet](#)

[IStakeRewardRow](#)

[IStakeRewardSheet](#)

[IWorldBossRewardRow](#)

[IWorldBossRewardSheet](#)

# Enums

[CreateAvatarFavSheet.Target](#)

[RewardType](#)

[StakeRegularRewardSheet.StakeRewardType](#)

The reward type of stake. Do not use the whole element of this enum at once in actions.  
(e.g., count, loop) Do versioning when you change or remove the elements of this enum.

# Class ActionBuffSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActionBuffSheet : Sheet<int, ActionBuffSheet.Row>, IDictionary<int, ActionBuffSheet.Row>, ICollection<KeyValuePair<int, ActionBuffSheet.Row>>, IEnumerable<KeyValuePair<int, ActionBuffSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ActionBuffSheet.Row>](#) ← ActionBuffSheet

## Implements

[IDictionary<int, ActionBuffSheet.Row>](#),  
[ICollection<KeyValuePair<int, ActionBuffSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ActionBuffSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, ActionBuffSheet.Row>.Name](#) , [Sheet<int, ActionBuffSheet.Row>.OrderedList](#) ,  
[Sheet<int, ActionBuffSheet.Row>.First](#) , [Sheet<int, ActionBuffSheet.Row>.Last](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Keys](#) , [Sheet<int, ActionBuffSheet.Row>.Values](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Count](#) , [Sheet<int, ActionBuffSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ActionBuffSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.TryGetValue\(int, out ActionBuffSheet.Row, bool\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.AddRow\(int, ActionBuffSheet.Row\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Add\(int, ActionBuffSheet.Row\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.TryGetValue\(int, out ActionBuffSheet.Row\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Add\(KeyValuePair<int, ActionBuffSheet.Row>\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Contains\(KeyValuePair<int, ActionBuffSheet.Row>\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.CopyTo\(KeyValuePair<int, ActionBuffSheet.Row>\[\], int\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Remove\(KeyValuePair<int, ActionBuffSheet.Row>\)](#) ,  
[Sheet<int, ActionBuffSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ActionBuffSheet()

```
public ActionBuffSheet()
```

# Class ActionBuffSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ActionBuffSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← ActionBuffSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ATKPowerRatio

```
public decimal ATKPowerRatio { get; }
```

Property Value

[decimal](#)

### ActionBuffType

```
public ActionBuffType ActionBuffType { get; }
```

Property Value

[ActionBuffType](#)

## Chance

```
public int Chance { get; }
```

Property Value

[int](#)

## Duration

```
public int Duration { get; }
```

Property Value

[int](#)

## ElementType

```
public ElementType ElementType { get; }
```

Property Value

[ElementType](#)

## GroupId

```
public int GroupId { get; }
```

Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int](#)

Key

```
public override int Key { get; }
```

Property Value

[int](#)

TargetType

```
public SkillTargetType TargetType { get; }
```

Property Value

[SkillTargetType](#)

Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class ArenaSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaSheet : Sheet<int, ArenaSheet.Row>, IDictionary<int,
ArenaSheet.Row>, ICollection<KeyValuePair<int, ArenaSheet.Row>>,
IEnumerable<KeyValuePair<int, ArenaSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ArenaSheet.Row>](#) ← ArenaSheet

## Implements

[IDictionary<int, ArenaSheet.Row>](#),  
[ICollection<KeyValuePair<int, ArenaSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ArenaSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, ArenaSheet.Row>.Name](#) , [Sheet<int, ArenaSheet.Row>.OrderedList](#) ,  
[Sheet<int, ArenaSheet.Row>.First](#) , [Sheet<int, ArenaSheet.Row>.Last](#) ,  
[Sheet<int, ArenaSheet.Row>.Keys](#) , [Sheet<int, ArenaSheet.Row>.Values](#) ,  
[Sheet<int, ArenaSheet.Row>.Count](#) , [Sheet<int, ArenaSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ArenaSheet.Row>.this\[int\]](#) , [Sheet<int, ArenaSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ArenaSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ArenaSheet.Row>.TryGetValue\(int, out ArenaSheet.Row, bool\)](#) ,  
[Sheet<int, ArenaSheet.Row>.AddRow\(int, ArenaSheet.Row\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Add\(int, ArenaSheet.Row\)](#) ,  
[Sheet<int, ArenaSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, ArenaSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ArenaSheet.Row>.TryGetValue\(int, out ArenaSheet.Row\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Add\(KeyValuePair<int, ArenaSheet.Row>\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Contains\(KeyValuePair<int, ArenaSheet.Row>\)](#) ,  
[Sheet<int, ArenaSheet.Row>.CopyTo\(KeyValuePair<int, ArenaSheet.Row>\[\], int\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Remove\(KeyValuePair<int, ArenaSheet.Row>\)](#) ,  
[Sheet<int, ArenaSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## ArenaSheet()

```
public ArenaSheet()
```

# Methods

## AddRow(int, Row)

```
protected override void AddRow(int key, ArenaSheet.Row value)
```

### Parameters

key [int](#)

value [ArenaSheet.Row](#)

## GetRoundByBlockIndex(long)

```
public ArenaSheet.RoundData GetRoundByBlockIndex(long blockIndex)
```

### Parameters

blockIndex [long](#)

### Returns

[ArenaSheet.RoundData](#)

## GetRowByBlockIndex(long)

```
public ArenaSheet.Row GetRowByBlockIndex(long blockIndex)
```

## Parameters

**blockIndex** [long](#) ↗

## Returns

[ArenaSheet.Row](#)

# Class ArenaSheet.RoundData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaSheet.RoundData
```

## Inheritance

[object](#) ← ArenaSheet.RoundData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

RoundData(int, int, ArenaType, long, long, int, long, long, long, int, int)

```
public RoundData(int championshipId, int round, ArenaType arenaType, long
startBlockIndex, long endBlockIndex, int requiredMedalCount, long entranceFee, long
ticketPrice, long additionalTicketPrice, int maxPurchaseCount, int
maxPurchaseCountWithInterval, int medalId = 0)
```

## Parameters

championshipId [int](#)

round [int](#)

arenaType [ArenaType](#)

startBlockIndex [long](#)

endBlockIndex [long](#)

requiredMedalCount [int](#)

entranceFee [long ↗](#)

ticketPrice [long ↗](#)

additionalTicketPrice [long ↗](#)

maxPurchaseCount [int ↗](#)

maxPurchaseCountWithInterval [int ↗](#)

medalId [int ↗](#)

## Properties

### AdditionalTicketPrice

```
public long AdditionalTicketPrice { get; }
```

Property Value

[long ↗](#)

### ArenaType

```
public ArenaType ArenaType { get; }
```

Property Value

[ArenaType](#)

### ChampionshipId

```
public int ChampionshipId { get; }
```

Property Value

[int ↗](#)

## EndBlockIndex

```
public long EndBlockIndex { get; }
```

Property Value

[long](#) ↗

## EntranceFee

```
public long EntranceFee { get; }
```

Property Value

[long](#) ↗

## MaxPurchaseCount

```
public int MaxPurchaseCount { get; }
```

Property Value

[int](#) ↗

## MaxPurchaseCountWithInterval

```
public int MaxPurchaseCountWithInterval { get; }
```

Property Value

[int](#) ↗

## MedalId

```
public int MedalId { get; }
```

Property Value

[int](#) ↗

## RequiredMedalCount

```
public int RequiredMedalCount { get; }
```

Property Value

[int](#) ↗

## Round

```
public int Round { get; }
```

Property Value

[int](#) ↗

## StartBlockIndex

```
public long StartBlockIndex { get; }
```

Property Value

[long](#) ↗

## TicketPrice

```
public long TicketPrice { get; }
```

Property Value

[long](#)

## Methods

### IsTheRoundOpened(long)

```
public bool IsTheRoundOpened(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[bool](#)

# Class ArenaSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ArenaSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← ArenaSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ChampionshipId

```
public int ChampionshipId { get; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

# Round

```
public List<ArenaSheet.RoundData> Round { get; }
```

Property Value

[List](#)<[ArenaSheet](#).[RoundData](#)>

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

**TryGetChampionshipRound(out RoundData)**

```
public bool TryGetChampionshipRound(out ArenaSheet.RoundData roundData)
```

Parameters

roundData [ArenaSheet](#).[RoundData](#)

Returns

[bool](#)

**TryGetRound(int, out RoundData)**

```
public bool TryGetRound(int round, out ArenaSheet.RoundData roundData)
```

## Parameters

round [int](#)

roundData [ArenaSheet.RoundData](#)

## Returns

[bool](#)

# Class ArenaSimulatorSheets

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class ArenaSimulatorSheets : SimulatorSheets
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← [SimulatorSheets](#) ← ArenaSimulatorSheets

## Inherited Members

[SimulatorSheets.RuneOptionSheet](#) , [SimulatorSheets.RuneListSheet](#) ,  
[SimulatorSheets.RuneLevelBonusSheet](#) , [SimulatorSheetsV1.MaterialItemSheet](#) ,  
[SimulatorSheetsV1.SkillSheet](#) , [SimulatorSheetsV1.SkillBuffSheet](#) ,  
[SimulatorSheetsV1.SkillActionBuffSheet](#) , [SimulatorSheetsV1.ActionBuffSheet](#) ,  
[SimulatorSheetsV1.StatBuffSheet](#) , [SimulatorSheetsV1.CharacterSheet](#) ,  
[SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

ArenaSimulatorSheets(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet, CostumeStatSheet,  
WeeklyArenaRewardSheet, RuneOptionSheet,  
RuneListSheet, RuneLevelBonusSheet)

```
public ArenaSimulatorSheets(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, CostumeStatSheet  
costumeStatSheet, WeeklyArenaRewardSheet weeklyArenaRewardSheet,
```

```
RuneOptionSheet runeOptionSheet, RuneListSheet runeListSheet,  
RuneLevelBonusSheet runeLevelBonusSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

costumeStatSheet [CostumeStatSheet](#)

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

runeOptionSheet [RuneOptionSheet](#)

runeListSheet [RuneListSheet](#)

runeLevelBonusSheet [RuneLevelBonusSheet](#)

## Properties

### CostumeStatSheet

```
public CostumeStatSheet CostumeStatSheet { get; }
```

## Property Value

[CostumeStatSheet](#)

# WeeklyArenaRewardSheet

```
public WeeklyArenaRewardSheet WeeklyArenaRewardSheet { get; }
```

Property Value

[WeeklyArenaRewardSheet](#)

# Class ArenaSimulatorSheetsV1

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class ArenaSimulatorSheetsV1 : SimulatorSheetsV1
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← ArenaSimulatorSheetsV1

## Inherited Members

[SimulatorSheetsV1.MaterialItemSheet](#) , [SimulatorSheetsV1.SkillSheet](#) ,  
[SimulatorSheetsV1.SkillBuffSheet](#) , [SimulatorSheetsV1.SkillActionBuffSheet](#) ,  
[SimulatorSheetsV1.ActionBuffSheet](#) , [SimulatorSheetsV1.StatBuffSheet](#) ,  
[SimulatorSheetsV1.CharacterSheet](#) , [SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

ArenaSimulatorSheetsV1(MaterialItemSheet, SkillSheet, SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet, ActionBuffSheet, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet, CostumeStatSheet, WeeklyArenaRewardSheet)

```
public ArenaSimulatorSheetsV1(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, CostumeStatSheet  
costumeStatSheet, WeeklyArenaRewardSheet weeklyArenaRewardSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

costumeStatSheet [CostumeStatSheet](#)

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

## Properties

### CostumeStatSheet

```
public CostumeStatSheet CostumeStatSheet { get; }
```

Property Value

[CostumeStatSheet](#)

### WeeklyArenaRewardSheet

```
public WeeklyArenaRewardSheet WeeklyArenaRewardSheet { get; }
```

Property Value

[WeeklyArenaRewardSheet](#)

# Class ArenaSimulatorSheetsV100291

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class ArenaSimulatorSheetsV100291 : SimulatorSheetsV100291
```

## Inheritance

[object](#) ← [SimulatorSheetsV100291](#) ← ArenaSimulatorSheetsV100291

## Inherited Members

[SimulatorSheetsV100291.MaterialItemSheet](#) , [SimulatorSheetsV100291.SkillSheet](#) ,  
[SimulatorSheetsV100291.SkillBuffSheet](#) , [SimulatorSheetsV100291.BuffSheet](#) ,  
[SimulatorSheetsV100291.CharacterSheet](#) , [SimulatorSheetsV100291.CharacterLevelSheet](#) ,  
[SimulatorSheetsV100291.EquipmentItemSetEffectSheet](#) ,  
[SimulatorSheetsV100291.ToSimulatorSheetsV1\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

ArenaSimulatorSheetsV100291(MaterialItemSheet, SkillSheet, SkillBuffSheet, BuffSheet, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet, CostumeStatSheet, WeeklyArenaRewardSheet)

```
public ArenaSimulatorSheetsV100291(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, BuffSheet buffSheet, CharacterSheet  
characterSheet, CharacterLevelSheet characterLevelSheet, EquipmentItemSetEffectSheet  
equipmentItemSetEffectSheet, CostumeStatSheet costumeStatSheet,  
WeeklyArenaRewardSheet weeklyArenaRewardSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

buffSheet [BuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

costumeStatSheet [CostumeStatSheet](#)

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

## Properties

### CostumeStatSheet

```
public CostumeStatSheet CostumeStatSheet { get; }
```

Property Value

[CostumeStatSheet](#)

### WeeklyArenaRewardSheet

```
public WeeklyArenaRewardSheet WeeklyArenaRewardSheet { get; }
```

Property Value

[WeeklyArenaRewardSheet](#)

## Methods

### ToArenaSimulatorSheetsV1()

```
public ArenaSimulatorSheetsV1 ToArenaSimulatorSheetsV1()
```

Returns

[ArenaSimulatorSheetsV1](#)

# Class AvatarSheets

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class AvatarSheets
```

## Inheritance

[object](#) ← AvatarSheets

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

AvatarSheets(WorldSheet, QuestSheet,  
QuestRewardSheet, QuestItemRewardSheet,  
EquipmentItemRecipeSheet,  
EquipmentItemSubRecipeSheet)

```
public AvatarSheets(WorldSheet worldSheet, QuestSheet questSheet,  
QuestRewardSheet questRewardSheet, QuestItemRewardSheet questItemRewardSheet,  
EquipmentItemRecipeSheet equipmentItemRecipeSheet, EquipmentItemSubRecipeSheet  
equipmentItemSubRecipeSheet)
```

## Parameters

worldSheet [WorldSheet](#)

questSheet [QuestSheet](#)

questRewardSheet [QuestRewardSheet](#)

questItemRewardSheet [QuestItemRewardSheet](#)

equipmentItemRecipeSheet [EquipmentItemRecipeSheet](#)

equipmentItemSubRecipeSheet [EquipmentItemSubRecipeSheet](#)

## Fields

### EquipmentItemRecipeSheet

```
public readonly EquipmentItemRecipeSheet EquipmentItemRecipeSheet
```

Field Value

[EquipmentItemRecipeSheet](#)

### EquipmentItemSubRecipeSheet

```
public readonly EquipmentItemSubRecipeSheet EquipmentItemSubRecipeSheet
```

Field Value

[EquipmentItemSubRecipeSheet](#)

### QuestItemRewardSheet

```
public readonly QuestItemRewardSheet QuestItemRewardSheet
```

Field Value

[QuestItemRewardSheet](#)

### QuestRewardSheet

```
public readonly QuestRewardSheet QuestRewardSheet
```

Field Value

## [QuestRewardSheet](#)

### QuestSheet

```
public readonly QuestSheet QuestSheet
```

Field Value

[QuestSheet](#)

### WorldSheet

```
public readonly WorldSheet WorldSheet
```

Field Value

[WorldSheet](#)

# Class BuffLinkSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class BuffLinkSheet : Sheet<int, BuffLinkSheet.Row>, IDictionary<int, BuffLinkSheet.Row>, ICollection<KeyValuePair<int, BuffLinkSheet.Row>>, IEnumerable<KeyValuePair<int, BuffLinkSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, BuffLinkSheet.Row>](#) ← [BuffLinkSheet](#)

## Implements

[IDictionary<int, BuffLinkSheet.Row>](#),  
[ICollection<KeyValuePair<int, BuffLinkSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, BuffLinkSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, BuffLinkSheet.Row>.Name](#) , [Sheet<int, BuffLinkSheet.Row>.OrderedList](#) ,  
[Sheet<int, BuffLinkSheet.Row>.First](#) , [Sheet<int, BuffLinkSheet.Row>.Last](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Keys](#) , [Sheet<int, BuffLinkSheet.Row>.Values](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Count](#) , [Sheet<int, BuffLinkSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, BuffLinkSheet.Row>.this\[int\]](#) , [Sheet<int, BuffLinkSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.TryGetValue\(int, out BuffLinkSheet.Row, bool\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.AddRow\(int, BuffLinkSheet.Row\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Add\(int, BuffLinkSheet.Row\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.TryGetValue\(int, out BuffLinkSheet.Row\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Add\(KeyValuePair<int, BuffLinkSheet.Row>\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Contains\(KeyValuePair<int, BuffLinkSheet.Row>\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.CopyTo\(KeyValuePair<int, BuffLinkSheet.Row>\[\], int\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Remove\(KeyValuePair<int, BuffLinkSheet.Row>\)](#) ,  
[Sheet<int, BuffLinkSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## BuffLinkSheet()

```
public BuffLinkSheet()
```

# Class BuffLinkSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuffLinkSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← BuffLinkSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BuffId

```
public int BuffId { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

# LinkedBuffId

```
public int LinkedBuffId { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class BuffSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuffSheet : Sheet<int, BuffSheet.Row>, IDictionary<int, BuffSheet.Row>,
ICollection<KeyValuePair<int, BuffSheet.Row>>, IEnumerable<KeyValuePair<int,
BuffSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, BuffSheet.Row>](#) ← [BuffSheet](#)

## Implements

[IDictionary<int, BuffSheet.Row>](#), [ICollection<KeyValuePair<int, BuffSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, BuffSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, BuffSheet.Row>.Name](#) , [Sheet<int, BuffSheet.Row>.OrderedList](#) ,  
[Sheet<int, BuffSheet.Row>.First](#) , [Sheet<int, BuffSheet.Row>.Last](#) ,  
[Sheet<int, BuffSheet.Row>.Keys](#) , [Sheet<int, BuffSheet.Row>.Values](#) ,  
[Sheet<int, BuffSheet.Row>.Count](#) , [Sheet<int, BuffSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, BuffSheet.Row>.this\[int\]](#) , [Sheet<int, BuffSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, BuffSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, BuffSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, BuffSheet.Row>.TryGetValue\(int, out BuffSheet.Row, bool\)](#) ,  
[Sheet<int, BuffSheet.Row>.AddRow\(int, BuffSheet.Row\)](#) ,  
[Sheet<int, BuffSheet.Row>.Add\(int, BuffSheet.Row\)](#) ,  
[Sheet<int, BuffSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, BuffSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, BuffSheet.Row>.TryGetValue\(int, out BuffSheet.Row\)](#) ,  
[Sheet<int, BuffSheet.Row>.Add\(KeyValuePair<int, BuffSheet.Row>\)](#) ,  
[Sheet<int, BuffSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, BuffSheet.Row>.Contains\(KeyValuePair<int, BuffSheet.Row>\)](#) ,  
[Sheet<int, BuffSheet.Row>.CopyTo\(KeyValuePair<int, BuffSheet.Row>\[\], int\)](#) ,  
[Sheet<int, BuffSheet.Row>.Remove\(KeyValuePair<int, BuffSheet.Row>\)](#) ,  
[Sheet<int, BuffSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## BuffSheet()

```
public BuffSheet()
```

# Class BuffSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class BuffSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← BuffSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Chance

100: 100%

```
public int Chance { get; }
```

### Property Value

[int](#)

### Duration

Turn count.

```
public int Duration { get; }
```

### Property Value

[int](#)

## GroupId

GroupId යනු GroupId සංඛ්‍යා නිස් ම ම ම ම ම ම ම ම ම ම ම ම ම ම . Id යනුවෙනු GroupId යනු, ම ම ම . ම ම ම ම Id ය ම ම ම ම ම ම ම .

```
public int GroupId { get; }
```

## Property Value

[int](#)

## IconResource

```
public string IconResource { get; }
```

## Property Value

[string](#)

## Id

```
public int Id { get; }
```

## Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## StatModifier

```
public StatModifier StatModifier { get; }
```

Property Value

[StatModifier](#)

## TargetType

```
public SkillTargetType TargetType { get; }
```

Property Value

[SkillTargetType](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CharacterLevelSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CharacterLevelSheet : Sheet<int, CharacterLevelSheet.Row>,
IDictionary<int, CharacterLevelSheet.Row>, ICollection<KeyValuePair<int,
CharacterLevelSheet.Row>>, IEnumerable<KeyValuePair<int, CharacterLevelSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CharacterLevelSheet.Row>](#) ← CharacterLevelSheet

## Implements

[IDictionary<int, CharacterLevelSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, CharacterLevelSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, CharacterLevelSheet.Row>>](#) , [IEnumerable](#) , [ISheet](#)

## Inherited Members

[Sheet<int, CharacterLevelSheet.Row>.Name](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.OrderedList](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.First](#) , [Sheet<int, CharacterLevelSheet.Row>.Last](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Keys](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Values](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Count](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.TryGetValue\(int, out CharacterLevelSheet.Row, bool\)](#) ,  
  
[Sheet<int, CharacterLevelSheet.Row>.AddRow\(int, CharacterLevelSheet.Row\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Add\(int, CharacterLevelSheet.Row\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.TryGetValue\(int, out CharacterLevelSheet.Row\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Add\(KeyValuePair<int, CharacterLevelSheet.Row>\)](#) ,  
[Sheet<int, CharacterLevelSheet.Row>.Clear\(\)](#) ,

```
Sheet<int, CharacterLevelSheet.Row>.Contains(KeyValuePair<int,
CharacterLevelSheet.Row>) ,
Sheet<int, CharacterLevelSheet.Row>.CopyTo(KeyValuePair<int,
CharacterLevelSheet.Row>[], int) ,
Sheet<int, CharacterLevelSheet.Row>.Remove(KeyValuePair<int,
CharacterLevelSheet.Row>) ,
Sheet<int, CharacterLevelSheet.Row>.Serialize() , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Constructors

### CharacterLevelSheet()

```
public CharacterLevelSheet()
```

## Methods

### GetLevel(long)

```
public int GetLevel(long exp)
```

#### Parameters

exp long

#### Returns

int

### TryGetLevel(long, out int)

```
public bool TryGetLevel(long exp, out int level)
```

#### Parameters

`exp` [long](#)

`level` [int](#)

Returns

[bool](#)

# Class CharacterLevelSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CharacterLevelSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CharacterLevelSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Exp

```
public long Exp { get; }
```

Property Value

[long](#)

### ExpNeed

```
public long ExpNeed { get; }
```

Property Value

[long](#)

## Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## Level

```
public int Level { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CharacterSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CharacterSheet : Sheet<int, CharacterSheet.Row>, IDictionary<int,
CharacterSheet.Row>, ICollection<KeyValuePair<int, CharacterSheet.Row>>,
IEnumerable<KeyValuePair<int, CharacterSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CharacterSheet.Row>](#) ← CharacterSheet

## Implements

[IDictionary<int, CharacterSheet.Row>](#),  
[ICollection<KeyValuePair<int, CharacterSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CharacterSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CharacterSheet.Row>.Name](#) , [Sheet<int, CharacterSheet.Row>.OrderedList](#) ,  
[Sheet<int, CharacterSheet.Row>.First](#) , [Sheet<int, CharacterSheet.Row>.Last](#) ,  
[Sheet<int, CharacterSheet.Row>.Keys](#) , [Sheet<int, CharacterSheet.Row>.Values](#) ,  
[Sheet<int, CharacterSheet.Row>.Count](#) , [Sheet<int, CharacterSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CharacterSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CharacterSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CharacterSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CharacterSheet.Row>.TryGetValue\(int, out CharacterSheet.Row, bool\)](#) ,  
[Sheet<int, CharacterSheet.Row>.AddRow\(int, CharacterSheet.Row\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Add\(int, CharacterSheet.Row\)](#) ,  
[Sheet<int, CharacterSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CharacterSheet.Row>.TryGetValue\(int, out CharacterSheet.Row\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Add\(KeyValuePair<int, CharacterSheet.Row>\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Contains\(KeyValuePair<int, CharacterSheet.Row>\)](#) ,  
[Sheet<int, CharacterSheet.Row>.CopyTo\(KeyValuePair<int, CharacterSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Remove\(KeyValuePair<int, CharacterSheet.Row>\)](#) ,  
[Sheet<int, CharacterSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CharacterSheet()

```
public CharacterSheet()
```

# Class CharacterSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CharacterSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CharacterSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[CharacterSheetExtension.ToStats\(CharacterSheet.Row, int\)](#)

## Properties

### ATK

```
public decimal ATK { get; }
```

Property Value

[decimal](#)

### AttackRange

```
public float AttackRange { get; }
```

Property Value

[float](#)

## CRI

```
public decimal CRI { get; }
```

Property Value

[decimal](#)

## DEF

```
public decimal DEF { get; }
```

Property Value

[decimal](#)

## ElementType

```
public ElementType ElementType { get; }
```

Property Value

[ElementType](#)

## HIT

```
public decimal HIT { get; }
```

Property Value

[decimal](#)

## HP

```
public decimal HP { get; }
```

Property Value

[decimal](#) ↗

## Id

```
public int Id { get; }
```

Property Value

[int](#) ↗

## Key

```
public override int Key { get; }
```

Property Value

[int](#) ↗

## LvATK

```
public decimal LvATK { get; }
```

Property Value

[decimal](#) ↗

## LvCRI

```
public decimal LvCRI { get; }
```

Property Value

[decimal](#) ↗

## LvDEF

```
public decimal LvDEF { get; }
```

Property Value

[decimal](#) ↗

## LvHIT

```
public decimal LvHIT { get; }
```

Property Value

[decimal](#) ↗

## LvHP

```
public decimal LvHP { get; }
```

Property Value

[decimal](#) ↗

## LvSPD

```
public decimal LvSPD { get; }
```

Property Value

[decimal](#) ↗

## RunSpeed

```
public float RunSpeed { get; }
```

Property Value

[float](#) ↗

## SPD

```
public decimal SPD { get; }
```

Property Value

[decimal](#) ↗

## SizeType

```
public SizeType SizeType { get; }
```

Property Value

[SizeType](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CharacterSheetExtension

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public static class CharacterSheetExtension
```

## Inheritance

[object](#) ← CharacterSheetExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ToStats(Row, int)

```
public static StatsMap ToStats(this CharacterSheet.Row row, int level)
```

#### Parameters

row [CharacterSheet.Row](#)

level [int](#)

#### Returns

[StatsMap](#)

# Class CollectQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CollectQuestSheet : Sheet<int, CollectQuestSheet.Row>, IDictionary<int, CollectQuestSheet.Row>, ICollection<KeyValuePair<int, CollectQuestSheet.Row>>, IEnumerable<KeyValuePair<int, CollectQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CollectQuestSheet.Row>](#) ← CollectQuestSheet

## Implements

[IDictionary<int, CollectQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, CollectQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CollectQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CollectQuestSheet.Row>.Name](#) ,  
[Sheet<int, CollectQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, CollectQuestSheet.Row>.First](#) , [Sheet<int, CollectQuestSheet.Row>.Last](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Keys](#) , [Sheet<int, CollectQuestSheet.Row>.Values](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Count](#) ,  
[Sheet<int, CollectQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CollectQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.TryGetValue\(int, out CollectQuestSheet.Row, bool\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.AddRow\(int, CollectQuestSheet.Row\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Add\(int, CollectQuestSheet.Row\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.TryGetValue\(int, out CollectQuestSheet.Row\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Add\(KeyValuePair<int, CollectQuestSheet.Row>\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CollectQuestSheet.Row>.Contains\(KeyValuePair<int, CollectQuestSheet.Row>\)](#) ,

```
Sheet<int, CollectQuestSheet.Row>.CopyTo(KeyValuePair<int, CollectQuestSheet.Row>[],  
int) ,  
Sheet<int, CollectQuestSheet.Row>.Remove(KeyValuePair<int, CollectQuestSheet.Row>) ,  
Sheet<int, CollectQuestSheet.Row>.Serialize() , object.Equals(object) ,  
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,  
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Constructors

### CollectQuestSheet()

```
public CollectQuestSheet()
```

# Class CollectQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CollectQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← CollectQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ItemId

```
public int ItemId { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CollectionSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class CollectionSheet : Sheet<int, CollectionSheet.Row>, IDictionary<int, CollectionSheet.Row>, ICollection<KeyValuePair<int, CollectionSheet.Row>>, IEnumerable<KeyValuePair<int, CollectionSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CollectionSheet.Row>](#) ← CollectionSheet

## Implements

[IDictionary<int, CollectionSheet.Row>](#),  
[ICollection<KeyValuePair<int, CollectionSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CollectionSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CollectionSheet.Row>.Name](#) , [Sheet<int, CollectionSheet.Row>.OrderedList](#) ,  
[Sheet<int, CollectionSheet.Row>.First](#) , [Sheet<int, CollectionSheet.Row>.Last](#) ,  
[Sheet<int, CollectionSheet.Row>.Keys](#) , [Sheet<int, CollectionSheet.Row>.Values](#) ,  
[Sheet<int, CollectionSheet.Row>.Count](#) , [Sheet<int, CollectionSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CollectionSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CollectionSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CollectionSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CollectionSheet.Row>.TryGetValue\(int, out CollectionSheet.Row, bool\)](#) ,  
[Sheet<int, CollectionSheet.Row>.AddRow\(int, CollectionSheet.Row\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Add\(int, CollectionSheet.Row\)](#) ,  
[Sheet<int, CollectionSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CollectionSheet.Row>.TryGetValue\(int, out CollectionSheet.Row\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Add\(KeyValuePair<int, CollectionSheet.Row>\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Contains\(KeyValuePair<int, CollectionSheet.Row>\)](#) ,  
[Sheet<int, CollectionSheet.Row>.CopyTo\(KeyValuePair<int, CollectionSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Remove\(KeyValuePair<int, CollectionSheet.Row>\)](#) ,  
[Sheet<int, CollectionSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CollectionSheet()

```
public CollectionSheet()
```

# Class CollectionSheet.RequiredMaterial

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class CollectionSheet.RequiredMaterial
```

## Inheritance

[object](#) ← CollectionSheet.RequiredMaterial

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### Count

```
public int Count
```

#### Field Value

[int](#)

### ItemId

```
public int ItemId
```

#### Field Value

[int](#)

### Level

```
public int Level
```

Field Value

[int](#)

## SkillContains

```
public bool SkillContains
```

Field Value

[bool](#)

## Methods

### GetMaterial(IEnumerable<ICollectionMaterial>)

Retrieves the [ICollectionMaterial](#) object from the given collection of materials based on the item ID and count.

```
public ICollectionMaterial GetMaterial(IEnumerable<ICollectionMaterial> materials)
```

Parameters

**materials** [IEnumerable](#)<[ICollectionMaterial](#)>

The collection of materials to search.

Returns

[ICollectionMaterial](#)

The [ICollectionMaterial](#) object if found; otherwise, an exception is thrown.

### Validate(INonFungibleItem)

```
public bool Validate(INonFungibleItem nonFungibleItem)
```

Parameters

nonFungibleItem [INonFungibleItem](#)

Returns

[bool](#)

# Class CollectionSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class CollectionSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CollectionSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Materials

```
public List<CollectionSheet.RequiredMaterial> Materials
```

#### Field Value

[List<CollectionSheet.RequiredMaterial>](#)

### StatModifiers

```
public List<StatModifier> StatModifiers
```

#### Field Value

[List<StatModifier>](#)

# Properties

## Id

```
public int Id { get; }
```

## Property Value

[int](#)

## Key

```
public override int Key { get; }
```

## Property Value

[int](#)

# Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CombinationEquipmentQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationEquipmentQuestSheet : Sheet<int,
CombinationEquipmentQuestSheet.Row>, IDictionary<int,
CombinationEquipmentQuestSheet.Row>, ICollection<KeyValuePair<int,
CombinationEquipmentQuestSheet.Row>>, IEnumerable<KeyValuePair<int,
CombinationEquipmentQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int](#), [CombinationEquipmentQuestSheet.Row](#) <–  
CombinationEquipmentQuestSheet

## Implements

[IDictionary](#)<[int](#), [CombinationEquipmentQuestSheet.Row](#)>,  
[ICollection](#)<[KeyValuePair](#)<[int](#), [CombinationEquipmentQuestSheet.Row](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [CombinationEquipmentQuestSheet.Row](#)>>,  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CombinationEquipmentQuestSheet.Row>.Name](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.First](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Last](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Keys](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Values](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Count](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.TryGetValue\(int, out  
CombinationEquipmentQuestSheet.Row, bool\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.AddRow\(int,  
CombinationEquipmentQuestSheet.Row\)](#) ,

[Sheet<int, CombinationEquipmentQuestSheet.Row>.Add\(int, CombinationEquipmentQuestSheet.Row\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.TryGetValue\(int, out CombinationEquipmentQuestSheet.Row\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Add\(KeyValuePair<int, CombinationEquipmentQuestSheet.Row>\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Contains\(KeyValuePair<int, CombinationEquipmentQuestSheet.Row>\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.CopyTo\(KeyValuePair<int, CombinationEquipmentQuestSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Remove\(KeyValuePair<int, CombinationEquipmentQuestSheet.Row>\)](#) ,  
[Sheet<int, CombinationEquipmentQuestSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CombinationEquipmentQuestSheet()

```
public CombinationEquipmentQuestSheet()
```

# Class CombinationEquipmentQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CombinationEquipmentQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← CombinationEquipmentQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Geoal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Recipeld

```
public int RecipeId { get; }
```

### Property Value

[int](#)

## Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList<string>](#)

# Class CombinationQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationQuestSheet : Sheet<int, CombinationQuestSheet.Row>,
IDictionary<int, CombinationQuestSheet.Row>, ICollection<KeyValuePair<int,
CombinationQuestSheet.Row>>, IEnumerable<KeyValuePair<int,
CombinationQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CombinationQuestSheet.Row>](#) ← CombinationQuestSheet

## Implements

[IDictionary<int, CombinationQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, CombinationQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CombinationQuestSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CombinationQuestSheet.Row>.Name](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.First](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Last](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Keys](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Values](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Count](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.TryGetValue\(int, out CombinationQuestSheet.Row, bool\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.AddRow\(int, CombinationQuestSheet.Row\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Add\(int, CombinationQuestSheet.Row\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CombinationQuestSheet.Row>.Remove\(int\)](#) ,

```
Sheet<int, CombinationQuestSheet.Row>.TryGetValue(int, out
CombinationQuestSheet.Row),
Sheet<int, CombinationQuestSheet.Row>.Add(KeyValuePair<int,
CombinationQuestSheet.Row>),
Sheet<int, CombinationQuestSheet.Row>.Clear(),
Sheet<int, CombinationQuestSheet.Row>.Contains(KeyValuePair<int,
CombinationQuestSheet.Row>),
Sheet<int, CombinationQuestSheet.Row>.CopyTo(KeyValuePair<int,
CombinationQuestSheet.Row>[], int),
Sheet<int, CombinationQuestSheet.Row>.Remove(KeyValuePair<int,
CombinationQuestSheet.Row>),
Sheet<int, CombinationQuestSheet.Row>.Serialize(), object.Equals(object)↗ ,
object.Equals(object, object)↗ , object.GetHashCode()↗ , object.GetType()↗ ,
object.MemberwiseClone()↗ , object.ReferenceEquals(object, object)↗ , object.ToString()↗
```

## Constructors

### CombinationQuestSheet()

```
public CombinationQuestSheet()
```

# Class CombinationQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CombinationQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← CombinationQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ItemSubType

```
public ItemSubType ItemSubType { get; }
```

### Property Value

[ItemSubType](#)

### ItemType

```
public ItemType ItemType { get; }
```

Property Value

[ItemType](#)

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList<string>](#)

# Class ConsumableItemRecipeSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ConsumableItemRecipeSheet : Sheet<int, ConsumableItemRecipeSheet.Row>,
IDictionary<int, ConsumableItemRecipeSheet.Row>, ICollection<KeyValuePair<int,
ConsumableItemRecipeSheet.Row>>, IEnumerable<KeyValuePair<int,
ConsumableItemRecipeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ConsumableItemRecipeSheet.Row>](#) ← [ConsumableItemRecipeSheet](#)

## Implements

[IDictionary<int, ConsumableItemRecipeSheet.Row>](#),  
[ICollection<KeyValuePair<int, ConsumableItemRecipeSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ConsumableItemRecipeSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, ConsumableItemRecipeSheet.Row>.Name](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.OrderedList](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.First](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Last](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Keys](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Values](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Count](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.TryGetValue\(int, out ConsumableItemRecipeSheet.Row, bool\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.AddRow\(int, ConsumableItemRecipeSheet.Row\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Add\(int, ConsumableItemRecipeSheet.Row\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.ContainsKey\(int\)](#) ,

[Sheet<int, ConsumableItemRecipeSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.TryGetValue\(int, out ConsumableItemRecipeSheet.Row\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Add\(KeyValuePair<int, ConsumableItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Contains\(KeyValuePair<int, ConsumableItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.CopyTo\(KeyValuePair<int, ConsumableItemRecipeSheet.Row>\[\], int\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Remove\(KeyValuePair<int, ConsumableItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, ConsumableItemRecipeSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ConsumableItemRecipeSheet()

```
public ConsumableItemRecipeSheet()
```

# Class ConsumableItemRecipeSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ConsumableItemRecipeSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← ConsumableItemRecipeSheet.Row

## Derived

[EventConsumableItemRecipeSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Materials

```
public List<EquipmentItemSubRecipeSheet.MaterialInfo> Materials { get; }
```

Property Value

[List](#)<[EquipmentItemSubRecipeSheet](#).[MaterialInfo](#)>

## RequiredActionPoint

```
public int RequiredActionPoint { get; }
```

Property Value

[int](#)

## RequiredBlockIndex

```
public long RequiredBlockIndex { get; }
```

Property Value

[long](#)

## RequiredGold

```
public decimal RequiredGold { get; }
```

Property Value

[decimal](#)

# ResultConsumableItemId

```
public int ResultConsumableItemId { get; }
```

Property Value

[int](#)

## Methods

### GetAllMaterials()

```
public IEnumerable<EquipmentItemSubRecipeSheet.MaterialInfo> GetAllMaterials()
```

Returns

[IEnumerable](#)<[EquipmentItemSubRecipeSheet.MaterialInfo](#)>

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class ConsumableItemSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ConsumableItemSheet : Sheet<int, ConsumableItemSheet.Row>,
IDictionary<int, ConsumableItemSheet.Row>, ICollection<KeyValuePair<int,
ConsumableItemSheet.Row>>, IEnumerable<KeyValuePair<int, ConsumableItemSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ConsumableItemSheet.Row>](#) ← [ConsumableItemSheet](#)

## Implements

[IDictionary<int, ConsumableItemSheet.Row>](#),  
[ICollection<KeyValuePair<int, ConsumableItemSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ConsumableItemSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, ConsumableItemSheet.Row>.Name](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.OrderedList](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.First](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Last](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Keys](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Values](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Count](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.TryGetValue\(int, out ConsumableItemSheet.Row, bool\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.AddRow\(int, ConsumableItemSheet.Row\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Add\(int, ConsumableItemSheet.Row\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.TryGetValue\(int, out ConsumableItemSheet.Row\)](#) ,

[Sheet<int, ConsumableItemSheet.Row>.Add\(KeyValuePair<int, ConsumableItemSheet.Row>\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Contains\(KeyValuePair<int, ConsumableItemSheet.Row>\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.CopyTo\(KeyValuePair<int, ConsumableItemSheet.Row>\[\], int\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Remove\(KeyValuePair<int, ConsumableItemSheet.Row>\)](#) ,  
[Sheet<int, ConsumableItemSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ConsumableItemSheet()

```
public ConsumableItemSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, ConsumableItemSheet.Row value)
```

## Parameters

key [int](#)

value [ConsumableItemSheet.Row](#)

# Class ConsumableItemSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ConsumableItemSheet.Row : ItemSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [ItemSheet.RowBase](#) ← [ItemSheet.Row](#) ←

ConsumableItemSheet.Row

## Implements

[IState](#)

## Inherited Members

[ItemSheet.Row.Key](#) , [ItemSheet.Row.Id](#) , [ItemSheet.Row.ItemSubType](#) ,  
[ItemSheet.Row.Grade](#) , [ItemSheet.Row.ElementalType](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Row()

```
public Row()
```

### Row(Dictionary)

```
public Row(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

# Properties

## ItemType

```
public override ItemType ItemType { get; }
```

### Property Value

[ItemType](#)

## Stats

```
public List<DecimalStat> Stats { get; }
```

### Property Value

[List](#)<[DecimalStat](#)>

## Methods

### Deserialize(Dictionary)

```
public static ConsumableItemSheet.Row Deserialize(Dictionary serialized)
```

### Parameters

**serialized** Dictionary

### Returns

[ConsumableItemSheet.Row](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList<string>](#)

# Class CostumeItemSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CostumeItemSheet : Sheet<int, CostumeItemSheet.Row>, IDictionary<int,
CostumeItemSheet.Row>, ICollection<KeyValuePair<int, CostumeItemSheet.Row>>,
IEnumerable<KeyValuePair<int, CostumeItemSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CostumeItemSheet.Row>](#) ← CostumeItemSheet

## Implements

[IDictionary<int, CostumeItemSheet.Row>](#),  
[ICollection<KeyValuePair<int, CostumeItemSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CostumeItemSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CostumeItemSheet.Row>.Name](#) ,  
[Sheet<int, CostumeItemSheet.Row>.OrderedList](#) ,  
[Sheet<int, CostumeItemSheet.Row>.First](#) , [Sheet<int, CostumeItemSheet.Row>.Last](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Keys](#) , [Sheet<int, CostumeItemSheet.Row>.Values](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Count](#) ,  
[Sheet<int, CostumeItemSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CostumeItemSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.TryGetValue\(int, out CostumeItemSheet.Row, bool\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.AddRow\(int, CostumeItemSheet.Row\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Add\(int, CostumeItemSheet.Row\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.TryGetValue\(int, out CostumeItemSheet.Row\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Add\(KeyValuePair<int, CostumeItemSheet.Row>\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CostumeItemSheet.Row>.Contains\(KeyValuePair<int, CostumeItemSheet.Row>\)](#) ,

```
Sheet<int, CostumeItemSheet.Row>.CopyTo(KeyValuePair<int, CostumeItemSheet.Row>[],  
int) ,  
Sheet<int, CostumeItemSheet.Row>.Remove(KeyValuePair<int, CostumeItemSheet.Row>)  
,
```

`Sheet<int, CostumeItemSheet.Row>.Serialize()` , `object.Equals(object)` ,  
`object.Equals(object, object)` , `object.GetHashCode()` , `object.GetType()` ,  
`object.MemberwiseClone()` , `object.ReferenceEquals(object, object)` , `object.ToString()`

## Constructors

### CostumeItemSheet()

```
public CostumeItemSheet()
```

# Class CostumeItemSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CostumeItemSheet.Row : ItemSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [ItemSheet.RowBase](#) ← [ItemSheet.Row](#) ←

CostumeItemSheet.Row

## Inherited Members

[ItemSheet.Row.Key](#) , [ItemSheet.Row.Id](#) , [ItemSheet.Row.ItemSubType](#) ,  
[ItemSheet.Row.Grade](#) , [ItemSheet.Row.ElementalType](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Row()

```
public Row()
```

### Row(Dictionary)

```
public Row(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Properties

## ItemType

```
public override ItemType ItemType { get; }
```

Property Value

[ItemType](#)

## SpineResourcePath

```
public string SpineResourcePath { get; }
```

Property Value

[string](#)

## Methods

### Deserialize(Dictionary)

```
public static CostumeItemSheet.Row Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[CostumeItemSheet.Row](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList<string>](#)

# Class CostumeStatSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CostumeStatSheet : Sheet<int, CostumeStatSheet.Row>, IDictionary<int,
CostumeStatSheet.Row>, ICollection<KeyValuePair<int, CostumeStatSheet.Row>>,
IEnumerable<KeyValuePair<int, CostumeStatSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CostumeStatSheet.Row>](#) ← CostumeStatSheet

## Implements

[IDictionary<int, CostumeStatSheet.Row>](#),  
[ICollection<KeyValuePair<int, CostumeStatSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CostumeStatSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CostumeStatSheet.Row>.Name](#) ,  
[Sheet<int, CostumeStatSheet.Row>.OrderedList](#) ,  
[Sheet<int, CostumeStatSheet.Row>.First](#) , [Sheet<int, CostumeStatSheet.Row>.Last](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Keys](#) , [Sheet<int, CostumeStatSheet.Row>.Values](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Count](#) ,  
[Sheet<int, CostumeStatSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CostumeStatSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.TryGetValue\(int, out CostumeStatSheet.Row, bool\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.AddRow\(int, CostumeStatSheet.Row\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Add\(int, CostumeStatSheet.Row\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.TryGetValue\(int, out CostumeStatSheet.Row\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Add\(KeyValuePair<int, CostumeStatSheet.Row>\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CostumeStatSheet.Row>.Contains\(KeyValuePair<int, CostumeStatSheet.Row>\)](#) ,

```
Sheet<int, CostumeStatSheet.Row>.CopyTo(KeyValuePair<int, CostumeStatSheet.Row>[],  
int) ,  
Sheet<int, CostumeStatSheet.Row>.Remove(KeyValuePair<int, CostumeStatSheet.Row>) ,  
Sheet<int, CostumeStatSheet.Row>.Serialize() , object.Equals(object) ,  
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,  
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Constructors

### CostumeStatSheet()

```
public CostumeStatSheet()
```

# Class CostumeStatSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CostumeStatSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CostumeStatSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CostumeId

```
public int CostumeId { get; }
```

Property Value

[int](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Stat

```
public decimal Stat { get; }
```

Property Value

[decimal](#)

## StatType

```
public StatType StatType { get; }
```

Property Value

[StatType](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CreateAvatarFavSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class CreateAvatarFavSheet : Sheet<string, CreateAvatarFavSheet.Row>,  
IDictionary<string, CreateAvatarFavSheet.Row>, ICollection<KeyValuePair<string,  
CreateAvatarFavSheet.Row>>, IEnumerable<KeyValuePair<string,  
CreateAvatarFavSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<string, CreateAvatarFavSheet.Row>](#) ← CreateAvatarFavSheet

## Implements

[IDictionary<string, CreateAvatarFavSheet.Row>](#),  
[ICollection<KeyValuePair<string, CreateAvatarFavSheet.Row>>](#),  
[IEnumerable<KeyValuePair<string, CreateAvatarFavSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<string, CreateAvatarFavSheet.Row>.Name](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.OrderedList](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.First](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Last](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Keys](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Values](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Count](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.IsReadOnly](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.this\[string\]](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Set<T>\(Sheet<string, T>, bool\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.TryGetValue\(string, out CreateAvatarFavSheet.Row, bool\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.AddRow\(string, CreateAvatarFavSheet.Row\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Add\(string, CreateAvatarFavSheet.Row\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.ContainsKey\(string\)](#) ,  
[Sheet<string, CreateAvatarFavSheet.Row>.Remove\(string\)](#) ,

```
Sheet<string, CreateAvatarFavSheet.Row>.TryGetValue(string, out
CreateAvatarFavSheet.Row) ,
Sheet<string, CreateAvatarFavSheet.Row>.Add(KeyValuePair<string,
CreateAvatarFavSheet.Row>) ,
Sheet<string, CreateAvatarFavSheet.Row>.Clear() ,
Sheet<string, CreateAvatarFavSheet.Row>.Contains(KeyValuePair<string,
CreateAvatarFavSheet.Row>) ,
Sheet<string, CreateAvatarFavSheet.Row>.CopyTo(KeyValuePair<string,
CreateAvatarFavSheet.Row>[], int) ,
Sheet<string, CreateAvatarFavSheet.Row>.Remove(KeyValuePair<string,
CreateAvatarFavSheet.Row>) ,
Sheet<string, CreateAvatarFavSheet.Row>.Serialize() , object.Equals(object)✉ ,
object.Equals(object, object)✉ , object.GetHashCode()✉ , object.GetType()✉ ,
object.MemberwiseClone()✉ , object.ReferenceEquals(object, object)✉ , object.ToString()✉
```

## Constructors

### CreateAvatarFavSheet()

```
public CreateAvatarFavSheet()
```

# Class CreateAvatarFavSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class CreateAvatarFavSheet.Row : SheetRow<string>
```

## Inheritance

[object](#) ← [SheetRow<string>](#) ← CreateAvatarFavSheet.Row

## Inherited Members

[SheetRow<string>.Validate\(\)](#) , [SheetRow<string>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<string>.Equals\(object\)](#) , [SheetRow<string>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Currency

```
public Currency Currency { get; }
```

#### Property Value

Currency

### Key

```
public override string Key { get; }
```

#### Property Value

[string](#)

# Quantity

```
public int Quantity { get; }
```

Property Value

[int](#)

# Target

```
public CreateAvatarFavSheet.Target Target { get; }
```

Property Value

[CreateAvatarFavSheet.Target](#)

# Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Enum CreateAvatarFavSheet.Target

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public enum CreateAvatarFavSheet.Target
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Agent = 0

Avatar = 1

# Class CreateAvatarItemSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class CreateAvatarItemSheet : Sheet<int, CreateAvatarItemSheet.Row>,
IDictionary<int, CreateAvatarItemSheet.Row>, ICollection<KeyValuePair<int,
CreateAvatarItemSheet.Row>>, IEnumerable<KeyValuePair<int,
CreateAvatarItemSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CreateAvatarItemSheet.Row>](#) ← CreateAvatarItemSheet

## Implements

[IDictionary<int, CreateAvatarItemSheet.Row>](#),  
[ICollection<KeyValuePair<int, CreateAvatarItemSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CreateAvatarItemSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CreateAvatarItemSheet.Row>.Name](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.OrderedList](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.First](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Last](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Keys](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Values](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Count](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.TryGetValue\(int, out CreateAvatarItemSheet.Row, bool\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.AddRow\(int, CreateAvatarItemSheet.Row\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Add\(int, CreateAvatarItemSheet.Row\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, CreateAvatarItemSheet.Row>.TryGetValue\(int, out CreateAvatarItemSheet.Row\)](#),  
,

[Sheet<int, CreateAvatarItemSheet.Row>.Add\(KeyValuePair<int, CreateAvatarItemSheet.Row>\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Contains\(KeyValuePair<int, CreateAvatarItemSheet.Row>\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.CopyTo\(KeyValuePair<int, CreateAvatarItemSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Remove\(KeyValuePair<int, CreateAvatarItemSheet.Row>\)](#) ,  
[Sheet<int, CreateAvatarItemSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### CreateAvatarItemSheet()

```
public CreateAvatarItemSheet()
```

# Class CreateAvatarItemSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CreateAvatarItemSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < CreateAvatarItemSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Count

```
public int Count { get; }
```

### Property Value

[int](#)

### ItemId

```
public int ItemId { get; }
```

### Property Value

[int](#)

# Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class DeBuffLimitSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class DeBuffLimitSheet : Sheet<int, DeBuffLimitSheet.Row>, IDictionary<int, DeBuffLimitSheet.Row>, ICollection<KeyValuePair<int, DeBuffLimitSheet.Row>>, IEnumerable<KeyValuePair<int, DeBuffLimitSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, DeBuffLimitSheet.Row>](#) ← DeBuffLimitSheet

## Implements

[IDictionary<int, DeBuffLimitSheet.Row>](#),  
[ICollection<KeyValuePair<int, DeBuffLimitSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, DeBuffLimitSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, DeBuffLimitSheet.Row>.Name](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.OrderedList](#) , [Sheet<int, DeBuffLimitSheet.Row>.First](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Last](#) , [Sheet<int, DeBuffLimitSheet.Row>.Keys](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Values](#) , [Sheet<int, DeBuffLimitSheet.Row>.Count](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.this\[int\]](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.TryGetValue\(int, out DeBuffLimitSheet.Row, bool\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.AddRow\(int, DeBuffLimitSheet.Row\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Add\(int, DeBuffLimitSheet.Row\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.TryGetValue\(int, out DeBuffLimitSheet.Row\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Add\(KeyValuePair<int, DeBuffLimitSheet.Row>\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Contains\(KeyValuePair<int, DeBuffLimitSheet.Row>\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.CopyTo\(KeyValuePair<int, DeBuffLimitSheet.Row>\[\], int\)](#) ,  
[Sheet<int, DeBuffLimitSheet.Row>.Remove\(KeyValuePair<int, DeBuffLimitSheet.Row>\)](#) ,

[Sheet<int, DeBuffLimitSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### DeBuffLimitSheet()

```
public DeBuffLimitSheet()
```

# Class DeBuffLimitSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class DeBuffLimitSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← DeBuffLimitSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### GroupId

```
public int GroupId { get; set; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Value

```
public int Value { get; set; }
```

Property Value

[int](#)

## Methods

### GetModifier(StatType)

```
public StatModifier GetModifier(StatType statType)
```

Parameters

statType [StatType](#)

Returns

[StatModifier](#)

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EnemySkillSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EnemySkillSheet : Sheet<int, EnemySkillSheet.Row>, IDictionary<int, EnemySkillSheet.Row>, ICollection<KeyValuePair<int, EnemySkillSheet.Row>>, IEnumerable<KeyValuePair<int, EnemySkillSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EnemySkillSheet.Row>](#) ← [EnemySkillSheet](#)

## Implements

[IDictionary<int, EnemySkillSheet.Row>](#),  
[ICollection<KeyValuePair<int, EnemySkillSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EnemySkillSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, EnemySkillSheet.Row>.Name](#) , [Sheet<int, EnemySkillSheet.Row>.OrderedList](#) ,  
[Sheet<int, EnemySkillSheet.Row>.First](#) , [Sheet<int, EnemySkillSheet.Row>.Last](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Keys](#) , [Sheet<int, EnemySkillSheet.Row>.Values](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Count](#) , [Sheet<int, EnemySkillSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EnemySkillSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.TryGetValue\(int, out EnemySkillSheet.Row, bool\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.AddRow\(int, EnemySkillSheet.Row\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Add\(int, EnemySkillSheet.Row\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.TryGetValue\(int, out EnemySkillSheet.Row\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Add\(KeyValuePair<int, EnemySkillSheet.Row>\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Contains\(KeyValuePair<int, EnemySkillSheet.Row>\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.CopyTo\(KeyValuePair<int, EnemySkillSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Remove\(KeyValuePair<int, EnemySkillSheet.Row>\)](#) ,  
[Sheet<int, EnemySkillSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### EnemySkillSheet()

```
public EnemySkillSheet()
```

# Class EnemySkillSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EnemySkillSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EnemySkillSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### characterId

```
public int characterId
```

### Field Value

[int](#)

### id

```
public int id
```

### Field Value

[int](#)

## skillId

```
public int skillId
```

### Field Value

[int](#)

## Properties

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EnhancementCostSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EnhancementCostSheet : Sheet<int, EnhancementCostSheet.Row>,
IDictionary<int, EnhancementCostSheet.Row>, ICollection<KeyValuePair<int,
EnhancementCostSheet.Row>>, IEnumerable<KeyValuePair<int,
EnhancementCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EnhancementCostSheet.Row>](#) ← [EnhancementCostSheet](#)

## Implements

[IDictionary<int, EnhancementCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, EnhancementCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EnhancementCostSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, EnhancementCostSheet.Row>.Name](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.First](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Last](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Keys](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Values](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Count](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.TryGetValue\(int, out](#)  
[EnhancementCostSheet.Row, bool\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.AddRow\(int, EnhancementCostSheet.Row\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Add\(int, EnhancementCostSheet.Row\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, EnhancementCostSheet.Row>.TryGetValue\(int, out EnhancementCostSheet.Row\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Add\(KeyValuePair<int, EnhancementCostSheet.Row>\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Contains\(KeyValuePair<int, EnhancementCostSheet.Row>\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.CopyTo\(KeyValuePair<int, EnhancementCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Remove\(KeyValuePair<int, EnhancementCostSheet.Row>\)](#) ,  
[Sheet<int, EnhancementCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### EnhancementCostSheet()

```
public EnhancementCostSheet()
```

# Class EnhancementCostSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EnhancementCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < EnhancementCostSheet.Row

## Derived

[EnhancementCostSheetV2.Row](#), [EnhancementCostSheetV3.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Cost

```
public BigInteger Cost { get; }
```

#### Property Value

[BigInteger](#)

### Grade

```
public int Grade { get; }
```

#### Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int ↗](#)

## ItemSubType

```
public ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

## Key

```
public override int Key { get; }
```

Property Value

[int ↗](#)

## Level

```
public int Level { get; }
```

Property Value

[int ↗](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EnhancementCostSheetV2

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EnhancementCostSheetV2 : Sheet<int, EnhancementCostSheetV2.Row>,
IDictionary<int, EnhancementCostSheetV2.Row>, ICollection<KeyValuePair<int,
EnhancementCostSheetV2.Row>>, IEnumerable<KeyValuePair<int,
EnhancementCostSheetV2.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EnhancementCostSheetV2.Row>](#) ← [EnhancementCostSheetV2](#)

## Implements

[IDictionary<int, EnhancementCostSheetV2.Row>](#),  
[ICollection<KeyValuePair<int, EnhancementCostSheetV2.Row>>](#),  
[IEnumerable<KeyValuePair<int, EnhancementCostSheetV2.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, EnhancementCostSheetV2.Row>.Name](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.OrderedList](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.First](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Last](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Keys](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Values](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Count](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.IsReadOnly](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.this\[int\]](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.TryGetValue\(int, out](#)  
[EnhancementCostSheetV2.Row, bool\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.AddRow\(int, EnhancementCostSheetV2.Row\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Add\(int, EnhancementCostSheetV2.Row\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Remove\(int\)](#) ,

[Sheet<int, EnhancementCostSheetV2.Row>.TryGetValue\(int, out EnhancementCostSheetV2.Row\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Add\(KeyValuePair<int, EnhancementCostSheetV2.Row>\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Clear\(\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Contains\(KeyValuePair<int, EnhancementCostSheetV2.Row>\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.CopyTo\(KeyValuePair<int, EnhancementCostSheetV2.Row>\[\], int\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Remove\(KeyValuePair<int, EnhancementCostSheetV2.Row>\)](#) ,  
[Sheet<int, EnhancementCostSheetV2.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### EnhancementCostSheetV2()

```
public EnhancementCostSheetV2()
```

# Class EnhancementCostSheetV2.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EnhancementCostSheetV2.Row : EnhancementCostSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [EnhancementCostSheet.Row](#) ←

EnhancementCostSheetV2.Row

## Inherited Members

[EnhancementCostSheet.Row.Key](#) , [EnhancementCostSheet.Row.Id](#) ,  
[EnhancementCostSheet.Row.ItemSubType](#) , [EnhancementCostSheet.Row.Grade](#) ,  
[EnhancementCostSheet.Row.Level](#) , [EnhancementCostSheet.Row.Cost](#) ,  
[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BaseStatGrowthMax

```
public int BaseStatGrowthMax { get; }
```

Property Value

[int](#)

### BaseStatGrowthMin

```
public int BaseStatGrowthMin { get; }
```

Property Value

[int](#)

## ExtraSkillChanceGrowthMax

```
public int ExtraSkillChanceGrowthMax { get; }
```

Property Value

[int](#)

## ExtraSkillChanceGrowthMin

```
public int ExtraSkillChanceGrowthMin { get; }
```

Property Value

[int](#)

## ExtraSkillDamageGrowthMax

```
public int ExtraSkillDamageGrowthMax { get; }
```

Property Value

[int](#)

## ExtraSkillDamageGrowthMin

```
public int ExtraSkillDamageGrowthMin { get; }
```

Property Value

[int](#)

## ExtraStatGrowthMax

```
public int ExtraStatGrowthMax { get; }
```

Property Value

[int ↗](#)

## ExtraStatGrowthMin

```
public int ExtraStatGrowthMin { get; }
```

Property Value

[int ↗](#)

## FailRatio

```
public int FailRatio { get; }
```

Property Value

[int ↗](#)

## FailRequiredBlockIndex

```
public int FailRequiredBlockIndex { get; }
```

Property Value

[int ↗](#)

## GreatSuccessRatio

```
public int GreatSuccessRatio { get; }
```

Property Value

[int](#)

## GreatSuccessRequiredBlockIndex

```
public int GreatSuccessRequiredBlockIndex { get; }
```

Property Value

[int](#)

## SuccessRatio

```
public int SuccessRatio { get; }
```

Property Value

[int](#)

## SuccessRequiredBlockIndex

```
public int SuccessRequiredBlockIndex { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EnhancementCostSheetV3

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EnhancementCostSheetV3 : Sheet<int, EnhancementCostSheetV3.Row>,
IDictionary<int, EnhancementCostSheetV3.Row>, ICollection<KeyValuePair<int,
EnhancementCostSheetV3.Row>>, IEnumerable<KeyValuePair<int,
EnhancementCostSheetV3.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EnhancementCostSheetV3.Row>](#) ← [EnhancementCostSheetV3](#)

## Implements

[IDictionary<int, EnhancementCostSheetV3.Row>](#) ,  
[ICollection<KeyValuePair<int, EnhancementCostSheetV3.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, EnhancementCostSheetV3.Row>>](#) , [IEnumerable](#) ,  
[ISheet](#)

## Inherited Members

[Sheet<int, EnhancementCostSheetV3.Row>.Name](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.OrderedList](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.First](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Last](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Keys](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Values](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Count](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.IsReadOnly](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.this\[int\]](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.TryGetValue\(int, out](#)  
[EnhancementCostSheetV3.Row, bool\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.AddRow\(int, EnhancementCostSheetV3.Row\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Add\(int, EnhancementCostSheetV3.Row\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EnhancementCostSheetV3.Row>.Remove\(int\)](#) ,

```
Sheet<int, EnhancementCostSheetV3.Row>.TryGetValue\(int, out
EnhancementCostSheetV3.Row\) ,
Sheet<int, EnhancementCostSheetV3.Row>.Add\(KeyValuePair<int,
EnhancementCostSheetV3.Row>\) ,
Sheet<int, EnhancementCostSheetV3.Row>.Clear\(\) ,
Sheet<int, EnhancementCostSheetV3.Row>.Contains\(KeyValuePair<int,
EnhancementCostSheetV3.Row>\) ,
Sheet<int, EnhancementCostSheetV3.Row>.CopyTo\(KeyValuePair<int,
EnhancementCostSheetV3.Row>\[\], int\) ,
Sheet<int, EnhancementCostSheetV3.Row>.Remove\(KeyValuePair<int,
EnhancementCostSheetV3.Row>\) ,
Sheet<int, EnhancementCostSheetV3.Row>.Serialize\(\) , object.Equals\(object\) ,
object.Equals\(object, object\) , object.GetHashCode\(\) , object.GetType\(\) ,
object.MemberwiseClone\(\) , object.ReferenceEquals\(object, object\) , object.ToString\(\)
```

## Constructors

### EnhancementCostSheetV3()

```
public EnhancementCostSheetV3()
```

## Methods

### GetHammerExp(int)

```
public long GetHammerExp(int hammerId)
```

#### Parameters

hammerId [int](#)

#### Returns

[long](#)

# Class EnhancementCostSheetV3.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EnhancementCostSheetV3.Row : EnhancementCostSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [EnhancementCostSheet.Row](#) ←

EnhancementCostSheetV3.Row

## Inherited Members

[EnhancementCostSheet.Row.Key](#) , [EnhancementCostSheet.Row.Id](#) ,  
[EnhancementCostSheet.Row.ItemSubType](#) , [EnhancementCostSheet.Row.Grade](#) ,  
[EnhancementCostSheet.Row.Level](#) , [EnhancementCostSheet.Row.Cost](#) ,  
[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BaseStatGrowthMax

```
public int BaseStatGrowthMax { get; }
```

Property Value

[int](#)

### BaseStatGrowthMin

```
public int BaseStatGrowthMin { get; }
```

Property Value

[int](#)

## Exp

```
public long Exp { get; }
```

Property Value

[long](#)

## ExtraSkillChanceGrowthMax

```
public int ExtraSkillChanceGrowthMax { get; }
```

Property Value

[int](#)

## ExtraSkillChanceGrowthMin

```
public int ExtraSkillChanceGrowthMin { get; }
```

Property Value

[int](#)

## ExtraSkillDamageGrowthMax

```
public int ExtraSkillDamageGrowthMax { get; }
```

Property Value

[int](#)

## ExtraSkillDamageGrowthMin

```
public int ExtraSkillDamageGrowthMin { get; }
```

Property Value

[int](#) ↗

## ExtraStatGrowthMax

```
public int ExtraStatGrowthMax { get; }
```

Property Value

[int](#) ↗

## ExtraStatGrowthMin

```
public int ExtraStatGrowthMin { get; }
```

Property Value

[int](#) ↗

## RequiredBlockIndex

```
public int RequiredBlockIndex { get; }
```

Property Value

[int](#) ↗

## Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList<string>](#)

# Class EquipmentItemOptionSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemOptionSheet : Sheet<int, EquipmentItemOptionSheet.Row>,
IDictionary<int, EquipmentItemOptionSheet.Row>, ICollection<KeyValuePair<int,
EquipmentItemOptionSheet.Row>>, IEnumerable<KeyValuePair<int,
EquipmentItemOptionSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EquipmentItemOptionSheet.Row>](#) ← [EquipmentItemOptionSheet](#)

## Implements

[IDictionary<int, EquipmentItemOptionSheet.Row>](#),  
[ICollection<KeyValuePair<int, EquipmentItemOptionSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EquipmentItemOptionSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, EquipmentItemOptionSheet.Row>.Name](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.OrderedList](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.First](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Last](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Keys](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Values](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Count](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.TryGetValue\(int, out](#)  
[EquipmentItemOptionSheet.Row, bool\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.AddRow\(int, EquipmentItemOptionSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Add\(int, EquipmentItemOptionSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, EquipmentItemOptionSheet.Row>.TryGetValue\(int, out EquipmentItemOptionSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Add\(KeyValuePair<int, EquipmentItemOptionSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Contains\(KeyValuePair<int, EquipmentItemOptionSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.CopyTo\(KeyValuePair<int, EquipmentItemOptionSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Remove\(KeyValuePair<int, EquipmentItemOptionSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemOptionSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### EquipmentItemOptionSheet()

```
public EquipmentItemOptionSheet()
```

# Class EquipmentItemOptionSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemOptionSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EquipmentItemOptionSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

#### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

## ReferencedStatType

```
public StatType ReferencedStatType { get; }
```

Property Value

[StatType](#)

## SkillChanceMax

```
public int SkillChanceMax { get; }
```

Property Value

[int](#)

## SkillChanceMin

```
public int SkillChanceMin { get; }
```

Property Value

[int](#)

## SkillDamageMax

```
public int SkillDamageMax { get; }
```

Property Value

[int](#)

## SkillDamageMin

```
public int SkillDamageMin { get; }
```

Property Value

[int](#)

## SkillId

```
public int SkillId { get; }
```

Property Value

[int](#)

## StatDamageRatioMax

```
public int StatDamageRatioMax { get; }
```

Property Value

[int](#)

## StatDamageRatioMin

```
public int StatDamageRatioMin { get; }
```

Property Value

[int](#)

## StatMax

```
public int StatMax { get; }
```

Property Value

[int](#)

## StatMin

```
public int StatMin { get; }
```

Property Value

[int](#)

## StatType

```
public StatType StatType { get; }
```

Property Value

[StatType](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EquipmentItemRecipeSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemRecipeSheet : Sheet<int, EquipmentItemRecipeSheet.Row>,
IDictionary<int, EquipmentItemRecipeSheet.Row>, ICollection<KeyValuePair<int,
EquipmentItemRecipeSheet.Row>>, IEnumerable<KeyValuePair<int,
EquipmentItemRecipeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EquipmentItemRecipeSheet.Row>](#) ← [EquipmentItemRecipeSheet](#)

## Implements

[IDictionary<int, EquipmentItemRecipeSheet.Row>](#),  
[ICollection<KeyValuePair<int, EquipmentItemRecipeSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EquipmentItemRecipeSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, EquipmentItemRecipeSheet.Row>.Name](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.OrderedList](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.First](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Last](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Keys](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Values](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Count](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.TryGetValue\(int, out EquipmentItemRecipeSheet.Row, bool\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.AddRow\(int, EquipmentItemRecipeSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Add\(int, EquipmentItemRecipeSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EquipmentItemRecipeSheet.Row>.Remove\(int\)](#) ,

```
Sheet<int, EquipmentItemRecipeSheet.Row>.TryGetValue(int, out
EquipmentItemRecipeSheet.Row) ,
Sheet<int, EquipmentItemRecipeSheet.Row>.Add(KeyValuePair<int,
EquipmentItemRecipeSheet.Row>) ,
Sheet<int, EquipmentItemRecipeSheet.Row>.Clear() ,
Sheet<int, EquipmentItemRecipeSheet.Row>.Contains(KeyValuePair<int,
EquipmentItemRecipeSheet.Row>) ,
Sheet<int, EquipmentItemRecipeSheet.Row>.CopyTo(KeyValuePair<int,
EquipmentItemRecipeSheet.Row>[], int) ,
Sheet<int, EquipmentItemRecipeSheet.Row>.Remove(KeyValuePair<int,
EquipmentItemRecipeSheet.Row>) ,
Sheet<int, EquipmentItemRecipeSheet.Row>.Serialize() , object.Equals(object)✉ ,
object.Equals(object, object)✉ , object.GetHashCode()✉ , object.GetType()✉ ,
object.MemberwiseClone()✉ , object.ReferenceEquals(object, object)✉ , object.ToString()✉
```

## Constructors

### EquipmentItemRecipeSheet()

```
public EquipmentItemRecipeSheet()
```

# Class EquipmentItemRecipeSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemRecipeSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EquipmentItemRecipeSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CRYSTAL

```
public int CRYSTAL { get; }
```

Property Value

[int](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## ItemSubType

```
public ItemSubType? ItemSubType { get; }
```

Property Value

[ItemSubType?](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## MaterialCount

```
public int MaterialCount { get; }
```

Property Value

[int](#)

## MaterialId

```
public int MaterialId { get; }
```

Property Value

[int](#)

## RequiredActionPoint

```
public int RequiredActionPoint { get; }
```

Property Value

[int](#)

## RequiredBlockIndex

```
public long RequiredBlockIndex { get; }
```

Property Value

[long](#)

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#)

## ResultEquipmentId

```
public int ResultEquipmentId { get; }
```

Property Value

[int](#)

## SubRecipIds

```
public List<int> SubRecipeIds { get; }
```

Property Value

[List](#)<[int](#)>

## UnlockStage

```
public int UnlockStage { get; }
```

Property Value

[int](#)

## Methods

### GetAllMaterials(EquipmentItemSubRecipeSheetV2, CraftType)

```
public IEnumerable<EquipmentItemSubRecipeSheet.MaterialInfo>
GetAllMaterials(EquipmentItemSubRecipeSheetV2 sheet, CraftType subRecipeType
= CraftType.Normal)
```

Parameters

sheet [EquipmentItemSubRecipeSheetV2](#)

subRecipeType [CraftType](#)

Returns

[IEnumerable](#)<[EquipmentItemSubRecipeSheet.MaterialInfo](#)>

## IsMimisBrunnrSubRecipe(int?)

```
[Obsolete("It is obsoleted. Check  
EquipmentItemSubRecipeSheetV2.IsMimisbrunnrSubRecipe.")]  
public bool IsMimisBrunnrSubRecipe(int? subRecipeId)
```

## Parameters

subRecipeId [int](#)?

## Returns

[bool](#)

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EquipmentItemSetEffectSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EquipmentItemSetEffectSheet : Sheet<int,
EquipmentItemSetEffectSheet.Row>, IDictionary<int, EquipmentItemSetEffectSheet.Row>,
ICollection<KeyValuePair<int, EquipmentItemSetEffectSheet.Row>>,
IEnumerable<KeyValuePair<int, EquipmentItemSetEffectSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EquipmentItemSetEffectSheet.Row>](#) ←  
EquipmentItemSetEffectSheet

## Implements

[IDictionary<int, EquipmentItemSetEffectSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, EquipmentItemSetEffectSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, EquipmentItemSetEffectSheet.Row>>](#) ,  
[IEnumerable](#) , [ISheet](#)

## Inherited Members

[Sheet<int, EquipmentItemSetEffectSheet.Row>.Name](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.OrderedList](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.First](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Last](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Keys](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Values](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Count](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.TryGetValue\(int, out EquipmentItemSetEffectSheet.Row, bool\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.AddRow\(int, EquipmentItemSetEffectSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Add\(int, EquipmentItemSetEffectSheet.Row\)](#) ,

[Sheet<int, EquipmentItemSetEffectSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.TryGetValue\(int, out EquipmentItemSetEffectSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Add\(KeyValuePair<int, EquipmentItemSetEffectSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Contains\(KeyValuePair<int, EquipmentItemSetEffectSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.CopyTo\(KeyValuePair<int, EquipmentItemSetEffectSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Remove\(KeyValuePair<int, EquipmentItemSetEffectSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSetEffectSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Extension Methods

[SetEffectExtension.GetSetEffectRows\(EquipmentItemSetEffectSheet, IEnumerable<Equipment>\)](#)

## Constructors

### EquipmentItemSetEffectSheet()

```
public EquipmentItemSetEffectSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, EquipmentItemSetEffectSheet.Row value)
```

## Parameters

key [int](#) ↴

value [EquipmentItemSetEffectSheet.Row](#)

# Class EquipmentItemSetEffectSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EquipmentItemSetEffectSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EquipmentItemSetEffectSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

# StatModifiers

```
public Dictionary<int, StatModifier> StatModifiers { get; }
```

Property Value

[Dictionary](#)<[int](#), [StatModifier](#)>

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EquipmentItemSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EquipmentItemSheet : Sheet<int, EquipmentItemSheet.Row>,
IDictionary<int, EquipmentItemSheet.Row>, ICollection<KeyValuePair<int,
EquipmentItemSheet.Row>>, IEnumerable<KeyValuePair<int, EquipmentItemSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EquipmentItemSheet.Row>](#) ← [EquipmentItemSheet](#)

## Implements

[IDictionary<int, EquipmentItemSheet.Row>](#),  
[ICollection<KeyValuePair<int, EquipmentItemSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EquipmentItemSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, EquipmentItemSheet.Row>.Name](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.OrderedList](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.First](#) , [Sheet<int, EquipmentItemSheet.Row>.Last](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Keys](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Values](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Count](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.TryGetValue\(int, out EquipmentItemSheet.Row, bool\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.AddRow\(int, EquipmentItemSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Add\(int, EquipmentItemSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.TryGetValue\(int, out EquipmentItemSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Add\(KeyValuePair<int, EquipmentItemSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSheet.Row>.Clear\(\)](#) ,

```
Sheet<int, EquipmentItemSheet.Row>.Contains(KeyValuePair<int,
EquipmentItemSheet.Row>),
Sheet<int, EquipmentItemSheet.Row>.CopyTo(KeyValuePair<int,
EquipmentItemSheet.Row>[], int),
Sheet<int, EquipmentItemSheet.Row>.Remove(KeyValuePair<int,
EquipmentItemSheet.Row>),
Sheet<int, EquipmentItemSheet.Row>.Serialize(), object.Equals(object)↗ ,
object.Equals(object, object)↗ , object.GetHashCode()↗ , object.GetType()↗ ,
object.MemberwiseClone()↗ , object.ReferenceEquals(object, object)↗ , object.ToString()↗
```

## Constructors

### EquipmentItemSheet()

```
public EquipmentItemSheet()
```

# Class EquipmentItemSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EquipmentItemSheet.Row : ItemSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [ItemSheet.RowBase](#) ← [ItemSheet.Row](#) ← EquipmentItemSheet.Row

## Inherited Members

[ItemSheet.Row.Key](#) , [ItemSheet.Row.Id](#) , [ItemSheet.Row.ItemSubType](#) ,  
[ItemSheet.Row.Grade](#) , [ItemSheet.Row.ElementalType](#) ,  
[ItemSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Row()

```
public Row()
```

### Row(Dictionary)

```
public Row(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Properties

### AttackRange

```
public decimal AttackRange { get; }
```

Property Value

[decimal](#) ↗

### Exp

```
public long? Exp { get; }
```

Property Value

[long](#) ↗?

### ItemType

```
public override ItemType ItemType { get; }
```

Property Value

[ItemType](#)

### SetId

```
public int SetId { get; }
```

Property Value

[int](#) ↗

# SpineResourcePath

```
public string SpineResourcePath { get; }
```

Property Value

[string](#)

## Stat

```
public DecimalStat Stat { get; }
```

Property Value

[DecimalStat](#)

## Methods

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EquipmentItemSubRecipeSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemSubRecipeSheet : Sheet<int,  
EquipmentItemSubRecipeSheet.Row>, IDictionary<int, EquipmentItemSubRecipeSheet.Row>,  
ICollection<KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>>,  
IEnumerable<KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EquipmentItemSubRecipeSheet.Row>](#) ←  
EquipmentItemSubRecipeSheet

## Implements

[IDictionary<int, EquipmentItemSubRecipeSheet.Row>](#),  
[ICollection<KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>>](#),  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Name](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.OrderedList](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.First](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Last](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Keys](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Values](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Count](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.TryGetValue\(int, out  
EquipmentItemSubRecipeSheet.Row, bool\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.AddRow\(int,  
EquipmentItemSubRecipeSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Add\(int,  
EquipmentItemSubRecipeSheet.Row\)](#) ,

[Sheet<int, EquipmentItemSubRecipeSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.TryGetValue\(int, out EquipmentItemSubRecipeSheet.Row\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Add\(KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Contains\(KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.CopyTo\(KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Remove\(KeyValuePair<int, EquipmentItemSubRecipeSheet.Row>\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### EquipmentItemSubRecipeSheet()

```
public EquipmentItemSubRecipeSheet()
```

# Struct EquipmentItemSubRecipeSheet.MaterialInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public readonly struct EquipmentItemSubRecipeSheet.MaterialInfo
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### MaterialInfo(int, int)

```
public MaterialInfo(int id, int count)
```

#### Parameters

id [int](#)

count [int](#)

## Fields

### Count

```
public readonly int Count
```

#### Field Value

[int](#) ↗

Id

```
public readonly int Id
```

Field Value

[int](#) ↗

# Struct EquipmentItemSubRecipeSheet.OptionInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public readonly struct EquipmentItemSubRecipeSheet.OptionInfo
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### OptionInfo(int, decimal)

```
public OptionInfo(int id, decimal ratio)
```

## Parameters

id [int](#)

ratio [decimal](#)

## Fields

### Id

```
public readonly int Id
```

## Field Value

[int](#)

# Ratio

```
public readonly decimal Ratio
```

Field Value

[decimal](#) ↗

# Class EquipmentItemSubRecipeSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemSubRecipeSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EquipmentItemSubRecipeSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

#### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

## Materials

```
public List<EquipmentItemSubRecipeSheet.MaterialInfo> Materials { get; }
```

Property Value

[List](#)<[EquipmentItemSubRecipeSheet](#).[MaterialInfo](#)>

## MaxOptionLimit

```
public int MaxOptionLimit { get; }
```

Property Value

[int](#)

## Options

```
public List<EquipmentItemSubRecipeSheet.OptionInfo> Options { get; }
```

Property Value

[List](#)<[EquipmentItemSubRecipeSheet](#).[OptionInfo](#)>

## RequiredActionPoint

```
public int RequiredActionPoint { get; }
```

Property Value

[int](#)

## RequiredBlockIndex

```
public long RequiredBlockIndex { get; }
```

Property Value

[long](#) ↗

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#) ↗

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#) ↗ <[string](#) ↗>

# Class EquipmentItemSubRecipeSheetV2

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemSubRecipeSheetV2 : Sheet<int,
EquipmentItemSubRecipeSheetV2.Row>, IDictionary<int,
EquipmentItemSubRecipeSheetV2.Row>, ICollection<KeyValuePair<int,
EquipmentItemSubRecipeSheetV2.Row>>, IEnumerable<KeyValuePair<int,
EquipmentItemSubRecipeSheetV2.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EquipmentItemSubRecipeSheetV2.Row>](#) ←  
EquipmentItemSubRecipeSheetV2

## Implements

[IDictionary<int, EquipmentItemSubRecipeSheetV2.Row>](#),  
[ICollection<KeyValuePair<int, EquipmentItemSubRecipeSheetV2.Row>>](#),  
[IEnumerable<KeyValuePair<int, EquipmentItemSubRecipeSheetV2.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Name](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.OrderedList](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.First](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Last](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Keys](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Values](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Count](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.IsReadOnly](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.this\[int\]](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.TryGetValue\(int, out EquipmentItemSubRecipeSheetV2.Row, bool\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.AddRow\(int, EquipmentItemSubRecipeSheetV2.Row\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Add\(int, EquipmentItemSubRecipeSheetV2.Row\)](#) ,

[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Remove\(int\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.TryGetValue\(int, out EquipmentItemSubRecipeSheetV2.Row\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Add\(KeyValuePair<int, EquipmentItemSubRecipeSheetV2.Row>\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Clear\(\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Contains\(KeyValuePair<int, EquipmentItemSubRecipeSheetV2.Row>\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.CopyTo\(KeyValuePair<int, EquipmentItemSubRecipeSheetV2.Row>\[\], int\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Remove\(KeyValuePair<int, EquipmentItemSubRecipeSheetV2.Row>\)](#) ,  
[Sheet<int, EquipmentItemSubRecipeSheetV2.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### EquipmentItemSubRecipeSheetV2()

```
public EquipmentItemSubRecipeSheetV2()
```

# Struct EquipmentItemSubRecipeSheetV2.OptionInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public readonly struct EquipmentItemSubRecipeSheetV2.OptionInfo
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### OptionInfo(int, int, int)

```
public OptionInfo(int id, int ratio, int requiredBlockIndex = 0)
```

## Parameters

id [int](#)

ratio [int](#)

requiredBlockIndex [int](#)

## Fields

### Id

```
public readonly int Id
```

Field Value

[int](#) ↗

Ratio

```
public readonly int Ratio
```

Field Value

[int](#) ↗

RequiredBlockIndex

```
public readonly int RequiredBlockIndex
```

Field Value

[int](#) ↗

# Class EquipmentItemSubRecipeSheetV2.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class EquipmentItemSubRecipeSheetV2.Row : SheetRow<int>
```

## Inheritance

[object](#) ↗ ← [SheetRow<int](#) ↗ < EquipmentItemSubRecipeSheetV2.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) ↗ , [object.GetType\(\)](#) ↗ , [object.MemberwiseClone\(\)](#) ↗ ,  
[object.ReferenceEquals\(object, object\)](#) ↗ , [object.ToString\(\)](#) ↗

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#) ↗

### IsMimisbrunnrSubRecipe

```
public bool? IsMimisbrunnrSubRecipe { get; }
```

### Property Value

[bool](#) ↗?

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Materials

```
public List<EquipmentItemSubRecipeSheet.MaterialInfo> Materials { get; }
```

Property Value

[List](#)<[EquipmentItemSubRecipeSheet](#).[MaterialInfo](#)>

## Options

```
public List<EquipmentItemSubRecipeSheetV2.OptionInfo> Options { get; }
```

Property Value

[List](#)<[EquipmentItemSubRecipeSheetV2](#).[OptionInfo](#)>

## RequiredActionPoint

```
public int RequiredActionPoint { get; }
```

Property Value

[int](#)

## RequiredBlockIndex

```
public long RequiredBlockIndex { get; }
```

Property Value

[long](#) ↗

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#) ↗

## RewardHammerPoint

```
public int? RewardHammerPoint { get; }
```

Property Value

[int](#) ↗?

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#) ↗<[string](#) ↗>

# Class GameConfigSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GameConfigSheet : Sheet<string, GameConfigSheet.Row>,
IDictionary<string, GameConfigSheet.Row>, ICollection<KeyValuePair<string,
GameConfigSheet.Row>>, IEnumerable<KeyValuePair<string, GameConfigSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<string, GameConfigSheet.Row>](#) ← [GameConfigSheet](#)

## Implements

[IDictionary<string, GameConfigSheet.Row>](#),  
[ICollection<KeyValuePair<string, GameConfigSheet.Row>>](#),  
[IEnumerable<KeyValuePair<string, GameConfigSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<string, GameConfigSheet.Row>.Name](#) ,  
[Sheet<string, GameConfigSheet.Row>.OrderedList](#) ,  
[Sheet<string, GameConfigSheet.Row>.First](#) , [Sheet<string, GameConfigSheet.Row>.Last](#) ,  
[Sheet<string, GameConfigSheet.Row>.Keys](#) ,  
[Sheet<string, GameConfigSheet.Row>.Values](#) ,  
[Sheet<string, GameConfigSheet.Row>.Count](#) ,  
[Sheet<string, GameConfigSheet.Row>.IsReadOnly](#) ,  
[Sheet<string, GameConfigSheet.Row>.this\[string\]](#) ,  
[Sheet<string, GameConfigSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.Set<T>\(Sheet<string, T>, bool\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.TryGetValue\(string, out GameConfigSheet.Row, bool\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.AddRow\(string, GameConfigSheet.Row\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.Add\(string, GameConfigSheet.Row\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.ContainsKey\(string\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.Remove\(string\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.TryGetValue\(string, out GameConfigSheet.Row\)](#) ,  
[Sheet<string, GameConfigSheet.Row>.Add\(KeyValuePair<string, GameConfigSheet.Row>\)](#) ,

```
Sheet<string, GameConfigSheet.Row>.Clear() ,  
Sheet<string, GameConfigSheet.Row>.Contains(KeyValuePair<string,  
GameConfigSheet.Row>) ,  
Sheet<string, GameConfigSheet.Row>.CopyTo(KeyValuePair<string,  
GameConfigSheet.Row>[], int) ,  
Sheet<string, GameConfigSheet.Row>.Remove(KeyValuePair<string,  
GameConfigSheet.Row>) ,  
Sheet<string, GameConfigSheet.Row>.Serialize() , object.Equals(object) ↗ ,  
object.Equals(object, object) ↗ , object.GetHashCode() ↗ , object.GetType() ↗ ,  
object.MemberwiseClone() ↗ , object.ReferenceEquals(object, object) ↗ , object.ToString() ↗
```

## Constructors

### GameConfigSheet()

```
public GameConfigSheet()
```

# Class GameConfigSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GameConfigSheet.Row : SheetRow<string>
```

## Inheritance

[object](#) ← [SheetRow<string>](#) ← GameConfigSheet.Row

## Inherited Members

[SheetRow<string>.Validate\(\)](#) , [SheetRow<string>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<string>.Equals\(object\)](#) , [SheetRow<string>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override string Key { get; }
```

#### Property Value

[string](#)

### Value

```
public string Value { get; }
```

#### Property Value

[string](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class GeneralQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GeneralQuestSheet : Sheet<int, GeneralQuestSheet.Row>, IDictionary<int, GeneralQuestSheet.Row>, ICollection<KeyValuePair<int, GeneralQuestSheet.Row>>, IEnumerable<KeyValuePair<int, GeneralQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, GeneralQuestSheet.Row>](#) ← GeneralQuestSheet

## Implements

[IDictionary<int, GeneralQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, GeneralQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, GeneralQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, GeneralQuestSheet.Row>.Name](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.First](#) , [Sheet<int, GeneralQuestSheet.Row>.Last](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Keys](#) , [Sheet<int, GeneralQuestSheet.Row>.Values](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Count](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.TryGetValue\(int, out GeneralQuestSheet.Row, bool\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.AddRow\(int, GeneralQuestSheet.Row\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Add\(int, GeneralQuestSheet.Row\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.TryGetValue\(int, out GeneralQuestSheet.Row\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Add\(KeyValuePair<int, GeneralQuestSheet.Row>\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, GeneralQuestSheet.Row>.Contains\(KeyValuePair<int, GeneralQuestSheet.Row>\)](#) ,

```
Sheet<int, GeneralQuestSheet.Row>.CopyTo(KeyValuePair<int, GeneralQuestSheet.Row>
[], int) ,
Sheet<int, GeneralQuestSheet.Row>.Remove(KeyValuePair<int, GeneralQuestSheet.Row>)

,
Sheet<int, GeneralQuestSheet.Row>.Serialize() , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Constructors

### GeneralQuestSheet()

```
public GeneralQuestSheet()
```

# Class GeneralQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GeneralQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← GeneralQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Event

```
public QuestEventType Event { get; }
```

### Property Value

[QuestEventType](#)

## Methods

[Set\(IReadOnlyList<string>\)](#)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class GoldQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GoldQuestSheet : Sheet<int, GoldQuestSheet.Row>, IDictionary<int,
GoldQuestSheet.Row>, ICollection<KeyValuePair<int, GoldQuestSheet.Row>>,
IEnumerable<KeyValuePair<int, GoldQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, GoldQuestSheet.Row>](#) ← [GoldQuestSheet](#)

## Implements

[IDictionary<int, GoldQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, GoldQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, GoldQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, GoldQuestSheet.Row>.Name](#) , [Sheet<int, GoldQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, GoldQuestSheet.Row>.First](#) , [Sheet<int, GoldQuestSheet.Row>.Last](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Keys](#) , [Sheet<int, GoldQuestSheet.Row>.Values](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Count](#) , [Sheet<int, GoldQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, GoldQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.TryGetValue\(int, out GoldQuestSheet.Row, bool\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.AddRow\(int, GoldQuestSheet.Row\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Add\(int, GoldQuestSheet.Row\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.TryGetValue\(int, out GoldQuestSheet.Row\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Add\(KeyValuePair<int, GoldQuestSheet.Row>\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Contains\(KeyValuePair<int, GoldQuestSheet.Row>\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.CopyTo\(KeyValuePair<int, GoldQuestSheet.Row>\[\], int\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Remove\(KeyValuePair<int, GoldQuestSheet.Row>\)](#) ,  
[Sheet<int, GoldQuestSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### GoldQuestSheet()

```
public GoldQuestSheet()
```

# Class GoldQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GoldQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← GoldQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Type

```
public TradeType Type { get; }
```

### Property Value

[TradeType](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Interface ISheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public interface ISheet
```

## Methods

### Set(string, bool)

```
void Set(string csv, bool isReversed = false)
```

#### Parameters

csv [string](#)

isReversed [bool](#)

# Interface IStakeRewardRow

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public interface IStakeRewardRow
```

## Properties

### Level

```
int Level { get; }
```

Property Value

[int](#)

### RequiredGold

```
long RequiredGold { get; }
```

Property Value

[long](#)

# Interface IStakeRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public interface IStakeRewardSheet : ISheet
```

## Inherited Members

[ISheet.Set\(string, bool\)](#)

## Extension Methods

[SheetsExtensions.FindLevelByStakedAmount\(IStakeRewardSheet, Address, FungibleAssetValue\)](#)

## Properties

### OrderedRows

```
IReadOnlyList<IStakeRewardRow> OrderedRows { get; }
```

### Property Value

[IReadOnlyList](#)<[IStakeRewardRow](#)>

# Interface IWorldBossRewardRow

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public interface IWorldBossRewardRow
```

## Properties

### BossId

```
int BossId { get; }
```

#### Property Value

[int](#)

### Circle

```
int Circle { get; }
```

#### Property Value

[int](#)

### Crystal

```
int Crystal { get; }
```

#### Property Value

[int](#)

## Rank

```
int Rank { get; }
```

Property Value

[int](#) ↗

## Rune

```
int Rune { get; }
```

Property Value

[int](#) ↗

# Interface IWorldBossRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public interface IWorldBossRewardSheet : ISheet
```

## Inherited Members

[ISheet.Set\(string, bool\)](#)

## Properties

### OrderedRows

IReadOnlyList<IWorldBossRewardRow> OrderedRows { get; }

Property Value

[IReadOnlyList](#)<[IWorldBossRewardRow](#)>

# Class ItemConfigForGradeSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemConfigForGradeSheet : Sheet<int, ItemConfigForGradeSheet.Row>,
IDictionary<int, ItemConfigForGradeSheet.Row>, ICollection<KeyValuePair<int,
ItemConfigForGradeSheet.Row>>, IEnumerable<KeyValuePair<int,
ItemConfigForGradeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ItemConfigForGradeSheet.Row>](#) ← [ItemConfigForGradeSheet](#)

## Implements

[IDictionary<int, ItemConfigForGradeSheet.Row>](#),  
[ICollection<KeyValuePair<int, ItemConfigForGradeSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ItemConfigForGradeSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, ItemConfigForGradeSheet.Row>.Name](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.OrderedList](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.First](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Last](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Keys](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Values](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Count](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.TryGetValue\(int, out  
ItemConfigForGradeSheet.Row, bool\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.AddRow\(int, ItemConfigForGradeSheet.Row\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Add\(int, ItemConfigForGradeSheet.Row\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ItemConfigForGradeSheet.Row>.Remove\(int\)](#) ,

```
Sheet<int, ItemConfigForGradeSheet.Row>.TryGetValue(int, out
ItemConfigForGradeSheet.Row) ,
Sheet<int, ItemConfigForGradeSheet.Row>.Add(KeyValuePair<int,
ItemConfigForGradeSheet.Row>) ,
Sheet<int, ItemConfigForGradeSheet.Row>.Clear() ,
Sheet<int, ItemConfigForGradeSheet.Row>.Contains(KeyValuePair<int,
ItemConfigForGradeSheet.Row>) ,
Sheet<int, ItemConfigForGradeSheet.Row>.CopyTo(KeyValuePair<int,
ItemConfigForGradeSheet.Row>[], int) ,
Sheet<int, ItemConfigForGradeSheet.Row>.Remove(KeyValuePair<int,
ItemConfigForGradeSheet.Row>) ,
Sheet<int, ItemConfigForGradeSheet.Row>.Serialize() , object.Equals(object)✉ ,
object.Equals(object, object)✉ , object.GetHashCode()✉ , object.GetType()✉ ,
object.MemberwiseClone()✉ , object.ReferenceEquals(object, object)✉ , object.ToString()✉
```

## Constructors

### ItemConfigForGradeSheet()

```
public ItemConfigForGradeSheet()
```

# Class ItemConfigForGradeSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemConfigForGradeSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← ItemConfigForGradeSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### EnhancementLimit

```
public int EnhancementLimit { get; }
```

Property Value

[int](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## MonsterPartsCountForCombination

```
public int MonsterPartsCountForCombination { get; }
```

Property Value

[int](#)

## MonsterPartsCountForCombinationWithNCG

```
public int MonsterPartsCountForCombinationWithNCG { get; }
```

Property Value

[int](#)

## RandomBuffSkillMaxCountForCombination

```
public int RandomBuffSkillMaxCountForCombination { get; }
```

Property Value

[int](#)

## RandomBuffSkillMinCountForCombination

```
public int RandomBuffSkillMinCountForCombination { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class ItemEnhancementQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancementQuestSheet : Sheet<int, ItemEnhancementQuestSheet.Row>,
IDictionary<int, ItemEnhancementQuestSheet.Row>, ICollection<KeyValuePair<int,
ItemEnhancementQuestSheet.Row>>, IEnumerable<KeyValuePair<int,
ItemEnhancementQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ItemEnhancementQuestSheet.Row>](#) ←  
ItemEnhancementQuestSheet

## Implements

[IDictionary<int, ItemEnhancementQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, ItemEnhancementQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ItemEnhancementQuestSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, ItemEnhancementQuestSheet.Row>.Name](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.First](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Last](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Keys](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Values](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Count](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.TryGetValue\(int, out ItemEnhancementQuestSheet.Row, bool\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.AddRow\(int, ItemEnhancementQuestSheet.Row\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Add\(int, ItemEnhancementQuestSheet.Row\)](#) ,

[Sheet<int, ItemEnhancementQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.TryGetValue\(int, out ItemEnhancementQuestSheet.Row\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Add\(KeyValuePair<int, ItemEnhancementQuestSheet.Row>\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Contains\(KeyValuePair<int, ItemEnhancementQuestSheet.Row>\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.CopyTo\(KeyValuePair<int, ItemEnhancementQuestSheet.Row>\[\], int\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Remove\(KeyValuePair<int, ItemEnhancementQuestSheet.Row>\)](#) ,  
[Sheet<int, ItemEnhancementQuestSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### ItemEnhancementQuestSheet()

```
public ItemEnhancementQuestSheet()
```

# Class ItemEnhancementQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemEnhancementQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← ItemEnhancementQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Geo](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Count

```
public int Count { get; }
```

### Property Value

[int](#)

### Grade

```
public int Grade { get; }
```

Property Value

[int](#)

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class ItemGradeQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemGradeQuestSheet : Sheet<int, ItemGradeQuestSheet.Row>,
IDictionary<int, ItemGradeQuestSheet.Row>, ICollection<KeyValuePair<int,
ItemGradeQuestSheet.Row>>, IEnumerable<KeyValuePair<int, ItemGradeQuestSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ItemGradeQuestSheet.Row>](#) ← [ItemGradeQuestSheet](#)

## Implements

[IDictionary<int, ItemGradeQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, ItemGradeQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ItemGradeQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, ItemGradeQuestSheet.Row>.Name](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.First](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Last](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Keys](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Values](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Count](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.TryGetValue\(int, out ItemGradeQuestSheet.Row, bool\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.AddRow\(int, ItemGradeQuestSheet.Row\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Add\(int, ItemGradeQuestSheet.Row\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ItemGradeQuestSheet.Row>.TryGetValue\(int, out ItemGradeQuestSheet.Row\)](#) ,

```
Sheet<int, ItemGradeQuestSheet.Row>.Add(KeyValuePair<int,
ItemGradeQuestSheet.Row>),
Sheet<int, ItemGradeQuestSheet.Row>.Clear() ,
Sheet<int, ItemGradeQuestSheet.Row>.Contains(KeyValuePair<int,
ItemGradeQuestSheet.Row>),
Sheet<int, ItemGradeQuestSheet.Row>.CopyTo(KeyValuePair<int,
ItemGradeQuestSheet.Row>[], int) ,
Sheet<int, ItemGradeQuestSheet.Row>.Remove(KeyValuePair<int,
ItemGradeQuestSheet.Row>),
Sheet<int, ItemGradeQuestSheet.Row>.Serialize() , object.Equals(object)✉ ,
object.Equals(object, object)✉ , object.GetHashCode()✉ , object.GetType()✉ ,
object.MemberwiseClone()✉ , object.ReferenceEquals(object, object)✉ , object.ToString()✉
```

## Constructors

### ItemGradeQuestSheet()

```
public ItemGradeQuestSheet()
```

# Class ItemGradeQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemGradeQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← ItemGradeQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Grade

```
public int Grade { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class ItemRequirementSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class ItemRequirementSheet : Sheet<int, ItemRequirementSheet.Row>,
IDictionary<int, ItemRequirementSheet.Row>, ICollection<KeyValuePair<int,
ItemRequirementSheet.Row>>, IEnumerable<KeyValuePair<int,
ItemRequirementSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ItemRequirementSheet.Row>](#) ← [ItemRequirementSheet](#)

## Implements

[IDictionary<int, ItemRequirementSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, ItemRequirementSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, ItemRequirementSheet.Row>>](#) , [IEnumerable](#) ,  
[ISheet](#)

## Inherited Members

[Sheet<int, ItemRequirementSheet.Row>.Name](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.OrderedList](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.First](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Last](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Keys](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Values](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Count](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.TryGetValue\(int, out ItemRequirementSheet.Row, bool\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.AddRow\(int, ItemRequirementSheet.Row\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Add\(int, ItemRequirementSheet.Row\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ItemRequirementSheet.Row>.TryGetValue\(int, out ItemRequirementSheet.Row\)](#) ,

```
Sheet<int, ItemRequirementSheet.Row>.Add(KeyValuePair<int,
ItemRequirementSheet.Row>),
Sheet<int, ItemRequirementSheet.Row>.Clear() ,
Sheet<int, ItemRequirementSheet.Row>.Contains(KeyValuePair<int,
ItemRequirementSheet.Row>),
Sheet<int, ItemRequirementSheet.Row>.CopyTo(KeyValuePair<int,
ItemRequirementSheet.Row>[], int) ,
Sheet<int, ItemRequirementSheet.Row>.Remove(KeyValuePair<int,
ItemRequirementSheet.Row>),
Sheet<int, ItemRequirementSheet.Row>.Serialize() , object.Equals(object) ↴ ,
object.Equals(object, object) ↴ , object.GetHashCode() ↴ , object.GetType() ↴ ,
object.MemberwiseClone() ↴ , object.ReferenceEquals(object, object) ↴ , object.ToString() ↴
```

## Constructors

### ItemRequirementSheet()

```
public ItemRequirementSheet()
```

# Class ItemRequirementSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class ItemRequirementSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← ItemRequirementSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ItemId

```
public int ItemId { get; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Level

```
public int Level { get; }
```

Property Value

[int](#)

## MimisLevel

```
public int MimisLevel { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class ItemSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemSheet : Sheet<int, ItemSheet.Row>, IDictionary<int, ItemSheet.Row>,
ICollection<KeyValuePair<int, ItemSheet.Row>>, IEnumerable<KeyValuePair<int,
ItemSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ItemSheet.Row>](#) ← ItemSheet

## Implements

[IDictionary<int, ItemSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, ItemSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, ItemSheet.Row>>](#) , [IEnumerable](#) , [ISheet](#)

## Inherited Members

[Sheet<int, ItemSheet.Row>.Name](#) , [Sheet<int, ItemSheet.Row>.OrderedList](#) ,  
[Sheet<int, ItemSheet.Row>.First](#) , [Sheet<int, ItemSheet.Row>.Last](#) ,  
[Sheet<int, ItemSheet.Row>.Keys](#) , [Sheet<int, ItemSheet.Row>.Values](#) ,  
[Sheet<int, ItemSheet.Row>.Count](#) , [Sheet<int, ItemSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ItemSheet.Row>.this\[int\]](#) , [Sheet<int, ItemSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ItemSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ItemSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ItemSheet.Row>.TryGetValue\(int, out ItemSheet.Row, bool\)](#) ,  
[Sheet<int, ItemSheet.Row>.AddRow\(int, ItemSheet.Row\)](#) ,  
[Sheet<int, ItemSheet.Row>.Add\(int, ItemSheet.Row\)](#) ,  
[Sheet<int, ItemSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, ItemSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, ItemSheet.Row>.TryGetValue\(int, out ItemSheet.Row\)](#) ,  
[Sheet<int, ItemSheet.Row>.Add\(KeyValuePair<int, ItemSheet.Row>\)](#) ,  
[Sheet<int, ItemSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, ItemSheet.Row>.Contains\(KeyValuePair<int, ItemSheet.Row>\)](#) ,  
[Sheet<int, ItemSheet.Row>.CopyTo\(KeyValuePair<int, ItemSheet.Row>\[\], int\)](#) ,  
[Sheet<int, ItemSheet.Row>.Remove\(KeyValuePair<int, ItemSheet.Row>\)](#) ,  
[Sheet<int, ItemSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## ItemSheet()

```
public ItemSheet()
```

# Class ItemSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemSheet.Row : ItemSheet.RowBase
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [ItemSheet.RowBase](#) ← ItemSheet.Row

## Derived

[ConsumableItemSheet.Row](#), [CostumeItemSheet.Row](#), [EquipmentItemSheet.Row](#),  
[MaterialItemSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Row()

```
public Row()
```

### Row(Dictionary)

```
public Row(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

# Properties

## ElementType

```
public ElementType ElementType { get; }
```

Property Value

[ElementType](#)

## Grade

```
public int Grade { get; }
```

Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## ItemSubType

```
public ItemSubType ItemSubType { get; protected set; }
```

Property Value

[ItemSubType](#)

## ItemType

```
public virtual ItemType ItemType { get; }
```

Property Value

[ItemType](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Deserialize(Dictionary)

```
public static ItemSheet.Row Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[ItemSheet.Row](#)

### Serialize()

```
public override IValue Serialize()
```

Returns

IValue

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList<string>](#)

# Class ItemSheet.RowBase

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class ItemSheet.RowBase : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← ItemSheet.RowBase

## Derived

[ItemSheet.Row](#)

## Inherited Members

[SheetRow<int>.Key](#) , [SheetRow<int>.Set\(IReadOnlyList<string>\)](#) ,  
[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Methods

### Serialize()

```
public abstract IValue Serialize()
```

## Returns

IValue

# Class ItemTypeCollectQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemTypeCollectQuestSheet : Sheet<int, ItemTypeCollectQuestSheet.Row>,
IDictionary<int, ItemTypeCollectQuestSheet.Row>, ICollection<KeyValuePair<int,
ItemTypeCollectQuestSheet.Row>>, IEnumerable<KeyValuePair<int,
ItemTypeCollectQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, ItemTypeCollectQuestSheet.Row>](#) ← [ItemTypeCollectQuestSheet](#)

## Implements

[IDictionary<int, ItemTypeCollectQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, ItemTypeCollectQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, ItemTypeCollectQuestSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, ItemTypeCollectQuestSheet.Row>.Name](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.First](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Last](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Keys](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Values](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Count](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.TryGetValue\(int, out ItemTypeCollectQuestSheet.Row, bool\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.AddRow\(int, ItemTypeCollectQuestSheet.Row\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Add\(int, ItemTypeCollectQuestSheet.Row\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, ItemTypeCollectQuestSheet.Row>.Remove\(int\)](#) ,

```
Sheet<int, ItemTypeCollectQuestSheet.Row>.TryGetValue(int, out
ItemTypeCollectQuestSheet.Row),
Sheet<int, ItemTypeCollectQuestSheet.Row>.Add(KeyValuePair<int,
ItemTypeCollectQuestSheet.Row>),
Sheet<int, ItemTypeCollectQuestSheet.Row>.Clear(),
Sheet<int, ItemTypeCollectQuestSheet.Row>.Contains(KeyValuePair<int,
ItemTypeCollectQuestSheet.Row>),
Sheet<int, ItemTypeCollectQuestSheet.Row>.CopyTo(KeyValuePair<int,
ItemTypeCollectQuestSheet.Row>[], int),
Sheet<int, ItemTypeCollectQuestSheet.Row>.Remove(KeyValuePair<int,
ItemTypeCollectQuestSheet.Row>),
Sheet<int, ItemTypeCollectQuestSheet.Row>.Serialize(), object.Equals(object)✉,
object.Equals(object, object)✉, object.GetHashCode()✉, object.GetType()✉,
object.MemberwiseClone()✉, object.ReferenceEquals(object, object)✉, object.ToString()✉
```

## Constructors

### ItemTypeCollectQuestSheet()

```
public ItemTypeCollectQuestSheet()
```

# Class ItemTypeCollectQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class ItemTypeCollectQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← ItemTypeCollectQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Geoal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ItemType

```
public ItemType ItemType { get; }
```

## Property Value

[ItemType](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class MaterialItemSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MaterialItemSheet : Sheet<int, MaterialItemSheet.Row>, IDictionary<int,
MaterialItemSheet.Row>, ICollection<KeyValuePair<int, MaterialItemSheet.Row>>,
IEnumerable<KeyValuePair<int, MaterialItemSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, MaterialItemSheet.Row>](#) ← [MaterialItemSheet](#)

## Implements

[IDictionary<int, MaterialItemSheet.Row>](#),  
[ICollection<KeyValuePair<int, MaterialItemSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, MaterialItemSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, MaterialItemSheet.Row>.Name](#) ,  
[Sheet<int, MaterialItemSheet.Row>.OrderedList](#) ,  
[Sheet<int, MaterialItemSheet.Row>.First](#) , [Sheet<int, MaterialItemSheet.Row>.Last](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Keys](#) , [Sheet<int, MaterialItemSheet.Row>.Values](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Count](#) ,  
[Sheet<int, MaterialItemSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, MaterialItemSheet.Row>.this\[int\]](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.TryGetValue\(int, out MaterialItemSheet.Row, bool\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.AddRow\(int, MaterialItemSheet.Row\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Add\(int, MaterialItemSheet.Row\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.TryGetValue\(int, out MaterialItemSheet.Row\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Add\(KeyValuePair<int, MaterialItemSheet.Row>\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, MaterialItemSheet.Row>.Contains\(KeyValuePair<int, MaterialItemSheet.Row>\)](#) ,

```
Sheet<int, MaterialItemSheet.Row>.CopyTo(KeyValuePair<int, MaterialItemSheet.Row>[],
int) ,
Sheet<int, MaterialItemSheet.Row>.Remove(KeyValuePair<int, MaterialItemSheet.Row>) ,
Sheet<int, MaterialItemSheet.Row>.Serialize() , object.Equals(object)❑ ,
object.Equals(object, object)❑ , object.GetHashCode()❑ , object.GetType()❑ ,
object.MemberwiseClone()❑ , object.ReferenceEquals(object, object)❑ , object.ToString()❑
```

## Extension Methods

```
MaterialItemSheetExtensions.ValidateFromAction(MaterialItemSheet,
List<EquipmentItemSubRecipeSheet.MaterialInfo>, Dictionary<int, int>, string)
```

## Constructors

### MaterialItemSheet()

```
public MaterialItemSheet()
```

# Class MaterialItemSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MaterialItemSheet.Row : ItemSheet.Row, ISerializable
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [ItemSheet.RowBase](#) ← [ItemSheet.Row](#) ←  
MaterialItemSheet.Row

## Implements

[ISerializable](#)

## Inherited Members

[ItemSheet.Row.Key](#) , [ItemSheet.Row.Id](#) , [ItemSheet.Row.ItemSubType](#) ,  
[ItemSheet.Row.Grade](#) , [ItemSheet.Row.ElementalType](#) , [ItemSheet.Row.Serialize\(\)](#) ,  
[ItemSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Row()

```
public Row()
```

### Row(Dictionary)

```
public Row(Dictionary serialized)
```

## Parameters

serialized Dictionary

# Row(SerializationInfo, StreamingContext)

```
protected Row(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## Properties

### ItemId

```
public HashDigest<SHA256> ItemId { get; }
```

#### Property Value

HashDigest<[SHA256](#)>

### ItemType

```
public override ItemType ItemType { get; }
```

#### Property Value

[ItemType](#)

## Methods

### GetObjectData(SerializationInfo, StreamingContext)

Populates a [SerializationInfo](#) with the data needed to serialize the target object.

```
public void GetObjectData(SerializationInfo info, StreamingContext context)
```

## Parameters

**info** [SerializationInfo](#)

The [SerializationInfo](#) to populate with data.

**context** [StreamingContext](#)

The destination (see [StreamingContext](#)) for this serialization.

## Exceptions

[SecurityException](#)

The caller does not have the required permission.

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

**fields** [IReadOnlyList](#)<[string](#)>

# Class MimisbrunnrSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MimisbrunnrSheet : Sheet<int, MimisbrunnrSheet.Row>, IDictionary<int, MimisbrunnrSheet.Row>, ICollection<KeyValuePair<int, MimisbrunnrSheet.Row>>, IEnumerable<KeyValuePair<int, MimisbrunnrSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, MimisbrunnrSheet.Row>](#) ← [MimisbrunnrSheet](#)

## Implements

[IDictionary<int, MimisbrunnrSheet.Row>](#),  
[ICollection<KeyValuePair<int, MimisbrunnrSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, MimisbrunnrSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, MimisbrunnrSheet.Row>.Name](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.OrderedList](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.First](#) , [Sheet<int, MimisbrunnrSheet.Row>.Last](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Keys](#) , [Sheet<int, MimisbrunnrSheet.Row>.Values](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Count](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.this\[int\]](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.TryGetValue\(int, out MimisbrunnrSheet.Row, bool\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.AddRow\(int, MimisbrunnrSheet.Row\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Add\(int, MimisbrunnrSheet.Row\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.TryGetValue\(int, out MimisbrunnrSheet.Row\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Add\(KeyValuePair<int, MimisbrunnrSheet.Row>\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, MimisbrunnrSheet.Row>.Contains\(KeyValuePair<int, MimisbrunnrSheet.Row>\)](#) ,

```
Sheet<int, MimisbrunnrSheet.Row>.CopyTo(KeyValuePair<int, MimisbrunnrSheet.Row>[],
int) ,
Sheet<int, MimisbrunnrSheet.Row>.Remove(KeyValuePair<int, MimisbrunnrSheet.Row>) ,
Sheet<int, MimisbrunnrSheet.Row>.Serialize() , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString() ,
```

## Constructors

### MimisbrunnrSheet()

```
public MimisbrunnrSheet()
```

# Class MimisbrunnrSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MimisbrunnrSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← MimisbrunnrSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ElementalTypes

```
public List<ElementalType> ElementalTypes { get; }
```

Property Value

[List<ElementalType>](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

# Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class MonsterCollectionRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionRewardSheet : Sheet<int,
MonsterCollectionRewardSheet.Row>, IDictionary<int,
MonsterCollectionRewardSheet.Row>, ICollection<KeyValuePair<int,
MonsterCollectionRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
MonsterCollectionRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, MonsterCollectionRewardSheet.Row>](#) ←

MonsterCollectionRewardSheet

## Implements

[IDictionary<int, MonsterCollectionRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, MonsterCollectionRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, MonsterCollectionRewardSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, MonsterCollectionRewardSheet.Row>.Name](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.First](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.Last](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.Keys](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.Values](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.Count](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.TryGetValue\(int, out](#)  
[MonsterCollectionRewardSheet.Row, bool\)](#) ,  
[Sheet<int, MonsterCollectionRewardSheet.Row>.AddRow\(int,](#)  
[MonsterCollectionRewardSheet.Row\)](#) ,

```
Sheet<int, MonsterCollectionRewardSheet.Row>.Add(int,
MonsterCollectionRewardSheet.Row),
Sheet<int, MonsterCollectionRewardSheet.Row>.ContainsKey(int),
Sheet<int, MonsterCollectionRewardSheet.Row>.Remove(int),
Sheet<int, MonsterCollectionRewardSheet.Row>.TryGetValue(int, out
MonsterCollectionRewardSheet.Row),
Sheet<int, MonsterCollectionRewardSheet.Row>.Add(KeyValuePair<int,
MonsterCollectionRewardSheet.Row>),
Sheet<int, MonsterCollectionRewardSheet.Row>.Clear(),
Sheet<int, MonsterCollectionRewardSheet.Row>.Contains(KeyValuePair<int,
MonsterCollectionRewardSheet.Row>),
Sheet<int, MonsterCollectionRewardSheet.Row>.CopyTo(KeyValuePair<int,
MonsterCollectionRewardSheet.Row>[], int),
Sheet<int, MonsterCollectionRewardSheet.Row>.Remove(KeyValuePair<int,
MonsterCollectionRewardSheet.Row>),
Sheet<int, MonsterCollectionRewardSheet.Row>.Serialize(), object.Equals(object)✉,
object.Equals(object, object)✉, object.GetHashCode()✉, object.GetType()✉,
object.MemberwiseClone()✉, object.ReferenceEquals(object, object)✉, object.ToString()✉
```

## Constructors

### MonsterCollectionRewardSheet()

```
public MonsterCollectionRewardSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, MonsterCollectionRewardSheet.Row value)
```

## Parameters

key [int✉](#)

value [MonsterCollectionRewardSheet.Row](#)

# Class MonsterCollectionRewardSheet.RewardInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class MonsterCollectionRewardSheet.RewardInfo
```

## Inheritance

[object](#) ← MonsterCollectionRewardSheet.RewardInfo

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RewardInfo(Dictionary)

```
public RewardInfo(Dictionary dictionary)
```

#### Parameters

**dictionary** Dictionary

### RewardInfo(int, int)

```
public RewardInfo(int itemId, int quantity)
```

#### Parameters

**itemId** [int](#)

**quantity** [int](#)

## RewardInfo(params string[])

```
public RewardInfo(params string[] fields)
```

### Parameters

fields [string](#)[]

## Fields

### ItemId

```
public readonly int ItemId
```

### Field Value

[int](#)

### Quantity

```
public readonly int Quantity
```

### Field Value

[int](#)

## Methods

### Equals(RewardInfo)

```
protected bool Equals(MonsterCollectionRewardSheet.RewardInfo other)
```

Parameters

other [MonsterCollectionRewardSheet.RewardInfo](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class MonsterCollectionRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← MonsterCollectionRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

### MonsterCollectionLevel

```
public int MonsterCollectionLevel { get; }
```

### Property Value

[int](#)

# Rewards

```
public List<MonsterCollectionRewardSheet.RewardInfo> Rewards { get; }
```

Property Value

[List](#)<[MonsterCollectionRewardSheet](#).[RewardInfo](#)>

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class MonsterCollectionSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionSheet : Sheet<int, MonsterCollectionSheet.Row>,
IDictionary<int, MonsterCollectionSheet.Row>, ICollection<KeyValuePair<int,
MonsterCollectionSheet.Row>>, IEnumerable<KeyValuePair<int,
MonsterCollectionSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, MonsterCollectionSheet.Row>](#) ← [MonsterCollectionSheet](#)

## Implements

[IDictionary<int, MonsterCollectionSheet.Row>](#),  
[ICollection<KeyValuePair<int, MonsterCollectionSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, MonsterCollectionSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, MonsterCollectionSheet.Row>.Name](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.OrderedList](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.First](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Last](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Keys](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Values](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Count](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.this\[int\]](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.TryGetValue\(int, out MonsterCollectionSheet.Row, bool\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.AddRow\(int, MonsterCollectionSheet.Row\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Add\(int, MonsterCollectionSheet.Row\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, MonsterCollectionSheet.Row>.TryGetValue\(int, out MonsterCollectionSheet.Row\)](#)  
,

[Sheet<int, MonsterCollectionSheet.Row>.Add\(KeyValuePair<int, MonsterCollectionSheet.Row>\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Contains\(KeyValuePair<int, MonsterCollectionSheet.Row>\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.CopyTo\(KeyValuePair<int, MonsterCollectionSheet.Row>\[\], int\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Remove\(KeyValuePair<int, MonsterCollectionSheet.Row>\)](#) ,  
[Sheet<int, MonsterCollectionSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### MonsterCollectionSheet()

```
public MonsterCollectionSheet()
```

# Class MonsterCollectionSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterCollectionSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← MonsterCollectionSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

### Level

```
public int Level { get; }
```

#### Property Value

[int](#)

## RequiredGold

```
public int RequiredGold { get; }
```

Property Value

[int](#)

## RewardId

```
public int RewardId { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class MonsterQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterQuestSheet : Sheet<int, MonsterQuestSheet.Row>, IDictionary<int, 
MonsterQuestSheet.Row>, ICollection<KeyValuePair<int, MonsterQuestSheet.Row>>, 
IEnumerable<KeyValuePair<int, MonsterQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, MonsterQuestSheet.Row>](#) ← [MonsterQuestSheet](#)

## Implements

[IDictionary<int, MonsterQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, MonsterQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, MonsterQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, MonsterQuestSheet.Row>.Name](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.First](#) , [Sheet<int, MonsterQuestSheet.Row>.Last](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Keys](#) , [Sheet<int, MonsterQuestSheet.Row>.Values](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Count](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.TryGetValue\(int, out MonsterQuestSheet.Row, bool\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.AddRow\(int, MonsterQuestSheet.Row\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Add\(int, MonsterQuestSheet.Row\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.TryGetValue\(int, out MonsterQuestSheet.Row\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Add\(KeyValuePair<int, MonsterQuestSheet.Row>\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, MonsterQuestSheet.Row>.Contains\(KeyValuePair<int, MonsterQuestSheet.Row>\)](#) ,

```
Sheet<int, MonsterQuestSheet.Row>.CopyTo(KeyValuePair<int, MonsterQuestSheet.Row>
[], int) ,
Sheet<int, MonsterQuestSheet.Row>.Remove(KeyValuePair<int,
MonsterQuestSheet.Row>),
Sheet<int, MonsterQuestSheet.Row>.Serialize() , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Constructors

### MonsterQuestSheet()

```
public MonsterQuestSheet()
```

# Class MonsterQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class MonsterQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← MonsterQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### MonsterId

```
public int MonsterId { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class PetCostSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetCostSheet : Sheet<int, PetCostSheet.Row>, IDictionary<int,
PetCostSheet.Row>, ICollection<KeyValuePair<int, PetCostSheet.Row>>,
IEnumerable<KeyValuePair<int, PetCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, PetCostSheet.Row>](#) ← PetCostSheet

## Implements

[IDictionary<int, PetCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, PetCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, PetCostSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, PetCostSheet.Row>.Name](#) , [Sheet<int, PetCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, PetCostSheet.Row>.First](#) , [Sheet<int, PetCostSheet.Row>.Last](#) ,  
[Sheet<int, PetCostSheet.Row>.Keys](#) , [Sheet<int, PetCostSheet.Row>.Values](#) ,  
[Sheet<int, PetCostSheet.Row>.Count](#) , [Sheet<int, PetCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, PetCostSheet.Row>.this\[int\]](#) , [Sheet<int, PetCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, PetCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, PetCostSheet.Row>.TryGetValue\(int, out PetCostSheet.Row, bool\)](#) ,  
[Sheet<int, PetCostSheet.Row>.AddRow\(int, PetCostSheet.Row\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Add\(int, PetCostSheet.Row\)](#) ,  
[Sheet<int, PetCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, PetCostSheet.Row>.TryGetValue\(int, out PetCostSheet.Row\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Add\(KeyValuePair<int, PetCostSheet.Row>\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Contains\(KeyValuePair<int, PetCostSheet.Row>\)](#) ,  
[Sheet<int, PetCostSheet.Row>.CopyTo\(KeyValuePair<int, PetCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Remove\(KeyValuePair<int, PetCostSheet.Row>\)](#) ,  
[Sheet<int, PetCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### PetCostSheet()

```
public PetCostSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, PetCostSheet.Row value)
```

## Parameters

key [int](#)

value [PetCostSheet.Row](#)

# Class PetCostSheet.PetCostData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetCostSheet.PetCostData
```

## Inheritance

[object](#) ← PetCostSheet.PetCostData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### PetCostData(int, int, int)

```
public PetCostData(int level, int soulStoneQuantity, int ncgQuantity)
```

## Parameters

level [int](#)

soulStoneQuantity [int](#)

ncgQuantity [int](#)

## Properties

### Level

```
public int Level { get; }
```

Property Value

[int ↗](#)

## NcgQuantity

```
public int NcgQuantity { get; }
```

Property Value

[int ↗](#)

## SoulStoneQuantity

```
public int SoulStoneQuantity { get; }
```

Property Value

[int ↗](#)

# Class PetCostSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← PetCostSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Cost

```
public List<PetCostSheet.PetCostData> Cost { get; }
```

### Property Value

[List<PetCostSheet.PetCostData>](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## PetId

```
public int PetId { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

### TryGetCost(int, out PetCostData)

```
public bool TryGetCost(int level, out PetCostSheet.PetCostData costData)
```

Parameters

level [int](#)

costData [PetCostSheet.PetCostData](#)

Returns

[bool](#)

# Class PetSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetSheet : Sheet<int, PetSheet.Row>, IDictionary<int, PetSheet.Row>,
ICollection<KeyValuePair<int, PetSheet.Row>>, IEnumerable<KeyValuePair<int,
PetSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, PetSheet.Row>](#) ← [PetSheet](#)

## Implements

[IDictionary<int, PetSheet.Row>](#), [ICollection<KeyValuePair<int, PetSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, PetSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, PetSheet.Row>.Name](#) , [Sheet<int, PetSheet.Row>.OrderedList](#) ,  
[Sheet<int, PetSheet.Row>.First](#) , [Sheet<int, PetSheet.Row>.Last](#) ,  
[Sheet<int, PetSheet.Row>.Keys](#) , [Sheet<int, PetSheet.Row>.Values](#) ,  
[Sheet<int, PetSheet.Row>.Count](#) , [Sheet<int, PetSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, PetSheet.Row>.this\[int\]](#) , [Sheet<int, PetSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, PetSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, PetSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, PetSheet.Row>.TryGetValue\(int, out PetSheet.Row, bool\)](#) ,  
[Sheet<int, PetSheet.Row>.AddRow\(int, PetSheet.Row\)](#) ,  
[Sheet<int, PetSheet.Row>.Add\(int, PetSheet.Row\)](#) ,  
[Sheet<int, PetSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, PetSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, PetSheet.Row>.TryGetValue\(int, out PetSheet.Row\)](#) ,  
[Sheet<int, PetSheet.Row>.Add\(KeyValuePair<int, PetSheet.Row>\)](#) ,  
[Sheet<int, PetSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, PetSheet.Row>.Contains\(KeyValuePair<int, PetSheet.Row>\)](#) ,  
[Sheet<int, PetSheet.Row>.CopyTo\(KeyValuePair<int, PetSheet.Row>\[\], int\)](#) ,  
[Sheet<int, PetSheet.Row>.Remove\(KeyValuePair<int, PetSheet.Row>\)](#) ,  
[Sheet<int, PetSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## PetSheet()

```
public PetSheet()
```

# Class PetSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class PetSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← PetSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Grade

```
public int Grade { get; }
```

Property Value

[int](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## SoulStoneTicker

```
public string SoulStoneTicker { get; }
```

Property Value

[string](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class QuestItemRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestItemRewardSheet : Sheet<int, QuestItemRewardSheet.Row>,
IDictionary<int, QuestItemRewardSheet.Row>, ICollection<KeyValuePair<int,
QuestItemRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
QuestItemRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, QuestItemRewardSheet.Row>](#) ← [QuestItemRewardSheet](#)

## Implements

[IDictionary<int, QuestItemRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, QuestItemRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, QuestItemRewardSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, QuestItemRewardSheet.Row>.Name](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.First](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Last](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Keys](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Values](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Count](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.TryGetValue\(int, out QuestItemRewardSheet.Row, bool\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.AddRow\(int, QuestItemRewardSheet.Row\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Add\(int, QuestItemRewardSheet.Row\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, QuestItemRewardSheet.Row>.TryGetValue\(int, out QuestItemRewardSheet.Row\)](#)  
,

[Sheet<int, QuestItemRewardSheet.Row>.Add\(KeyValuePair<int, QuestItemRewardSheet.Row>\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Contains\(KeyValuePair<int, QuestItemRewardSheet.Row>\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.CopyTo\(KeyValuePair<int, QuestItemRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Remove\(KeyValuePair<int, QuestItemRewardSheet.Row>\)](#) ,  
[Sheet<int, QuestItemRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### QuestItemRewardSheet()

```
public QuestItemRewardSheet()
```

# Class QuestItemRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestItemRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← QuestItemRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Count

```
public int Count { get; }
```

### Property Value

[int](#)

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

## ItemId

```
public int ItemId { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class QuestRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestRewardSheet : Sheet<int, QuestRewardSheet.Row>, IDictionary<int, QuestRewardSheet.Row>, ICollection<KeyValuePair<int, QuestRewardSheet.Row>>, IEnumerable<KeyValuePair<int, QuestRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, QuestRewardSheet.Row>](#) ← [QuestRewardSheet](#)

## Implements

[IDictionary<int, QuestRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, QuestRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, QuestRewardSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, QuestRewardSheet.Row>.Name](#) ,  
[Sheet<int, QuestRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, QuestRewardSheet.Row>.First](#) , [Sheet<int, QuestRewardSheet.Row>.Last](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Keys](#) , [Sheet<int, QuestRewardSheet.Row>.Values](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Count](#) ,  
[Sheet<int, QuestRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, QuestRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.TryGetValue\(int, out QuestRewardSheet.Row, bool\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.AddRow\(int, QuestRewardSheet.Row\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Add\(int, QuestRewardSheet.Row\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.TryGetValue\(int, out QuestRewardSheet.Row\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Add\(KeyValuePair<int, QuestRewardSheet.Row>\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, QuestRewardSheet.Row>.Contains\(KeyValuePair<int, QuestRewardSheet.Row>\)](#) ,

```
Sheet<int, QuestRewardSheet.Row>.CopyTo\(KeyValuePair<int, QuestRewardSheet.Row>\[\], int\),  
Sheet<int, QuestRewardSheet.Row>.Remove\\(KeyValuePair<int, QuestRewardSheet.Row>\\),  
,  
Sheet<int, QuestRewardSheet.Row>.Serialize\\(\\), object.Equals\\(object\\),  
object.Equals\\(object, object\\), object.GetHashCode\\(\\), object.GetType\\(\\),  
object.MemberwiseClone\\(\\), object.ReferenceEquals\\(object, object\\), object.ToString\\(\\)
```

## Constructors

### QuestRewardSheet()

```
public QuestRewardSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, QuestRewardSheet.Row value)
```

## Parameters

key [int](#)

value [QuestRewardSheet.Row](#)

# Class QuestRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← QuestRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## Rewards

```
public List<int> Rewards { get; }
```

Property Value

[List](#)<[int](#)>

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class QuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestSheet : Sheet<int, QuestSheet.Row>, IDictionary<int,
QuestSheet.Row>, ICollection<KeyValuePair<int, QuestSheet.Row>>,
IEnumerable<KeyValuePair<int, QuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, QuestSheet.Row>](#) ← QuestSheet

## Implements

[IDictionary<int, QuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, QuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, QuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, QuestSheet.Row>.Name](#) , [Sheet<int, QuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, QuestSheet.Row>.First](#) , [Sheet<int, QuestSheet.Row>.Last](#) ,  
[Sheet<int, QuestSheet.Row>.Keys](#) , [Sheet<int, QuestSheet.Row>.Values](#) ,  
[Sheet<int, QuestSheet.Row>.Count](#) , [Sheet<int, QuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, QuestSheet.Row>.this\[int\]](#) , [Sheet<int, QuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, QuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, QuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, QuestSheet.Row>.TryGetValue\(int, out QuestSheet.Row, bool\)](#) ,  
[Sheet<int, QuestSheet.Row>.AddRow\(int, QuestSheet.Row\)](#) ,  
[Sheet<int, QuestSheet.Row>.Add\(int, QuestSheet.Row\)](#) ,  
[Sheet<int, QuestSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, QuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, QuestSheet.Row>.TryGetValue\(int, out QuestSheet.Row\)](#) ,  
[Sheet<int, QuestSheet.Row>.Add\(KeyValuePair<int, QuestSheet.Row>\)](#) ,  
[Sheet<int, QuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, QuestSheet.Row>.Contains\(KeyValuePair<int, QuestSheet.Row>\)](#) ,  
[Sheet<int, QuestSheet.Row>.CopyTo\(KeyValuePair<int, QuestSheet.Row>\[\], int\)](#) ,  
[Sheet<int, QuestSheet.Row>.Remove\(KeyValuePair<int, QuestSheet.Row>\)](#) ,  
[Sheet<int, QuestSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## QuestSheet()

```
public QuestSheet()
```

# Class QuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class QuestSheet.Row : SheetRow<int>, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < QuestSheet.Row

## Implements

[IState](#)

## Derived

[CollectQuestSheet.Row](#), [CombinationEquipmentQuestSheet.Row](#),  
[CombinationQuestSheet.Row](#), [GeneralQuestSheet.Row](#), [GoldQuestSheet.Row](#),  
[ItemEnhancementQuestSheet.Row](#), [ItemGradeQuestSheet.Row](#),  
[ItemTypeCollectQuestSheet.Row](#), [MonsterQuestSheet.Row](#), [TradeQuestSheet.Row](#),  
[WorldQuestSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Properties

## Goal

```
public int Goal { get; }
```

## Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## QuestRewardId

```
public int QuestRewardId { get; }
```

Property Value

[int](#)

## Methods

### Deserialize(Dictionary)

```
public static QuestSheet.Row Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[QuestSheet.Row](#)

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList<string>](#)

# Class RaidSimulatorSheets

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RaidSimulatorSheets : SimulatorSheets
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← [SimulatorSheets](#) ← RaidSimulatorSheets

## Inherited Members

[SimulatorSheets.RuneOptionSheet](#) , [SimulatorSheets.RuneListSheet](#) ,  
[SimulatorSheets.RuneLevelBonusSheet](#) , [SimulatorSheetsV1.MaterialItemSheet](#) ,  
[SimulatorSheetsV1.SkillSheet](#) , [SimulatorSheetsV1.SkillBuffSheet](#) ,  
[SimulatorSheetsV1.SkillActionBuffSheet](#) , [SimulatorSheetsV1.ActionBuffSheet](#) ,  
[SimulatorSheetsV1.StatBuffSheet](#) , [SimulatorSheetsV1.CharacterSheet](#) ,  
[SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RaidSimulatorSheets(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet,  
WorldBossCharacterSheet,  
WorldBossActionPatternSheet,  
WorldBossBattleRewardSheet, RuneWeightSheet,  
RuneSheet, RuneOptionSheet, RuneListSheet,  
RuneLevelBonusSheet)

```
public RaidSimulatorSheets(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,
```

```
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, WorldBossCharacterSheet
worldBossCharacterSheet, WorldBossActionPatternSheet worldBossActionPatternSheet,
WorldBossBattleRewardSheet worldBossBattleRewardSheet, RuneWeightSheet
runeWeightSheet, RuneSheet runeSheet, RuneOptionSheet runeOptionSheet, RuneListSheet
runeListSheet, RuneLevelBonusSheet runeLevelBonusSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

worldBossCharacterSheet [WorldBossCharacterSheet](#)

worldBossActionPatternSheet [WorldBossActionPatternSheet](#)

worldBossBattleRewardSheet [WorldBossBattleRewardSheet](#)

runeWeightSheet [RuneWeightSheet](#)

runeSheet [RuneSheet](#)

runeOptionSheet [RuneOptionSheet](#)

runeListSheet [RuneListSheet](#)

runeLevelBonusSheet [RuneLevelBonusSheet](#)

## Properties

## RuneSheet

```
public RuneSheet RuneSheet { get; }
```

Property Value

[RuneSheet](#)

## RuneWeightSheet

```
public RuneWeightSheet RuneWeightSheet { get; }
```

Property Value

[RuneWeightSheet](#)

## WorldBossActionPatternSheet

```
public WorldBossActionPatternSheet WorldBossActionPatternSheet { get; }
```

Property Value

[WorldBossActionPatternSheet](#)

## WorldBossBattleRewardSheet

```
public WorldBossBattleRewardSheet WorldBossBattleRewardSheet { get; }
```

Property Value

[WorldBossBattleRewardSheet](#)

## WorldBossCharacterSheet

```
public WorldBossCharacterSheet WorldBossCharacterSheet { get; }
```

Property Value

[WorldBossCharacterSheet](#)

# Class RaidSimulatorSheetsV1

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RaidSimulatorSheetsV1 : SimulatorSheetsV1
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← RaidSimulatorSheetsV1

## Inherited Members

[SimulatorSheetsV1.MaterialItemSheet](#) , [SimulatorSheetsV1.SkillSheet](#) ,  
[SimulatorSheetsV1.SkillBuffSheet](#) , [SimulatorSheetsV1.SkillActionBuffSheet](#) ,  
[SimulatorSheetsV1.ActionBuffSheet](#) , [SimulatorSheetsV1.StatBuffSheet](#) ,  
[SimulatorSheetsV1.CharacterSheet](#) , [SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RaidSimulatorSheetsV1(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet,  
WorldBossCharacterSheet,  
WorldBossActionPatternSheet,  
WorldBossBattleRewardSheet, RuneWeightSheet,  
RuneSheet)

```
public RaidSimulatorSheetsV1(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, WorldBossCharacterSheet  
worldBossCharacterSheet, WorldBossActionPatternSheet worldBossActionPatternSheet,
```

```
WorldBossBattleRewardSheet worldBossBattleRewardSheet, RuneWeightSheet  
runeWeightSheet, RuneSheet runeSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

worldBossCharacterSheet [WorldBossCharacterSheet](#)

worldBossActionPatternSheet [WorldBossActionPatternSheet](#)

worldBossBattleRewardSheet [WorldBossBattleRewardSheet](#)

runeWeightSheet [RuneWeightSheet](#)

runeSheet [RuneSheet](#)

## Properties

### RuneSheet

```
public RuneSheet RuneSheet { get; }
```

## Property Value

[RuneSheet](#)

## RuneWeightSheet

```
public RuneWeightSheet RuneWeightSheet { get; }
```

Property Value

[RuneWeightSheet](#)

## WorldBossActionPatternSheet

```
public WorldBossActionPatternSheet WorldBossActionPatternSheet { get; }
```

Property Value

[WorldBossActionPatternSheet](#)

## WorldBossBattleRewardSheet

```
public WorldBossBattleRewardSheet WorldBossBattleRewardSheet { get; }
```

Property Value

[WorldBossBattleRewardSheet](#)

## WorldBossCharacterSheet

```
public WorldBossCharacterSheet WorldBossCharacterSheet { get; }
```

Property Value

[WorldBossCharacterSheet](#)

# Class RankingSimulatorSheets

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RankingSimulatorSheets : SimulatorSheets
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← [SimulatorSheets](#) ← RankingSimulatorSheets

## Inherited Members

[SimulatorSheets.RuneOptionSheet](#) , [SimulatorSheets.RuneListSheet](#) ,  
[SimulatorSheets.RuneLevelBonusSheet](#) , [SimulatorSheetsV1.MaterialItemSheet](#) ,  
[SimulatorSheetsV1.SkillSheet](#) , [SimulatorSheetsV1.SkillBuffSheet](#) ,  
[SimulatorSheetsV1.SkillActionBuffSheet](#) , [SimulatorSheetsV1.ActionBuffSheet](#) ,  
[SimulatorSheetsV1.StatBuffSheet](#) , [SimulatorSheetsV1.CharacterSheet](#) ,  
[SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RankingSimulatorSheets(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet,  
WeeklyArenaRewardSheet, RuneOptionSheet,  
RuneListSheet, RuneLevelBonusSheet)

```
public RankingSimulatorSheets(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, WeeklyArenaRewardSheet
```

```
weeklyArenaRewardSheet, RuneOptionSheet runeOptionSheet, RuneListSheet  
runeListSheet, RuneLevelBonusSheet runeLevelBonusSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

runeOptionSheet [RuneOptionSheet](#)

runeListSheet [RuneListSheet](#)

runeLevelBonusSheet [RuneLevelBonusSheet](#)

## Fields

### WeeklyArenaRewardSheet

```
public readonly WeeklyArenaRewardSheet WeeklyArenaRewardSheet
```

## Field Value

[WeeklyArenaRewardSheet](#)

# Class RankingSimulatorSheetsV1

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RankingSimulatorSheetsV1 : SimulatorSheetsV1
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← RankingSimulatorSheetsV1

## Inherited Members

[SimulatorSheetsV1.MaterialItemSheet](#) , [SimulatorSheetsV1.SkillSheet](#) ,  
[SimulatorSheetsV1.SkillBuffSheet](#) , [SimulatorSheetsV1.SkillActionBuffSheet](#) ,  
[SimulatorSheetsV1.ActionBuffSheet](#) , [SimulatorSheetsV1.StatBuffSheet](#) ,  
[SimulatorSheetsV1.CharacterSheet](#) , [SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RankingSimulatorSheetsV1(MaterialItemSheet,  
SkillSheet, SkillBuffSheet, StatBuffSheet,  
SkillActionBuffSheet, ActionBuffSheet, CharacterSheet,  
CharacterLevelSheet, EquipmentItemSetEffectSheet,  
WeeklyArenaRewardSheet)

```
public RankingSimulatorSheetsV1(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, WeeklyArenaRewardSheet  
weeklyArenaRewardSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

## Fields

### WeeklyArenaRewardSheet

```
public readonly WeeklyArenaRewardSheet WeeklyArenaRewardSheet
```

Field Value

[WeeklyArenaRewardSheet](#)

# Class RankingSimulatorSheetsV100291

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RankingSimulatorSheetsV100291 : SimulatorSheetsV100291
```

## Inheritance

[object](#) ← [SimulatorSheetsV100291](#) ← RankingSimulatorSheetsV100291

## Inherited Members

[SimulatorSheetsV100291.MaterialItemSheet](#) , [SimulatorSheetsV100291.SkillSheet](#) ,  
[SimulatorSheetsV100291.SkillBuffSheet](#) , [SimulatorSheetsV100291.BuffSheet](#) ,  
[SimulatorSheetsV100291.CharacterSheet](#) , [SimulatorSheetsV100291.CharacterLevelSheet](#) ,  
[SimulatorSheetsV100291.EquipmentItemSetEffectSheet](#) ,  
[SimulatorSheetsV100291.ToSimulatorSheetsV1\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RankingSimulatorSheetsV100291(MaterialItemSheet, SkillSheet, SkillBuffSheet, BuffSheet, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet, WeeklyArenaRewardSheet)

```
public RankingSimulatorSheetsV100291(MaterialItemSheet materialItemSheet, SkillSheet skillSheet, SkillBuffSheet skillBuffSheet, BuffSheet buffSheet, CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet, EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, WeeklyArenaRewardSheet weeklyArenaRewardSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

buffSheet [BuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

weeklyArenaRewardSheet [WeeklyArenaRewardSheet](#)

## Fields

### WeeklyArenaRewardSheet

```
public readonly WeeklyArenaRewardSheet WeeklyArenaRewardSheet
```

## Field Value

[WeeklyArenaRewardSheet](#)

## Methods

### ToRankingSimulatorSheetsV1()

```
public RankingSimulatorSheetsV1 ToRankingSimulatorSheetsV1()
```

## Returns

[RankingSimulatorSheetsV1](#)

# Class RedeemCodeListSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RedeemCodeListSheet : Sheet<int, RedeemCodeListSheet.Row>,
IDictionary<int, RedeemCodeListSheet.Row>, ICollection<KeyValuePair<int,
RedeemCodeListSheet.Row>>, IEnumerable<KeyValuePair<int, RedeemCodeListSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RedeemCodeListSheet.Row>](#) ← [RedeemCodeListSheet](#)

## Implements

[IDictionary<int, RedeemCodeListSheet.Row>](#),  
[ICollection<KeyValuePair<int, RedeemCodeListSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RedeemCodeListSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RedeemCodeListSheet.Row>.Name](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.OrderedList](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.First](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Last](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Keys](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Values](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Count](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.TryGetValue\(int, out RedeemCodeListSheet.Row, bool\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.AddRow\(int, RedeemCodeListSheet.Row\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Add\(int, RedeemCodeListSheet.Row\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RedeemCodeListSheet.Row>.TryGetValue\(int, out RedeemCodeListSheet.Row\)](#) ,

```
Sheet<int, RedeemCodeListSheet.Row>.Add(KeyValuePair<int,
RedeemCodeListSheet.Row>),
Sheet<int, RedeemCodeListSheet.Row>.Clear(),
Sheet<int, RedeemCodeListSheet.Row>.Contains(KeyValuePair<int,
RedeemCodeListSheet.Row>),
Sheet<int, RedeemCodeListSheet.Row>.CopyTo(KeyValuePair<int,
RedeemCodeListSheet.Row>[], int),
Sheet<int, RedeemCodeListSheet.Row>.Remove(KeyValuePair<int,
RedeemCodeListSheet.Row>),
Sheet<int, RedeemCodeListSheet.Row>.Serialize(), object.Equals(object) ↴ ,
object.Equals(object, object) ↴ , object.GetHashCode() ↴ , object.GetType() ↴ ,
object.MemberwiseClone() ↴ , object.ReferenceEquals(object, object) ↴ , object.ToString() ↴
```

## Constructors

### RedeemCodeListSheet()

```
public RedeemCodeListSheet()
```

# Class RedeemCodeListSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RedeemCodeListSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RedeemCodeListSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## PublicKey

```
public PublicKey PublicKey { get; }
```

Property Value

PublicKey

## PublicKeyBinary

```
public Binary PublicKeyBinary { get; }
```

Property Value

Binary

## RewardId

```
public int RewardId { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class RedeemRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RedeemRewardSheet : Sheet<int, RedeemRewardSheet.Row>, IDictionary<int, RedeemRewardSheet.Row>, ICollection<KeyValuePair<int, RedeemRewardSheet.Row>>, IEnumerable<KeyValuePair<int, RedeemRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RedeemRewardSheet.Row>](#) ← [RedeemRewardSheet](#)

## Implements

[IDictionary<int, RedeemRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, RedeemRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RedeemRewardSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RedeemRewardSheet.Row>.Name](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.First](#) , [Sheet<int, RedeemRewardSheet.Row>.Last](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Keys](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Values](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Count](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.TryGetValue\(int, out RedeemRewardSheet.Row, bool\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.AddRow\(int, RedeemRewardSheet.Row\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Add\(int, RedeemRewardSheet.Row\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.TryGetValue\(int, out RedeemRewardSheet.Row\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Add\(KeyValuePair<int, RedeemRewardSheet.Row>\)](#)

[Sheet<int, RedeemRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Contains\(KeyValuePair<int, RedeemRewardSheet.Row>\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.CopyTo\(KeyValuePair<int, RedeemRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Remove\(KeyValuePair<int, RedeemRewardSheet.Row>\)](#) ,  
[Sheet<int, RedeemRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RedeemRewardSheet()

```
public RedeemRewardSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, RedeemRewardSheet.Row value)
```

## Parameters

key [int](#)

value [RedeemRewardSheet.Row](#)

# Class RedeemRewardSheet.RewardInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RedeemRewardSheet.RewardInfo : IState
```

## Inheritance

[object](#) ← RedeemRewardSheet.RewardInfo

## Implements

[IState](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RewardInfo(Dictionary)

```
public RewardInfo(Dictionary serialized)
```

#### Parameters

serialized Dictionary

### RewardInfo(params string[])

```
public RewardInfo(params string[] fields)
```

#### Parameters

fields [string](#)[]

# Fields

## ItemId

```
public readonly int? ItemId
```

### Field Value

[int](#)?

## Quantity

```
public readonly int Quantity
```

### Field Value

[int](#)

## Type

```
public readonly RewardType Type
```

### Field Value

[RewardType](#)

# Methods

## Equals(RewardInfo)

```
protected bool Equals(RedeemRewardSheet.RewardInfo other)
```

### Parameters

other [RedeemRewardSheet.RewardInfo](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

[obj](#) [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class RedeemRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RedeemRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RedeemRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

# Rewards

```
public List<RedeemRewardSheet.RewardInfo> Rewards { get; }
```

Property Value

[List](#)<[RedeemRewardSheet.RewardInfo](#)>

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Enum RewardType

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public enum RewardType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Gold = 1

Item = 0

# Class RuneCostSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneCostSheet : Sheet<int, RuneCostSheet.Row>, IDictionary<int,
RuneCostSheet.Row>, ICollection<KeyValuePair<int, RuneCostSheet.Row>>,
IEnumerable<KeyValuePair<int, RuneCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RuneCostSheet.Row>](#) ← [RuneCostSheet](#)

## Implements

[IDictionary<int, RuneCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, RuneCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RuneCostSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RuneCostSheet.Row>.Name](#) , [Sheet<int, RuneCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, RuneCostSheet.Row>.First](#) , [Sheet<int, RuneCostSheet.Row>.Last](#) ,  
[Sheet<int, RuneCostSheet.Row>.Keys](#) , [Sheet<int, RuneCostSheet.Row>.Values](#) ,  
[Sheet<int, RuneCostSheet.Row>.Count](#) , [Sheet<int, RuneCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RuneCostSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RuneCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.TryGetValue\(int, out RuneCostSheet.Row, bool\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.AddRow\(int, RuneCostSheet.Row\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Add\(int, RuneCostSheet.Row\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.TryGetValue\(int, out RuneCostSheet.Row\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Add\(KeyValuePair<int, RuneCostSheet.Row>\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Contains\(KeyValuePair<int, RuneCostSheet.Row>\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.CopyTo\(KeyValuePair<int, RuneCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Remove\(KeyValuePair<int, RuneCostSheet.Row>\)](#) ,  
[Sheet<int, RuneCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RuneCostSheet()

```
public RuneCostSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, RuneCostSheet.Row value)
```

## Parameters

key [int](#)

value [RuneCostSheet.Row](#)

# Class RuneCostSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RuneCostSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Cost

```
public List<RuneCostSheet.RuneCostData> Cost { get; }
```

#### Property Value

[List<RuneCostSheet.RuneCostData>](#)

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

# Runeld

```
public int RuneId { get; }
```

Property Value

[int](#)

## Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

**TryGetCost(int, out RuneCostData)**

```
public bool TryGetCost(int level, out RuneCostSheet.RuneCostData costData)
```

Parameters

level [int](#)

costData [RuneCostSheet.RuneCostData](#)

Returns

[bool](#)

# Class RuneCostSheet.RuneCostData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneCostSheet.RuneCostData
```

## Inheritance

[object](#) ← RuneCostSheet.RuneCostData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RuneCostData(int, int, int, int, int)

```
public RuneCostData(int level, int runeStoneQuantity, int crystalQuantity, int
ncgQuantity, int levelUpSuccessRate)
```

## Parameters

level [int](#)

runeStoneQuantity [int](#)

crystalQuantity [int](#)

ncgQuantity [int](#)

levelUpSuccessRate [int](#)

## Properties

## CrystalQuantity

```
public int CrystalQuantity { get; }
```

Property Value

[int](#)

## Level

```
public int Level { get; }
```

Property Value

[int](#)

## LevelUpSuccessRate

```
public int LevelUpSuccessRate { get; }
```

Property Value

[int](#)

## NcgQuantity

```
public int NcgQuantity { get; }
```

Property Value

[int](#)

## RuneStoneQuantity

```
public int RuneStoneQuantity { get; }
```

Property Value

[int](#) ↗

# Class RuneOptionSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneOptionSheet : Sheet<int, RuneOptionSheet.Row>, IDictionary<int,
RuneOptionSheet.Row>, ICollection<KeyValuePair<int, RuneOptionSheet.Row>>,
IEnumerable<KeyValuePair<int, RuneOptionSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RuneOptionSheet.Row>](#) ← [RuneOptionSheet](#)

## Implements

[IDictionary<int, RuneOptionSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, RuneOptionSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, RuneOptionSheet.Row>>](#) , [IEnumerable](#) , [ISheet](#)

## Inherited Members

[Sheet<int, RuneOptionSheet.Row>.Name](#) ,  
[Sheet<int, RuneOptionSheet.Row>.OrderedList](#) , [Sheet<int, RuneOptionSheet.Row>.First](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Last](#) , [Sheet<int, RuneOptionSheet.Row>.Keys](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Values](#) , [Sheet<int, RuneOptionSheet.Row>.Count](#) ,  
[Sheet<int, RuneOptionSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RuneOptionSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.TryGetValue\(int, out RuneOptionSheet.Row, bool\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.AddRow\(int, RuneOptionSheet.Row\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Add\(int, RuneOptionSheet.Row\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.TryGetValue\(int, out RuneOptionSheet.Row\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Add\(KeyValuePair<int, RuneOptionSheet.Row>\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.Contains\(KeyValuePair<int, RuneOptionSheet.Row>\)](#) ,  
[Sheet<int, RuneOptionSheet.Row>.CopyTo\(KeyValuePair<int, RuneOptionSheet.Row>\[\], int\)](#) ,

```
Sheet<int, RuneOptionSheet.Row>.Remove\(KeyValuePair<int, RuneOptionSheet.Row>\) ,  
Sheet<int, RuneOptionSheet.Row>.Serialize\(\) , object.Equals\(object\) ,  
object.Equals\(object, object\) , object.GetHashCode\(\) , object.GetType\(\) ,  
object.MemberwiseClone\(\) , object.ReferenceEquals\(object, object\) , object.ToString\(\)
```

## Constructors

### RuneOptionSheet()

```
public RuneOptionSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, RuneOptionSheet.Row value)
```

#### Parameters

key [int](#)

value [RuneOptionSheet.Row](#)

### TryGetOptionInfo(int, int, out RuneOptionInfo)

```
public bool TryGetOptionInfo(int runeId, int level, out  
RuneOptionSheet.Row.RuneOptionInfo optionInfo)
```

#### Parameters

runeId [int](#)

level [int](#)

optionInfo [RuneOptionSheet.Row.RuneOptionInfo](#)

## Returns

[bool](#) ↗

# Class RuneOptionSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class RuneOptionSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RuneOptionSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

### LevelOptionMap

```
public Dictionary<int, RuneOptionSheet.Row.RuneOptionInfo> LevelOptionMap { get; }
```

### Property Value

[Dictionary](#)<[int](#), [RuneOptionSheet.Row.RuneOptionInfo](#)>

# Runeld

```
public int RuneId { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class RuneOptionSheet.Row.RuneOptionInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RuneOptionSheet.Row.RuneOptionInfo
```

## Inheritance

[object](#) ← RuneOptionSheet.Row.RuneOptionInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

RuneOptionInfo(int, List<(DecimalStat, OperationType)>)

```
public RuneOptionInfo(int cp, List<(DecimalStat, StatModifier.OperationType)> stats)
```

### Parameters

cp [int](#)

stats [List](#)<(DecimalStat [stat](#), StatModifier.OperationType [operationType](#))>

RuneOptionInfo(int, List<(DecimalStat, OperationType)>, int, int, int, decimal, OperationType, StatType, StatReferenceType, int)

```
public RuneOptionInfo(int cp, List<(DecimalStat, StatModifier.OperationType)> stats,  
int skillId, int skillCooldown, int skillChance, decimal skillValue,
```

```
StatModifier.OperationType skillValueType, StatType skillStatType, StatReferenceType  
statReferenceType, int buffDuration)
```

## Parameters

cp [int](#)

stats [List](#)<([DecimalStat](#) [stat](#), [StatModifier.OperationType](#) [operationType](#))>

skillId [int](#)

skillCooldown [int](#)

skillChance [int](#)

skillValue [decimal](#)

skillValueType [StatModifier.OperationType](#)

skillStatType [StatType](#)

statReferenceType [StatReferenceType](#)

buffDuration [int](#)

## Properties

### BuffDuration

```
public int BuffDuration { get; set; }
```

### Property Value

[int](#)

### Cp

```
public int Cp { get; }
```

Property Value

[int](#)

## SkillChance

```
public int SkillChance { get; set; }
```

Property Value

[int](#)

## SkillCooldown

```
public int SkillCooldown { get; set; }
```

Property Value

[int](#)

## SkillId

```
public int SkillId { get; set; }
```

Property Value

[int](#)

## SkillStatType

```
public StatType SkillStatType { get; set; }
```

Property Value

## [StatType](#)

### SkillValue

```
public decimal SkillValue { get; set; }
```

Property Value

[decimal](#)

### SkillValueType

```
public StatModifier.OperationType SkillValueType { get; set; }
```

Property Value

[StatModifier.OperationType](#)

### StatReferenceType

```
public StatReferenceType StatReferenceType { get; set; }
```

Property Value

[StatReferenceType](#)

### Stats

```
public List<(DecimalStat stat, StatModifier.OperationType operationType)> Stats {  
    get; set; }
```

Property Value

[List](#)<([DecimalStat](#) [stat](#), [StatModifier.OperationType](#) [operationType](#))>



# Class RuneSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RuneSheet : Sheet<int, RuneSheet.Row>, IDictionary<int, RuneSheet.Row>,  
ICollection<KeyValuePair<int, RuneSheet.Row>>, IEnumerable<KeyValuePair<int,  
RuneSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RuneSheet.Row>](#) ← [RuneSheet](#)

## Implements

[IDictionary<int, RuneSheet.Row>](#),  
[ICollection<KeyValuePair<int, RuneSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RuneSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RuneSheet.Row>.Name](#) , [Sheet<int, RuneSheet.Row>.OrderedList](#) ,  
[Sheet<int, RuneSheet.Row>.First](#) , [Sheet<int, RuneSheet.Row>.Last](#) ,  
[Sheet<int, RuneSheet.Row>.Keys](#) , [Sheet<int, RuneSheet.Row>.Values](#) ,  
[Sheet<int, RuneSheet.Row>.Count](#) , [Sheet<int, RuneSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RuneSheet.Row>.this\[int\]](#) , [Sheet<int, RuneSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RuneSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RuneSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RuneSheet.Row>.TryGetValue\(int, out RuneSheet.Row, bool\)](#) ,  
[Sheet<int, RuneSheet.Row>.AddRow\(int, RuneSheet.Row\)](#) ,  
[Sheet<int, RuneSheet.Row>.Add\(int, RuneSheet.Row\)](#) ,  
[Sheet<int, RuneSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, RuneSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RuneSheet.Row>.TryGetValue\(int, out RuneSheet.Row\)](#) ,  
[Sheet<int, RuneSheet.Row>.Add\(KeyValuePair<int, RuneSheet.Row>\)](#) ,  
[Sheet<int, RuneSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, RuneSheet.Row>.Contains\(KeyValuePair<int, RuneSheet.Row>\)](#) ,  
[Sheet<int, RuneSheet.Row>.CopyTo\(KeyValuePair<int, RuneSheet.Row>\[\], int\)](#) ,  
[Sheet<int, RuneSheet.Row>.Remove\(KeyValuePair<int, RuneSheet.Row>\)](#) ,  
[Sheet<int, RuneSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## RuneSheet()

```
public RuneSheet()
```

# Class RuneSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RuneSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RuneSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Id

```
public int Id
```

#### Field Value

[int](#)

### Ticker

```
public string Ticker
```

#### Field Value

[string](#)

# Properties

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class RuneWeightSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RuneWeightSheet : Sheet<int, RuneWeightSheet.Row>, IDictionary<int, RuneWeightSheet.Row>, ICollection<KeyValuePair<int, RuneWeightSheet.Row>>, IEnumerable<KeyValuePair<int, RuneWeightSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RuneWeightSheet.Row>](#) ← [RuneWeightSheet](#)

## Implements

[IDictionary<int, RuneWeightSheet.Row>](#),  
[ICollection<KeyValuePair<int, RuneWeightSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RuneWeightSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RuneWeightSheet.Row>.Name](#) ,  
[Sheet<int, RuneWeightSheet.Row>.OrderedList](#) , [Sheet<int, RuneWeightSheet.Row>.First](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Last](#) , [Sheet<int, RuneWeightSheet.Row>.Keys](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Values](#) , [Sheet<int, RuneWeightSheet.Row>.Count](#) ,  
[Sheet<int, RuneWeightSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RuneWeightSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.TryGetValue\(int, out RuneWeightSheet.Row, bool\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.AddRow\(int, RuneWeightSheet.Row\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Add\(int, RuneWeightSheet.Row\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.TryGetValue\(int, out RuneWeightSheet.Row\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Add\(KeyValuePair<int, RuneWeightSheet.Row>\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Contains\(KeyValuePair<int, RuneWeightSheet.Row>\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.CopyTo\(KeyValuePair<int, RuneWeightSheet.Row>\[\], int\)](#) ,  
[Sheet<int, RuneWeightSheet.Row>.Remove\(KeyValuePair<int, RuneWeightSheet.Row>\)](#) ,

[Sheet<int, RuneWeightSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RuneWeightSheet()

```
public RuneWeightSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, RuneWeightSheet.Row value)
```

## Parameters

key [int](#)

value [RuneWeightSheet.Row](#)

# Class RuneWeightSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RuneWeightSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RuneWeightSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BossId

```
public int BossId
```

#### Field Value

[int](#)

### Id

```
public int Id
```

#### Field Value

[int](#)

# Rank

```
public int Rank
```

## Field Value

[int](#)

# RuneInfos

```
public List<RuneWeightSheet.RuneInfo> RuneInfos
```

## Field Value

[List](#)<[RuneWeightSheet.RuneInfo](#)>

# Properties

## Key

```
public override int Key { get; }
```

## Property Value

[int](#)

# Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields  [IReadOnlyList](#)<[string](#)>

# Class RuneWeightSheet.RuneInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class RuneWeightSheet.RuneInfo
```

## Inheritance

[object](#) ← RuneWeightSheet.RuneInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RuneInfo(int, decimal)

```
public RuneInfo(int runeId, decimal weight)
```

## Parameters

runeId [int](#)

weight [decimal](#)

## Fields

### Runeld

```
public int RuneId
```

## Field Value

[int](#) ↗

## Weight

`public decimal Weight`

Field Value

[decimal](#) ↗

# Class SetEffectExtension

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public static class SetEffectExtension
```

## Inheritance

[object](#) ← SetEffectExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

GetSetEffectRows(EquipmentItemSetEffectSheet,  
IEnumerable<Equipment>)

```
public static List<EquipmentItemSetEffectSheet.Row> GetSetEffectRows(this  
EquipmentItemSetEffectSheet sheet, IEnumerable<Equipment> equipments)
```

### Parameters

sheet [EquipmentItemSetEffectSheet](#)

equipments [IEnumerable](#)<[Equipment](#)>

### Returns

[List](#)<[EquipmentItemSetEffectSheet](#).Row>

# Class SheetRowColumnException

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SheetRowColumnException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SheetRowColumnException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SheetRowColumnException(SerializationInfo, StreamingContext)

```
protected SheetRowColumnException(SerializationInfo info, StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SheetRowColumnException(string)

```
public SheetRowColumnException(string message)
```

### Parameters

message [string](#)

# Class SheetRowNotFoundException

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SheetRowNotFoundException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SheetRowNotFoundException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SheetRowNotFoundException(SerializationInfo, StreamingContext)

```
protected SheetRowNotFoundException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SheetRowNotFoundException(string, int)

```
public SheetRowNotFoundException(string sheetName, int intKey)
```

### Parameters

sheetName [string](#)

intKey [int](#)

## SheetRowNotFoundException(string, long)

```
public SheetRowNotFoundException(string sheetName, long longKey)
```

### Parameters

sheetName [string](#)

longKey [long](#)

## SheetRowNotFoundException(string, string)

```
public SheetRowNotFoundException(string sheetName, string key)
```

### Parameters

sheetName [string](#)

key [string](#)

## SheetRowNotFoundException(string, string, int)

```
public SheetRowNotFoundException(string addressesHex, string sheetName, int intKey)
```

### Parameters

addressesHex [string](#)

sheetName [string](#)

intKey [int](#)

## SheetRowNotFoundException(string, string, string)

```
public SheetRowNotFoundException(string sheetName, string condition, string value)
```

### Parameters

sheetName [string](#)

condition [string](#)

value [string](#)

## SheetRowNotFoundException(string, string, string, int)

```
public SheetRowNotFoundException(string actionType, string addressesHex, string sheetName, int intKey)
```

### Parameters

actionType [string](#)

addressesHex [string](#)

sheetName [string](#)

intKey [int](#)

# Class SheetRowValidateException

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SheetRowValidateException : Exception, ISerializable
```

## Inheritance

[object](#) ← [Exception](#) ← SheetRowValidateException

## Implements

[ISerializable](#)

## Inherited Members

[Exception.GetBaseException\(\)](#) ,  
[Exception.GetObjectData\(SerializationInfo, StreamingContext\)](#) , [Exception.GetType\(\)](#) ,  
[Exception.ToString\(\)](#) , [Exception.Data](#) , [Exception.HelpLink](#) , [Exception.HResult](#) ,  
[Exception.InnerException](#) , [Exception.Message](#) , [Exception.Source](#) ,  
[Exception.StackTrace](#) , [Exception.TargetSite](#) , [Exception.SerializeObjectState](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### SheetRowValidateException(SerializationInfo, StreamingContext)

```
protected SheetRowValidateException(SerializationInfo info,
StreamingContext context)
```

## Parameters

info [SerializationInfo](#)

context [StreamingContext](#)

## SheetRowValidateException(string)

```
public SheetRowValidateException(string message)
```

### Parameters

message [string](#)

# Class SheetRow<T>

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class SheetRow<T>
```

## Type Parameters

T

## Inheritance

[object](#) ← SheetRow<T>

## Derived

[ActionBuffSheet.Row](#), [AdventureBossContributionRewardSheet.Row](#),  
[AdventureBossFloorFirstRewardSheet.Row](#), [AdventureBossFloorPointSheet.Row](#),  
[AdventureBossFloorSheet.Row](#), [AdventureBossFloorWaveSheet.Row](#),  
[AdventureBossNcgRewardRatioSheet.Row](#), [AdventureBossSheet.Row](#),  
[AdventureBossUnlockFloorCostSheet.Row](#), [AdventureBossWantedRewardSheet.Row](#),  
[ArenaSheet.Row](#), [BuffLinkSheet.Row](#), [BuffSheet.Row](#), [CharacterLevelSheet.Row](#),  
[CharacterSheet.Row](#), [CollectionSheet.Row](#), [ConsumableItemRecipeSheet.Row](#),  
[CostumeStatSheet.Row](#), [CreateAvatarFavSheet.Row](#), [CreateAvatarItemSheet.Row](#),  
[CrystalEquipmentGrindingSheet.Row](#), [CrystalFluctuationSheet.Row](#),  
[CrystalHammerPointSheet.Row](#), [CrystalMaterialCostSheet.Row](#),  
[CrystalMonsterCollectionMultiplierSheet.Row](#), [CrystalRandomBuffSheet.Row](#),  
[CrystalStageBuffGachaSheet.Row](#), [CustomEquipmentCraftIconSheet.Row](#),  
[CustomEquipmentCraftOptionSheet.Row](#), [CustomEquipmentCraftRecipeSheet.Row](#),  
[CustomEquipmentCraftRecipeSkillSheet.Row](#),  
[CustomEquipmentCraftRelationshipSheet.Row](#), [DeBuffLimitSheet.Row](#),  
[EnemySkillSheet.Row](#), [EnhancementCostSheet.Row](#), [EquipmentItemOptionSheet.Row](#),  
[EquipmentItemRecipeSheet.Row](#), [EquipmentItemSetEffectSheet.Row](#),  
[EquipmentItemSubRecipeSheet.Row](#), [EquipmentItemSubRecipeSheetV2.Row](#),  
[EventMaterialItemRecipeSheet.Row](#), [EventScheduleSheet.Row](#), [GameConfigSheet.Row](#),  
[LoadIntoMyGaragesCostSheet.Row](#), [GrandFinaleParticipantsSheet.Row](#),  
[GrandFinaleScheduleSheet.Row](#), [ItemConfigForGradeSheet.Row](#),  
[ItemRequirementSheet.Row](#), [ItemSheet.RowBase](#), [MimisbrunnrSheet.Row](#),  
[MonsterCollectionRewardSheet.Row](#), [MonsterCollectionSheet.Row](#), [PetOptionSheet.Row](#),

[PetCostSheet.Row](#), [PetSheet.Row](#), [QuestItemRewardSheet.Row](#), [QuestRewardSheet.Row](#),  
[QuestSheet.Row](#), [RedeemCodeListSheet.Row](#), [RedeemRewardSheet.Row](#),  
[RuneLevelBonusSheet.Row](#), [RuneListSheet.Row](#), [RuneCostSheet.Row](#),  
[RuneOptionSheet.Row](#), [RuneSheet.Row](#), [RuneWeightSheet.Row](#), [SkillActionBuffSheet.Row](#),  
[SkillBuffSheet.Row](#), [SkillSheet.Row](#), [StageDialogSheet.Row](#), [StageSheet.Row](#),  
[StageWaveSheet.Row](#), [StakePolicySheet.Row](#), [StakeAchievementRewardSheet.Row](#),  
[StakeActionPointCoefficientSheet.Row](#), [StakeRegularFixedRewardSheet.Row](#),  
[StakeRegularRewardSheet.Row](#), [StatBuffSheet.Row](#), [SummonSheet.Row](#),  
[SweepRequiredCPSheet.Row](#), [UnlockCombinationSlotCostSheet.Row](#),  
[WeeklyArenaRewardSheet.Row](#), [WorldBossActionPatternSheet.Row](#),  
[WorldBossBattleRewardSheet.Row](#), [WorldBossCharacterSheet.Row](#),  
[WorldBossGlobalHpSheet.Row](#), [WorldBossKillRewardSheet.Row](#), [WorldBossListSheet.Row](#),  
[WorldBossRankRewardSheet.Row](#), [WorldBossRankingRewardSheet.Row](#), [WorldSheet.Row](#),  
[WorldUnlockSheet.Row](#)

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Properties

## Key

```
public abstract T Key { get; }
```

## Property Value

T

# Methods

## EndOfSheetInitialize()

```
public virtual void EndOfSheetInitialize()
```

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

**obj** [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Set(IReadOnlyList<string>)

```
public abstract void Set(IReadOnlyList<string> fields)
```

Parameters

**fields** [IReadOnlyList](#)<[string](#)>

## Validate()

```
public virtual void Validate()
```

# Class Sheet< TKey, TValue >

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public abstract class Sheet< TKey, TValue > : IDictionary< TKey, TValue >,
ICollection< KeyValuePair< TKey, TValue > >, IEnumerable< KeyValuePair< TKey, TValue > >,
IEnumerable, ISheet where TKey : notnull where TValue : SheetRow< TKey >, new()
```

## Type Parameters

TKey

TValue

## Inheritance

[object](#) ← Sheet< TKey, TValue >

## Implements

[IDictionary](#)< TKey, TValue >, [ICollection](#)< [KeyValuePair](#)< TKey, TValue > >,  
[IEnumerable](#)< [KeyValuePair](#)< TKey, TValue > >, [IEnumerable](#), [ISheet](#)

## Derived

[ActionBuffSheet](#), [AdventureBossContributionRewardSheet](#),  
[AdventureBossFloorFirstRewardSheet](#), [AdventureBossFloorPointSheet](#),  
[AdventureBossFloorSheet](#), [AdventureBossFloorWaveSheet](#),  
[AdventureBossNcgRewardRatioSheet](#), [AdventureBossSheet](#),  
[AdventureBossUnlockFloorCostSheet](#), [AdventureBossWantedRewardSheet](#), [ArenaSheet](#),  
[BuffLinkSheet](#), [BuffSheet](#), [CharacterLevelSheet](#), [CharacterSheet](#), [CollectQuestSheet](#),  
[CollectionSheet](#), [CombinationEquipmentQuestSheet](#), [CombinationQuestSheet](#),  
[ConsumableItemRecipeSheet](#), [ConsumableItemSheet](#), [CostumeItemSheet](#),  
[CostumeStatSheet](#), [CreateAvatarFavSheet](#), [CreateAvatarItemSheet](#),  
[CrystalEquipmentGrindingSheet](#), [CrystalFluctuationSheet](#), [CrystalHammerPointSheet](#),  
[CrystalMaterialCostSheet](#), [CrystalMonsterCollectionMultiplierSheet](#),  
[CrystalRandomBuffSheet](#), [CrystalStageBuffGachaSheet](#), [CustomEquipmentCraftIconSheet](#),  
[CustomEquipmentCraftOptionSheet](#), [CustomEquipmentCraftRecipeSheet](#),  
[CustomEquipmentCraftRecipeSkillSheet](#), [CustomEquipmentCraftRelationshipSheet](#),  
[DeBuffLimitSheet](#), [EnemySkillSheet](#), [EnhancementCostSheet](#), [EnhancementCostSheetV2](#),  
[EnhancementCostSheetV3](#), [EquipmentItemOptionSheet](#), [EquipmentItemRecipeSheet](#),

[EquipmentItemSetEffectSheet](#), [EquipmentItemSheet](#), [EquipmentItemSubRecipeSheet](#),  
[EquipmentItemSubRecipeSheetV2](#), [EventConsumableItemRecipeSheet](#),  
[EventDungeonSheet](#), [EventDungeonStageSheet](#), [EventDungeonStageWaveSheet](#),  
[EventMaterialItemRecipeSheet](#), [EventScheduleSheet](#), [GameConfigSheet](#),  
[LoadIntoMyGaragesCostSheet](#), [GeneralQuestSheet](#), [GoldQuestSheet](#),  
[GrandFinaleParticipantsSheet](#), [GrandFinaleScheduleSheet](#), [ItemConfigForGradeSheet](#),  
[ItemEnhancementQuestSheet](#), [ItemGradeQuestSheet](#), [ItemRequirementSheet](#), [ItemSheet](#),  
[ItemTypeCollectQuestSheet](#), [MaterialItemSheet](#), [MimisbrunnrSheet](#),  
[MonsterCollectionRewardSheet](#), [MonsterCollectionSheet](#), [MonsterQuestSheet](#),  
[PetOptionSheet](#), [PetCostSheet](#), [PetSheet](#), [QuestItemRewardSheet](#), [QuestRewardSheet](#),  
[QuestSheet](#), [RedeemCodeListSheet](#), [RedeemRewardSheet](#), [RuneLevelBonusSheet](#),  
[RuneListSheet](#), [RuneCostSheet](#), [RuneOptionSheet](#), [RuneSheet](#), [RuneWeightSheet](#),  
[SkillActionBuffSheet](#), [SkillBuffSheet](#), [SkillSheet](#), [StageDialogSheet](#), [StageSheet](#),  
[StageWaveSheet](#), [StakePolicySheet](#), [StakeAchievementRewardSheet](#),  
[StakeActionPointCoefficientSheet](#), [StakeRegularFixedRewardSheet](#),  
[StakeRegularRewardSheet](#), [StatBuffSheet](#), [SummonSheet](#), [SweepRequiredCPSheet](#),  
[TradeQuestSheet](#), [UnlockCombinationSlotCostSheet](#), [WeeklyArenaRewardSheet](#),  
[WorldBossActionPatternSheet](#), [WorldBossBattleRewardSheet](#), [WorldBossCharacterSheet](#),  
[WorldBossGlobalHpSheet](#), [WorldBossKillRewardSheet](#), [WorldBossListSheet](#),  
[WorldBossRankRewardSheet](#), [WorldBossRankingRewardSheet](#), [WorldQuestSheet](#),  
[WorldSheet](#), [WorldUnlockSheet](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

### Sheet(string)

```
protected Sheet(string name)
```

## Parameters

name [string](#)

# Properties

## Count

Gets the number of elements contained in the [ICollection<T>](#).

```
public int Count { get; }
```

### Property Value

[int](#)

The number of elements contained in the [ICollection<T>](#).

## First

```
public TValue? First { get; }
```

### Property Value

TValue

## IsReadOnly

Gets a value indicating whether the [ICollection<T>](#) is read-only.

```
public bool IsReadOnly { get; }
```

### Property Value

[bool](#)

[true](#) if the [ICollection<T>](#) is read-only; otherwise, [false](#).

## this[TKey]

Gets or sets the element with the specified key.

```
public TValue this[TKey key] { get; set; }
```

## Parameters

### key TKey

The key of the element to get or set.

## Property Value

### TValue

The element with the specified key.

## Exceptions

### [ArgumentNullException](#)

key is [null](#).

### [KeyNotFoundException](#)

The property is retrieved and key is not found.

### [NotSupportedException](#)

The property is set and the [IDictionary<TKey, TValue>](#) is read-only.

## Keys

Gets an [ICollection<T>](#) containing the keys of the [IDictionary<TKey, TValue>](#).

```
public ICollection<TKey> Keys { get; }
```

## Property Value

### [ICollection<TKey>](#)

An [ICollection<T>](#) containing the keys of the object that implements [IDictionary<TKey, TValue>](#).

## Last

```
public TValue? Last { get; }
```

Property Value

TValue

## Name

```
public string Name { get; }
```

Property Value

[string](#)

## OrderedList

```
public IReadOnlyList<TValue>? OrderedDict { get; }
```

Property Value

[IReadOnlyList](#)<TValue>

## Values

Gets an [ICollection](#)<T> containing the values in the [IDictionary](#)< TKey, TValue >.

```
public ICollection<TValue> Values { get; }
```

Property Value

[ICollection](#)<TValue>

An [ICollection<T>](#) containing the values in the object that implements [IDictionary< TKey, TValue >](#).

## Methods

### Add(KeyValuePair< TKey, TValue >)

Adds an item to the [ICollection<T>](#).

```
public void Add(KeyValuePair< TKey, TValue > item)
```

#### Parameters

**item** [KeyValuePair](#)< TKey, TValue >

The object to add to the [ICollection<T>](#).

#### Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

### Add(TKey, TValue)

Adds an element with the provided key and value to the [IDictionary< TKey, TValue >](#).

```
public void Add(TKey key, TValue value)
```

#### Parameters

**key** TKey

The object to use as the key of the element to add.

**value** TValue

The object to use as the value of the element to add.

## Exceptions

### [ArgumentNullException](#)

`key` is [null](#).

### [ArgumentException](#)

An element with the same key already exists in the [IDictionary<TKey, TValue>](#).

### [NotSupportedException](#)

The [IDictionary<TKey, TValue>](#) is read-only.

## AddRow(TKey, TValue)

```
protected virtual void AddRow(TKey key, TValue value)
```

### Parameters

`key` TKey

`value` TValue

## Clear()

Removes all items from the [ICollection<T>](#).

```
public void clear()
```

## Exceptions

### [NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Contains(KeyValuePair<TKey, TValue>)

Determines whether the [ICollection<T>](#) contains a specific value.

```
public bool Contains(KeyValuePair<TKey, TValue> item)
```

## Parameters

**item** [KeyValuePair](#)<TKey, TValue>

The object to locate in the [ICollection](#)<T>.

## Returns

[bool](#)

[true](#) if **item** is found in the [ICollection](#)<T>; otherwise, [false](#).

## ContainsKey(TKey)

Determines whether the [IDictionary](#)<TKey, TValue> contains an element with the specified key.

```
public bool ContainsKey(TKey key)
```

## Parameters

**key** TKey

The key to locate in the [IDictionary](#)<TKey, TValue>.

## Returns

[bool](#)

[true](#) if the [IDictionary](#)<TKey, TValue> contains an element with the key; otherwise, [false](#).

## Exceptions

[ArgumentNullException](#)

**key** is [null](#).

## CopyTo(KeyValuePair<TKey, TValue>[], int)

Copies the elements of the [ICollection<T>](#) to an [Array](#), starting at a particular [Array](#) index.

```
public void CopyTo(KeyValuePair<TKey, TValue>[] array, int arrayIndex)
```

### Parameters

**array** [KeyValuePair](#)<TKey, TValue>[]

The one-dimensional [Array](#) that is the destination of the elements copied from [ICollection<T>](#). The [Array](#) must have zero-based indexing.

**arrayIndex** [int](#)

The zero-based index in **array** at which copying begins.

### Exceptions

[ArgumentNullException](#)

**array** is [null](#).

[ArgumentOutOfRangeException](#)

**arrayIndex** is less than 0.

[ArgumentException](#)

The number of elements in the source [ICollection<T>](#) is greater than the available space from **arrayIndex** to the end of the destination **array**.

## GetEnumerator()

```
public IEnumrator<TValue> GetEnumerator()
```

### Returns

[IEnumrator](#)<TValue>

## Remove(KeyValuePair<TKey, TValue>)

Removes the first occurrence of a specific object from the [ICollection<T>](#).

```
public bool Remove(KeyValuePair<TKey, TValue> item)
```

### Parameters

**item** [KeyValuePair](#)<TKey, TValue>

The object to remove from the [ICollection<T>](#).

### Returns

[bool](#)

[true](#) if **item** was successfully removed from the [ICollection<T>](#); otherwise, [false](#). This method also returns [false](#) if **item** is not found in the original [ICollection<T>](#).

### Exceptions

[NotSupportedException](#)

The [ICollection<T>](#) is read-only.

## Remove(TKey)

Removes the element with the specified key from the [IDictionary<TKey, TValue>](#).

```
public bool Remove(TKey key)
```

### Parameters

**key** TKey

The key of the element to remove.

### Returns

[bool](#)

[true](#) if the element is successfully removed; otherwise, [false](#). This method also returns [false](#) if `key` was not found in the original [IDictionary<TKey, TValue>](#).

## Exceptions

### [ArgumentNullException](#)

`key` is [null](#).

### [NotSupportedException](#)

The [IDictionary<TKey, TValue>](#) is read-only.

## Serialize()

```
public IValue Serialize()
```

## Returns

IValue

## Set(string, bool)

```
public virtual void Set(string csv, bool isReversed = false)
```

## Parameters

`csv` [string](#)

`isReversed` [bool](#)

true: csv의 column과 row의 역순을 적용합니다.

## Exceptions

### [ArgumentNullException](#)

### [InvalidOperationException](#)

## Set<T>(Sheet<TKey, T>, bool)

```
public void Set<T>(Sheet<TKey, T> sheet, bool executePostSet = true) where T : TValue, new()
```

### Parameters

sheet [Sheet](#)<TKey, T>

executePostSet [bool](#)

### Type Parameters

T

## TryGetValue(TKey, out TValue)

Gets the value associated with the specified key.

```
public bool TryGetValue(TKey key, out TValue value)
```

### Parameters

key TKey

The key whose value to get.

value TValue

When this method returns, the value associated with the specified key, if the key is found; otherwise, the default value for the type of the [value](#) parameter. This parameter is passed uninitialized.

### Returns

[bool](#)

[true](#) if the object that implements [IDictionary](#)<TKey, TValue> contains an element with the specified key; otherwise, [false](#).

## Exceptions

### [ArgumentNullException](#)

key is [null](#).

## TryGetValue(TKey, out TValue, bool)

```
public bool TryGetValue(TKey key, out TValue value, bool throwException)
```

## Parameters

key TKey

value TValue

throwException [bool](#)

## Returns

[bool](#)

# Class SimulatorSheets

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class SimulatorSheets : SimulatorSheetsV1
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← SimulatorSheets

## Derived

[ArenaSimulatorSheets](#), [RaidSimulatorSheets](#), [RankingSimulatorSheets](#),  
[StageSimulatorSheets](#)

## Inherited Members

[SimulatorSheetsV1.MaterialItemSheet](#) , [SimulatorSheetsV1.SkillSheet](#) ,  
[SimulatorSheetsV1.SkillBuffSheet](#) , [SimulatorSheetsV1.SkillActionBuffSheet](#) ,  
[SimulatorSheetsV1.ActionBuffSheet](#) , [SimulatorSheetsV1.StatBuffSheet](#) ,  
[SimulatorSheetsV1.CharacterSheet](#) , [SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

SimulatorSheets(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet, RuneOptionSheet,  
RuneListSheet, RuneLevelBonusSheet)

```
public SimulatorSheets(MaterialItemSheet materialItemSheet, SkillSheet skillSheet,  
SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet, SkillActionBuffSheet  
skillActionBuffSheet, ActionBuffSheet actionBuffSheet, CharacterSheet  
characterSheet, CharacterLevelSheet characterLevelSheet, EquipmentItemSetEffectSheet  
equipmentItemSetEffectSheet, RuneOptionSheet runeOptionSheet, RuneListSheet  
runeListSheet, RuneLevelBonusSheet runeLevelBonusSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

runeOptionSheet [RuneOptionSheet](#)

runeListSheet [RuneListSheet](#)

runeLevelBonusSheet [RuneLevelBonusSheet](#)

## Fields

### RuneLevelBonusSheet

```
public readonly RuneLevelBonusSheet RuneLevelBonusSheet
```

#### Field Value

[RuneLevelBonusSheet](#)

### RuneListSheet

```
public readonly RuneListSheet RuneListSheet
```

Field Value

[RuneListSheet](#)

## RuneOptionSheet

```
public readonly RuneOptionSheet RuneOptionSheet
```

Field Value

[RuneOptionSheet](#)

# Class SimulatorSheetsV1

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class SimulatorSheetsV1
```

## Inheritance

[object](#) ← SimulatorSheetsV1

## Derived

[ArenaSimulatorSheetsV1](#), [RaidSimulatorSheetsV1](#), [RankingSimulatorSheetsV1](#),  
[SimulatorSheets](#), [StageSimulatorSheetsV1](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

SimulatorSheetsV1(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet)

```
public SimulatorSheetsV1(MaterialItemSheet materialItemSheet, SkillSheet skillSheet,  
SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

## Fields

### ActionBuffSheet

```
public readonly ActionBuffSheet ActionBuffSheet
```

Field Value

[ActionBuffSheet](#)

### CharacterLevelSheet

```
public readonly CharacterLevelSheet CharacterLevelSheet
```

Field Value

[CharacterLevelSheet](#)

### CharacterSheet

```
public readonly CharacterSheet CharacterSheet
```

Field Value

[CharacterSheet](#)

## EquipmentItemSetEffectSheet

```
public readonly EquipmentItemSetEffectSheet EquipmentItemSetEffectSheet
```

Field Value

[EquipmentItemSetEffectSheet](#)

## MaterialItemSheet

```
public readonly MaterialItemSheet MaterialItemSheet
```

Field Value

[MaterialItemSheet](#)

## SkillActionBuffSheet

```
public readonly SkillActionBuffSheet SkillActionBuffSheet
```

Field Value

[SkillActionBuffSheet](#)

## SkillBuffSheet

```
public readonly SkillBuffSheet SkillBuffSheet
```

Field Value

[SkillBuffSheet](#)

## SkillSheet

```
public readonly SkillSheet SkillSheet
```

Field Value

[SkillSheet](#)

## StatBuffSheet

```
public readonly StatBuffSheet StatBuffSheet
```

Field Value

[StatBuffSheet](#)

# Class SimulatorSheetsV100291

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class SimulatorSheetsV100291
```

## Inheritance

[object](#) ← SimulatorSheetsV100291

## Derived

[ArenaSimulatorSheetsV100291](#), [RankingSimulatorSheetsV100291](#),  
[StageSimulatorSheetsV100291](#)

## Inherited Members

[object.Equals\(object\)](#), [object.Equals\(object, object\)](#), [object.GetHashCode\(\)](#),  
[object.GetType\(\)](#), [object.MemberwiseClone\(\)](#), [object.ReferenceEquals\(object, object\)](#),  
[object.ToString\(\)](#)

## Constructors

SimulatorSheetsV100291(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, BuffSheet, CharacterSheet,  
CharacterLevelSheet, EquipmentItemSetEffectSheet)

```
public SimulatorSheetsV100291(MaterialItemSheet materialItemSheet,  
SkillSheet skillSheet, SkillBuffSheet skillBuffSheet, BuffSheet buffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

buffSheet [BuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

## Fields

### BuffSheet

```
public readonly BuffSheet BuffSheet
```

Field Value

[BuffSheet](#)

### CharacterLevelSheet

```
public readonly CharacterLevelSheet CharacterLevelSheet
```

Field Value

[CharacterLevelSheet](#)

### CharacterSheet

```
public readonly CharacterSheet CharacterSheet
```

Field Value

[CharacterSheet](#)

### EquipmentItemSetEffectSheet

```
public readonly EquipmentItemSetEffectSheet EquipmentItemSetEffectSheet
```

Field Value

[EquipmentItemSetEffectSheet](#)

## MaterialItemSheet

```
public readonly MaterialItemSheet MaterialItemSheet
```

Field Value

[MaterialItemSheet](#)

## SkillBuffSheet

```
public readonly SkillBuffSheet SkillBuffSheet
```

Field Value

[SkillBuffSheet](#)

## SkillSheet

```
public readonly SkillSheet SkillSheet
```

Field Value

[SkillSheet](#)

## Methods

### ToSimulatorSheetsV1()

```
public SimulatorSheetsV1 ToSimulatorSheetsV1()
```

Returns

[SimulatorSheetsV1](#)

# Class SkillActionBuffSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SkillActionBuffSheet : Sheet<int, SkillActionBuffSheet.Row>,
IDictionary<int, SkillActionBuffSheet.Row>, ICollection<KeyValuePair<int,
SkillActionBuffSheet.Row>>, IEnumerable<KeyValuePair<int,
SkillActionBuffSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, SkillActionBuffSheet.Row>](#) ← [SkillActionBuffSheet](#)

## Implements

[IDictionary<int, SkillActionBuffSheet.Row>](#),  
[ICollection<KeyValuePair<int, SkillActionBuffSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, SkillActionBuffSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, SkillActionBuffSheet.Row>.Name](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.OrderedList](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.First](#) , [Sheet<int, SkillActionBuffSheet.Row>.Last](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Keys](#) , [Sheet<int, SkillActionBuffSheet.Row>.Values](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Count](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.this\[int\]](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.TryGetValue\(int, out SkillActionBuffSheet.Row, bool\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.AddRow\(int, SkillActionBuffSheet.Row\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Add\(int, SkillActionBuffSheet.Row\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.TryGetValue\(int, out SkillActionBuffSheet.Row\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Add\(KeyValuePair<int, SkillActionBuffSheet.Row>\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, SkillActionBuffSheet.Row>.Contains\(KeyValuePair<int, SkillActionBuffSheet.Row>\)](#) ,

```
Sheet<int, SkillActionBuffSheet.Row>.CopyTo\(KeyValuePair<int, SkillActionBuffSheet.Row>\[\], int\) ,  
Sheet<int, SkillActionBuffSheet.Row>.Remove\(KeyValuePair<int, SkillActionBuffSheet.Row>\) ,  
Sheet<int, SkillActionBuffSheet.Row>.Serialize\(\) , object.Equals\(object\) ,  
object.Equals\(object, object\) , object.GetHashCode\(\) , object.GetType\(\) ,  
object.MemberwiseClone\(\) , object.ReferenceEquals\(object, object\) , object.ToString\(\)
```

## Constructors

### SkillActionBuffSheet()

```
public SkillActionBuffSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, SkillActionBuffSheet.Row value)
```

## Parameters

key [int](#)

value [SkillActionBuffSheet.Row](#)

# Class SkillActionBuffSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SkillActionBuffSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← SkillActionBuffSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BuffIds

```
public List<int> BuffIds { get; }
```

### Property Value

[List<int>](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

# SkillId

```
public int SkillId { get; }
```

Property Value

[int](#)

## Methods

EndOfSheetInitialize()

```
public override void EndOfSheetInitialize()
```

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class SkillBuffSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SkillBuffSheet : Sheet<int, SkillBuffSheet.Row>, IDictionary<int, SkillBuffSheet.Row>, ICollection<KeyValuePair<int, SkillBuffSheet.Row>>, IEnumerable<KeyValuePair<int, SkillBuffSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, SkillBuffSheet.Row>](#) ← [SkillBuffSheet](#)

## Implements

[IDictionary<int, SkillBuffSheet.Row>](#),  
[ICollection<KeyValuePair<int, SkillBuffSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, SkillBuffSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, SkillBuffSheet.Row>.Name](#) , [Sheet<int, SkillBuffSheet.Row>.OrderedList](#) ,  
[Sheet<int, SkillBuffSheet.Row>.First](#) , [Sheet<int, SkillBuffSheet.Row>.Last](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Keys](#) , [Sheet<int, SkillBuffSheet.Row>.Values](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Count](#) , [Sheet<int, SkillBuffSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, SkillBuffSheet.Row>.this\[int\]](#) , [Sheet<int, SkillBuffSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.TryGetValue\(int, out SkillBuffSheet.Row, bool\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.AddRow\(int, SkillBuffSheet.Row\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Add\(int, SkillBuffSheet.Row\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.TryGetValue\(int, out SkillBuffSheet.Row\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Add\(KeyValuePair<int, SkillBuffSheet.Row>\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Contains\(KeyValuePair<int, SkillBuffSheet.Row>\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.CopyTo\(KeyValuePair<int, SkillBuffSheet.Row>\[\], int\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Remove\(KeyValuePair<int, SkillBuffSheet.Row>\)](#) ,  
[Sheet<int, SkillBuffSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SkillBuffSheet()

```
public SkillBuffSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, SkillBuffSheet.Row value)
```

## Parameters

key [int](#)

value [SkillBuffSheet.Row](#)

# Class SkillBuffSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SkillBuffSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← SkillBuffSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BuffIds

```
public List<int> BuffIds { get; }
```

### Property Value

[List<int>](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## SkillId

```
public int SkillId { get; }
```

Property Value

[int](#)

## Methods

### EndOfSheetInitialize()

```
public override void EndOfSheetInitialize()
```

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class SkillSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SkillSheet : Sheet<int, SkillSheet.Row>, IDictionary<int,
SkillSheet.Row>, ICollection<KeyValuePair<int, SkillSheet.Row>>,
IEnumerable<KeyValuePair<int, SkillSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, SkillSheet.Row>](#) ← SkillSheet

## Implements

[IDictionary<int, SkillSheet.Row>](#), [ICollection<KeyValuePair<int, SkillSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, SkillSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, SkillSheet.Row>.Name](#) , [Sheet<int, SkillSheet.Row>.OrderedList](#) ,  
[Sheet<int, SkillSheet.Row>.First](#) , [Sheet<int, SkillSheet.Row>.Last](#) ,  
[Sheet<int, SkillSheet.Row>.Keys](#) , [Sheet<int, SkillSheet.Row>.Values](#) ,  
[Sheet<int, SkillSheet.Row>.Count](#) , [Sheet<int, SkillSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, SkillSheet.Row>.this\[int\]](#) , [Sheet<int, SkillSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, SkillSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, SkillSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, SkillSheet.Row>.TryGetValue\(int, out SkillSheet.Row, bool\)](#) ,  
[Sheet<int, SkillSheet.Row>.AddRow\(int, SkillSheet.Row\)](#) ,  
[Sheet<int, SkillSheet.Row>.Add\(int, SkillSheet.Row\)](#) ,  
[Sheet<int, SkillSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, SkillSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, SkillSheet.Row>.TryGetValue\(int, out SkillSheet.Row\)](#) ,  
[Sheet<int, SkillSheet.Row>.Add\(KeyValuePair<int, SkillSheet.Row>\)](#) ,  
[Sheet<int, SkillSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, SkillSheet.Row>.Contains\(KeyValuePair<int, SkillSheet.Row>\)](#) ,  
[Sheet<int, SkillSheet.Row>.CopyTo\(KeyValuePair<int, SkillSheet.Row>\[\], int\)](#) ,  
[Sheet<int, SkillSheet.Row>.Remove\(KeyValuePair<int, SkillSheet.Row>\)](#) ,  
[Sheet<int, SkillSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## SkillSheet()

```
public SkillSheet()
```

# Class SkillSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class SkillSheet.Row : SheetRow<int>, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < SkillSheet.Row

## Implements

[IState](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### Row()

```
public Row()
```

### Row(Dictionary)

```
public Row(Dictionary serialized)
```

## Parameters

**serialized** Dictionary

## Properties

### Combo

```
public bool Combo { get; }
```

Property Value

[bool](#) ↗

### Cooldown

```
public int Cooldown { get; }
```

Property Value

[int](#) ↗

### ElementType

```
public ElementType ElementType { get; }
```

Property Value

[ElementType](#)

### HitCount

```
public int HitCount { get; }
```

Property Value

[int](#) ↗

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## SkillCategory

```
public SkillCategory SkillCategory { get; }
```

Property Value

[SkillCategory](#)

## SkillTargetType

```
public SkillTargetType SkillTargetType { get; }
```

Property Value

[SkillTargetType](#)

## SkillType

```
public SkillType SkillType { get; }
```

Property Value

[SkillType](#)

## Methods

### Deserialize(Dictionary)

```
public static SkillSheet.Row Deserialize(Dictionary serialized)
```

Parameters

**serialized** Dictionary

Returns

[SkillSheet.Row](#)

### Serialize()

```
public IValue Serialize()
```

Returns

IValue

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class StageDialogSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageDialogSheet : Sheet<int, StageDialogSheet.Row>, IDictionary<int,
StageDialogSheet.Row>, ICollection<KeyValuePair<int, StageDialogSheet.Row>>,
IEnumerable<KeyValuePair<int, StageDialogSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StageDialogSheet.Row>](#) ← StageDialogSheet

## Implements

[IDictionary<int, StageDialogSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, StageDialogSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, StageDialogSheet.Row>>](#) , [IEnumerable](#) , [ISheet](#)

## Inherited Members

[Sheet<int, StageDialogSheet.Row>.Name](#) ,  
[Sheet<int, StageDialogSheet.Row>.OrderedList](#) , [Sheet<int, StageDialogSheet.Row>.First](#) ,  
[Sheet<int, StageDialogSheet.Row>.Last](#) , [Sheet<int, StageDialogSheet.Row>.Keys](#) ,  
[Sheet<int, StageDialogSheet.Row>.Values](#) , [Sheet<int, StageDialogSheet.Row>.Count](#) ,  
[Sheet<int, StageDialogSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StageDialogSheet.Row>.this\[int\]](#) ,  
[Sheet<int, StageDialogSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.TryGetValue\(int, out StageDialogSheet.Row, bool\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.AddRow\(int, StageDialogSheet.Row\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.Add\(int, StageDialogSheet.Row\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.TryGetValue\(int, out StageDialogSheet.Row\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.Add\(KeyValuePair<int, StageDialogSheet.Row>\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.Contains\(KeyValuePair<int, StageDialogSheet.Row>\)](#) ,  
[Sheet<int, StageDialogSheet.Row>.CopyTo\(KeyValuePair<int, StageDialogSheet.Row>\[\], int\)](#) ,

```
Sheet<int, StageDialogSheet.Row>.Remove(KeyValuePair<int, StageDialogSheet.Row>),  
Sheet<int, StageDialogSheet.Row>.Serialize(), object.Equals(object)»,  
object.Equals(object, object)», object.GetHashCode()», object.GetType()»,  
object.MemberwiseClone()», object.ReferenceEquals(object, object)», object.ToString()»
```

## Constructors

### StageDialogSheet()

```
public StageDialogSheet()
```

# Class StageDialogSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageDialogSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← StageDialogSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### DialogId

```
public int DialogId { get; }
```

Property Value

[int](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class StageSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageSheet : Sheet<int, StageSheet.Row>, IDictionary<int,
StageSheet.Row>, ICollection<KeyValuePair<int, StageSheet.Row>>,
IEnumerable<KeyValuePair<int, StageSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StageSheet.Row>](#) ← StageSheet

## Implements

[IDictionary<int, StageSheet.Row>](#),  
[ICollection<KeyValuePair<int, StageSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, StageSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, StageSheet.Row>.Name](#) , [Sheet<int, StageSheet.Row>.OrderedList](#) ,  
[Sheet<int, StageSheet.Row>.First](#) , [Sheet<int, StageSheet.Row>.Last](#) ,  
[Sheet<int, StageSheet.Row>.Keys](#) , [Sheet<int, StageSheet.Row>.Values](#) ,  
[Sheet<int, StageSheet.Row>.Count](#) , [Sheet<int, StageSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StageSheet.Row>.this\[int\]](#) , [Sheet<int, StageSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StageSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StageSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StageSheet.Row>.TryGetValue\(int, out StageSheet.Row, bool\)](#) ,  
[Sheet<int, StageSheet.Row>.AddRow\(int, StageSheet.Row\)](#) ,  
[Sheet<int, StageSheet.Row>.Add\(int, StageSheet.Row\)](#) ,  
[Sheet<int, StageSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, StageSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StageSheet.Row>.TryGetValue\(int, out StageSheet.Row\)](#) ,  
[Sheet<int, StageSheet.Row>.Add\(KeyValuePair<int, StageSheet.Row>\)](#) ,  
[Sheet<int, StageSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StageSheet.Row>.Contains\(KeyValuePair<int, StageSheet.Row>\)](#) ,  
[Sheet<int, StageSheet.Row>.CopyTo\(KeyValuePair<int, StageSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StageSheet.Row>.Remove\(KeyValuePair<int, StageSheet.Row>\)](#) ,  
[Sheet<int, StageSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## StageSheet()

```
public StageSheet()
```

# Class StageSheet.RewardData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageSheet.RewardData
```

## Inheritance

[object](#) ← StageSheet.RewardData

## Derived

[WeeklyArenaRewardSheet.RewardData](#)

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RewardData(int, decimal, int, int)

```
public RewardData(int itemId, decimal ratio, int min, int max)
```

## Parameters

itemId [int](#)

ratio [decimal](#)

min [int](#)

max [int](#)

## Properties

## ItemId

```
public int ItemId { get; }
```

Property Value

[int](#) ↗

## Max

```
public int Max { get; }
```

Property Value

[int](#) ↗

## Min

```
public int Min { get; }
```

Property Value

[int](#) ↗

## Ratio

```
public decimal Ratio { get; }
```

Property Value

[decimal](#) ↗

# Class StageSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← StageSheet.Row

## Derived

[EventDungeonStageSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BGM

```
public string BGM { get; }
```

#### Property Value

[string](#)

### Background

```
public string Background { get; }
```

#### Property Value

[string](#)

## CostAP

```
public int CostAP { get; }
```

Property Value

[int](#)

## DropItemMax

```
public int DropItemMax { get; }
```

Property Value

[int](#)

## DropItemMin

```
public int DropItemMin { get; }
```

Property Value

[int](#)

## EnemyInitialStatModifiers

```
public List<StatModifier> EnemyInitialStatModifiers { get; }
```

Property Value

[List](#)<[StatModifier](#)>

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Rewards

```
public List<StageSheet.RewardData> Rewards { get; }
```

Property Value

[List](#)<[StageSheet.RewardData](#)>

## TurnLimit

```
public int TurnLimit { get; }
```

Property Value

[int](#)

## Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList<string>](#)

# Class StageSimulatorSheets

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class StageSimulatorSheets : SimulatorSheets
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← [SimulatorSheets](#) ← StageSimulatorSheets

## Inherited Members

[SimulatorSheets.RuneOptionSheet](#) , [SimulatorSheets.RuneListSheet](#) ,  
[SimulatorSheets.RuneLevelBonusSheet](#) , [SimulatorSheetsV1.MaterialItemSheet](#) ,  
[SimulatorSheetsV1.SkillSheet](#) , [SimulatorSheetsV1.SkillBuffSheet](#) ,  
[SimulatorSheetsV1.SkillActionBuffSheet](#) , [SimulatorSheetsV1.ActionBuffSheet](#) ,  
[SimulatorSheetsV1.StatBuffSheet](#) , [SimulatorSheetsV1.CharacterSheet](#) ,  
[SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

StageSimulatorSheets(MaterialItemSheet, SkillSheet,  
SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet,  
ActionBuffSheet, CharacterSheet, CharacterLevelSheet,  
EquipmentItemSetEffectSheet, StageSheet,  
StageWaveSheet, EnemySkillSheet, RuneOptionSheet,  
RuneListSheet, RuneLevelBonusSheet)

```
public StageSimulatorSheets(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, StageSheet  
stageSheet, StageWaveSheet stageWaveSheet, EnemySkillSheet enemySkillSheet,
```

```
RuneOptionSheet runeOptionSheet, RuneListSheet runeListSheet,  
RuneLevelBonusSheet runeLevelBonusSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

stageSheet [StageSheet](#)

stageWaveSheet [StageWaveSheet](#)

enemySkillSheet [EnemySkillSheet](#)

runeOptionSheet [RuneOptionSheet](#)

runeListSheet [RuneListSheet](#)

runeLevelBonusSheet [RuneLevelBonusSheet](#)

## Fields

### EnemySkillSheet

```
public readonly EnemySkillSheet EnemySkillSheet
```

## Field Value

## StageSheet

```
public readonly StageSheet StageSheet
```

Field Value

[StageSheet](#)

## StageWaveSheet

```
public readonly StageWaveSheet StageWaveSheet
```

Field Value

[StageWaveSheet](#)

# Class StageSimulatorSheetsV1

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class StageSimulatorSheetsV1 : SimulatorSheetsV1
```

## Inheritance

[object](#) ← [SimulatorSheetsV1](#) ← StageSimulatorSheetsV1

## Inherited Members

[SimulatorSheetsV1.MaterialItemSheet](#) , [SimulatorSheetsV1.SkillSheet](#) ,  
[SimulatorSheetsV1.SkillBuffSheet](#) , [SimulatorSheetsV1.SkillActionBuffSheet](#) ,  
[SimulatorSheetsV1.ActionBuffSheet](#) , [SimulatorSheetsV1.StatBuffSheet](#) ,  
[SimulatorSheetsV1.CharacterSheet](#) , [SimulatorSheetsV1.CharacterLevelSheet](#) ,  
[SimulatorSheetsV1.EquipmentItemSetEffectSheet](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

StageSimulatorSheetsV1(MaterialItemSheet, SkillSheet, SkillBuffSheet, StatBuffSheet, SkillActionBuffSheet, ActionBuffSheet, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet, StageSheet, StageWaveSheet, EnemySkillSheet)

```
public StageSimulatorSheetsV1(MaterialItemSheet materialItemSheet, SkillSheet  
skillSheet, SkillBuffSheet skillBuffSheet, StatBuffSheet statBuffSheet,  
SkillActionBuffSheet skillActionBuffSheet, ActionBuffSheet actionBuffSheet,  
CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet,  
EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, StageSheet stageSheet,  
StageWaveSheet stageWaveSheet, EnemySkillSheet enemySkillSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

statBuffSheet [StatBuffSheet](#)

skillActionBuffSheet [SkillActionBuffSheet](#)

actionBuffSheet [ActionBuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

stageSheet [StageSheet](#)

stageWaveSheet [StageWaveSheet](#)

enemySkillSheet [EnemySkillSheet](#)

## Fields

### EnemySkillSheet

```
public readonly EnemySkillSheet EnemySkillSheet
```

Field Value

[EnemySkillSheet](#)

### StageSheet

```
public readonly StageSheet StageSheet
```

Field Value

[StageSheet](#)

# StageWaveSheet

```
public readonly StageWaveSheet StageWaveSheet
```

Field Value

[StageWaveSheet](#)

# Class StageSimulatorSheetsV100291

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class StageSimulatorSheetsV100291 : SimulatorSheetsV100291
```

## Inheritance

[object](#) ← [SimulatorSheetsV100291](#) ← StageSimulatorSheetsV100291

## Inherited Members

[SimulatorSheetsV100291.MaterialItemSheet](#) , [SimulatorSheetsV100291.SkillSheet](#) ,  
[SimulatorSheetsV100291.SkillBuffSheet](#) , [SimulatorSheetsV100291.BuffSheet](#) ,  
[SimulatorSheetsV100291.CharacterSheet](#) , [SimulatorSheetsV100291.CharacterLevelSheet](#) ,  
[SimulatorSheetsV100291.EquipmentItemSetEffectSheet](#) ,  
[SimulatorSheetsV100291.ToSimulatorSheetsV1\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

StageSimulatorSheetsV100291(MaterialItemSheet, SkillSheet, SkillBuffSheet, BuffSheet, CharacterSheet, CharacterLevelSheet, EquipmentItemSetEffectSheet, StageSheet, StageWaveSheet, EnemySkillSheet)

```
public StageSimulatorSheetsV100291(MaterialItemSheet materialItemSheet, SkillSheet skillSheet, SkillBuffSheet skillBuffSheet, BuffSheet buffSheet, CharacterSheet characterSheet, CharacterLevelSheet characterLevelSheet, EquipmentItemSetEffectSheet equipmentItemSetEffectSheet, StageSheet stageSheet, StageWaveSheet stageWaveSheet, EnemySkillSheet enemySkillSheet)
```

## Parameters

materialItemSheet [MaterialItemSheet](#)

skillSheet [SkillSheet](#)

skillBuffSheet [SkillBuffSheet](#)

buffSheet [BuffSheet](#)

characterSheet [CharacterSheet](#)

characterLevelSheet [CharacterLevelSheet](#)

equipmentItemSetEffectSheet [EquipmentItemSetEffectSheet](#)

stageSheet [StageSheet](#)

stageWaveSheet [StageWaveSheet](#)

enemySkillSheet [EnemySkillSheet](#)

## Fields

### EnemySkillSheet

```
public readonly EnemySkillSheet EnemySkillSheet
```

#### Field Value

[EnemySkillSheet](#)

### StageSheet

```
public readonly StageSheet StageSheet
```

#### Field Value

[StageSheet](#)

### StageWaveSheet

```
public readonly StageWaveSheet StageWaveSheet
```

Field Value

[StageWaveSheet](#)

## Methods

### ToStageSimulatorSheetsV1()

```
public StageSimulatorSheetsV1 ToStageSimulatorSheetsV1()
```

Returns

[StageSimulatorSheetsV1](#)

# Class StageWaveSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageWaveSheet : Sheet<int, StageWaveSheet.Row>, IDictionary<int,
StageWaveSheet.Row>, ICollection<KeyValuePair<int, StageWaveSheet.Row>>,
IEnumerable<KeyValuePair<int, StageWaveSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StageWaveSheet.Row>](#) ← StageWaveSheet

## Implements

[IDictionary<int, StageWaveSheet.Row>](#),  
[ICollection<KeyValuePair<int, StageWaveSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, StageWaveSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, StageWaveSheet.Row>.Name](#) , [Sheet<int, StageWaveSheet.Row>.OrderedList](#) ,  
[Sheet<int, StageWaveSheet.Row>.First](#) , [Sheet<int, StageWaveSheet.Row>.Last](#) ,  
[Sheet<int, StageWaveSheet.Row>.Keys](#) , [Sheet<int, StageWaveSheet.Row>.Values](#) ,  
[Sheet<int, StageWaveSheet.Row>.Count](#) , [Sheet<int, StageWaveSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StageWaveSheet.Row>.this\[int\]](#) ,  
[Sheet<int, StageWaveSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.TryGetValue\(int, out StageWaveSheet.Row, bool\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.AddRow\(int, StageWaveSheet.Row\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Add\(int, StageWaveSheet.Row\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.TryGetValue\(int, out StageWaveSheet.Row\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Add\(KeyValuePair<int, StageWaveSheet.Row>\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Contains\(KeyValuePair<int, StageWaveSheet.Row>\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.CopyTo\(KeyValuePair<int, StageWaveSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StageWaveSheet.Row>.Remove\(KeyValuePair<int, StageWaveSheet.Row>\)](#) ,

[Sheet<int, StageWaveSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### StageWaveSheet()

```
public StageWaveSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, StageWaveSheet.Row value)
```

## Parameters

key [int](#)

value [StageWaveSheet.Row](#)

# Class StageWaveSheet.MonsterData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageWaveSheet.MonsterData
```

## Inheritance

[object](#) ← StageWaveSheet.MonsterData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### MonsterData(int, int, int)

```
public MonsterData(int characterId, int level, int count)
```

#### Parameters

characterId [int](#)

level [int](#)

count [int](#)

## Properties

### CharacterId

```
public int CharacterId { get; }
```

Property Value

[int ↗](#)

Count

```
public int Count { get; }
```

Property Value

[int ↗](#)

Level

```
public int Level { get; }
```

Property Value

[int ↗](#)

# Class StageWaveSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageWaveSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← StageWaveSheet.Row

## Derived

[EventDungeonStageWaveSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Properties

## HasBoss

```
public bool HasBoss { get; }
```

## Property Value

[bool](#)

## Key

```
public override int Key { get; }
```

## Property Value

[int](#)

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## TotalMonsterIds

```
public List<int> TotalMonsterIds { get; }
```

Property Value

[List](#)<[int](#)>

## Waves

```
public List<StageWaveSheet.WaveData> Waves { get; }
```

Property Value

[List](#)<[StageWaveSheet](#).[WaveData](#)>

## Methods

### EndOfSheetInitialize()

```
public override void EndOfSheetInitialize()
```

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class StageWaveSheet.WaveData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StageWaveSheet.WaveData
```

## Inheritance

[object](#) ← StageWaveSheet.WaveData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### WaveData(int, List<MonsterData>, bool)

```
public WaveData(int number, List<StageWaveSheet.MonsterData> monsters, bool hasBoss)
```

## Parameters

number [int](#)

monsters [List](#)<[StageWaveSheet.MonsterData](#)>

hasBoss [bool](#)

## Properties

### HasBoss

```
public bool HasBoss { get; }
```

Property Value

[bool](#)

## Monsters

```
public List<StageWaveSheet.MonsterData> Monsters { get; }
```

Property Value

[List](#)<[StageWaveSheet](#).[MonsterData](#)>

## Number

```
public int Number { get; }
```

Property Value

[int](#)

# Class StakeAchievementRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeAchievementRewardSheet : Sheet<int,
StakeAchievementRewardSheet.Row>, IDictionary<int, StakeAchievementRewardSheet.Row>,
ICollection<KeyValuePair<int, StakeAchievementRewardSheet.Row>>,
IEnumerable<KeyValuePair<int, StakeAchievementRewardSheet.Row>>, IEnumerable,
IStakeRewardSheet, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StakeAchievementRewardSheet.Row>](#) ←

StakeAchievementRewardSheet

## Implements

[IDictionary<int, StakeAchievementRewardSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, StakeAchievementRewardSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, StakeAchievementRewardSheet.Row>>](#) ,  
[IEnumerable](#) , [IStakeRewardSheet](#) , [ISheet](#)

## Inherited Members

[Sheet<int, StakeAchievementRewardSheet.Row>.Name](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.First](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Last](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Keys](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Values](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Count](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.TryGetValue\(int, out](#)  
[StakeAchievementRewardSheet.Row, bool\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.AddRow\(int,](#)  
[StakeAchievementRewardSheet.Row\)](#) ,

[Sheet<int, StakeAchievementRewardSheet.Row>.Add\(int, StakeAchievementRewardSheet.Row\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.TryGetValue\(int, out StakeAchievementRewardSheet.Row\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Add\(KeyValuePair<int, StakeAchievementRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Contains\(KeyValuePair<int, StakeAchievementRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.CopyTo\(KeyValuePair<int, StakeAchievementRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Remove\(KeyValuePair<int, StakeAchievementRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeAchievementRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[SheetsExtensions.FindLevelByStakedAmount\(IStakeRewardSheet, Address, FungibleAssetValue\)](#)

## Constructors

### StakeAchievementRewardSheet()

```
public StakeAchievementRewardSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IStakeRewardRow> OrderedRows { get; }
```

### Property Value

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, StakeAchievementRewardSheet.Row value)
```

#### Parameters

key [int](#)

value [StakeAchievementRewardSheet.Row](#)

### FindStep(int, long)

```
public int FindStep(int level, long stakedBlockPeriod)
```

#### Parameters

level [int](#)

stakedBlockPeriod [long](#)

#### Returns

[int](#)

# Class StakeAchievementRewardSheet.RewardInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class StakeAchievementRewardSheet.RewardInfo
```

## Inheritance

[object](#) ← StakeAchievementRewardSheet.RewardInfo

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RewardInfo(Dictionary)

```
public RewardInfo(Dictionary dictionary)
```

#### Parameters

**dictionary** Dictionary

### RewardInfo(int, int)

```
public RewardInfo(int itemId, int quantity)
```

#### Parameters

**itemId** [int](#)

quantity [int](#)

## RewardInfo(params string[])

```
public RewardInfo(params string[] fields)
```

### Parameters

fields [string](#)[]

## Fields

### ItemId

```
public readonly int ItemId
```

### Field Value

[int](#)

### Quantity

```
public readonly int Quantity
```

### Field Value

[int](#)

## Methods

### Equals(RewardInfo)

```
protected bool Equals(StakeAchievementRewardSheet.RewardInfo other)
```

Parameters

other [StakeAchievementRewardSheet.RewardInfo](#)

Returns

[bool](#)

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## Serialize()

```
public IValue Serialize()
```

Returns

IValue

# Class StakeAchievementRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeAchievementRewardSheet.Row : SheetRow<int>, IStakeRewardRow
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < StakeAchievementRewardSheet.Row

## Implements

[IStakeRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

### Level

```
public int Level { get; }
```

#### Property Value

[int](#)

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#)

## Steps

```
public List<StakeAchievementRewardSheet.Step> Steps { get; }
```

Property Value

[List](#)<[StakeAchievementRewardSheet](#).[Step](#)>

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class StakeAchievementRewardSheet.Step

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class StakeAchievementRewardSheet.Step
```

## Inheritance

[object](#) ← StakeAchievementRewardSheet.Step

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### Step(long, long, List<RewardInfo>)

```
public Step(long requiredGold, long requiredBlockIndex,  
List<StakeAchievementRewardSheet.RewardInfo> rewards)
```

## Parameters

requiredGold [long](#)

requiredBlockIndex [long](#)

rewards [List](#)<[StakeAchievementRewardSheet.RewardInfo](#)>

## Properties

### RequiredBlockIndex

```
public long RequiredBlockIndex { get; }
```

Property Value

[long](#) ↴

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#) ↴

## Rewards

```
public List<StakeAchievementRewardSheet.RewardInfo> Rewards { get; }
```

Property Value

[List](#) ↴ <[StakeAchievementRewardSheet.RewardInfo](#)>

# Class StakeActionPointCoefficientSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeActionPointCoefficientSheet : Sheet<int,
StakeActionPointCoefficientSheet.Row>, IDictionary<int,
StakeActionPointCoefficientSheet.Row>, ICollection<KeyValuePair<int,
StakeActionPointCoefficientSheet.Row>>, IEnumerable<KeyValuePair<int,
StakeActionPointCoefficientSheet.Row>>, IEnumerable, IStakeRewardSheet, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StakeActionPointCoefficientSheet.Row>](#) ←  
StakeActionPointCoefficientSheet

## Implements

[IDictionary<int, StakeActionPointCoefficientSheet.Row>](#),  
[ICollection<KeyValuePair<int, StakeActionPointCoefficientSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, StakeActionPointCoefficientSheet.Row>>](#),  
[IEnumerable, IStakeRewardSheet, ISheet](#)

## Inherited Members

[Sheet<int, StakeActionPointCoefficientSheet.Row>.Name](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.OrderedList](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.First](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Last](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Keys](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Values](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Count](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.this\[int\]](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.TryGetValue\(int, out StakeActionPointCoefficientSheet.Row, bool\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.AddRow\(int, StakeActionPointCoefficientSheet.Row\)](#) ,

[Sheet<int, StakeActionPointCoefficientSheet.Row>.Add\(int, StakeActionPointCoefficientSheet.Row\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.TryGetValue\(int, out StakeActionPointCoefficientSheet.Row\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Add\(KeyValuePair<int, StakeActionPointCoefficientSheet.Row>\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Contains\(KeyValuePair<int, StakeActionPointCoefficientSheet.Row>\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.CopyTo\(KeyValuePair<int, StakeActionPointCoefficientSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Remove\(KeyValuePair<int, StakeActionPointCoefficientSheet.Row>\)](#) ,  
[Sheet<int, StakeActionPointCoefficientSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[SheetsExtensions.FindLevelByStakedAmount\(IStakeRewardSheet, Address, FungibleAssetValue\)](#) ,  
[SheetsExtensions.GetActionPointByStaking\(StakeActionPointCoefficientSheet, int, int, int\)](#)

## Constructors

### StakeActionPointCoefficientSheet()

```
public StakeActionPointCoefficientSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IStakeRewardRow> OrderedRows { get; }
```

### Property Value

[IReadOnlyList](#)<[IStakeRewardRow](#)>

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, StakeActionPointCoefficientSheet.Row value)
```

#### Parameters

key [int](#)

value [StakeActionPointCoefficientSheet.Row](#)

# Class StakeActionPointCoefficientSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class StakeActionPointCoefficientSheet.Row : SheetRow<int>, IStakeRewardRow
```

## Inheritance

[object](#) ↗ ← [SheetRow<int>](#) ← StakeActionPointCoefficientSheet.Row

## Implements

[IStakeRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) ↗ , [object.GetType\(\)](#) ↗ , [object.MemberwiseClone\(\)](#) ↗ ,  
[object.ReferenceEquals\(object, object\)](#) ↗ , [object.ToString\(\)](#) ↗

## Properties

### Coefficient

```
public int Coefficient { get; }
```

Property Value

[int](#) ↗

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Level

```
public int Level { get; }
```

Property Value

[int](#)

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class StakeRegularFixedRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

This sheet is used for setting the regular rewards for staking. The difference between this sheet and [StakeRegularRewardSheet](#) is that the [StakeRegularFixedRewardSheet.RewardInfo](#) of this sheet has a fixed count of reward.

```
[Serializable]
public class StakeRegularFixedRewardSheet : Sheet<int,
StakeRegularFixedRewardSheet.Row>, IDictionary<int,
StakeRegularFixedRewardSheet.Row>, ICollection<KeyValuePair<int,
StakeRegularFixedRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
StakeRegularFixedRewardSheet.Row>>, IEnumerable, IStakeRewardSheet, ISheet
```

## Inheritance

```
object ↪ Sheet<int ↪ , StakeRegularFixedRewardSheet.Row> ↪
StakeRegularFixedRewardSheet
```

## Implements

```
IDictionary<int ↪ , StakeRegularFixedRewardSheet.Row> ,
ICollection<KeyValuePair<int ↪ , StakeRegularFixedRewardSheet.Row>> ,
IEnumerable<KeyValuePair<int ↪ , StakeRegularFixedRewardSheet.Row>> ,
IEnumerable , IStakeRewardSheet , ISheet
```

## Inherited Members

```
Sheet<int, StakeRegularFixedRewardSheet.Row>.Name ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.OrderedList ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.First ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.Last ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.Keys ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.Values ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.Count ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.IsReadOnly ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.this[int] ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.Set(string, bool) ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.Set<T>(Sheet<int, T>, bool) ,
Sheet<int, StakeRegularFixedRewardSheet.Row>.GetEnumerator() ,
```

[Sheet<int, StakeRegularFixedRewardSheet.Row>.TryGetValue\(int, out StakeRegularFixedRewardSheet.Row, bool\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.AddRow\(int, StakeRegularFixedRewardSheet.Row\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Add\(int, StakeRegularFixedRewardSheet.Row\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.TryGetValue\(int, out StakeRegularFixedRewardSheet.Row\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Add\(KeyValuePair<int, StakeRegularFixedRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Contains\(KeyValuePair<int, StakeRegularFixedRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.CopyTo\(KeyValuePair<int, StakeRegularFixedRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Remove\(KeyValuePair<int, StakeRegularFixedRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeRegularFixedRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[SheetsExtensions.FindLevelByStakedAmount\(IStakeRewardSheet, Address, FungibleAssetValue\)](#)

## Constructors

### StakeRegularFixedRewardSheet()

```
public StakeRegularFixedRewardSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IStakeRewardRow> OrderedRows { get; }
```

Property Value

[IReadOnlyList](#)<[IStakeRewardRow](#)>

## Methods

AddRow(int, Row)

```
protected override void AddRow(int key, StakeRegularFixedRewardSheet.Row value)
```

Parameters

key [int](#)

value [StakeRegularFixedRewardSheet.Row](#)

# Class StakeRegularFixedRewardSheet.RewardInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeRegularFixedRewardSheet.RewardInfo
```

## Inheritance

[object](#) ← StakeRegularFixedRewardSheet.RewardInfo

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RewardInfo(int, int)

```
public RewardInfo(int itemId, int count)
```

#### Parameters

itemId [int](#)

count [int](#)

### RewardInfo(params string[])

```
public RewardInfo(params string[] fields)
```

#### Parameters

`fields string[][]`

## Fields

### Count

`public readonly int Count`

#### Field Value

`int`

### ItemId

`public readonly int ItemId`

#### Field Value

`int`

## Methods

### Equals(RewardInfo)

`protected bool Equals(StakeRegularFixedRewardSheet.RewardInfo other)`

#### Parameters

`other StakeRegularFixedRewardSheet.RewardInfo`

#### Returns

`bool`

## Equals(object)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object obj)
```

Parameters

obj [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

# Class StakeRegularFixedRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeRegularFixedRewardSheet.Row : SheetRow<int>, IStakeRewardRow
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← StakeRegularFixedRewardSheet.Row

## Implements

[IStakeRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

### Level

```
public int Level { get; }
```

#### Property Value

[int](#)

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#)

## Rewards

```
public List<StakeRegularFixedRewardSheet.RewardInfo> Rewards { get; }
```

Property Value

[List](#)<[StakeRegularFixedRewardSheet.RewardInfo](#)>

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class StakeRegularRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

This sheet is used for setting the regular rewards for staking. The difference between this sheet and [StakeRegularFixedRewardSheet](#) is that the [StakeRegularRewardSheet.RewardInfo](#) of this sheet has a rate and a reward type. The count of reward is calculated by the rate and the amount of staked currency.

```
[Serializable]
public class StakeRegularRewardSheet : Sheet<int, StakeRegularRewardSheet.Row>,
IDictionary<int, StakeRegularRewardSheet.Row>, ICollection<KeyValuePair<int,
StakeRegularRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
StakeRegularRewardSheet.Row>>, IEnumerable, IStakeRewardSheet, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StakeRegularRewardSheet.Row>](#) ← [StakeRegularRewardSheet](#)

## Implements

[IDictionary<int, StakeRegularRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, StakeRegularRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, StakeRegularRewardSheet.Row>>](#), [IEnumerable](#),  
[IStakeRewardSheet](#), [ISheet](#)

## Inherited Members

[Sheet<int, StakeRegularRewardSheet.Row>.Name](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.First](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Last](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Keys](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Values](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Count](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.TryGetValue\(int, out](#)  
[StakeRegularRewardSheet.Row, bool\)](#) ,

[Sheet<int, StakeRegularRewardSheet.Row>.AddRow\(int, StakeRegularRewardSheet.Row\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Add\(int, StakeRegularRewardSheet.Row\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.TryGetValue\(int, out StakeRegularRewardSheet.Row\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Add\(KeyValuePair<int, StakeRegularRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Contains\(KeyValuePair<int, StakeRegularRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.CopyTo\(KeyValuePair<int, StakeRegularRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Remove\(KeyValuePair<int, StakeRegularRewardSheet.Row>\)](#) ,  
[Sheet<int, StakeRegularRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Extension Methods

[SheetsExtensions.FindLevelByStakedAmount\(IStakeRewardSheet, Address, FungibleAssetValue\)](#)

## Constructors

### StakeRegularRewardSheet()

```
public StakeRegularRewardSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IStakeRewardRow> OrderedRows { get; }
```

### Property Value

[IReadOnlyList](#)<[IStakeRewardRow](#)>

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, StakeRegularRewardSheet.Row value)
```

#### Parameters

key [int](#)

value [StakeRegularRewardSheet.Row](#)

# Class StakeRegularRewardSheet.RewardInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class StakeRegularRewardSheet.RewardInfo
```

## Inheritance

[object](#) ← StakeRegularRewardSheet.RewardInfo

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

RewardInfo(int, int, StakeRewardType, string, int?, decimal)

```
public RewardInfo(int itemId, int rate = 0, StakeRegularRewardSheet.StakeRewardType  
type = StakeRewardType.Item, string currencyTicker = null, int?  
currencyDecimalPlaces = null, decimal decimalRate = 0)
```

## Parameters

itemId [int](#)

rate [int](#)

type [StakeRegularRewardSheet.StakeRewardType](#)

currencyTicker [string](#)

currencyDecimalPlaces [int](#)?

decimalRate [decimal](#)

## RewardInfo(params string[])

```
public RewardInfo(params string[] fields)
```

### Parameters

fields [string](#)[]

## Fields

### CurrencyDecimalPlaces

The decimal places of currency. This field is only used when [Type](#) is [Currency](#).

```
public readonly int? CurrencyDecimalPlaces
```

### Field Value

[int](#)?

### CurrencyTicker

The ticker of currency. This field is only used when [Type](#) is [Currency](#).

```
public readonly string CurrencyTicker
```

### Field Value

[string](#)[]

### DecimalRate

The decimal rate of reward. This field is used from [v3](#)(in blockchain state) or later. This field is set to [Rate](#) when [ClaimStakeReward.V2](#) or earlier.

```
public readonly decimal DecimalRate
```

Field Value

[decimal](#)

## ItemId

```
public readonly int ItemId
```

Field Value

[int](#)

## Rate

The rate of reward.

```
[Obsolete("This field is used from `ClaimStakeReward.V2` or earlier. Use  
`DecimalRate` instead. Deprecated this field because we need to support  
decimal rate.")]  
public readonly int Rate
```

Field Value

[int](#)

## Tradable

```
public readonly bool Tradable
```

Field Value

[bool](#)

# Type

`public readonly StakeRegularRewardSheet.StakeRewardType Type`

## Field Value

[StakeRegularRewardSheet.StakeRewardType](#)

# Methods

## Equals(RewardInfo)

`protected bool Equals(StakeRegularRewardSheet.RewardInfo other)`

### Parameters

`other` [StakeRegularRewardSheet.RewardInfo](#)

### Returns

[bool](#) ↗

## Equals(object)

Determines whether the specified object is equal to the current object.

`public override bool Equals(object obj)`

### Parameters

`obj` [object](#) ↗

The object to compare with the current object.

### Returns

[bool](#) ↗

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

# Class StakeRegularRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakeRegularRewardSheet.Row : SheetRow<int>, IStakeRewardRow
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < StakeRegularRewardSheet.Row

## Implements

[IStakeRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

#### Property Value

[int](#)

### Level

```
public int Level { get; }
```

#### Property Value

[int](#)

## RequiredGold

```
public long RequiredGold { get; }
```

Property Value

[long](#)

## Rewards

```
public List<StakeRegularRewardSheet.RewardInfo> Rewards { get; }
```

Property Value

[List](#)<[StakeRegularRewardSheet.RewardInfo](#)>

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Enum StakeRegularRewardSheet.StakeRewardType

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

The reward type of stake. Do not use the whole element of this enum at once in actions.  
(e.g., count, loop) Do versioning when you change or remove the elements of this enum.

```
public enum StakeRegularRewardSheet.StakeRewardType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Currency = 2

Item = 0

Rune = 1

# Class StatBuffSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatBuffSheet : Sheet<int, StatBuffSheet.Row>, IDictionary<int,
StatBuffSheet.Row>, ICollection<KeyValuePair<int, StatBuffSheet.Row>>,
IEnumerable<KeyValuePair<int, StatBuffSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, StatBuffSheet.Row>](#) ← [StatBuffSheet](#)

## Implements

[IDictionary<int, StatBuffSheet.Row>](#),  
[ICollection<KeyValuePair<int, StatBuffSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, StatBuffSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, StatBuffSheet.Row>.Name](#) , [Sheet<int, StatBuffSheet.Row>.OrderedList](#) ,  
[Sheet<int, StatBuffSheet.Row>.First](#) , [Sheet<int, StatBuffSheet.Row>.Last](#) ,  
[Sheet<int, StatBuffSheet.Row>.Keys](#) , [Sheet<int, StatBuffSheet.Row>.Values](#) ,  
[Sheet<int, StatBuffSheet.Row>.Count](#) , [Sheet<int, StatBuffSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, StatBuffSheet.Row>.this\[int\]](#) , [Sheet<int, StatBuffSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.TryGetValue\(int, out StatBuffSheet.Row, bool\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.AddRow\(int, StatBuffSheet.Row\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Add\(int, StatBuffSheet.Row\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.TryGetValue\(int, out StatBuffSheet.Row\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Add\(KeyValuePair<int, StatBuffSheet.Row>\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Contains\(KeyValuePair<int, StatBuffSheet.Row>\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.CopyTo\(KeyValuePair<int, StatBuffSheet.Row>\[\], int\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Remove\(KeyValuePair<int, StatBuffSheet.Row>\)](#) ,  
[Sheet<int, StatBuffSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### StatBuffSheet()

```
public StatBuffSheet()
```

# Class StatBuffSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StatBuffSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← StatBuffSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Chance

100: 100%

```
public int Chance { get; }
```

### Property Value

[int](#)

### Duration

Turn count.

```
public int Duration { get; }
```

### Property Value

int

# GroupId

```
public int GroupId { get; }
```

## Property Value

int ↗

Id

```
public int Id { get; }
```

## Property Value

int ☈

# IsEnhanceable

```
public bool IsEnhanceable { get; }
```

## Property Value

## bool

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## MaxStack

`public int MaxStack { get; }`

Property Value

[int](#)

## OperationType

`public StatModifier.OperationType OperationType { get; }`

Property Value

[StatModifier.OperationType](#)

## StatType

`public StatType StatType { get; }`

Property Value

[StatType](#)

## TargetType

`public SkillTargetType TargetType { get; }`

Property Value

## [SkillTargetType](#)

### Value

```
public long Value { get; }
```

Property Value

[long](#)

### Methods

**Set(IReadOnlyList<string>)**

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

**fields** [IReadOnlyList](#)<[string](#)>

# Class SweepRequiredCPSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class SweepRequiredCPSheet : Sheet<int, SweepRequiredCPSheet.Row>,
IDictionary<int, SweepRequiredCPSheet.Row>, ICollection<KeyValuePair<int,
SweepRequiredCPSheet.Row>>, IEnumerable<KeyValuePair<int,
SweepRequiredCPSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, SweepRequiredCPSheet.Row>](#) ← [SweepRequiredCPSheet](#)

## Implements

[IDictionary<int, SweepRequiredCPSheet.Row>](#),  
[ICollection<KeyValuePair<int, SweepRequiredCPSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, SweepRequiredCPSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, SweepRequiredCPSheet.Row>.Name](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.OrderedList](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.First](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Last](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Keys](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Values](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Count](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.this\[int\]](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.TryGetValue\(int, out SweepRequiredCPSheet.Row, bool\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.AddRow\(int, SweepRequiredCPSheet.Row\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Add\(int, SweepRequiredCPSheet.Row\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, SweepRequiredCPSheet.Row>.TryGetValue\(int, out SweepRequiredCPSheet.Row\)](#),  
,

[Sheet<int, SweepRequiredCPSheet.Row>.Add\(KeyValuePair<int, SweepRequiredCPSheet.Row>\)](#),  
[Sheet<int, SweepRequiredCPSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Contains\(KeyValuePair<int, SweepRequiredCPSheet.Row>\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.CopyTo\(KeyValuePair<int, SweepRequiredCPSheet.Row>\[\], int\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Remove\(KeyValuePair<int, SweepRequiredCPSheet.Row>\)](#) ,  
[Sheet<int, SweepRequiredCPSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SweepRequiredCPSheet()

```
public SweepRequiredCPSheet()
```

# Class SweepRequiredCPSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class SweepRequiredCPSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← SweepRequiredCPSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

### RequiredCP

```
public int RequiredCP { get; }
```

Property Value

[int](#)

## StageId

```
public int StageId { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class TableExtensions

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public static class TableExtensions
```

## Inheritance

[object](#) ← TableExtensions

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ParseBigInteger(string)

```
public static BigInteger ParseBigInteger(string value)
```

#### Parameters

value [string](#)

#### Returns

[BigInteger](#)

### ParseBool(string, bool)

```
public static bool ParseBool(string value, bool defaultValue)
```

#### Parameters

`value` [string](#)

`defaultValue` [bool](#)

Returns

[bool](#)

## ParseDecimal(string)

```
public static decimal ParseDecimal(string value)
```

Parameters

`value` [string](#)

Returns

[decimal](#)

## ParseDecimal(string, decimal)

```
public static decimal ParseDecimal(string value, decimal defaultValue)
```

Parameters

`value` [string](#)

`defaultValue` [decimal](#)

Returns

[decimal](#)

## ParseInt(string)

```
public static int ParseInt(string value)
```

Parameters

value [string](#)

Returns

[int](#)

## ParseInt(string, int)

```
public static int ParseInt(string value, int defaultValue)
```

Parameters

value [string](#)

defaultValue [int](#)

Returns

[int](#)

## ParseLong(string)

```
public static long ParseLong(string value)
```

Parameters

value [string](#)

Returns

[long](#)

## ParseLong(string, long)

```
public static long ParseLong(string value, long defaultValue)
```

Parameters

value [string](#)

defaultValue [long](#)

Returns

[long](#)

## TryParseDecimal(string, out decimal)

```
public static bool TryParseDecimal(string value, out decimal result)
```

Parameters

value [string](#)

result [decimal](#)

Returns

[bool](#)

## TryParseFloat(string, out float)

```
public static bool TryParseFloat(string value, out float result)
```

Parameters

value [string](#)

result [float](#)

Returns

[bool](#)

## TryParseInt(string, out int)

```
public static bool TryParseInt(string value, out int result)
```

Parameters

[value](#) [string](#)

[result](#) [int](#)

Returns

[bool](#)

## TryParseLong(string, out long)

```
public static bool TryParseLong(string value, out long result)
```

Parameters

[value](#) [string](#)

[result](#) [long](#)

Returns

[bool](#)

# Class TradeQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TradeQuestSheet : Sheet<int, TradeQuestSheet.Row>, IDictionary<int, TradeQuestSheet.Row>, ICollection<KeyValuePair<int, TradeQuestSheet.Row>>, IEnumerable<KeyValuePair<int, TradeQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, TradeQuestSheet.Row>](#) ← TradeQuestSheet

## Implements

[IDictionary<int, TradeQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, TradeQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, TradeQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, TradeQuestSheet.Row>.Name](#) , [Sheet<int, TradeQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, TradeQuestSheet.Row>.First](#) , [Sheet<int, TradeQuestSheet.Row>.Last](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Keys](#) , [Sheet<int, TradeQuestSheet.Row>.Values](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Count](#) , [Sheet<int, TradeQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, TradeQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.TryGetValue\(int, out TradeQuestSheet.Row, bool\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.AddRow\(int, TradeQuestSheet.Row\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Add\(int, TradeQuestSheet.Row\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.TryGetValue\(int, out TradeQuestSheet.Row\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Add\(KeyValuePair<int, TradeQuestSheet.Row>\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Contains\(KeyValuePair<int, TradeQuestSheet.Row>\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.CopyTo\(KeyValuePair<int, TradeQuestSheet.Row>\[\], int\)](#) ,  
[Sheet<int, TradeQuestSheet.Row>.Remove\(KeyValuePair<int, TradeQuestSheet.Row>\)](#) ,

[Sheet<int, TradeQuestSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### TradeQuestSheet()

```
public TradeQuestSheet()
```

# Class TradeQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class TradeQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← TradeQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Serialize\(\)](#) ,  
[QuestSheet.Row.Deserialize\(Dictionary\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Type

```
public TradeType Type { get; }
```

### Property Value

[TradeType](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class UnlockCombinationSlotCostSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class UnlockCombinationSlotCostSheet : Sheet<int,
UnlockCombinationSlotCostSheet.Row>, IDictionary<int,
UnlockCombinationSlotCostSheet.Row>, ICollection<KeyValuePair<int,
UnlockCombinationSlotCostSheet.Row>>, IEnumerable<KeyValuePair<int,
UnlockCombinationSlotCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, UnlockCombinationSlotCostSheet.Row>](#) ←

UnlockCombinationSlotCostSheet

## Implements

[IDictionary<int, UnlockCombinationSlotCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, UnlockCombinationSlotCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, UnlockCombinationSlotCostSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Name](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.First](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Last](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Keys](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Values](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Count](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.this\[int\]](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.TryGetValue\(int, out](#)  
[UnlockCombinationSlotCostSheet.Row, bool\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.AddRow\(int,](#)  
[UnlockCombinationSlotCostSheet.Row\)](#) ,

[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Add\(int, UnlockCombinationSlotCostSheet.Row\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.TryGetValue\(int, out UnlockCombinationSlotCostSheet.Row\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Add\(KeyValuePair<int, UnlockCombinationSlotCostSheet.Row>\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Contains\(KeyValuePair<int, UnlockCombinationSlotCostSheet.Row>\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.CopyTo\(KeyValuePair<int, UnlockCombinationSlotCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Remove\(KeyValuePair<int, UnlockCombinationSlotCostSheet.Row>\)](#) ,  
[Sheet<int, UnlockCombinationSlotCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### UnlockCombinationSlotCostSheet()

```
public UnlockCombinationSlotCostSheet()
```

# Class UnlockCombinationSlotCostSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class UnlockCombinationSlotCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [UnlockCombinationSlotCostSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### CrystalPrice

```
public int CrystalPrice
```

#### Field Value

[int](#)

### GoldenDustPrice

```
public int GoldenDustPrice
```

#### Field Value

[int](#)

## NcgPrice

`public BigInteger NcgPrice`

Field Value

[BigInteger](#)

## RubyDustPrice

`public int RubyDustPrice`

Field Value

[int](#)

## SlotId

`public int SlotId`

Field Value

[int](#)

## Properties

### Key

`public override int Key { get; }`

Property Value

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class WeeklyArenaRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WeeklyArenaRewardSheet : Sheet<int, WeeklyArenaRewardSheet.Row>,
IDictionary<int, WeeklyArenaRewardSheet.Row>, ICollection<KeyValuePair<int,
WeeklyArenaRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
WeeklyArenaRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WeeklyArenaRewardSheet.Row>](#) ← [WeeklyArenaRewardSheet](#)

## Implements

[IDictionary<int, WeeklyArenaRewardSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, WeeklyArenaRewardSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, WeeklyArenaRewardSheet.Row>>](#) , [IEnumerable](#) ,  
[ISheet](#)

## Inherited Members

[Sheet<int, WeeklyArenaRewardSheet.Row>.Name](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.First](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Last](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Keys](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Values](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Count](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.TryGetValue\(int, out WeeklyArenaRewardSheet.Row, bool\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.AddRow\(int, WeeklyArenaRewardSheet.Row\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Add\(int, WeeklyArenaRewardSheet.Row\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, WeeklyArenaRewardSheet.Row>.TryGetValue\(int, out WeeklyArenaRewardSheet.Row\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Add\(KeyValuePair<int, WeeklyArenaRewardSheet.Row>\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Contains\(KeyValuePair<int, WeeklyArenaRewardSheet.Row>\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.CopyTo\(KeyValuePair<int, WeeklyArenaRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Remove\(KeyValuePair<int, WeeklyArenaRewardSheet.Row>\)](#) ,  
[Sheet<int, WeeklyArenaRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### WeeklyArenaRewardSheet()

```
public WeeklyArenaRewardSheet()
```

# Class

# WeeklyArenaRewardSheet.RewardData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WeeklyArenaRewardSheet.RewardData : StageSheet.RewardData
```

## Inheritance

[object](#) ← [StageSheet.RewardData](#) ← WeeklyArenaRewardSheet.RewardData

## Inherited Members

[StageSheet.RewardData.ItemId](#) , [StageSheet.RewardData.Ratio](#) ,  
[StageSheet.RewardData.Min](#) , [StageSheet.RewardData.Max](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RewardData(int, decimal, int, int, int)

```
public RewardData(int itemId, decimal ratio, int min, int max, int requiredLevel)
```

## Parameters

itemId [int](#)

ratio [decimal](#)

min [int](#)

max [int](#)

requiredLevel [int](#)

# Properties

## RequiredLevel

```
public int RequiredLevel { get; }
```

Property Value

[int](#) ↗

# Class WeeklyArenaRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WeeklyArenaRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← WeeklyArenaRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

# Reward

```
public WeeklyArenaRewardSheet.RewardData Reward { get; }
```

Property Value

[WeeklyArenaRewardSheet.RewardData](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList<string>](#)

# Class WorldBossActionPatternSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldBossActionPatternSheet : Sheet<int,
WorldBossActionPatternSheet.Row>, IDictionary<int, WorldBossActionPatternSheet.Row>,
ICollection<KeyValuePair<int, WorldBossActionPatternSheet.Row>>,
IEnumerable<KeyValuePair<int, WorldBossActionPatternSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int](#), [WorldBossActionPatternSheet.Row](#) < WorldBossActionPatternSheet

## Implements

[IDictionary](#)<[int](#), [WorldBossActionPatternSheet.Row](#)>,  
[ICollection](#)<[KeyValuePair](#)<[int](#), [WorldBossActionPatternSheet.Row](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [WorldBossActionPatternSheet.Row](#)>>, [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, WorldBossActionPatternSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.First](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.TryGetValue\(int, out WorldBossActionPatternSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.AddRow\(int, WorldBossActionPatternSheet.Row\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Add\(int, WorldBossActionPatternSheet.Row\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.ContainsKey\(int\)](#) ,

[Sheet<int, WorldBossActionPatternSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.TryGetValue\(int, out WorldBossActionPatternSheet.Row\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Add\(KeyValuePair<int, WorldBossActionPatternSheet.Row>\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Contains\(KeyValuePair<int, WorldBossActionPatternSheet.Row>\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.CopyTo\(KeyValuePair<int, WorldBossActionPatternSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Remove\(KeyValuePair<int, WorldBossActionPatternSheet.Row>\)](#) ,  
[Sheet<int, WorldBossActionPatternSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### WorldBossActionPatternSheet()

```
public WorldBossActionPatternSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, WorldBossActionPatternSheet.Row value)
```

## Parameters

key [int](#)

value [WorldBossActionPatternSheet.Row](#)

# Class WorldBossActionPatternSheet.ActionPatternData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class WorldBossActionPatternSheet.ActionPatternData
```

## Inheritance

[object](#) ← WorldBossActionPatternSheet.ActionPatternData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Fields

### SkillIds

```
public readonly List<int> SkillIds
```

#### Field Value

[List](#)<[int](#)>

### Wave

```
public int Wave
```

#### Field Value

[int ↗](#)

# Class WorldBossActionPatternSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldBossActionPatternSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < WorldBossActionPatternSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BossId

```
public int BossId
```

### Field Value

[int](#)

## Patterns

```
public List<WorldBossActionPatternSheet.ActionPatternData> Patterns
```

### Field Value

[List](#)<[WorldBossActionPatternSheet.ActionPatternData](#)>

# Properties

## Key

```
public override int Key { get; }
```

## Property Value

[int](#)

# Methods

## EndOfSheetInitialize()

```
public override void EndOfSheetInitialize()
```

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class WorldBossBattleRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossBattleRewardSheet : Sheet<int,
WorldBossBattleRewardSheet.Row>, IDictionary<int, WorldBossBattleRewardSheet.Row>,
ICollection<KeyValuePair<int, WorldBossBattleRewardSheet.Row>>,
IEnumerable<KeyValuePair<int, WorldBossBattleRewardSheet.Row>>, IEnumerable,
IWorldBossRewardSheet, ISheet
```

## Inheritance

[object](#) ← [Sheet<int](#), [WorldBossBattleRewardSheet.Row](#) < WorldBossBattleRewardSheet

## Implements

[IDictionary](#)<[int](#), [WorldBossBattleRewardSheet.Row](#)>,  
[ICollection](#)<[KeyValuePair](#)<[int](#), [WorldBossBattleRewardSheet.Row](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [WorldBossBattleRewardSheet.Row](#)>>, [IEnumerable](#),  
[IWorldBossRewardSheet](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldBossBattleRewardSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.First](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.TryGetValue\(int, out WorldBossBattleRewardSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.AddRow\(int, WorldBossBattleRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.Add\(int, WorldBossBattleRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossBattleRewardSheet.Row>.ContainsKey\(int\)](#) ,

```
Sheet<int, WorldBossBattleRewardSheet.Row>.Remove(int) ,
Sheet<int, WorldBossBattleRewardSheet.Row>.TryGetValue(int, out
WorldBossBattleRewardSheet.Row) ,
Sheet<int, WorldBossBattleRewardSheet.Row>.Add(KeyValuePair<int,
WorldBossBattleRewardSheet.Row>) ,
Sheet<int, WorldBossBattleRewardSheet.Row>.Clear() ,
Sheet<int, WorldBossBattleRewardSheet.Row>.Contains(KeyValuePair<int,
WorldBossBattleRewardSheet.Row>) ,
Sheet<int, WorldBossBattleRewardSheet.Row>.CopyTo(KeyValuePair<int,
WorldBossBattleRewardSheet.Row>[], int) ,
Sheet<int, WorldBossBattleRewardSheet.Row>.Remove(KeyValuePair<int,
WorldBossBattleRewardSheet.Row>) ,
Sheet<int, WorldBossBattleRewardSheet.Row>.Serialize() , object.Equals(object)✉ ,
object.Equals(object, object)✉ , object.GetHashCode()✉ , object.GetType()✉ ,
object.MemberwiseClone()✉ , object.ReferenceEquals(object, object)✉ , object.ToString()✉
```

## Constructors

### WorldBossBattleRewardSheet()

```
public WorldBossBattleRewardSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IWorldBossRewardRow> OrderedRows { get; }
```

### Property Value

[IReadOnlyList](#)✉ <[IWorldBossRewardRow](#)>

# Class WorldBossBattleRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossBattleRewardSheet.Row : SheetRow<int>, IWorldBossRewardRow
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < WorldBossBattleRewardSheet.Row

## Implements

[IWorldBossRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Id

```
public int Id
```

### Field Value

[int](#)

### RuneMax

```
public int RuneMax
```

### Field Value

[int](#)

## RuneMin

```
public int RuneMin
```

Field Value

[int](#)

## Properties

### BossId

```
public int BossId { get; }
```

Property Value

[int](#)

### Circle

```
public int Circle { get; }
```

Property Value

[int](#)

### Crystal

```
public int Crystal { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Rank

```
public int Rank { get; }
```

Property Value

[int](#)

## Rune

```
public int Rune { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields  [IReadOnlyList<string>](#)

## SetRune(IRandom)

**public void SetRune(IRandom random)**

### Parameters

**random IRandom**

# Class WorldBossCharacterSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldBossCharacterSheet : Sheet<int, WorldBossCharacterSheet.Row>,
IDictionary<int, WorldBossCharacterSheet.Row>, ICollection<KeyValuePair<int,
WorldBossCharacterSheet.Row>>, IEnumerable<KeyValuePair<int,
WorldBossCharacterSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldBossCharacterSheet.Row>](#) ← [WorldBossCharacterSheet](#)

## Implements

[IDictionary<int, WorldBossCharacterSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldBossCharacterSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldBossCharacterSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, WorldBossCharacterSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.First](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.TryGetValue\(int, out](#)  
[WorldBossCharacterSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.AddRow\(int, WorldBossCharacterSheet.Row\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Add\(int, WorldBossCharacterSheet.Row\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldBossCharacterSheet.Row>.Remove\(int\)](#) ,

```
Sheet<int, WorldBossCharacterSheet.Row>.TryGetValue\(int, out
WorldBossCharacterSheet.Row\) ,
Sheet<int, WorldBossCharacterSheet.Row>.Add\(KeyValuePair<int,
WorldBossCharacterSheet.Row>\) ,
Sheet<int, WorldBossCharacterSheet.Row>.Clear\(\) ,
Sheet<int, WorldBossCharacterSheet.Row>.Contains\(KeyValuePair<int,
WorldBossCharacterSheet.Row>\) ,
Sheet<int, WorldBossCharacterSheet.Row>.CopyTo\(KeyValuePair<int,
WorldBossCharacterSheet.Row>\[\], int\) ,
Sheet<int, WorldBossCharacterSheet.Row>.Remove\(KeyValuePair<int,
WorldBossCharacterSheet.Row>\) ,
Sheet<int, WorldBossCharacterSheet.Row>.Serialize\(\) , object.Equals\(object\) ,
object.Equals\(object, object\) , object.GetHashCode\(\) , object.GetType\(\) ,
object.MemberwiseClone\(\) , object.ReferenceEquals\(object, object\) , object.ToString\(\)
```

## Constructors

### WorldBossCharacterSheet()

```
public WorldBossCharacterSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, WorldBossCharacterSheet.Row value)
```

## Parameters

key [int](#)

value [WorldBossCharacterSheet.Row](#)

# Class WorldBossCharacterSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldBossCharacterSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← WorldBossCharacterSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BossId

```
public int BossId { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

# WaveStats

```
public List<WorldBossCharacterSheet.WaveStatData> WaveStats { get; }
```

## Property Value

[List](#) <[WorldBossCharacterSheet](#).[WaveStatData](#)>

## Methods

### EndOfSheetInitialize()

```
public override void EndOfSheetInitialize()
```

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#) <[string](#)>

# Class WorldBossCharacterSheet.WaveStatData

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class WorldBossCharacterSheet.WaveStatData
```

## Inheritance

[object](#) ← WorldBossCharacterSheet.WaveStatData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Extension Methods

[WorldBossStatSheetExtension.ToStats\(WorldBossCharacterSheet.WaveStatData\)](#).

## Properties

### ATK

```
public decimal ATK { get; set; }
```

Property Value

[decimal](#)

### CRI

```
public decimal CRI { get; set; }
```

Property Value

[decimal](#) ↗

## DEF

```
public decimal DEF { get; set; }
```

Property Value

[decimal](#) ↗

## ElementType

```
public ElementType ElementType { get; set; }
```

Property Value

[ElementType](#)

## EnrageSkillId

```
public int EnrageSkillId { get; set; }
```

Property Value

[int](#) ↗

## EnrageTurn

```
public int EnrageTurn { get; set; }
```

Property Value

[int](#) ↗

## HIT

```
public decimal HIT { get; set; }
```

Property Value

[decimal](#) ↗

## HP

```
public decimal HP { get; set; }
```

Property Value

[decimal](#) ↗

## Level

```
public int Level { get; set; }
```

Property Value

[int](#) ↗

## SPD

```
public decimal SPD { get; set; }
```

Property Value

[decimal](#) ↗

## TurnLimit

```
public int TurnLimit { get; set; }
```

Property Value

[int ↗](#)

## Wave

```
public int Wave { get; set; }
```

Property Value

[int ↗](#)

# Class WorldBossGlobalHpSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldBossGlobalHpSheet : Sheet<int, WorldBossGlobalHpSheet.Row>,
IDictionary<int, WorldBossGlobalHpSheet.Row>, ICollection<KeyValuePair<int,
WorldBossGlobalHpSheet.Row>>, IEnumerable<KeyValuePair<int,
WorldBossGlobalHpSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldBossGlobalHpSheet.Row>](#) ← [WorldBossGlobalHpSheet](#)

## Implements

[IDictionary<int, WorldBossGlobalHpSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldBossGlobalHpSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldBossGlobalHpSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, WorldBossGlobalHpSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.First](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.TryGetValue\(int, out WorldBossGlobalHpSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.AddRow\(int, WorldBossGlobalHpSheet.Row\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Add\(int, WorldBossGlobalHpSheet.Row\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, WorldBossGlobalHpSheet.Row>.TryGetValue\(int, out WorldBossGlobalHpSheet.Row\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Add\(KeyValuePair<int, WorldBossGlobalHpSheet.Row>\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Contains\(KeyValuePair<int, WorldBossGlobalHpSheet.Row>\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.CopyTo\(KeyValuePair<int, WorldBossGlobalHpSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Remove\(KeyValuePair<int, WorldBossGlobalHpSheet.Row>\)](#) ,  
[Sheet<int, WorldBossGlobalHpSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### WorldBossGlobalHpSheet()

```
public WorldBossGlobalHpSheet()
```

# Class WorldBossGlobalHpSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossGlobalHpSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← WorldBossGlobalHpSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Hp

```
public BigInteger Hp
```

#### Field Value

[BigInteger](#)

### Level

```
public int Level
```

#### Field Value

[int](#)

# Properties

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class WorldBossKillRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossKillRewardSheet : Sheet<int, WorldBossKillRewardSheet.Row>,  
IDictionary<int, WorldBossKillRewardSheet.Row>, ICollection<KeyValuePair<int,  
WorldBossKillRewardSheet.Row>>, IEnumerable<KeyValuePair<int,  
WorldBossKillRewardSheet.Row>>, IEnumerable, IWorldBossRewardSheet, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldBossKillRewardSheet.Row>](#) ← [WorldBossKillRewardSheet](#)

## Implements

[IDictionary<int, WorldBossKillRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldBossKillRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldBossKillRewardSheet.Row>>](#), [IEnumerable](#),  
[IWorldBossRewardSheet](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldBossKillRewardSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.First](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.TryGetValue\(int, out](#)  
[WorldBossKillRewardSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.AddRow\(int, WorldBossKillRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Add\(int, WorldBossKillRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, WorldBossKillRewardSheet.Row>.TryGetValue\(int, out WorldBossKillRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Add\(KeyValuePair<int, WorldBossKillRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Contains\(KeyValuePair<int, WorldBossKillRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.CopyTo\(KeyValuePair<int, WorldBossKillRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Remove\(KeyValuePair<int, WorldBossKillRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossKillRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### WorldBossKillRewardSheet()

```
public WorldBossKillRewardSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IWorldBossRewardRow> OrderedRows { get; }
```

## Property Value

[IReadOnlyList](#) ↴ <[IWorldBossRewardRow](#)>

# Class WorldBossKillRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossKillRewardSheet.Row : SheetRow<int>, IWorldBossRewardRow
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < WorldBossKillRewardSheet.Row

## Implements

[IWorldBossRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Id

```
public int Id
```

### Field Value

[int](#)

### RuneMax

```
public int RuneMax
```

### Field Value

[int](#)

## RuneMin

```
public int RuneMin
```

Field Value

[int](#)

## Properties

### BossId

```
public int BossId { get; }
```

Property Value

[int](#)

### Circle

```
public int Circle { get; }
```

Property Value

[int](#)

### Crystal

```
public int Crystal { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Rank

```
public int Rank { get; }
```

Property Value

[int](#)

## Rune

```
public int Rune { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields  [IReadOnlyList<string>](#)

## SetRune(IRandom)

**public void SetRune(IRandom random)**

### Parameters

**random IRandom**

# Class WorldBossListSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossListSheet : Sheet<int, WorldBossListSheet.Row>,
IDictionary<int, WorldBossListSheet.Row>, ICollection<KeyValuePair<int,
WorldBossListSheet.Row>>, IEnumerable<KeyValuePair<int, WorldBossListSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldBossListSheet.Row>](#) ← [WorldBossListSheet](#)

## Implements

[IDictionary<int, WorldBossListSheet.Row>](#) ,  
[ICollection<KeyValuePair<int, WorldBossListSheet.Row>>](#) ,  
[IEnumerable<KeyValuePair<int, WorldBossListSheet.Row>>](#) , [IEnumerable](#) , [ISheet](#)

## Inherited Members

[Sheet<int, WorldBossListSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossListSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossListSheet.Row>.First](#) , [Sheet<int, WorldBossListSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Keys](#) , [Sheet<int, WorldBossListSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossListSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossListSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.TryGetValue\(int, out WorldBossListSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.AddRow\(int, WorldBossListSheet.Row\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Add\(int, WorldBossListSheet.Row\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.TryGetValue\(int, out WorldBossListSheet.Row\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Add\(KeyValuePair<int, WorldBossListSheet.Row>\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldBossListSheet.Row>.Contains\(KeyValuePair<int, WorldBossListSheet.Row>\)](#) ,

```
Sheet<int, WorldBossListSheet.Row>.CopyTo(KeyValuePair<int, WorldBossListSheet.Row>
[], int) ,
Sheet<int, WorldBossListSheet.Row>.Remove(KeyValuePair<int, WorldBossListSheet.Row>),
Sheet<int, WorldBossListSheet.Row>.Serialize() , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Extension Methods

```
SheetsExtensions.FindPreviousRaidIdByBlockIndex(WorldBossListSheet, long) ,
SheetsExtensions.FindPreviousRowByBlockIndex(WorldBossListSheet, long) ,
SheetsExtensions.FindRaidIdByBlockIndex(WorldBossListSheet, long) ,
SheetsExtensions.FindRowByBlockIndex(WorldBossListSheet, long).
```

## Constructors

### WorldBossListSheet()

```
public WorldBossListSheet()
```

# Class WorldBossListSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossListSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← WorldBossListSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AdditionalTicketPrice

```
public int AdditionalTicketPrice
```

#### Field Value

[int](#)

### BossId

```
public int BossId
```

#### Field Value

[int](#)

## EndedBlockIndex

```
public long EndedBlockIndex
```

Field Value

[long](#) ↗

## EntranceFee

```
public int EntranceFee
```

Field Value

[int](#) ↗

## Id

```
public int Id
```

Field Value

[int](#) ↗

## MaxPurchaseCount

```
public int MaxPurchaseCount
```

Field Value

[int](#) ↗

## StartedBlockIndex

```
public long StartedBlockIndex
```

Field Value

[long](#)

## TicketPrice

```
public int TicketPrice
```

Field Value

[int](#)

## Properties

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class WorldBossRankRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossRankRewardSheet : Sheet<int, WorldBossRankRewardSheet.Row>,  
IDictionary<int, WorldBossRankRewardSheet.Row>, ICollection<KeyValuePair<int,  
WorldBossRankRewardSheet.Row>>, IEnumerable<KeyValuePair<int,  
WorldBossRankRewardSheet.Row>>, IEnumerable, IWorldBossRewardSheet, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldBossRankRewardSheet.Row>](#) ← [WorldBossRankRewardSheet](#)

## Implements

[IDictionary<int, WorldBossRankRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldBossRankRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldBossRankRewardSheet.Row>>](#), [IEnumerable](#),  
[IWorldBossRewardSheet](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldBossRankRewardSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.First](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.TryGetValue\(int, out  
WorldBossRankRewardSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.AddRow\(int,  
WorldBossRankRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Add\(int, WorldBossRankRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, WorldBossRankRewardSheet.Row>.TryGetValue\(int, out WorldBossRankRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Add\(KeyValuePair<int, WorldBossRankRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Contains\(KeyValuePair<int, WorldBossRankRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.CopyTo\(KeyValuePair<int, WorldBossRankRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Remove\(KeyValuePair<int, WorldBossRankRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossRankRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### WorldBossRankRewardSheet()

```
public WorldBossRankRewardSheet()
```

## Properties

### OrderedRows

```
public IReadOnlyList<IWorldBossRewardRow> OrderedRows { get; }
```

## Property Value

[IReadOnlyList](#) ↴ <[IWorldBossRewardRow](#)>

# Class WorldBossRankRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossRankRewardSheet.Row : SheetRow<int>, IWorldBossRewardRow
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < WorldBossRankRewardSheet.Row

## Implements

[IWorldBossRewardRow](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Id

```
public int Id
```

### Field Value

[int](#)

## Properties

### BossId

```
public int BossId { get; }
```

Property Value

[int ↗](#)

## Circle

`public int Circle { get; }`

Property Value

[int ↗](#)

## Crystal

`public int Crystal { get; }`

Property Value

[int ↗](#)

## Key

`public override int Key { get; }`

Property Value

[int ↗](#)

## Rank

`public int Rank { get; }`

Property Value

[int](#)

## Rune

```
public int Rune { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class WorldBossRankingRewardSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossRankingRewardSheet : Sheet<int,  
WorldBossRankingRewardSheet.Row>, IDictionary<int, WorldBossRankingRewardSheet.Row>,  
ICollection<KeyValuePair<int, WorldBossRankingRewardSheet.Row>>,  
IEnumerable<KeyValuePair<int, WorldBossRankingRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldBossRankingRewardSheet.Row>](#) ←  
WorldBossRankingRewardSheet

## Implements

[IDictionary<int, WorldBossRankingRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldBossRankingRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldBossRankingRewardSheet.Row>>](#),  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldBossRankingRewardSheet.Row>.Name](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.First](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Last](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Keys](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Values](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Count](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.TryGetValue\(int, out  
WorldBossRankingRewardSheet.Row, bool\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.AddRow\(int,  
WorldBossRankingRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Add\(int,  
WorldBossRankingRewardSheet.Row\)](#) ,

[Sheet<int, WorldBossRankingRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.TryGetValue\(int, out WorldBossRankingRewardSheet.Row\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Add\(KeyValuePair<int, WorldBossRankingRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Contains\(KeyValuePair<int, WorldBossRankingRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.CopyTo\(KeyValuePair<int, WorldBossRankingRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Remove\(KeyValuePair<int, WorldBossRankingRewardSheet.Row>\)](#) ,  
[Sheet<int, WorldBossRankingRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### WorldBossRankingRewardSheet()

```
public WorldBossRankingRewardSheet()
```

## Methods

### FindRow(int, int, int)

```
public WorldBossRankingRewardSheet.Row FindRow(int bossId, int ranking, int rate)
```

## Parameters

bossId [int](#) ↴

ranking [int](#) ↴

rate [int](#) ↴

## Returns

[WorldBossRankingRewardSheet.Row](#)

# Class WorldBossRankingRewardSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public class WorldBossRankingRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < WorldBossRankingRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BossId

```
public int BossId
```

#### Field Value

[int](#)

### Crystal

```
public int Crystal
```

#### Field Value

[int](#)

## Id

```
public int Id
```

## Field Value

[int](#)

## Materials

```
public List<(int itemId, int quantity)> Materials
```

## Field Value

[List](#)<(int id, int count)>

## RankingMax

```
public int RankingMax
```

## Field Value

[int](#)

## RankingMin

```
public int RankingMin
```

## Field Value

[int](#)

## RateMax

```
public int RateMax
```

Field Value

[int](#)

## RateMin

```
public int RateMin
```

Field Value

[int](#)

## Runes

```
public List<WorldBossRankingRewardSheet.Row.RuneInfo> Runes
```

Field Value

[List](#)<[WorldBossRankingRewardSheet](#).[Row](#).[RuneInfo](#)>

## Properties

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

## GetRewards(RuneSheet, MaterialItemSheet)

```
public List<FungibleAssetValue> GetRewards(RuneSheet runeSheet,  
MaterialItemSheet materialSheet)
```

### Parameters

runeSheet [RuneSheet](#)

materialSheet [MaterialItemSheet](#)

### Returns

[List](#)<FungibleAssetValue>

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Struct WorldBossRankingRewardSheet.Row.RuneInfo

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public struct WorldBossRankingRewardSheet.Row.RuneInfo
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### RuneInfo(int, int)

```
public RuneInfo(int id, int qty)
```

## Parameters

**id** [int](#)

**qty** [int](#)

## Fields

### Runeld

```
public int RuneId
```

## Field Value

[int](#) ↗

## RuneQty

`public int RuneQty`

Field Value

[int](#) ↗

# Class WorldBossStatSheetExtension

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
public static class WorldBossStatSheetExtension
```

## Inheritance

[object](#) ← WorldBossStatSheetExtension

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### ToStats(WaveStatData)

```
public static StatsMap ToStats(this WorldBossCharacterSheet.WaveStatData statData)
```

#### Parameters

statData [WorldBossCharacterSheet.WaveStatData](#)

#### Returns

[StatsMap](#)

# Class WorldQuestSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldQuestSheet : Sheet<int, WorldQuestSheet.Row>, IDictionary<int, WorldQuestSheet.Row>, ICollection<KeyValuePair<int, WorldQuestSheet.Row>>, IEnumerable<KeyValuePair<int, WorldQuestSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldQuestSheet.Row>](#) ← [WorldQuestSheet](#)

## Implements

[IDictionary<int, WorldQuestSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldQuestSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldQuestSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldQuestSheet.Row>.Name](#) , [Sheet<int, WorldQuestSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldQuestSheet.Row>.First](#) , [Sheet<int, WorldQuestSheet.Row>.Last](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Keys](#) , [Sheet<int, WorldQuestSheet.Row>.Values](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Count](#) , [Sheet<int, WorldQuestSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldQuestSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.TryGetValue\(int, out WorldQuestSheet.Row, bool\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.AddRow\(int, WorldQuestSheet.Row\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Add\(int, WorldQuestSheet.Row\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.TryGetValue\(int, out WorldQuestSheet.Row\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Add\(KeyValuePair<int, WorldQuestSheet.Row>\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Contains\(KeyValuePair<int, WorldQuestSheet.Row>\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.CopyTo\(KeyValuePair<int, WorldQuestSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldQuestSheet.Row>.Remove\(KeyValuePair<int, WorldQuestSheet.Row>\)](#) ,

[Sheet<int, WorldQuestSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### WorldQuestSheet()

```
public WorldQuestSheet()
```

# Class WorldQuestSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldQuestSheet.Row : QuestSheet.Row, IState
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [QuestSheet.Row](#) ← WorldQuestSheet.Row

## Implements

[IState](#)

## Inherited Members

[QuestSheet.Row.Key](#) , [QuestSheet.Row.Id](#) , [QuestSheet.Row.Goal](#) ,  
[QuestSheet.Row.QuestRewardId](#) , [QuestSheet.Row.Set\(IReadOnlyList<string>\)](#) ,  
[QuestSheet.Row.Serialize\(\)](#) , [QuestSheet.Row.Deserialize\(Dictionary\)](#) ,  
[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Class WorldSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldSheet : Sheet<int, WorldSheet.Row>, IDictionary<int,
WorldSheet.Row>, ICollection<KeyValuePair<int, WorldSheet.Row>>,
IEnumerable<KeyValuePair<int, WorldSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldSheet.Row>](#) ← [WorldSheet](#)

## Implements

[IDictionary<int, WorldSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldSheet.Row>.Name](#) , [Sheet<int, WorldSheet.Row>.OrderedList](#) ,  
[Sheet<int, WorldSheet.Row>.First](#) , [Sheet<int, WorldSheet.Row>.Last](#) ,  
[Sheet<int, WorldSheet.Row>.Keys](#) , [Sheet<int, WorldSheet.Row>.Values](#) ,  
[Sheet<int, WorldSheet.Row>.Count](#) , [Sheet<int, WorldSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldSheet.Row>.this\[int\]](#) , [Sheet<int, WorldSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldSheet.Row>.TryGetValue\(int, out WorldSheet.Row, bool\)](#) ,  
[Sheet<int, WorldSheet.Row>.AddRow\(int, WorldSheet.Row\)](#) ,  
[Sheet<int, WorldSheet.Row>.Add\(int, WorldSheet.Row\)](#) ,  
[Sheet<int, WorldSheet.Row>.ContainsKey\(int\)](#) , [Sheet<int, WorldSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, WorldSheet.Row>.TryGetValue\(int, out WorldSheet.Row\)](#) ,  
[Sheet<int, WorldSheet.Row>.Add\(KeyValuePair<int, WorldSheet.Row>\)](#) ,  
[Sheet<int, WorldSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldSheet.Row>.Contains\(KeyValuePair<int, WorldSheet.Row>\)](#) ,  
[Sheet<int, WorldSheet.Row>.CopyTo\(KeyValuePair<int, WorldSheet.Row>\[\], int\)](#) ,  
[Sheet<int, WorldSheet.Row>.Remove\(KeyValuePair<int, WorldSheet.Row>\)](#) ,  
[Sheet<int, WorldSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Constructors

## WorldSheet()

```
public WorldSheet()
```

# Class WorldSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← WorldSheet.Row

## Derived

[EventDungeonSheet.Row](#)

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Name

`public string Name { get; }`

### Property Value

[string](#)

## StageBegin

`public int StageBegin { get; }`

### Property Value

[int](#)

## StageEnd

`public int StageEnd { get; }`

### Property Value

[int](#)

## StagesCount

`public int StagesCount { get; }`

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class WorldUnlockSheet

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldUnlockSheet : Sheet<int, WorldUnlockSheet.Row>, IDictionary<int, WorldUnlockSheet.Row>, ICollection<KeyValuePair<int, WorldUnlockSheet.Row>>, IEnumerable<KeyValuePair<int, WorldUnlockSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, WorldUnlockSheet.Row>](#) ← [WorldUnlockSheet](#)

## Implements

[IDictionary<int, WorldUnlockSheet.Row>](#),  
[ICollection<KeyValuePair<int, WorldUnlockSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, WorldUnlockSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, WorldUnlockSheet.Row>.Name](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.OrderedList](#) , [Sheet<int, WorldUnlockSheet.Row>.First](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Last](#) , [Sheet<int, WorldUnlockSheet.Row>.Keys](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Values](#) , [Sheet<int, WorldUnlockSheet.Row>.Count](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.this\[int\]](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.TryGetValue\(int, out WorldUnlockSheet.Row, bool\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.AddRow\(int, WorldUnlockSheet.Row\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Add\(int, WorldUnlockSheet.Row\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.TryGetValue\(int, out WorldUnlockSheet.Row\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Add\(KeyValuePair<int, WorldUnlockSheet.Row>\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.Contains\(KeyValuePair<int, WorldUnlockSheet.Row>\)](#) ,  
[Sheet<int, WorldUnlockSheet.Row>.CopyTo\(KeyValuePair<int, WorldUnlockSheet.Row>\[\], int\)](#) ,

```
Sheet<int, WorldUnlockSheet.Row>.Remove(KeyValuePair<int, WorldUnlockSheet.Row>),  
Sheet<int, WorldUnlockSheet.Row>.Serialize(), object.Equals(object) ↴ ,  
object.Equals(object, object) ↴ , object.GetHashCode() ↴ , object.GetType() ↴ ,  
object.MemberwiseClone() ↴ , object.ReferenceEquals(object, object) ↴ , object.ToString() ↴
```

## Constructors

### WorldUnlockSheet()

```
public WorldUnlockSheet()
```

## Methods

### TryGetUnlockedInformation(int, int, out List<int>)

```
public bool TryGetUnlockedInformation(int clearedWorldId, int clearedStageId, out  
List<int> worldIdsToUnlock)
```

## Parameters

clearedWorldId [int ↴](#)

clearedStageId [int ↴](#)

worldIdsToUnlock [List<int>](#)

## Returns

[bool ↴](#)

# Class WorldUnlockSheet.Row

Namespace: [Nekoyume.TableData](#)

Assembly: Lib9c.dll

```
[Serializable]
public class WorldUnlockSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← WorldUnlockSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CRYSTAL

```
public int CRYSTAL { get; }
```

Property Value

[int](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#) ↗

## StageId

```
public int StageId { get; }
```

Property Value

[int](#) ↗

## WorldId

```
public int WorldId { get; }
```

Property Value

[int](#) ↗

## WorldIdToUnlock

```
public int WorldIdToUnlock { get; }
```

Property Value

[int](#) ↗

## Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList<string>](#)

# Namespace Nekoyume.TableData. AdventureBoss Classes

[AdventureBossContributionRewardSheet](#)

[AdventureBossContributionRewardSheet.Row](#)

[AdventureBossFloorFirstRewardSheet](#)

[AdventureBossFloorFirstRewardSheet.Row](#)

[AdventureBossFloorPointSheet](#)

[AdventureBossFloorPointSheet.Row](#)

[AdventureBossFloorSheet](#)

[AdventureBossFloorSheet.RewardData](#)

[AdventureBossFloorSheet.Row](#)

[AdventureBossFloorWaveSheet](#)

[AdventureBossFloorWaveSheet.MonsterData](#)

[AdventureBossFloorWaveSheet.Row](#)

[AdventureBossFloorWaveSheet.WaveData](#)

[AdventureBossNcgRewardRatioSheet](#)

[AdventureBossNcgRewardRatioSheet.Row](#)

[AdventureBossSheet](#)

[AdventureBossSheet.RewardAmountData](#)

[AdventureBossSheet.RewardRatioData](#)

[AdventureBossSheet.Row](#)

[AdventureBossUnlockFloorCostSheet](#)

[AdventureBossUnlockFloorCostSheet.Row](#)

[AdventureBossWantedRewardSheet](#)

[AdventureBossWantedRewardSheet.Row](#)

# Class AdventureBossContributionRewardSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossContributionRewardSheet : Sheet<int,
AdventureBossContributionRewardSheet.Row>, IDictionary<int,
AdventureBossContributionRewardSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossContributionRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossContributionRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossContributionRewardSheet.Row>](#) ←  
AdventureBossContributionRewardSheet

## Implements

[IDictionary<int, AdventureBossContributionRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossContributionRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossContributionRewardSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossContributionRewardSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.TryGetValue\(int, out AdventureBossContributionRewardSheet.Row, bool\)](#) ,

[Sheet<int, AdventureBossContributionRewardSheet.Row>.AddRow\(int, AdventureBossContributionRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Add\(int, AdventureBossContributionRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.TryGetValue\(int, out AdventureBossContributionRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Add\(KeyValuePair<int, AdventureBossContributionRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Contains\(KeyValuePair<int, AdventureBossContributionRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossContributionRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Remove\(KeyValuePair<int, AdventureBossContributionRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossContributionRewardSheet.Row>.Serialize\(\)](#) ,  
[object.Equals\(object\) ↴](#) , [object.Equals\(object, object\) ↴](#) , [object.GetHashCode\(\) ↴](#) ,  
[object.GetType\(\) ↴](#) , [object.MemberwiseClone\(\) ↴](#) , [object.ReferenceEquals\(object, object\) ↴](#) ,  
[object.ToString\(\) ↴](#)

## Constructors

### AdventureBossContributionRewardSheet()

```
public AdventureBossContributionRewardSheet()
```

# Class AdventureBossContributionRewardSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossContributionRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossContributionRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AdventureBossId

```
public int AdventureBossId
```

Field Value

[int](#)

### Id

```
public int Id
```

Field Value

[int](#)

## Rewards

`public List<AdventureBossSheet.RewardRatioData> Rewards`

### Field Value

[List](#)<[AdventureBossSheet.RewardRatioData](#)>

## Properties

### Key

`public override int Key { get; }`

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

`public override void Set(IReadOnlyList<string> fields)`

### Parameters

`fields` [IReadOnlyList](#)<[string](#)>

# Class AdventureBossFloorFirstRewardSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorFirstRewardSheet : Sheet<int,
AdventureBossFloorFirstRewardSheet.Row>, IDictionary<int,
AdventureBossFloorFirstRewardSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossFloorFirstRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossFloorFirstRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

```
object ↪ Sheet<int ↪ , AdventureBossFloorFirstRewardSheet.Row> ↪
AdventureBossFloorFirstRewardSheet
```

## Implements

```
IDictionary<int ↪ , AdventureBossFloorFirstRewardSheet.Row>,
ICollection<KeyValuePair<int ↪ , AdventureBossFloorFirstRewardSheet.Row>>,
IEnumerable<KeyValuePair<int ↪ , AdventureBossFloorFirstRewardSheet.Row>>,
IEnumerable, ISheet
```

## Inherited Members

```
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Name ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.OrderedList ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.First ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Last ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Keys ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Values ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Count ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.IsReadOnly ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.this[int] ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Set(string, bool) ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Set<T>(Sheet<int, T>, bool) ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.GetEnumerator() ,
Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.TryGetValue(int, out
AdventureBossFloorFirstRewardSheet.Row, bool) ,
```

[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.AddRow\(int, AdventureBossFloorFirstRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Add\(int, AdventureBossFloorFirstRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.TryGetValue\(int, out AdventureBossFloorFirstRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Add\(KeyValuePair<int, AdventureBossFloorFirstRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Contains\(KeyValuePair<int, AdventureBossFloorFirstRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossFloorFirstRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Remove\(KeyValuePair<int, AdventureBossFloorFirstRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorFirstRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AdventureBossFloorFirstRewardSheet()

```
public AdventureBossFloorFirstRewardSheet()
```

# Class AdventureBossFloorFirstRewardSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossFloorFirstRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossFloorFirstRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### FloorId

```
public int FloorId
```

### Field Value

[int](#)

## Rewards

```
public List<AdventureBossSheet.RewardAmountData> Rewards
```

### Field Value

[List](#)<[AdventureBossSheet.RewardAmountData](#)>

# Properties

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

# Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class AdventureBossFloorPointSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorPointSheet : Sheet<int,
AdventureBossFloorPointSheet.Row>, IDictionary<int,
AdventureBossFloorPointSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossFloorPointSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossFloorPointSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossFloorPointSheet.Row>](#) ←

AdventureBossFloorPointSheet

## Implements

[IDictionary<int, AdventureBossFloorPointSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossFloorPointSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossFloorPointSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossFloorPointSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.TryGetValue\(int, out AdventureBossFloorPointSheet.Row, bool\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.AddRow\(int, AdventureBossFloorPointSheet.Row\)](#) ,

[Sheet<int, AdventureBossFloorPointSheet.Row>.Add\(int, AdventureBossFloorPointSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.TryGetValue\(int, out AdventureBossFloorPointSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Add\(KeyValuePair<int, AdventureBossFloorPointSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Contains\(KeyValuePair<int, AdventureBossFloorPointSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossFloorPointSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Remove\(KeyValuePair<int, AdventureBossFloorPointSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorPointSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AdventureBossFloorPointSheet()

```
public AdventureBossFloorPointSheet()
```

# Class AdventureBossFloorPointSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorPointSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < AdventureBossFloorPointSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### FloorId

```
public int FloorId
```

#### Field Value

[int](#)

### MaxPoint

```
public int MaxPoint
```

#### Field Value

[int](#)

## MinPoint

```
public int MinPoint
```

Field Value

[int](#)

## Properties

Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class AdventureBossFloorSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorSheet : Sheet<int, AdventureBossFloorSheet.Row>,
IDictionary<int, AdventureBossFloorSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossFloorSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossFloorSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossFloorSheet.Row>](#) ← AdventureBossFloorSheet

## Implements

[IDictionary<int, AdventureBossFloorSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossFloorSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossFloorSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossFloorSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.TryGetValue\(int, out AdventureBossFloorSheet.Row, bool\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.AddRow\(int, AdventureBossFloorSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Add\(int, AdventureBossFloorSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, AdventureBossFloorSheet.Row>.TryGetValue\(int, out AdventureBossFloorSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Add\(KeyValuePair<int, AdventureBossFloorSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Contains\(KeyValuePair<int, AdventureBossFloorSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossFloorSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Remove\(KeyValuePair<int, AdventureBossFloorSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### AdventureBossFloorSheet()

```
public AdventureBossFloorSheet()
```

# Class AdventureBossFloorSheet.RewardData

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossFloorSheet.RewardData
```

## Inheritance

[object](#) ← AdventureBossFloorSheet.RewardData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RewardData(string, int, int, int, int)

```
public RewardData(string itemType, int itemId, int min, int max, int ratio)
```

## Parameters

itemType [string](#)

itemId [int](#)

min [int](#)

max [int](#)

ratio [int](#)

## Properties

## ItemId

```
public int ItemId { get; }
```

Property Value

[int](#)

## ItemType

```
public string ItemType { get; }
```

Property Value

[string](#)

## Max

```
public int Max { get; }
```

Property Value

[int](#)

## Min

```
public int Min { get; }
```

Property Value

[int](#)

## Ratio

```
public int Ratio { get; }
```

Property Value

[int](#) ↗

# Class AdventureBossFloorSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossFloorSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### AdventureBossId

```
public int AdventureBossId { get; }
```

### Property Value

[int](#)

### BGM

```
public string BGM { get; }
```

### Property Value

[string](#)

## Background

```
public string Background { get; }
```

Property Value

[string](#)

## EnemyInitialStatModifiers

```
public List<StatModifier> EnemyInitialStatModifiers { get; }
```

Property Value

[List](#)<[StatModifier](#)>

## Floor

```
public int Floor { get; }
```

Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## MaxDropItem

```
public int MaxDropItem { get; }
```

Property Value

[int](#)

## MinDropItem

```
public int MinDropItem { get; }
```

Property Value

[int](#)

## Rewards

```
public List<AdventureBossFloorSheet.RewardData> Rewards { get; }
```

Property Value

[List](#)<[AdventureBossFloorSheet.RewardData](#)>

## StageBuffSkillId

```
public int StageBuffSkillId { get; }
```

Property Value

[int](#)

## TurnLimit

```
public int TurnLimit { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class AdventureBossFloorWaveSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorWaveSheet : Sheet<int,
AdventureBossFloorWaveSheet.Row>, IDictionary<int, AdventureBossFloorWaveSheet.Row>,
ICollection<KeyValuePair<int, AdventureBossFloorWaveSheet.Row>>,
IEnumerable<KeyValuePair<int, AdventureBossFloorWaveSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossFloorWaveSheet.Row>](#) ←  
AdventureBossFloorWaveSheet

## Implements

[IDictionary<int, AdventureBossFloorWaveSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossFloorWaveSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossFloorWaveSheet.Row>>](#),  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossFloorWaveSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.TryGetValue\(int, out AdventureBossFloorWaveSheet.Row, bool\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.AddRow\(int, AdventureBossFloorWaveSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Add\(int, AdventureBossFloorWaveSheet.Row\)](#) ,

[Sheet<int, AdventureBossFloorWaveSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.TryGetValue\(int, out AdventureBossFloorWaveSheet.Row\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Add\(KeyValuePair<int, AdventureBossFloorWaveSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Contains\(KeyValuePair<int, AdventureBossFloorWaveSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossFloorWaveSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Remove\(KeyValuePair<int, AdventureBossFloorWaveSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossFloorWaveSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### AdventureBossFloorWaveSheet()

```
public AdventureBossFloorWaveSheet()
```

# Class AdventureBossFloorWaveSheet.MonsterData

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorWaveSheet.MonsterData
```

## Inheritance

[object](#) ← AdventureBossFloorWaveSheet.MonsterData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### MonsterData(int, int, int)

```
public MonsterData(int characterId, int level, int count)
```

## Parameters

characterId [int](#)

level [int](#)

count [int](#)

## Properties

### CharacterId

```
public int CharacterId { get; }
```

Property Value

[int](#)

Count

```
public int Count { get; }
```

Property Value

[int](#)

Level

```
public int Level { get; }
```

Property Value

[int](#)

# Class AdventureBossFloorWaveSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossFloorWaveSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < AdventureBossFloorWaveSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### FloorId

```
public int FloorId { get; }
```

### Property Value

[int](#)

### HasBoss

```
public bool HasBoss { get; }
```

### Property Value

[bool](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## TotalMonsterIds

```
public List<int> TotalMonsterIds { get; }
```

Property Value

[List](#)<[int](#)>

## Waves

```
public List<AdventureBossFloorWaveSheet.WaveData> Waves { get; }
```

Property Value

[List](#)<[AdventureBossFloorWaveSheet](#).[WaveData](#)>

## Methods

### EndOfSheetInitialize()

```
public override void EndOfSheetInitialize()
```

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class AdventureBossFloorWaveSheet.WaveData

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossFloorWaveSheet.WaveData
```

## Inheritance

[object](#) ← AdventureBossFloorWaveSheet.WaveData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### WaveData(int, List<MonsterData>, bool)

```
public WaveData(int number, List<AdventureBossFloorWaveSheet.MonsterData> monsters,  
bool hasBoss)
```

## Parameters

number [int](#)

monsters [List](#)<[AdventureBossFloorWaveSheet.MonsterData](#)>

hasBoss [bool](#)

## Properties

### HasBoss

```
public bool HasBoss { get; }
```

Property Value

[bool](#)

## Monsters

```
public List<AdventureBossFloorWaveSheet.MonsterData> Monsters { get; }
```

Property Value

[List](#)<[AdventureBossFloorWaveSheet](#).[MonsterData](#)>

## Number

```
public int Number { get; }
```

Property Value

[int](#)

# Class AdventureBossNcgRewardRatioSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossNcgRewardRatioSheet : Sheet<int,
AdventureBossNcgRewardRatioSheet.Row>, IDictionary<int,
AdventureBossNcgRewardRatioSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossNcgRewardRatioSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossNcgRewardRatioSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossNcgRewardRatioSheet.Row>](#) ←  
AdventureBossNcgRewardRatioSheet

## Implements

[IDictionary<int, AdventureBossNcgRewardRatioSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossNcgRewardRatioSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossNcgRewardRatioSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.TryGetValue\(int, out AdventureBossNcgRewardRatioSheet.Row, bool\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.AddRow\(int, AdventureBossNcgRewardRatioSheet.Row\)](#) ,

[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Add\(int, AdventureBossNcgRewardRatioSheet.Row\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.TryGetValue\(int, out AdventureBossNcgRewardRatioSheet.Row\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Add\(KeyValuePair<int, AdventureBossNcgRewardRatioSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Contains\(KeyValuePair<int, AdventureBossNcgRewardRatioSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossNcgRewardRatioSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Remove\(KeyValuePair<int, AdventureBossNcgRewardRatioSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossNcgRewardRatioSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AdventureBossNcgRewardRatioSheet()

```
public AdventureBossNcgRewardRatioSheet()
```

# Class AdventureBossNcgRewardRatioSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossNcgRewardRatioSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossNcgRewardRatioSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### ItemId

```
public int ItemId
```

#### Field Value

[int](#)

### Ratio

```
public decimal Ratio
```

#### Field Value

[decimal](#)

# Properties

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class AdventureBossSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossSheet : Sheet<int, AdventureBossSheet.Row>,
IDictionary<int, AdventureBossSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossSheet.Row>>, IEnumerable<KeyValuePair<int, AdventureBossSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossSheet.Row>](#) ← AdventureBossSheet

## Implements

[IDictionary<int, AdventureBossSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossSheet.Row>.First](#) , [Sheet<int, AdventureBossSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.TryGetValue\(int, out AdventureBossSheet.Row, bool\)](#) ,  
  
[Sheet<int, AdventureBossSheet.Row>.AddRow\(int, AdventureBossSheet.Row\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Add\(int, AdventureBossSheet.Row\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.TryGetValue\(int, out AdventureBossSheet.Row\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Add\(KeyValuePair<int, AdventureBossSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossSheet.Row>.Clear\(\)](#) ,

```
Sheet<int, AdventureBossSheet.Row>.Contains(KeyValuePair<int,
AdventureBossSheet.Row>) ,
Sheet<int, AdventureBossSheet.Row>.CopyTo(KeyValuePair<int,
AdventureBossSheet.Row>[], int) ,
Sheet<int, AdventureBossSheet.Row>.Remove(KeyValuePair<int,
AdventureBossSheet.Row>) ,
Sheet<int, AdventureBossSheet.Row>.Serialize() , object.Equals(object) ,
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()
```

## Constructors

### AdventureBossSheet()

```
public AdventureBossSheet()
```

# Class AdventureBossSheet.RewardAmountData

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossSheet.RewardAmountData
```

## Inheritance

[object](#) ← AdventureBossSheet.RewardAmountData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RewardAmountData(string, int, int)

```
public RewardAmountData(string itemType, int itemId, int amount)
```

## Parameters

itemType [string](#)

itemId [int](#)

amount [int](#)

## Properties

### Amount

```
public int Amount { get; }
```

Property Value

[int](#)

## ItemId

```
public int ItemId { get; }
```

Property Value

[int](#)

## ItemType

```
public string ItemType { get; }
```

Property Value

[string](#)

# Class AdventureBossSheet.RewardRatioData

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossSheet.RewardRatioData
```

## Inheritance

[object](#) ← AdventureBossSheet.RewardRatioData

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### RewardRatioData(string, int, int)

```
public RewardRatioData(string itemType, int itemId, int ratio)
```

## Parameters

itemType [string](#)

itemId [int](#)

ratio [int](#)

## Properties

### ItemId

```
public int ItemId { get; }
```

Property Value

[int](#)

## ItemType

```
public string ItemType { get; }
```

Property Value

[string](#)

## Ratio

```
public int Ratio { get; }
```

Property Value

[int](#)

# Class AdventureBossSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### BossId

```
public int BossId { get; }
```

### Property Value

[int](#)

### ExploreAp

```
public int ExploreAp { get; }
```

### Property Value

[int](#)

## Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## SweepAp

```
public int SweepAp { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class AdventureBossUnlockFloorCostSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossUnlockFloorCostSheet : Sheet<int,
AdventureBossUnlockFloorCostSheet.Row>, IDictionary<int,
AdventureBossUnlockFloorCostSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossUnlockFloorCostSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossUnlockFloorCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossUnlockFloorCostSheet.Row>](#) ←  
AdventureBossUnlockFloorCostSheet

## Implements

[IDictionary<int, AdventureBossUnlockFloorCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossUnlockFloorCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossUnlockFloorCostSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.TryGetValue\(int, out AdventureBossUnlockFloorCostSheet.Row, bool\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.AddRow\(int, AdventureBossUnlockFloorCostSheet.Row\)](#) ,

[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Add\(int, AdventureBossUnlockFloorCostSheet.Row\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.TryGetValue\(int, out AdventureBossUnlockFloorCostSheet.Row\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Add\(KeyValuePair<int, AdventureBossUnlockFloorCostSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Contains\(KeyValuePair<int, AdventureBossUnlockFloorCostSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossUnlockFloorCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Remove\(KeyValuePair<int, AdventureBossUnlockFloorCostSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossUnlockFloorCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AdventureBossUnlockFloorCostSheet()

```
public AdventureBossUnlockFloorCostSheet()
```

# Class AdventureBossUnlockFloorCostSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossUnlockFloorCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossUnlockFloorCostSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### FloorId

```
public int FloorId
```

#### Field Value

[int](#)

### GoldenDustPrice

```
public int GoldenDustPrice
```

#### Field Value

[int](#)

# NcgPrice

`public BigInteger NcgPrice`

Field Value

[BigInteger](#)

## Properties

Key

`public override int Key { get; }`

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

`public override void Set(IReadOnlyList<string> fields)`

Parameters

`fields` [IReadOnlyList](#)<[string](#)>

# Class AdventureBossWantedRewardSheet

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]
public class AdventureBossWantedRewardSheet : Sheet<int,
AdventureBossWantedRewardSheet.Row>, IDictionary<int,
AdventureBossWantedRewardSheet.Row>, ICollection<KeyValuePair<int,
AdventureBossWantedRewardSheet.Row>>, IEnumerable<KeyValuePair<int,
AdventureBossWantedRewardSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, AdventureBossWantedRewardSheet.Row>](#) ←  
AdventureBossWantedRewardSheet

## Implements

[IDictionary<int, AdventureBossWantedRewardSheet.Row>](#),  
[ICollection<KeyValuePair<int, AdventureBossWantedRewardSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, AdventureBossWantedRewardSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, AdventureBossWantedRewardSheet.Row>.Name](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.OrderedList](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.First](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Last](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Keys](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Values](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Count](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.this\[int\]](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.TryGetValue\(int, out AdventureBossWantedRewardSheet.Row, bool\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.AddRow\(int, AdventureBossWantedRewardSheet.Row\)](#) ,

[Sheet<int, AdventureBossWantedRewardSheet.Row>.Add\(int, AdventureBossWantedRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.TryGetValue\(int, out AdventureBossWantedRewardSheet.Row\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Add\(KeyValuePair<int, AdventureBossWantedRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Contains\(KeyValuePair<int, AdventureBossWantedRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.CopyTo\(KeyValuePair<int, AdventureBossWantedRewardSheet.Row>\[\], int\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Remove\(KeyValuePair<int, AdventureBossWantedRewardSheet.Row>\)](#) ,  
[Sheet<int, AdventureBossWantedRewardSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AdventureBossWantedRewardSheet()

```
public AdventureBossWantedRewardSheet()
```

# Class AdventureBossWantedRewardSheet.Row

Namespace: [Nekoyume.TableData.AdventureBoss](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class AdventureBossWantedRewardSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← AdventureBossWantedRewardSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AdventureBossId

```
public int AdventureBossId
```

#### Field Value

[int](#)

### FixedReward

```
public AdventureBossSheet.RewardRatioData FixedReward
```

#### Field Value

[AdventureBossSheet.RewardRatioData](#)

## Id

```
public int Id
```

### Field Value

[int](#)

## RandomRewards

```
public List<AdventureBossSheet.RewardRatioData> RandomRewards
```

### Field Value

[List](#)<[AdventureBossSheet.RewardRatioData](#)>

## Properties

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields  [IReadOnlyList<string>](#)

# Namespace Nekoyume.TableData.Crystal

## Classes

[CrystalEquipmentGrindingSheet](#)

[CrystalEquipmentGrindingSheet.Row](#)

[CrystalFluctuationSheet](#)

[CrystalFluctuationSheet.Row](#)

[CrystalHammerPointSheet](#)

[CrystalHammerPointSheet.Row](#)

[CrystalMaterialCostSheet](#)

[CrystalMaterialCostSheet.Row](#)

[CrystalMonsterCollectionMultiplierSheet](#)

[CrystalMonsterCollectionMultiplierSheet.Row](#)

[CrystalRandomBuffSheet](#)

[CrystalRandomBuffSheet.Row](#)

[CrystalStageBuffGachaSheet](#)

[CrystalStageBuffGachaSheet.Row](#)

## Enums

[CrystalFluctuationSheet.ServiceType](#)

[CrystalRandomBuffSheet.Row.BuffRank](#)

# Class CrystalEquipmentGrindingSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalEquipmentGrindingSheet : Sheet<int,
CrystalEquipmentGrindingSheet.Row>, IDictionary<int,
CrystalEquipmentGrindingSheet.Row>, ICollection<KeyValuePair<int,
CrystalEquipmentGrindingSheet.Row>>, IEnumerable<KeyValuePair<int,
CrystalEquipmentGrindingSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalEquipmentGrindingSheet.Row>](#) ←  
CrystalEquipmentGrindingSheet

## Implements

[IDictionary<int, CrystalEquipmentGrindingSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalEquipmentGrindingSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalEquipmentGrindingSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Name](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.First](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Last](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Values](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Count](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.TryGetValue\(int, out CrystalEquipmentGrindingSheet.Row, bool\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.AddRow\(int, CrystalEquipmentGrindingSheet.Row\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Add\(int, CrystalEquipmentGrindingSheet.Row\)](#) ,

[Sheet<int, CrystalEquipmentGrindingSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.TryGetValue\(int, out CrystalEquipmentGrindingSheet.Row\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Add\(KeyValuePair<int, CrystalEquipmentGrindingSheet.Row>\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Contains\(KeyValuePair<int, CrystalEquipmentGrindingSheet.Row>\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.CopyTo\(KeyValuePair<int, CrystalEquipmentGrindingSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Remove\(KeyValuePair<int, CrystalEquipmentGrindingSheet.Row>\)](#) ,  
[Sheet<int, CrystalEquipmentGrindingSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### CrystalEquipmentGrindingSheet()

```
public CrystalEquipmentGrindingSheet()
```

# Class CrystalEquipmentGrindingSheet.Row

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalEquipmentGrindingSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalEquipmentGrindingSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### CRYSTAL

```
public int CRYSTAL
```

#### Field Value

[int](#)

### EnchantBaseId

```
public int EnchantBaseId
```

#### Field Value

[int](#)

## ItemId

```
public int ItemId
```

### Field Value

[int](#)

## RewardMaterials

```
public List<(int materialId, int count)> RewardMaterials
```

### Field Value

[List](#)<(int id, int count)>

## Properties

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields  [IReadOnlyList<string>](#)

# Class CrystalFluctuationSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalFluctuationSheet : Sheet<int, CrystalFluctuationSheet.Row>,  
IDictionary<int, CrystalFluctuationSheet.Row>, ICollection<KeyValuePair<int,  
CrystalFluctuationSheet.Row>>, IEnumerable<KeyValuePair<int,  
CrystalFluctuationSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalFluctuationSheet.Row>](#) ← [CrystalFluctuationSheet](#)

## Implements

[IDictionary<int, CrystalFluctuationSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalFluctuationSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalFluctuationSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CrystalFluctuationSheet.Row>.Name](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.First](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Last](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Values](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Count](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.TryGetValue\(int, out CrystalFluctuationSheet.Row, bool\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.AddRow\(int, CrystalFluctuationSheet.Row\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Add\(int, CrystalFluctuationSheet.Row\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, CrystalFluctuationSheet.Row>.TryGetValue\(int, out CrystalFluctuationSheet.Row\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Add\(KeyValuePair<int, CrystalFluctuationSheet.Row>\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Contains\(KeyValuePair<int, CrystalFluctuationSheet.Row>\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.CopyTo\(KeyValuePair<int, CrystalFluctuationSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Remove\(KeyValuePair<int, CrystalFluctuationSheet.Row>\)](#) ,  
[Sheet<int, CrystalFluctuationSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### CrystalFluctuationSheet()

```
public CrystalFluctuationSheet()
```

# Class CrystalFluctuationSheet.Row

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalFluctuationSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalFluctuationSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BlockInterval

```
public long BlockInterval
```

#### Field Value

[long](#)

### Id

```
public int Id
```

#### Field Value

[int](#)

## MaximumRate

```
public int MaximumRate
```

Field Value

[int](#)

## MinimumRate

```
public int MinimumRate
```

Field Value

[int](#)

## Type

```
public CrystalFluctuationSheet.ServiceType Type
```

Field Value

[CrystalFluctuationSheet.ServiceType](#)

## Properties

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Enum CrystalFluctuationSheet.ServiceType

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public enum CrystalFluctuationSheet.ServiceType
```

## Extension Methods

[StateExtensions.Serialize\(Enum\)](#)

## Fields

Combination = 0

# Class CrystalHammerPointSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalHammerPointSheet : Sheet<int, CrystalHammerPointSheet.Row>,  
IDictionary<int, CrystalHammerPointSheet.Row>, ICollection<KeyValuePair<int,  
CrystalHammerPointSheet.Row>>, IEnumerable<KeyValuePair<int,  
CrystalHammerPointSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalHammerPointSheet.Row>](#) ← [CrystalHammerPointSheet](#)

## Implements

[IDictionary<int, CrystalHammerPointSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalHammerPointSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalHammerPointSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CrystalHammerPointSheet.Row>.Name](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.First](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Last](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Values](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Count](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.TryGetValue\(int, out  
CrystalHammerPointSheet.Row, bool\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.AddRow\(int, CrystalHammerPointSheet.Row\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Add\(int, CrystalHammerPointSheet.Row\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, CrystalHammerPointSheet.Row>.TryGetValue\(int, out CrystalHammerPointSheet.Row\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Add\(KeyValuePair<int, CrystalHammerPointSheet.Row>\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Contains\(KeyValuePair<int, CrystalHammerPointSheet.Row>\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.CopyTo\(KeyValuePair<int, CrystalHammerPointSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Remove\(KeyValuePair<int, CrystalHammerPointSheet.Row>\)](#) ,  
[Sheet<int, CrystalHammerPointSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### CrystalHammerPointSheet()

```
public CrystalHammerPointSheet()
```

# Class CrystalHammerPointSheet.Row

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalHammerPointSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalHammerPointSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CRYSTAL

```
public int CRYSTAL { get; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## MaxPoint

```
public int MaxPoint { get; }
```

Property Value

[int](#)

## RecipId

```
public int RecipeId { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CrystalMaterialCostSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CrystalMaterialCostSheet : Sheet<int, CrystalMaterialCostSheet.Row>,
IDictionary<int, CrystalMaterialCostSheet.Row>, ICollection<KeyValuePair<int,
CrystalMaterialCostSheet.Row>>, IEnumerable<KeyValuePair<int,
CrystalMaterialCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalMaterialCostSheet.Row>](#) ← [CrystalMaterialCostSheet](#)

## Implements

[IDictionary<int, CrystalMaterialCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalMaterialCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalMaterialCostSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CrystalMaterialCostSheet.Row>.Name](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.First](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Last](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Values](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Count](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.TryGetValue\(int, out CrystalMaterialCostSheet.Row, bool\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.AddRow\(int, CrystalMaterialCostSheet.Row\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Add\(int, CrystalMaterialCostSheet.Row\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, CrystalMaterialCostSheet.Row>.TryGetValue\(int, out CrystalMaterialCostSheet.Row\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Add\(KeyValuePair<int, CrystalMaterialCostSheet.Row>\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Contains\(KeyValuePair<int, CrystalMaterialCostSheet.Row>\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.CopyTo\(KeyValuePair<int, CrystalMaterialCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Remove\(KeyValuePair<int, CrystalMaterialCostSheet.Row>\)](#) ,  
[Sheet<int, CrystalMaterialCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### CrystalMaterialCostSheet()

```
public CrystalMaterialCostSheet()
```

# Class CrystalMaterialCostSheet.Row

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalMaterialCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalMaterialCostSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CRYSTAL

```
public int CRYSTAL { get; }
```

Property Value

[int](#)

### ItemId

```
public int ItemId { get; }
```

Property Value

[int](#)

# Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CrystalMonsterCollectionMultiplierSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CrystalMonsterCollectionMultiplierSheet : Sheet<int,
CrystalMonsterCollectionMultiplierSheet.Row>, IDictionary<int,
CrystalMonsterCollectionMultiplierSheet.Row>, ICollection<KeyValuePair<int,
CrystalMonsterCollectionMultiplierSheet.Row>>, IEnumerable<KeyValuePair<int,
CrystalMonsterCollectionMultiplierSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>](#) ←  
CrystalMonsterCollectionMultiplierSheet

## Implements

[IDictionary<int, CrystalMonsterCollectionMultiplierSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalMonsterCollectionMultiplierSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalMonsterCollectionMultiplierSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Name](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.First](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Last](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Values](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Count](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.TryGetValue\(int, out CrystalMonsterCollectionMultiplierSheet.Row, bool\)](#) ,

[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.AddRow\(int, CrystalMonsterCollectionMultiplierSheet.Row\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Add\(int, CrystalMonsterCollectionMultiplierSheet.Row\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.TryGetValue\(int, out CrystalMonsterCollectionMultiplierSheet.Row\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Add\(KeyValuePair<int, CrystalMonsterCollectionMultiplierSheet.Row>\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Contains\(KeyValuePair<int, CrystalMonsterCollectionMultiplierSheet.Row>\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.CopyTo\(KeyValuePair<int, CrystalMonsterCollectionMultiplierSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Remove\(KeyValuePair<int, CrystalMonsterCollectionMultiplierSheet.Row>\)](#) ,  
[Sheet<int, CrystalMonsterCollectionMultiplierSheet.Row>.Serialize\(\)](#) ,  
[object.Equals\(object\) ↴](#) , [object.Equals\(object, object\) ↴](#) , [object.GetHashCode\(\) ↴](#) ,  
[object.GetType\(\) ↴](#) , [object.MemberwiseClone\(\) ↴](#) , [object.ReferenceEquals\(object, object\) ↴](#) ,  
[object.ToString\(\) ↴](#)

## Constructors

### CrystalMonsterCollectionMultiplierSheet()

```
public CrystalMonsterCollectionMultiplierSheet()
```

# Class CrystalMonsterCollectionMultiplierSheet.R OW

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CrystalMonsterCollectionMultiplierSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalMonsterCollectionMultiplierSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Level

```
public int Level
```

#### Field Value

[int](#)

### Multiplier

```
public int Multiplier
```

#### Field Value

[int](#)

## Properties

### Key

```
public override int Key { get; }
```

## Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CrystalRandomBuffSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CrystalRandomBuffSheet : Sheet<int, CrystalRandomBuffSheet.Row>,
IDictionary<int, CrystalRandomBuffSheet.Row>, ICollection<KeyValuePair<int,
CrystalRandomBuffSheet.Row>>, IEnumerable<KeyValuePair<int,
CrystalRandomBuffSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalRandomBuffSheet.Row>](#) ← [CrystalRandomBuffSheet](#)

## Implements

[IDictionary<int, CrystalRandomBuffSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalRandomBuffSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalRandomBuffSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CrystalRandomBuffSheet.Row>.Name](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.First](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Last](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Values](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Count](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.TryGetValue\(int, out CrystalRandomBuffSheet.Row, bool\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.AddRow\(int, CrystalRandomBuffSheet.Row\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Add\(int, CrystalRandomBuffSheet.Row\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalRandomBuffSheet.Row>.Remove\(int\)](#) ,

```
Sheet<int, CrystalRandomBuffSheet.Row>.TryGetValue(int, out
CrystalRandomBuffSheet.Row) ,
Sheet<int, CrystalRandomBuffSheet.Row>.Add(KeyValuePair<int,
CrystalRandomBuffSheet.Row>) ,
Sheet<int, CrystalRandomBuffSheet.Row>.Clear() ,
Sheet<int, CrystalRandomBuffSheet.Row>.Contains(KeyValuePair<int,
CrystalRandomBuffSheet.Row>) ,
Sheet<int, CrystalRandomBuffSheet.Row>.CopyTo(KeyValuePair<int,
CrystalRandomBuffSheet.Row>[], int) ,
Sheet<int, CrystalRandomBuffSheet.Row>.Remove(KeyValuePair<int,
CrystalRandomBuffSheet.Row>) ,
Sheet<int, CrystalRandomBuffSheet.Row>.Serialize() , object.Equals(object) ↴ ,
object.Equals(object, object) ↴ , object.GetHashCode() ↴ , object.GetType() ↴ ,
object.MemberwiseClone() ↴ , object.ReferenceEquals(object, object) ↴ , object.ToString() ↴
```

## Constructors

### CrystalRandomBuffSheet()

```
public CrystalRandomBuffSheet()
```

# Class CrystalRandomBuffSheet.Row

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalRandomBuffSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalRandomBuffSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Id

```
public int Id
```

#### Field Value

[int](#)

### Rank

```
public CrystalRandomBuffSheet.Row.BuffRank Rank
```

#### Field Value

[CrystalRandomBuffSheet.Row.BuffRank](#)

# Ratio

```
public decimal Ratio
```

## Field Value

[decimal](#) ↗

## SkillId

```
public int SkillId
```

## Field Value

[int](#) ↗

# Properties

## Key

```
public override int Key { get; }
```

## Property Value

[int](#) ↗

# Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields  [IReadOnlyList<string>](#)

# Enum CrystalRandomBuffSheet.Row.BuffRank

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public enum CrystalRandomBuffSheet.Row.BuffRank
```

## Extension Methods

[StateExtensions.Serialize\(Enumerable\)](#)

## Fields

A = 3

B = 4

S = 2

SS = 1

# Class CrystalStageBuffGachaSheet

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CrystalStageBuffGachaSheet : Sheet<int,
CrystalStageBuffGachaSheet.Row>, IDictionary<int, CrystalStageBuffGachaSheet.Row>,
ICollection<KeyValuePair<int, CrystalStageBuffGachaSheet.Row>>,
IEnumerable<KeyValuePair<int, CrystalStageBuffGachaSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, CrystalStageBuffGachaSheet.Row>](#) ← [CrystalStageBuffGachaSheet](#)

## Implements

[IDictionary<int, CrystalStageBuffGachaSheet.Row>](#),  
[ICollection<KeyValuePair<int, CrystalStageBuffGachaSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, CrystalStageBuffGachaSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, CrystalStageBuffGachaSheet.Row>.Name](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.OrderedList](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.First](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Last](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Keys](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Values](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Count](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.TryGetValue\(int, out CrystalStageBuffGachaSheet.Row, bool\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.AddRow\(int, CrystalStageBuffGachaSheet.Row\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Add\(int, CrystalStageBuffGachaSheet.Row\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, CrystalStageBuffGachaSheet.Row>.TryGetValue\(int, out CrystalStageBuffGachaSheet.Row\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Add\(KeyValuePair<int, CrystalStageBuffGachaSheet.Row>\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Contains\(KeyValuePair<int, CrystalStageBuffGachaSheet.Row>\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.CopyTo\(KeyValuePair<int, CrystalStageBuffGachaSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Remove\(KeyValuePair<int, CrystalStageBuffGachaSheet.Row>\)](#) ,  
[Sheet<int, CrystalStageBuffGachaSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CrystalStageBuffGachaSheet()

```
public CrystalStageBuffGachaSheet()
```

# Class CrystalStageBuffGachaSheet.Row

Namespace: [Nekoyume.TableData.Crystal](#)

Assembly: Lib9c.dll

```
public class CrystalStageBuffGachaSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CrystalStageBuffGachaSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### AdvancedCost

```
public int AdvancedCost
```

#### Field Value

[int](#)

### MaxStar

```
public int MaxStar
```

#### Field Value

[int](#)

## NormalCost

```
public int NormalCost
```

Field Value

[int](#)

## StageId

```
public int StageId
```

Field Value

[int](#)

## Properties

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields  [IReadOnlyList<string>](#)

# Namespace Nekoyume.TableData.Custom EquipmentCraft

## Classes

[CustomEquipmentCraftIconSheet](#)

[CustomEquipmentCraftIconSheet.Row](#)

[CustomEquipmentCraftOptionSheet](#)

[CustomEquipmentCraftOptionSheet.Row](#)

[CustomEquipmentCraftRecipeSheet](#)

[CustomEquipmentCraftRecipeSheet.Row](#)

[CustomEquipmentCraftRecipeSkillSheet](#)

[CustomEquipmentCraftRecipeSkillSheet.Row](#)

[CustomEquipmentCraftRelationshipSheet](#)

[CustomEquipmentCraftRelationshipSheet.Row](#)

## Structs

[CustomEquipmentCraftOptionSheet.SubStat](#)

[CustomEquipmentCraftRelationshipSheet.CpGroup](#)

[CustomEquipmentCraftRelationshipSheet.MaterialCost](#)

# Class CustomEquipmentCraftIconSheet

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CustomEquipmentCraftIconSheet : Sheet<int,
CustomEquipmentCraftIconSheet.Row>, IDictionary<int,
CustomEquipmentCraftIconSheet.Row>, ICollection<KeyValuePair<int,
CustomEquipmentCraftIconSheet.Row>>, IEnumerable<KeyValuePair<int,
CustomEquipmentCraftIconSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int](#), [CustomEquipmentCraftIconSheet.Row](#) <–  
CustomEquipmentCraftIconSheet

## Implements

[IDictionary](#)<[int](#), [CustomEquipmentCraftIconSheet.Row](#)>,  
[ICollection](#)<[KeyValuePair](#)<[int](#), [CustomEquipmentCraftIconSheet.Row](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [CustomEquipmentCraftIconSheet.Row](#)>>,  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Name](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.OrderedList](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.First](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Last](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Keys](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Values](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Count](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftIconSheet.Row, bool\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.AddRow\(int, CustomEquipmentCraftIconSheet.Row\)](#) ,

[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Add\(int, CustomEquipmentCraftIconSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftIconSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Add\(KeyValuePair<int, CustomEquipmentCraftIconSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Contains\(KeyValuePair<int, CustomEquipmentCraftIconSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.CopyTo\(KeyValuePair<int, CustomEquipmentCraftIconSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Remove\(KeyValuePair<int, CustomEquipmentCraftIconSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftIconSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CustomEquipmentCraftIconSheet()

```
public CustomEquipmentCraftIconSheet()
```

# Class CustomEquipmentCraftIconSheet.Row

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CustomEquipmentCraftIconSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CustomEquipmentCraftIconSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### IconId

```
public int IconId { get; }
```

### Property Value

[int](#)

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

## ItemSubType

```
public ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## RandomOnly

```
public bool RandomOnly { get; }
```

Property Value

[bool](#)

## Ratio

```
public int Ratio { get; }
```

Property Value

[int](#)

# RequiredRelationship

```
public int RequiredRelationship { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CustomEquipmentCraftOptionSheet

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CustomEquipmentCraftOptionSheet : Sheet<int,
CustomEquipmentCraftOptionSheet.Row>, IDictionary<int,
CustomEquipmentCraftOptionSheet.Row>, ICollection<KeyValuePair<int,
CustomEquipmentCraftOptionSheet.Row>>, IEnumerable<KeyValuePair<int,
CustomEquipmentCraftOptionSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

```
object ↪ Sheet<int, CustomEquipmentCraftOptionSheet.Row> ↪
CustomEquipmentCraftOptionSheet
```

## Implements

```
IDictionary<int, CustomEquipmentCraftOptionSheet.Row>,
ICollection<KeyValuePair<int, CustomEquipmentCraftOptionSheet.Row>>,
IEnumerable<KeyValuePair<int, CustomEquipmentCraftOptionSheet.Row>>,
IEnumerable, ISheet
```

## Inherited Members

```
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Name ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.OrderedList ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.First ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Last ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Keys ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Values ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Count ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.IsReadOnly ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.this[int] ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Set(string, bool) ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Set<T>(Sheet<int, T>, bool) ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.GetEnumerator() ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.TryGetValue(int, out
CustomEquipmentCraftOptionSheet.Row, bool) ,
Sheet<int, CustomEquipmentCraftOptionSheet.Row>.AddRow(int,
CustomEquipmentCraftOptionSheet.Row) ,
```

[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Add\(int, CustomEquipmentCraftOptionSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftOptionSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Add\(KeyValuePair<int, CustomEquipmentCraftOptionSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Contains\(KeyValuePair<int, CustomEquipmentCraftOptionSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.CopyTo\(KeyValuePair<int, CustomEquipmentCraftOptionSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Remove\(KeyValuePair<int, CustomEquipmentCraftOptionSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftOptionSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### CustomEquipmentCraftOptionSheet()

```
public CustomEquipmentCraftOptionSheet()
```

# Class CustomEquipmentCraftOptionSheet.Row

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CustomEquipmentCraftOptionSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CustomEquipmentCraftOptionSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### ItemSubType

```
public ItemSubType ItemSubType { get; }
```

### Property Value

[ItemSubType](#)

## Key

```
public override int Key { get; }
```

### Property Value

[int](#)

## Ratio

```
public int Ratio { get; }
```

### Property Value

[int](#)

## SubStatData

```
public List<CustomEquipmentCraftOptionSheet.SubStat> SubStatData { get; }
```

### Property Value

[List](#)<[CustomEquipmentCraftOptionSheet](#).[SubStat](#)>

## TotalOptionRatio

```
public int TotalOptionRatio { get; }
```

### Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Struct CustomEquipmentCraftOptionSheet.SubStat

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
public struct CustomEquipmentCraftOptionSheet.SubStat
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Fields

### Ratio

```
public int Ratio
```

Field Value

[int](#)

### StatType

```
public StatType StatType
```

Field Value

[StatType](#)

# Class CustomEquipmentCraftRecipeSheet

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CustomEquipmentCraftRecipeSheet : Sheet<int,
CustomEquipmentCraftRecipeSheet.Row>, IDictionary<int,
CustomEquipmentCraftRecipeSheet.Row>, ICollection<KeyValuePair<int,
CustomEquipmentCraftRecipeSheet.Row>>, IEnumerable<KeyValuePair<int,
CustomEquipmentCraftRecipeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int](#), [CustomEquipmentCraftRecipeSheet.Row](#) <–  
CustomEquipmentCraftRecipeSheet

## Implements

[IDictionary](#)<[int](#), [CustomEquipmentCraftRecipeSheet.Row](#)>,  
[ICollection](#)<[KeyValuePair](#)<[int](#), [CustomEquipmentCraftRecipeSheet.Row](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [CustomEquipmentCraftRecipeSheet.Row](#)>>,  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Name](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.OrderedList](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.First](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Last](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Keys](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Values](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Count](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftRecipeSheet.Row, bool\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.AddRow\(int, CustomEquipmentCraftRecipeSheet.Row\)](#) ,

[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Add\(int, CustomEquipmentCraftRecipeSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftRecipeSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Add\(KeyValuePair<int, CustomEquipmentCraftRecipeSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Contains\(KeyValuePair<int, CustomEquipmentCraftRecipeSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.CopyTo\(KeyValuePair<int, CustomEquipmentCraftRecipeSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Remove\(KeyValuePair<int, CustomEquipmentCraftRecipeSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSheet.Row>.Serialize\(\)](#) , object.Equals(object) ,  
object.Equals(object, object) , object.GetHashCode() , object.GetType() ,  
object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString() ,

## Constructors

### CustomEquipmentCraftRecipeSheet()

```
public CustomEquipmentCraftRecipeSheet()
```

# Class CustomEquipmentCraftRecipeSheet.Row

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CustomEquipmentCraftRecipeSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CustomEquipmentCraftRecipeSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CircleAmount

```
public int CircleAmount { get; }
```

### Property Value

[int](#)

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

## ItemSubType

```
public ItemSubType ItemSubType { get; }
```

Property Value

[ItemSubType](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## RequiredBlock

```
public long RequiredBlock { get; }
```

Property Value

[long](#)

## ScrollAmount

```
public int ScrollAmount { get; }
```

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class CustomEquipmentCraftRecipeSkillSheet

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CustomEquipmentCraftRecipeSkillSheet : Sheet<int,
CustomEquipmentCraftRecipeSkillSheet.Row>, IDictionary<int,
CustomEquipmentCraftRecipeSkillSheet.Row>, ICollection<KeyValuePair<int,
CustomEquipmentCraftRecipeSkillSheet.Row>>, IEnumerable<KeyValuePair<int,
CustomEquipmentCraftRecipeSkillSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

```
object ↪ Sheet<int ↪ , CustomEquipmentCraftRecipeSkillSheet.Row> ↪
CustomEquipmentCraftRecipeSkillSheet
```

## Implements

```
IDictionary<int ↪ , CustomEquipmentCraftRecipeSkillSheet.Row>,
ICollection<KeyValuePair<int ↪ , CustomEquipmentCraftRecipeSkillSheet.Row>>,
IEnumerable<KeyValuePair<int ↪ , CustomEquipmentCraftRecipeSkillSheet.Row>>,
IEnumerable, ISheet
```

## Inherited Members

```
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Name ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.OrderedList ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.First ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Last ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Keys ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Values ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Count ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.IsReadOnly ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.this[int] ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Set(string, bool) ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Set<T>(Sheet<int, T>, bool) ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.GetEnumerator() ,
Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.TryGetValue(int, out
CustomEquipmentCraftRecipeSkillSheet.Row, bool) ,
```

[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.AddRow\(int, CustomEquipmentCraftRecipeSkillSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Add\(int, CustomEquipmentCraftRecipeSkillSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftRecipeSkillSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Add\(KeyValuePair<int, CustomEquipmentCraftRecipeSkillSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Contains\(KeyValuePair<int, CustomEquipmentCraftRecipeSkillSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.CopyTo\(KeyValuePair<int, CustomEquipmentCraftRecipeSkillSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Remove\(KeyValuePair<int, CustomEquipmentCraftRecipeSkillSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRecipeSkillSheet.Row>.Serialize\(\)](#) ,  
[object.Equals\(object\) ↴](#) , [object.Equals\(object, object\) ↴](#) , [object.GetHashCode\(\) ↴](#) ,  
[object.GetType\(\) ↴](#) , [object.MemberwiseClone\(\) ↴](#) , [object.ReferenceEquals\(object, object\) ↴](#) ,  
[object.ToString\(\) ↴](#)

## Constructors

### CustomEquipmentCraftRecipeSkillSheet()

```
public CustomEquipmentCraftRecipeSkillSheet()
```

# Class CustomEquipmentCraftRecipeSkillSheet.Ro W

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CustomEquipmentCraftRecipeSkillSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CustomEquipmentCraftRecipeSkillSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### ItemOptionId

```
public int ItemOptionId { get; }
```

### Property Value

[int](#)

## ItemSubType

`public ItemSubType ItemSubType { get; }`

Property Value

[ItemSubType](#)

## Key

`public override int Key { get; }`

Property Value

[int](#)

## Ratio

`public int Ratio { get; }`

Property Value

[int](#)

## Methods

### Set(IReadOnlyList<string>)

`public override void Set(IReadOnlyList<string> fields)`

Parameters

fields  [IReadOnlyList<string>](#)

# Class CustomEquipmentCraftRelationshipSheet

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]
public class CustomEquipmentCraftRelationshipSheet : Sheet<int,
CustomEquipmentCraftRelationshipSheet.Row>, IDictionary<int,
CustomEquipmentCraftRelationshipSheet.Row>, ICollection<KeyValuePair<int,
CustomEquipmentCraftRelationshipSheet.Row>>, IEnumerable<KeyValuePair<int,
CustomEquipmentCraftRelationshipSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

```
object ↪ Sheet<int ↪ , CustomEquipmentCraftRelationshipSheet.Row> ↪
CustomEquipmentCraftRelationshipSheet
```

## Implements

```
IDictionary<int ↪ , CustomEquipmentCraftRelationshipSheet.Row>,
ICollection<KeyValuePair<int ↪ , CustomEquipmentCraftRelationshipSheet.Row>>,
IEnumerable<KeyValuePair<int ↪ , CustomEquipmentCraftRelationshipSheet.Row>>,
IEnumerable, ISheet
```

## Inherited Members

```
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Name ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.OrderedList ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.First ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Last ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Keys ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Values ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Count ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.IsReadOnly ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.this[int] ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Set(string, bool) ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Set<T>(Sheet<int, T>, bool) ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.GetEnumerator() ,
Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.TryGetValue(int, out
CustomEquipmentCraftRelationshipSheet.Row, bool) ,
```

[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.AddRow\(int, CustomEquipmentCraftRelationshipSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Add\(int, CustomEquipmentCraftRelationshipSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.TryGetValue\(int, out CustomEquipmentCraftRelationshipSheet.Row\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Add\(KeyValuePair<int, CustomEquipmentCraftRelationshipSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Contains\(KeyValuePair<int, CustomEquipmentCraftRelationshipSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.CopyTo\(KeyValuePair<int, CustomEquipmentCraftRelationshipSheet.Row>\[\], int\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Remove\(KeyValuePair<int, CustomEquipmentCraftRelationshipSheet.Row>\)](#) ,  
[Sheet<int, CustomEquipmentCraftRelationshipSheet.Row>.Serialize\(\)](#) ,  
[object.Equals\(object\) ↗](#) , [object.Equals\(object, object\) ↗](#) , [object.GetHashCode\(\) ↗](#) ,  
[object.GetType\(\) ↗](#) , [object.MemberwiseClone\(\) ↗](#) , [object.ReferenceEquals\(object, object\) ↗](#) ,  
[object.ToString\(\) ↗](#)

## Constructors

### CustomEquipmentCraftRelationshipSheet()

```
public CustomEquipmentCraftRelationshipSheet()
```

# Struct CustomEquipmentCraftRelationshipSheet. CpGroup

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
public struct CustomEquipmentCraftRelationshipSheet.CpGroup
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Fields

### MaxCp

```
public int MaxCp
```

Field Value

[int](#)

### MinCp

```
public int MinCp
```

Field Value

[int](#)

# Ratio

`public int Ratio`

Field Value

[int](#)

## Methods

### SelectCp(IRandom)

`public int SelectCp(IRandom random)`

Parameters

`random` IRandom

Returns

[int](#)

# Struct CustomEquipmentCraftRelationshipSheet. MaterialCost

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
public struct CustomEquipmentCraftRelationshipSheet.MaterialCost
```

## Inherited Members

[ValueType.Equals\(object\)](#) , [ValueType.GetHashCode\(\)](#) , [ValueType.ToString\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Fields

### Amount

```
public int Amount
```

#### Field Value

[int](#)

### ItemId

```
public int ItemId
```

#### Field Value

[int](#)

# Class CustomEquipmentCraftRelationshipSheet. Row

Namespace: [Nekoyume.TableData.CustomEquipmentCraft](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class CustomEquipmentCraftRelationshipSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← CustomEquipmentCraftRelationshipSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### ArmorItemId

```
public int ArmorItemId { get; }
```

Property Value

[int](#)

### BeltItemId

```
public int BeltItemId { get; }
```

Property Value

[int](#)

## CostMultiplier

`public long CostMultiplier { get; }`

Property Value

[long](#)

## CpGroups

`public List<CustomEquipmentCraftRelationshipSheet.CpGroup> CpGroups { get; }`

Property Value

[List](#)<[CustomEquipmentCraftRelationshipSheet](#).[CpGroup](#)>

## GoldAmount

`public BigInteger GoldAmount { get; }`

Property Value

[BigInteger](#)

## Key

`public override int Key { get; }`

Property Value

[int](#)

## MaterialCosts

```
public List<CustomEquipmentCraftRelationshipSheet.MaterialCost> MaterialCosts {  
    get; }
```

Property Value

[List](#)<[CustomEquipmentCraftRelationshipSheet](#).[MaterialCost](#)>

## NecklaceItemId

```
public int NecklaceItemId { get; }
```

Property Value

[int](#)

## Relationship

```
public int Relationship { get; }
```

Property Value

[int](#)

## RequiredBlockMultiplier

```
public long RequiredBlockMultiplier { get; }
```

Property Value

[long](#)

## RingItemId

```
public int RingItemId { get; }
```

Property Value

[int](#)

## WeaponItemId

```
public int WeaponItemId { get; }
```

Property Value

[int](#)

## Methods

### GetItemId(ItemSubType)

```
public int GetItemId(ItemSubType itemSubType)
```

Parameters

`itemSubType` [ItemSubType](#)

Returns

[int](#)

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields  [IReadOnlyList<string>](#)

# Namespace Nekoyume.TableData.Event Classes

[EventConsumableItemRecipeSheet](#)

[EventConsumableItemRecipeSheet.Row](#)

[EventDungeonSheet](#)

[EventDungeonSheet.Row](#)

[EventDungeonStageSheet](#)

[EventDungeonStageSheet.Row](#)

[EventDungeonStageWaveSheet](#)

[EventDungeonStageWaveSheet.Row](#)

[EventMaterialItemRecipeSheet](#)

[EventMaterialItemRecipeSheet.Row](#)

[EventScheduleSheet](#)

[EventScheduleSheet.Row](#)

# Class EventConsumableItemRecipeSheet

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventConsumableItemRecipeSheet : Sheet<int,
EventConsumableItemRecipeSheet.Row>, IDictionary<int,
EventConsumableItemRecipeSheet.Row>, ICollection<KeyValuePair<int,
EventConsumableItemRecipeSheet.Row>>, IEnumerable<KeyValuePair<int,
EventConsumableItemRecipeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EventConsumableItemRecipeSheet.Row>](#) ←  
EventConsumableItemRecipeSheet

## Implements

[IDictionary<int, EventConsumableItemRecipeSheet.Row>](#),  
[ICollection<KeyValuePair<int, EventConsumableItemRecipeSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EventConsumableItemRecipeSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, EventConsumableItemRecipeSheet.Row>.Name](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.OrderedList](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.First](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Last](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Keys](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Values](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Count](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.TryGetValue\(int, out  
EventConsumableItemRecipeSheet.Row, bool\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.AddRow\(int,  
EventConsumableItemRecipeSheet.Row\)](#) ,

[Sheet<int, EventConsumableItemRecipeSheet.Row>.Add\(int, EventConsumableItemRecipeSheet.Row\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.TryGetValue\(int, out EventConsumableItemRecipeSheet.Row\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Add\(KeyValuePair<int, EventConsumableItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Contains\(KeyValuePair<int, EventConsumableItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.CopyTo\(KeyValuePair<int, EventConsumableItemRecipeSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Remove\(KeyValuePair<int, EventConsumableItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, EventConsumableItemRecipeSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventRecipeExtensions.GetRecipeRows\(EventConsumableItemRecipeSheet, int\)](#) ,  
[EventRecipeExtensions.ValidateFromAction\(EventConsumableItemRecipeSheet, int, string, string\)](#)

## Constructors

### EventConsumableItemRecipeSheet()

```
public EventConsumableItemRecipeSheet()
```

# Class EventConsumableItemRecipeSheet.Row

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]  
public class EventConsumableItemRecipeSheet.Row : ConsumableItemRecipeSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) <— [ConsumableItemRecipeSheet.Row](#) ←  
EventConsumableItemRecipeSheet.Row

## Inherited Members

[ConsumableItemRecipeSheet.Row.Key](#) , [ConsumableItemRecipeSheet.Row.Id](#) ,  
[ConsumableItemRecipeSheet.Row.RequiredBlockIndex](#) ,  
[ConsumableItemRecipeSheet.Row.RequiredActionPoint](#) ,  
[ConsumableItemRecipeSheet.Row.RequiredGold](#) ,  
[ConsumableItemRecipeSheet.Row.Materials](#) ,  
[ConsumableItemRecipeSheet.Row.ResultConsumableItemId](#) ,  
[ConsumableItemRecipeSheet.Row.Set\(IReadOnlyList<string>\)](#) ,  
[ConsumableItemRecipeSheet.Row.GetAllMaterials\(\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Class EventDungeonSheet

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventDungeonSheet : Sheet<int, EventDungeonSheet.Row>, IDictionary<int, EventDungeonSheet.Row>, ICollection<KeyValuePair<int, EventDungeonSheet.Row>>, IEnumerable<KeyValuePair<int, EventDungeonSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EventDungeonSheet.Row>](#) ← [EventDungeonSheet](#)

## Implements

[IDictionary<int, EventDungeonSheet.Row>](#),  
[ICollection<KeyValuePair<int, EventDungeonSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EventDungeonSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, EventDungeonSheet.Row>.Name](#) ,  
[Sheet<int, EventDungeonSheet.Row>.OrderedList](#) ,  
[Sheet<int, EventDungeonSheet.Row>.First](#) , [Sheet<int, EventDungeonSheet.Row>.Last](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Keys](#) , [Sheet<int, EventDungeonSheet.Row>.Values](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Count](#) ,  
[Sheet<int, EventDungeonSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EventDungeonSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.TryGetValue\(int, out EventDungeonSheet.Row, bool\)](#) ,  
  
[Sheet<int, EventDungeonSheet.Row>.AddRow\(int, EventDungeonSheet.Row\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Add\(int, EventDungeonSheet.Row\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.TryGetValue\(int, out EventDungeonSheet.Row\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Add\(KeyValuePair<int, EventDungeonSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonSheet.Row>.Clear\(\)](#) ,

```
Sheet<int, EventDungeonSheet.Row>.Contains(KeyValuePair<int,
EventDungeonSheet.Row>),
Sheet<int, EventDungeonSheet.Row>.CopyTo(KeyValuePair<int, EventDungeonSheet.Row>
[], int),
Sheet<int, EventDungeonSheet.Row>.Remove(KeyValuePair<int,
EventDungeonSheet.Row>),
Sheet<int, EventDungeonSheet.Row>.Serialize(), object.Equals(object) ↴ ,
object.Equals(object, object) ↴ , object.GetHashCode() ↴ , object.GetType() ↴ ,
object.MemberwiseClone() ↴ , object.ReferenceEquals(object, object) ↴ , object.ToString() ↴
```

## Extension Methods

```
EventDungeonExtensions.TryGetRowByEventDungeonStageId(EventDungeonSheet, int, out
EventDungeonSheet.Row),
EventDungeonExtensions.TryGetRowByEventScheduleId(EventDungeonSheet, int, out Event
DungeonSheet.Row),
EventDungeonExtensions.ValidateFromAction(EventDungeonSheet, int, int, string, string)
```

## Constructors

### EventDungeonSheet()

```
public EventDungeonSheet()
```

# Class EventDungeonSheet.Row

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventDungeonSheet.Row : WorldSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [WorldSheet.Row](#) ← EventDungeonSheet.Row

## Inherited Members

[WorldSheet.Row.Key](#) , [WorldSheet.Row.Id](#) , [WorldSheet.Row.Name](#) ,  
[WorldSheet.Row.StageBegin](#) , [WorldSheet.Row.StageEnd](#) , [WorldSheet.Row.StagesCount](#) ,  
[WorldSheet.Row.Set\(IReadOnlyList<string>\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Class EventDungeonStageSheet

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventDungeonStageSheet : Sheet<int, EventDungeonStageSheet.Row>,
IDictionary<int, EventDungeonStageSheet.Row>, ICollection<KeyValuePair<int,
EventDungeonStageSheet.Row>>, IEnumerable<KeyValuePair<int,
EventDungeonStageSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EventDungeonStageSheet.Row>](#) ← [EventDungeonStageSheet](#)

## Implements

[IDictionary<int, EventDungeonStageSheet.Row>](#),  
[ICollection<KeyValuePair<int, EventDungeonStageSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EventDungeonStageSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, EventDungeonStageSheet.Row>.Name](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.OrderedList](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.First](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Last](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Keys](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Values](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Count](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.TryGetValue\(int, out](#)  
[EventDungeonStageSheet.Row, bool\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.AddRow\(int, EventDungeonStageSheet.Row\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Add\(int, EventDungeonStageSheet.Row\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, EventDungeonStageSheet.Row>.TryGetValue\(int, out EventDungeonStageSheet.Row\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Add\(KeyValuePair<int, EventDungeonStageSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Contains\(KeyValuePair<int, EventDungeonStageSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.CopyTo\(KeyValuePair<int, EventDungeonStageSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Remove\(KeyValuePair<int, EventDungeonStageSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonStageSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventDungeonExtensions.GetStageRows\(EventDungeonStageSheet, int, int\)](#) ,  
[EventDungeonExtensions.ValidateFromAction\(EventDungeonStageSheet, int, string, string\)](#)

## Constructors

### EventDungeonStageSheet()

```
public EventDungeonStageSheet()
```

# Class EventDungeonStageSheet.Row

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventDungeonStageSheet.Row : StageSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [StageSheet.Row](#) ← EventDungeonStageSheet.Row

## Inherited Members

[StageSheet.Row.Key](#) , [StageSheet.Row.Id](#) , [StageSheet.Row.CostAP](#) ,  
[StageSheet.Row.TurnLimit](#) , [StageSheet.Row.EnemyInitialStatModifiers](#) ,  
[StageSheet.Row.Background](#) , [StageSheet.Row.BGM](#) , [StageSheet.Row.Rewards](#) ,  
[StageSheet.Row.DropItemMin](#) , [StageSheet.Row.DropItemMax](#) ,  
[StageSheet.Row.Set\(IReadOnlyList<string>\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Equals\(object\)](#) ,  
[SheetRow<int>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventDungeonExtensions.GetStageNumber\(EventDungeonStageSheet.Row\)](#)

# Class EventDungeonStageWaveSheet

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventDungeonStageWaveSheet : Sheet<int,
EventDungeonStageWaveSheet.Row>, IDictionary<int, EventDungeonStageWaveSheet.Row>,
ICollection<KeyValuePair<int, EventDungeonStageWaveSheet.Row>>,
IEnumerable<KeyValuePair<int, EventDungeonStageWaveSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EventDungeonStageWaveSheet.Row>](#) ←  
EventDungeonStageWaveSheet

## Implements

[IDictionary<int, EventDungeonStageWaveSheet.Row>](#),  
[ICollection<KeyValuePair<int, EventDungeonStageWaveSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EventDungeonStageWaveSheet.Row>>](#),  
[IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, EventDungeonStageWaveSheet.Row>.Name](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.OrderedList](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.First](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Last](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Keys](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Values](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Count](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.TryGetValue\(int, out](#)  
[EventDungeonStageWaveSheet.Row, bool\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.AddRow\(int,](#)  
[EventDungeonStageWaveSheet.Row\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Add\(int,](#)  
[EventDungeonStageWaveSheet.Row\)](#) ,

[Sheet<int, EventDungeonStageWaveSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.TryGetValue\(int, out EventDungeonStageWaveSheet.Row\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Add\(KeyValuePair<int, EventDungeonStageWaveSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Contains\(KeyValuePair<int, EventDungeonStageWaveSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.CopyTo\(KeyValuePair<int, EventDungeonStageWaveSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Remove\(KeyValuePair<int, EventDungeonStageWaveSheet.Row>\)](#) ,  
[Sheet<int, EventDungeonStageWaveSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Extension Methods

[EventDungeonExtensions.GetStageWaveRows\(EventDungeonStageWaveSheet, int, int\)](#).

## Constructors

### EventDungeonStageWaveSheet()

```
public EventDungeonStageWaveSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, EventDungeonStageWaveSheet.Row value)
```

## Parameters

key [int](#) ↴

value [EventDungeonStageWaveSheet.Row](#)

# Class EventDungeonStageWaveSheet.Row

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventDungeonStageWaveSheet.Row : StageWaveSheet.Row
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← [StageWaveSheet.Row](#) ← EventDungeonStageWaveSheet.Row

## Inherited Members

[StageWaveSheet.Row.Key](#) , [StageWaveSheet.Row.StageId](#) , [StageWaveSheet.Row.Waves](#) ,  
[StageWaveSheet.Row.HasBoss](#) , [StageWaveSheet.Row.TotalMonsterIds](#) ,  
[StageWaveSheet.Row.Set\(IReadOnlyList<string>\)](#) ,  
[StageWaveSheet.Row.EndOfSheetInitialize\(\)](#) , [SheetRow<int>.Validate\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

# Class EventMaterialItemRecipeSheet

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventMaterialItemRecipeSheet : Sheet<int,
EventMaterialItemRecipeSheet.Row>, IDictionary<int,
EventMaterialItemRecipeSheet.Row>, ICollection<KeyValuePair<int,
EventMaterialItemRecipeSheet.Row>>, IEnumerable<KeyValuePair<int,
EventMaterialItemRecipeSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EventMaterialItemRecipeSheet.Row>](#) ←  
EventMaterialItemRecipeSheet

## Implements

[IDictionary<int, EventMaterialItemRecipeSheet.Row>](#),  
[ICollection<KeyValuePair<int, EventMaterialItemRecipeSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EventMaterialItemRecipeSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, EventMaterialItemRecipeSheet.Row>.Name](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.OrderedList](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.First](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Last](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Keys](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Values](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Count](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.TryGetValue\(int, out](#)  
[EventMaterialItemRecipeSheet.Row, bool\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.AddRow\(int,](#)  
[EventMaterialItemRecipeSheet.Row\)](#) ,

[Sheet<int, EventMaterialItemRecipeSheet.Row>.Add\(int, EventMaterialItemRecipeSheet.Row\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.TryGetValue\(int, out EventMaterialItemRecipeSheet.Row\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Add\(KeyValuePair<int, EventMaterialItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Contains\(KeyValuePair<int, EventMaterialItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.CopyTo\(KeyValuePair<int, EventMaterialItemRecipeSheet.Row>\[\], int\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Remove\(KeyValuePair<int, EventMaterialItemRecipeSheet.Row>\)](#) ,  
[Sheet<int, EventMaterialItemRecipeSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventRecipeExtensions.GetRecipeRows\(EventMaterialItemRecipeSheet, int\)](#) ,  
[EventRecipeExtensions.ValidateFromAction\(EventMaterialItemRecipeSheet, int, string, string\)](#)

## Constructors

### EventMaterialItemRecipeSheet()

```
public EventMaterialItemRecipeSheet()
```

# Class EventMaterialItemRecipeSheet.Row

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventMaterialItemRecipeSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EventMaterialItemRecipeSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventRecipeExtensions.ValidateFromAction\(EventMaterialItemRecipeSheet.Row,](#)  
[MaterialItemSheet, Dictionary<int, int>, string, string\)](#)

## Properties

### Id

```
public int Id { get; }
```

### Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

## RequiredMaterialsCount

```
public int RequiredMaterialsCount { get; }
```

Property Value

[int](#)

## RequiredMaterialsId

```
public List<int> RequiredMaterialsId { get; }
```

Property Value

[List](#)<[int](#)>

## ResultMaterialItemCount

```
public int ResultMaterialItemCount { get; }
```

Property Value

[int](#)

## ResultMaterialItemId

```
public int ResultMaterialItemId { get; }
```

Property Value

## Methods

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

#### Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class EventScheduleSheet

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
public class EventScheduleSheet : Sheet<int, EventScheduleSheet.Row>,
IDictionary<int, EventScheduleSheet.Row>, ICollection<KeyValuePair<int,
EventScheduleSheet.Row>>, IEnumerable<KeyValuePair<int, EventScheduleSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, EventScheduleSheet.Row>](#) ← [EventScheduleSheet](#)

## Implements

[IDictionary<int, EventScheduleSheet.Row>](#),  
[ICollection<KeyValuePair<int, EventScheduleSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, EventScheduleSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, EventScheduleSheet.Row>.Name](#) ,  
[Sheet<int, EventScheduleSheet.Row>.OrderedList](#) ,  
[Sheet<int, EventScheduleSheet.Row>.First](#) , [Sheet<int, EventScheduleSheet.Row>.Last](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Keys](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Values](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Count](#) ,  
[Sheet<int, EventScheduleSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, EventScheduleSheet.Row>.this\[int\]](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.TryGetValue\(int, out EventScheduleSheet.Row, bool\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.AddRow\(int, EventScheduleSheet.Row\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Add\(int, EventScheduleSheet.Row\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.TryGetValue\(int, out EventScheduleSheet.Row\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Add\(KeyValuePair<int, EventScheduleSheet.Row>\)](#) ,  
[Sheet<int, EventScheduleSheet.Row>.Clear\(\)](#) ,

```
Sheet<int, EventScheduleSheet.Row>.Contains\(KeyValuePair<int, EventScheduleSheet.Row>\) ,  
Sheet<int, EventScheduleSheet.Row>.CopyTo\(KeyValuePair<int, EventScheduleSheet.Row>\[\], int\) ,  
Sheet<int, EventScheduleSheet.Row>.Remove\(KeyValuePair<int, EventScheduleSheet.Row>\) ,  
Sheet<int, EventScheduleSheet.Row>.Serialize\(\) , object.Equals\(object\) ,  
object.Equals\(object, object\) , object.GetHashCode\(\) , object.GetType\(\) ,  
object.MemberwiseClone\(\) , object.ReferenceEquals\(object, object\) , object.ToString\(\)
```

## Extension Methods

```
EventScheduleExtensions.TryGetRowForDungeon\(EventScheduleSheet, long, out Event ScheduleSheet.Row\) ,  
EventScheduleExtensions.TryGetRowForRecipe\(EventScheduleSheet, long, out Event ScheduleSheet.Row\) ,  
EventScheduleExtensions.ValidateFromActionForDungeon\(EventScheduleSheet, long, int, int, string, string\) ,  
EventScheduleExtensions.ValidateFromActionForRecipe\(EventScheduleSheet, long, int, int, string, string\)
```

## Constructors

### EventScheduleSheet()

```
public EventScheduleSheet()
```

# Class EventScheduleSheet.Row

Namespace: [Nekoyume.TableData.Event](#)

Assembly: Lib9c.dll

```
[Serializable]
public class EventScheduleSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← EventScheduleSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[EventScheduleExtensions.GetDungeonTicketCost\(EventScheduleSheet.Row, int, Currency\)](#) ,  
[EventScheduleExtensions.GetDungeonTicketCostV1\(EventScheduleSheet.Row, int\)](#) ,  
[EventScheduleExtensions.GetStageExp\(EventScheduleSheet.Row, int, int\)](#)

## Properties

### DungeonEndBlockIndex

```
public long DungeonEndBlockIndex { get; }
```

#### Property Value

[long](#)

### DungeonExpSeedValue

```
public int DungeonExpSeedValue { get; }
```

Property Value

[int ↗](#)

## DungeonTicketAdditionalPrice

This value is divided by 10 and used. e.g.,

DecimalMath.DecimalEx.Floor(row.DungeonTicketAdditionalPrice \* .1m, 2);

```
public int DungeonTicketAdditionalPrice { get; }
```

Property Value

[int ↗](#)

## DungeonTicketPrice

This value is divided by 10 and used. e.g.,

DecimalMath.DecimalEx.Floor(row.DungeonTicketPrice \* .1m, 2);

```
public int DungeonTicketPrice { get; }
```

Property Value

[int ↗](#)

## DungeonTicketsMax

```
public int DungeonTicketsMax { get; }
```

Property Value

[int ↗](#)

## DungeonTicketsResetIntervalBlockRange

```
public int DungeonTicketsResetIntervalBlockRange { get; }
```

Property Value

[int](#)

Id

```
public int Id { get; }
```

Property Value

[int](#)

Key

```
public override int Key { get; }
```

Property Value

[int](#)

RecipeEndBlockIndex

```
public long RecipeEndBlockIndex { get; }
```

Property Value

[long](#)

StartBlockIndex

```
public long StartBlockIndex { get; }
```

Property Value

[long](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Namespace Nekoyume.TableData.Garages

## Classes

[LoadIntoMyGaragesCostSheet](#)

[LoadIntoMyGaragesCostSheet.Row](#)

# Class LoadIntoMyGaragesCostSheet

Namespace: [Nekoyume.TableData.Garages](#)

Assembly: Lib9c.dll

```
[Serializable]
public class LoadIntoMyGaragesCostSheet : Sheet<int,
LoadIntoMyGaragesCostSheet.Row>, IDictionary<int, LoadIntoMyGaragesCostSheet.Row>,
ICollection<KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>>,
IEnumerable<KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, LoadIntoMyGaragesCostSheet.Row>](#) ←  
LoadIntoMyGaragesCostSheet

## Implements

[IDictionary<int, LoadIntoMyGaragesCostSheet.Row>](#),  
[ICollection<KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>>](#),  
[IEnumerable, ISheet](#)

## Inherited Members

[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Name](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.OrderedList](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.First](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Last](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Keys](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Values](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Count](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.this\[int\]](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.TryGetValue\(int, out LoadIntoMyGaragesCostSheet.Row, bool\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.AddRow\(int, LoadIntoMyGaragesCostSheet.Row\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Add\(int, LoadIntoMyGaragesCostSheet.Row\)](#) ,

[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.TryGetValue\(int, out LoadIntoMyGaragesCostSheet.Row\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Add\(KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Contains\(KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.CopyTo\(KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>\[\], int\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Remove\(KeyValuePair<int, LoadIntoMyGaragesCostSheet.Row>\)](#) ,  
[Sheet<int, LoadIntoMyGaragesCostSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### LoadIntoMyGaragesCostSheet()

```
public LoadIntoMyGaragesCostSheet()
```

## Methods

### GetGarageCost(HashDigest<SHA256>, int)

```
public FungibleAssetValue GetGarageCost(HashDigest<SHA256> fungibleId, int count)
```

## Parameters

**fungibleId** HashDigest<[SHA256](#)>

**count** [int](#) ↴

## Returns

FungibleAssetValue

## GetGarageCost(FungibleAssetValue)

```
public FungibleAssetValue GetGarageCost(FungibleAssetValue fav)
```

Parameters

**fav** FungibleAssetValue

Returns

FungibleAssetValue

## GetGarageCost(IEnumerable<FungibleAssetValue>, IEnumerable<(HashDigest<SHA256> fungibleId, int count)>)

```
public FungibleAssetValue GetGarageCost(IEnumerable<FungibleAssetValue>  
fungibleAssetValues, IEnumerable<(HashDigest<SHA256> fungibleId, int  
count)> fungibleIdAndCounts)
```

Parameters

**fungibleAssetValues** IEnumerable<FungibleAssetValue>

**fungibleIdAndCounts** IEnumerable<(HashDigest<SHA256> **fungibleId**, **int** **count**)>

Returns

FungibleAssetValue

## GetGarageCost(IEnumerable<FungibleAssetValue>, IEnumerable<(int itemId, int count, bool tradable)>)

```
public FungibleAssetValue GetGarageCost(IEnumerable<FungibleAssetValue>
fungibleAssetValues, IEnumerable<(int itemId, int count, bool
tradable)> itemIdAndCounts)
```

Parameters

fungibleAssetValues [IEnumerable](#)<FungibleAssetValue>  
itemIdAndCounts [IEnumerable](#)<(int itemId, int count, bool tradable)>

Returns

FungibleAssetValue

## GetGarageCost(int, int)

```
public FungibleAssetValue GetGarageCost(int itemId, int count)
```

Parameters

itemId [int](#)  
count [int](#)

Returns

FungibleAssetValue

## GetGarageCostPerUnit(HashDigest<SHA256>)

```
public decimal GetGarageCostPerUnit(HashDigest<SHA256> fungibleId)
```

Parameters

fungibleId HashDigest<[SHA256](#)>

Returns

[decimal](#)

## GetGarageCostPerUnit(int)

```
public decimal GetGarageCostPerUnit(int itemId)
```

Parameters

[itemId](#) [int](#)

Returns

[decimal](#)

## GetGarageCostPerUnit(string)

```
public decimal GetGarageCostPerUnit(string currencyTicker)
```

Parameters

[currencyTicker](#) [string](#)

Returns

[decimal](#)

## HasCost(HashDigest<SHA256>)

```
public bool HasCost(HashDigest<SHA256> fungibleId)
```

Parameters

[fungibleId](#) HashDigest<[SHA256](#)>

Returns

[bool](#)

## HasCost(string)

`public bool HasCost(string currencyTicker)`

### Parameters

`currencyTicker` [string](#)

### Returns

[bool](#)

# Class LoadIntoMyGaragesCostSheet.Row

Namespace: [Nekoyume.TableData.Garages](#)

Assembly: Lib9c.dll

```
[Serializable]
public class LoadIntoMyGaragesCostSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) < LoadIntoMyGaragesCostSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### CurrencyTicker

```
public string CurrencyTicker { get; }
```

Property Value

[string](#)

### FungibleId

```
public HashDigest<SHA256>? FungibleId { get; }
```

Property Value

[HashDigest<SHA256>?](#)

## GarageCostPerUnit

```
public decimal GarageCostPerUnit { get; }
```

Property Value

[decimal](#) ↗

## Id

```
public int Id { get; }
```

Property Value

[int](#) ↗

## ItemId

```
public int ItemId { get; }
```

Property Value

[int](#) ↗

## Key

```
public override int Key { get; }
```

Property Value

[int](#) ↗

## Methods

## Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

### Parameters

fields [IReadOnlyList<string>](#)

# Namespace Nekoyume.TableData.Grand Finale

## Classes

[GrandFinaleParticipantsSheet](#)

[GrandFinaleParticipantsSheet.Row](#)

[GrandFinaleScheduleSheet](#)

[GrandFinaleScheduleSheet.Row](#)

# Class GrandFinaleParticipantsSheet

Namespace: [Nekoyume.TableData.GrandFinale](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GrandFinaleParticipantsSheet : Sheet<int,
GrandFinaleParticipantsSheet.Row>, IDictionary<int,
GrandFinaleParticipantsSheet.Row>, ICollection<KeyValuePair<int,
GrandFinaleParticipantsSheet.Row>>, IEnumerable<KeyValuePair<int,
GrandFinaleParticipantsSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int](#), [GrandFinaleParticipantsSheet.Row](#) < GrandFinaleParticipantsSheet

## Implements

[IDictionary](#)<[int](#), [GrandFinaleParticipantsSheet.Row](#)>,  
[ICollection](#)<[KeyValuePair](#)<[int](#), [GrandFinaleParticipantsSheet.Row](#)>>,  
[IEnumerable](#)<[KeyValuePair](#)<[int](#), [GrandFinaleParticipantsSheet.Row](#)>>, [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, GrandFinaleParticipantsSheet.Row>.Name](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.OrderedList](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.First](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Last](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Keys](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Values](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Count](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.this\[int\]](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.TryGetValue\(int, out GrandFinaleParticipantsSheet.Row, bool\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.AddRow\(int, GrandFinaleParticipantsSheet.Row\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Add\(int, GrandFinaleParticipantsSheet.Row\)](#)

,

[Sheet<int, GrandFinaleParticipantsSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.TryGetValue\(int, out GrandFinaleParticipantsSheet.Row\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Add\(KeyValuePair<int, GrandFinaleParticipantsSheet.Row>\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Contains\(KeyValuePair<int, GrandFinaleParticipantsSheet.Row>\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.CopyTo\(KeyValuePair<int, GrandFinaleParticipantsSheet.Row>\[\], int\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Remove\(KeyValuePair<int, GrandFinaleParticipantsSheet.Row>\)](#) ,  
[Sheet<int, GrandFinaleParticipantsSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### GrandFinaleParticipantsSheet()

```
public GrandFinaleParticipantsSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, GrandFinaleParticipantsSheet.Row value)
```

## Parameters

key [int](#) ↴

value [GrandFinaleParticipantsSheet.Row](#)

# Class GrandFinaleParticipantsSheet.Row

Namespace: [Nekoyume.TableData.GrandFinale](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GrandFinaleParticipantsSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← GrandFinaleParticipantsSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### GrandFinaleId

```
public int GrandFinaleId { get; }
```

Property Value

[int](#)

### Key

```
public override int Key { get; }
```

Property Value

[int](#)

# Participants

```
public List<Address> Participants { get; }
```

Property Value

[List](#)<Address>

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class GrandFinaleScheduleSheet

Namespace: [Nekoyume.TableData.GrandFinale](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GrandFinaleScheduleSheet : Sheet<int, GrandFinaleScheduleSheet.Row>,
IDictionary<int, GrandFinaleScheduleSheet.Row>, ICollection<KeyValuePair<int,
GrandFinaleScheduleSheet.Row>>, IEnumerable<KeyValuePair<int,
GrandFinaleScheduleSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, GrandFinaleScheduleSheet.Row>](#) ← [GrandFinaleScheduleSheet](#)

## Implements

[IDictionary<int, GrandFinaleScheduleSheet.Row>](#),  
[ICollection<KeyValuePair<int, GrandFinaleScheduleSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, GrandFinaleScheduleSheet.Row>>](#), [IEnumerable](#),  
[ISheet](#)

## Inherited Members

[Sheet<int, GrandFinaleScheduleSheet.Row>.Name](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.OrderedList](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.First](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Last](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Keys](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Values](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Count](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.this\[int\]](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.TryGetValue\(int, out GrandFinaleScheduleSheet.Row, bool\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.AddRow\(int, GrandFinaleScheduleSheet.Row\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Add\(int, GrandFinaleScheduleSheet.Row\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Remove\(int\)](#) ,

[Sheet<int, GrandFinaleScheduleSheet.Row>.TryGetValue\(int, out GrandFinaleScheduleSheet.Row\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Add\(KeyValuePair<int, GrandFinaleScheduleSheet.Row>\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Contains\(KeyValuePair<int, GrandFinaleScheduleSheet.Row>\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.CopyTo\(KeyValuePair<int, GrandFinaleScheduleSheet.Row>\[\], int\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Remove\(KeyValuePair<int, GrandFinaleScheduleSheet.Row>\)](#) ,  
[Sheet<int, GrandFinaleScheduleSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ↴ ,  
[object.Equals\(object, object\)](#) ↴ , [object.GetHashCode\(\)](#) ↴ , [object.GetType\(\)](#) ↴ ,  
[object.MemberwiseClone\(\)](#) ↴ , [object.ReferenceEquals\(object, object\)](#) ↴ , [object.ToString\(\)](#) ↴

## Constructors

### GrandFinaleScheduleSheet()

```
public GrandFinaleScheduleSheet()
```

## Methods

### GetRowByBlockIndex(long)

```
public GrandFinaleScheduleSheet.Row GetRowByBlockIndex(long blockIndex)
```

## Parameters

blockIndex [long](#) ↴

## Returns

[GrandFinaleScheduleSheet.Row](#)

# Class GrandFinaleScheduleSheet.Row

Namespace: [Nekoyume.TableData.GrandFinale](#)

Assembly: Lib9c.dll

```
[Serializable]
public class GrandFinaleScheduleSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← GrandFinaleScheduleSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### EndBlockIndex

```
public long EndBlockIndex { get; }
```

Property Value

[long](#)

### Id

```
public int Id { get; }
```

Property Value

[int](#)

## Key

```
public override int Key { get; }
```

Property Value

[int](#)

## StartBlockIndex

```
public long StartBlockIndex { get; }
```

Property Value

[long](#)

## Methods

### IsOpened(long)

```
public bool IsOpened(long blockIndex)
```

Parameters

blockIndex [long](#)

Returns

[bool](#)

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

## Parameters

fields  [IReadOnlyList<string>](#)

# Namespace Nekoyume.TableData.Pet

## Classes

[PetOptionSheet](#)

[PetOptionSheet.Row](#)

[PetOptionSheet.Row.PetOptionInfo](#)

# Class PetOptionSheet

Namespace: [Nekoyume.TableData.Pet](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetOptionSheet : Sheet<int, PetOptionSheet.Row>, IDictionary<int,
PetOptionSheet.Row>, ICollection<KeyValuePair<int, PetOptionSheet.Row>>,
IEnumerable<KeyValuePair<int, PetOptionSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, PetOptionSheet.Row>](#) ← PetOptionSheet

## Implements

[IDictionary<int, PetOptionSheet.Row>](#),  
[ICollection<KeyValuePair<int, PetOptionSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, PetOptionSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, PetOptionSheet.Row>.Name](#) , [Sheet<int, PetOptionSheet.Row>.OrderedList](#) ,  
[Sheet<int, PetOptionSheet.Row>.First](#) , [Sheet<int, PetOptionSheet.Row>.Last](#) ,  
[Sheet<int, PetOptionSheet.Row>.Keys](#) , [Sheet<int, PetOptionSheet.Row>.Values](#) ,  
[Sheet<int, PetOptionSheet.Row>.Count](#) , [Sheet<int, PetOptionSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, PetOptionSheet.Row>.this\[int\]](#) ,  
[Sheet<int, PetOptionSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.TryGetValue\(int, out PetOptionSheet.Row, bool\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.AddRow\(int, PetOptionSheet.Row\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Add\(int, PetOptionSheet.Row\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.TryGetValue\(int, out PetOptionSheet.Row\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Add\(KeyValuePair<int, PetOptionSheet.Row>\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Contains\(KeyValuePair<int, PetOptionSheet.Row>\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.CopyTo\(KeyValuePair<int, PetOptionSheet.Row>\[\], int\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Remove\(KeyValuePair<int, PetOptionSheet.Row>\)](#) ,  
[Sheet<int, PetOptionSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### PetOptionSheet()

```
public PetOptionSheet()
```

## Methods

### AddRow(int, Row)

```
protected override void AddRow(int key, PetOptionSheet.Row value)
```

## Parameters

key [int](#)

value [PetOptionSheet.Row](#)

# Class PetOptionSheet.Row

Namespace: [Nekoyume.TableData.Pet](#)

Assembly: Lib9c.dll

```
[Serializable]
public class PetOptionSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← PetOptionSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### Key

```
public override int Key { get; }
```

### Property Value

[int](#)

### LevelOptionMap

```
public Dictionary<int, PetOptionSheet.Row.PetOptionInfo> LevelOptionMap { get; }
```

### Property Value

[Dictionary](#)<[int](#), [PetOptionSheet.Row.PetOptionInfo](#)>

## PetId

```
public int PetId { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class PetOptionSheet.Row.PetOptionInfo

Namespace: [Nekoyume.TableData.Pet](#)

Assembly: Lib9c.dll

```
public class PetOptionSheet.Row.PetOptionInfo
```

## Inheritance

[object](#) ← PetOptionSheet.Row.PetOptionInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### PetOptionInfo(PetOptionType, decimal)

```
public PetOptionInfo(PetOptionType optionType, decimal optionValue)
```

## Parameters

optionType [PetOptionType](#)

optionValue [decimal](#)

## Properties

### OptionType

```
public PetOptionType OptionType { get; }
```

## Property Value

## OptionValue

```
public decimal OptionValue { get; }
```

Property Value

[decimal](#) ↗

# Namespace Nekoyume.TableData.Rune Classes

[RuneLevelBonusSheet](#)

[RuneLevelBonusSheet.Row](#)

[RuneListSheet](#)

[RuneListSheet.Row](#)

# Class RuneLevelBonusSheet

Namespace: [Nekoyume.TableData.Rune](#)

Assembly: Lib9c.dll

```
public class RuneLevelBonusSheet : Sheet<int, RuneLevelBonusSheet.Row>,
IDictionary<int, RuneLevelBonusSheet.Row>, ICollection<KeyValuePair<int,
RuneLevelBonusSheet.Row>>, IEnumerable<KeyValuePair<int, RuneLevelBonusSheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RuneLevelBonusSheet.Row>](#) ← [RuneLevelBonusSheet](#)

## Implements

[IDictionary<int, RuneLevelBonusSheet.Row>](#),  
[ICollection<KeyValuePair<int, RuneLevelBonusSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RuneLevelBonusSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RuneLevelBonusSheet.Row>.Name](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.OrderedList](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.First](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Last](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Keys](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Values](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Count](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.TryGetValue\(int, out RuneLevelBonusSheet.Row, bool\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.AddRow\(int, RuneLevelBonusSheet.Row\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Add\(int, RuneLevelBonusSheet.Row\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RuneLevelBonusSheet.Row>.TryGetValue\(int, out RuneLevelBonusSheet.Row\)](#) ,

```
Sheet<int, RuneLevelBonusSheet.Row>.Add(KeyValuePair<int,
RuneLevelBonusSheet.Row>),
Sheet<int, RuneLevelBonusSheet.Row>.Clear() ,
Sheet<int, RuneLevelBonusSheet.Row>.Contains(KeyValuePair<int,
RuneLevelBonusSheet.Row>),
Sheet<int, RuneLevelBonusSheet.Row>.CopyTo(KeyValuePair<int,
RuneLevelBonusSheet.Row>[], int) ,
Sheet<int, RuneLevelBonusSheet.Row>.Remove(KeyValuePair<int,
RuneLevelBonusSheet.Row>),
Sheet<int, RuneLevelBonusSheet.Row>.Serialize() , object.Equals(object)✉ ,
object.Equals(object, object)✉ , object.GetHashCode()✉ , object.GetType()✉ ,
object.MemberwiseClone()✉ , object.ReferenceEquals(object, object)✉ , object.ToString()✉
```

## Constructors

### RuneLevelBonusSheet()

```
public RuneLevelBonusSheet()
```

# Class RuneLevelBonusSheet.Row

Namespace: [Nekoyume.TableData.Rune](#)

Assembly: Lib9c.dll

```
public class RuneLevelBonusSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RuneLevelBonusSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### Bonus

```
public int Bonus
```

#### Field Value

[int](#)

### Id

```
public int Id
```

#### Field Value

[int](#)

# RuneLevel

```
public int RuneLevel
```

Field Value

[int](#)

## Properties

Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Class RuneListSheet

Namespace: [Nekoyume.TableData.Rune](#)

Assembly: Lib9c.dll

```
public class RuneListSheet : Sheet<int, RuneListSheet.Row>, IDictionary<int, RuneListSheet.Row>, ICollection<KeyValuePair<int, RuneListSheet.Row>>, IEnumerable<KeyValuePair<int, RuneListSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, RuneListSheet.Row>](#) ← [RuneListSheet](#)

## Implements

[IDictionary<int, RuneListSheet.Row>](#),  
[ICollection<KeyValuePair<int, RuneListSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, RuneListSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, RuneListSheet.Row>.Name](#) , [Sheet<int, RuneListSheet.Row>.OrderedList](#) ,  
[Sheet<int, RuneListSheet.Row>.First](#) , [Sheet<int, RuneListSheet.Row>.Last](#) ,  
[Sheet<int, RuneListSheet.Row>.Keys](#) , [Sheet<int, RuneListSheet.Row>.Values](#) ,  
[Sheet<int, RuneListSheet.Row>.Count](#) , [Sheet<int, RuneListSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, RuneListSheet.Row>.this\[int\]](#) ,  
[Sheet<int, RuneListSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, RuneListSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, RuneListSheet.Row>.TryGetValue\(int, out RuneListSheet.Row, bool\)](#) ,  
[Sheet<int, RuneListSheet.Row>.AddRow\(int, RuneListSheet.Row\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Add\(int, RuneListSheet.Row\)](#) ,  
[Sheet<int, RuneListSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, RuneListSheet.Row>.TryGetValue\(int, out RuneListSheet.Row\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Add\(KeyValuePair<int, RuneListSheet.Row>\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Contains\(KeyValuePair<int, RuneListSheet.Row>\)](#) ,  
[Sheet<int, RuneListSheet.Row>.CopyTo\(KeyValuePair<int, RuneListSheet.Row>\[\], int\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Remove\(KeyValuePair<int, RuneListSheet.Row>\)](#) ,  
[Sheet<int, RuneListSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### RuneListSheet()

```
public RuneListSheet()
```

# Class RuneListSheet.Row

Namespace: [Nekoyume.TableData.Rune](#)

Assembly: Lib9c.dll

```
public class RuneListSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← RuneListSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### BonusCoef

```
public int BonusCoef
```

#### Field Value

[int](#)

### Grade

```
public int Grade
```

#### Field Value

[int](#)

## Id

```
public int Id
```

Field Value

[int](#)

## RequiredLevel

```
public int RequiredLevel
```

Field Value

[int](#)

## RuneType

```
public int RuneType
```

Field Value

[int](#)

## UsePlace

```
public int UsePlace
```

Field Value

[int](#)

## Properties

# Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

# Namespace Nekoyume.TableData.Stake Classes

[StakePolicySheet](#)

[StakePolicySheet.Row](#)

# Class StakePolicySheet

Namespace: [Nekoyume.TableData.Stake](#)

Assembly: Lib9c.dll

```
public class StakePolicySheet : Sheet<string, StakePolicySheet.Row>,
IDictionary<string, StakePolicySheet.Row>, ICollection<KeyValuePair<string,
StakePolicySheet.Row>>, IEnumerable<KeyValuePair<string, StakePolicySheet.Row>>,
IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<string, StakePolicySheet.Row>](#) ← StakePolicySheet

## Implements

[IDictionary<string, StakePolicySheet.Row>](#),  
[ICollection<KeyValuePair<string, StakePolicySheet.Row>>](#),  
[IEnumerable<KeyValuePair<string, StakePolicySheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<string, StakePolicySheet.Row>.Name](#) ,  
[Sheet<string, StakePolicySheet.Row>.OrderedList](#) ,  
[Sheet<string, StakePolicySheet.Row>.First](#) , [Sheet<string, StakePolicySheet.Row>.Last](#) ,  
[Sheet<string, StakePolicySheet.Row>.Keys](#) , [Sheet<string, StakePolicySheet.Row>.Values](#) ,  
[Sheet<string, StakePolicySheet.Row>.Count](#) ,  
[Sheet<string, StakePolicySheet.Row>.IsReadOnly](#) ,  
[Sheet<string, StakePolicySheet.Row>.this\[string\]](#) ,  
[Sheet<string, StakePolicySheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.Set<T>\(Sheet<string, T>, bool\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.TryGetValue\(string, out StakePolicySheet.Row, bool\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.AddRow\(string, StakePolicySheet.Row\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.Add\(string, StakePolicySheet.Row\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.ContainsKey\(string\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.Remove\(string\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.TryGetValue\(string, out StakePolicySheet.Row\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.Add\(KeyValuePair<string, StakePolicySheet.Row>\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.Clear\(\)](#) ,  
[Sheet<string, StakePolicySheet.Row>.Contains\(KeyValuePair<string, StakePolicySheet.Row>\)](#) ,

```
Sheet<string, StakePolicySheet.Row>.CopyTo\(KeyValuePair<string, StakePolicySheet.Row>\[\], int\),
Sheet<string, StakePolicySheet.Row>.Remove\(KeyValuePair<string, StakePolicySheet.Row>\),
Sheet<string, StakePolicySheet.Row>.Serialize\(\), object.Equals\(object\) ↴ ,
object.Equals\(object, object\) ↴ , object.GetHashCode\(\) ↴ , object.GetType\(\) ↴ ,
object.MemberwiseClone\(\) ↴ , object.ReferenceEquals\(object, object\) ↴ , object.ToString\(\) ↴
```

## Constructors

### StakePolicySheet()

```
public StakePolicySheet()
```

## Fields

### RequiredAttrNames

```
public static readonly string[] RequiredAttrNames
```

#### Field Value

[string\[\]](#)

### SheetPrefixRules

```
public static readonly (string attrName, string value)[] SheetPrefixRules
```

#### Field Value

[\(string ↴ attrName ↴ , string ↴ value ↴ \)\[\]](#)

## Properties

## LockupIntervalValue

```
public long LockupIntervalValue { get; }
```

Property Value

[long](#) ↗

## RewardIntervalValue

```
public long RewardIntervalValue { get; }
```

Property Value

[long](#) ↗

## StakeRegularFixedRewardSheetValue

```
public string StakeRegularFixedRewardSheetValue { get; }
```

Property Value

[string](#) ↗

## StakeRegularRewardSheetValue

```
public string StakeRegularRewardSheetValue { get; }
```

Property Value

[string](#) ↗

## Methods

## Set(string, bool)

```
public override void Set(string csv, bool isReversed = false)
```

### Parameters

csv [string](#)

isReversed [bool](#)

true: csv의 column과 row의 순서를 역순으로 한다.

### Exceptions

[ArgumentNullException](#)

[InvalidDataException](#)

# Class StakePolicySheet.Row

Namespace: [Nekoyume.TableData.Stake](#)

Assembly: Lib9c.dll

```
[Serializable]
public class StakePolicySheet.Row : SheetRow<string>
```

## Inheritance

[object](#) ← [SheetRow<string>](#) ← StakePolicySheet.Row

## Inherited Members

[SheetRow<string>.EndOfSheetInitialize\(\)](#) , [SheetRow<string>.Equals\(object\)](#) ,  
[SheetRow<string>.GetHashCode\(\)](#) , [object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Properties

### AttrName

```
public string AttrName { get; }
```

### Property Value

[string](#)

### Key

```
public override string Key { get; }
```

### Property Value

[string](#)

## Value

```
public string Value { get; }
```

Property Value

[string](#)

## Methods

Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

Validate()

```
public override void Validate()
```

# Namespace Nekoyume.TableData. Summon

## Classes

[SummonSheet](#)

[SummonSheet.Row](#)

# Class SummonSheet

Namespace: [Nekoyume.TableData.Summon](#)

Assembly: Lib9c.dll

```
public class SummonSheet : Sheet<int, SummonSheet.Row>, IDictionary<int, SummonSheet.Row>, ICollection<KeyValuePair<int, SummonSheet.Row>>, IEnumerable<KeyValuePair<int, SummonSheet.Row>>, IEnumerable, ISheet
```

## Inheritance

[object](#) ← [Sheet<int, SummonSheet.Row>](#) ← SummonSheet

## Implements

[IDictionary<int, SummonSheet.Row>](#),  
[ICollection<KeyValuePair<int, SummonSheet.Row>>](#),  
[IEnumerable<KeyValuePair<int, SummonSheet.Row>>](#), [IEnumerable](#), [ISheet](#)

## Inherited Members

[Sheet<int, SummonSheet.Row>.Name](#) , [Sheet<int, SummonSheet.Row>.OrderedList](#) ,  
[Sheet<int, SummonSheet.Row>.First](#) , [Sheet<int, SummonSheet.Row>.Last](#) ,  
[Sheet<int, SummonSheet.Row>.Keys](#) , [Sheet<int, SummonSheet.Row>.Values](#) ,  
[Sheet<int, SummonSheet.Row>.Count](#) , [Sheet<int, SummonSheet.Row>.IsReadOnly](#) ,  
[Sheet<int, SummonSheet.Row>.this\[int\]](#) ,  
[Sheet<int, SummonSheet.Row>.Set\(string, bool\)](#) ,  
[Sheet<int, SummonSheet.Row>.Set<T>\(Sheet<int, T>, bool\)](#) ,  
[Sheet<int, SummonSheet.Row>.GetEnumerator\(\)](#) ,  
[Sheet<int, SummonSheet.Row>.TryGetValue\(int, out SummonSheet.Row, bool\)](#) ,  
[Sheet<int, SummonSheet.Row>.AddRow\(int, SummonSheet.Row\)](#) ,  
[Sheet<int, SummonSheet.Row>.Add\(int, SummonSheet.Row\)](#) ,  
[Sheet<int, SummonSheet.Row>.ContainsKey\(int\)](#) ,  
[Sheet<int, SummonSheet.Row>.Remove\(int\)](#) ,  
[Sheet<int, SummonSheet.Row>.TryGetValue\(int, out SummonSheet.Row\)](#) ,  
[Sheet<int, SummonSheet.Row>.Add\(KeyValuePair<int, SummonSheet.Row>\)](#) ,  
[Sheet<int, SummonSheet.Row>.Clear\(\)](#) ,  
[Sheet<int, SummonSheet.Row>.Contains\(KeyValuePair<int, SummonSheet.Row>\)](#) ,  
[Sheet<int, SummonSheet.Row>.CopyTo\(KeyValuePair<int, SummonSheet.Row>\[\], int\)](#) ,  
[Sheet<int, SummonSheet.Row>.Remove\(KeyValuePair<int, SummonSheet.Row>\)](#) ,  
[Sheet<int, SummonSheet.Row>.Serialize\(\)](#) , [object.Equals\(object\)](#) ,

[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### SummonSheet()

```
public SummonSheet()
```

# Class SummonSheet.Row

Namespace: [Nekoyume.TableData.Summon](#)

Assembly: Lib9c.dll

```
public class SummonSheet.Row : SheetRow<int>
```

## Inheritance

[object](#) ← [SheetRow<int>](#) ← SummonSheet.Row

## Inherited Members

[SheetRow<int>.Validate\(\)](#) , [SheetRow<int>.EndOfSheetInitialize\(\)](#) ,  
[SheetRow<int>.Equals\(object\)](#) , [SheetRow<int>.GetHashCode\(\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Fields

### MaxRecipeCount

```
public const int MaxRecipeCount = 100
```

#### Field Value

[int](#)

## Recipes

```
public readonly List<(int, int)> Recipes
```

#### Field Value

[List](#)<([int](#) [id](#), [int](#) [count](#))>

## Properties

### CostMaterial

```
public int CostMaterial { get; }
```

Property Value

[int](#)

### CostMaterialCount

```
public int CostMaterialCount { get; }
```

Property Value

[int](#)

### CostNcg

```
public int CostNcg { get; }
```

Property Value

[int](#)

### GroupId

```
public int GroupId { get; }
```

Property Value

[int](#)

# Key

```
public override int Key { get; }
```

Property Value

[int](#)

## Methods

### CumulativeRatio(int)

```
public int CumulativeRatio(int index)
```

Parameters

index [int](#)

Returns

[int](#)

### Set(IReadOnlyList<string>)

```
public override void Set(IReadOnlyList<string> fields)
```

Parameters

fields [IReadOnlyList](#)<[string](#)>

### TotalRatio()

```
public int TotalRatio()
```

Returns

[int](#) ↗

# Namespace Nekoyume.TypedAddress

## Structs

[AgentAddress](#)

[GuildAddress](#)

[PledgeAddress](#)

# Struct AgentAddress

Namespace: [Nekoyume.TypedAddress](#)

Assembly: Lib9c.dll

```
public readonly struct AgentAddress : IBencodable
```

## Implements

IBencodable

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Extension Methods

[AgentAddressExtensions.GetPledgeAddress\(AgentAddress\)](#)

## Constructors

### AgentAddress(IValue)

```
public AgentAddress(IValue value)
```

#### Parameters

**value** IValue

### AgentAddress(Address)

```
public AgentAddress(Address address)
```

#### Parameters

**address** Address

## AgentAddress(byte[])

```
public AgentAddress(byte[] bytes)
```

### Parameters

bytes [byte](#)[]

## AgentAddress(string)

```
public AgentAddress(string hex)
```

### Parameters

hex [string](#)

## Properties

### Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

### Property Value

IValue

### Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

## Methods

### Equals(AgentAddress)

```
public bool Equals(AgentAddress other)
```

#### Parameters

other [AgentAddress](#)

#### Returns

[bool](#)

### Equals(object)

Indicates whether this instance and a specified object are equal.

```
public override bool Equals(object obj)
```

#### Parameters

obj [object](#)

The object to compare with the current instance.

#### Returns

[bool](#)

[true](#) if `obj` and this instance are the same type and represent the same value; otherwise, [false](#).

### GetHashCode()

Returns the hash code for this instance.

```
public override int GetHashCode()
```

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## ToString()

Returns the fully qualified type name of this instance.

```
public override string ToString()
```

Returns

[string](#)

The fully qualified type name.

## Operators

### operator ==(AgentAddress, AgentAddress)

```
public static bool operator ==(AgentAddress left, AgentAddress right)
```

Parameters

[left](#) [AgentAddress](#)

[right](#) [AgentAddress](#)

Returns

[bool](#)

## implicit operator Address(AgentAddress)

```
public static implicit operator Address(AgentAddress agentAddress)
```

Parameters

agentAddress [AgentAddress](#)

Returns

Address

## operator !=(AgentAddress, AgentAddress)

```
public static bool operator !=(AgentAddress left, AgentAddress right)
```

Parameters

left [AgentAddress](#)

right [AgentAddress](#)

Returns

[bool](#) ↗

# Struct GuildAddress

Namespace: [Nekoyume.TypedAddress](#)

Assembly: Lib9c.dll

```
public readonly struct GuildAddress : IBencodable, IEquatable<GuildAddress>
```

## Implements

IBencodable, [IEquatable](#)<[GuildAddress](#)>

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### GuildAddress(IValue)

```
public GuildAddress(IValue value)
```

#### Parameters

**value** IValue

### GuildAddress(Address)

```
public GuildAddress(Address address)
```

#### Parameters

**address** Address

### GuildAddress(byte[])

```
public GuildAddress(byte[] bytes)
```

## Parameters

bytes [byte](#)[]

## GuildAddress(string)

```
public GuildAddress(string hex)
```

## Parameters

hex [string](#)

# Properties

## Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

## Property Value

IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

# Methods

## Equals(GuildAddress)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(GuildAddress other)
```

### Parameters

**other** [GuildAddress](#)

An object to compare with this object.

### Returns

[bool](#)

[true](#) if the current object is equal to the **other** parameter; otherwise, [false](#).

## Equals(object)

Indicates whether this instance and a specified object are equal.

```
public override bool Equals(object obj)
```

### Parameters

**obj** [object](#)

The object to compare with the current instance.

### Returns

[bool](#)

[true](#) if **obj** and this instance are the same type and represent the same value; otherwise, [false](#).

## GetHashCode()

Returns the hash code for this instance.

```
public override int GetHashCode()
```

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## ToString()

Returns the fully qualified type name of this instance.

```
public override string ToString()
```

Returns

[string](#)

The fully qualified type name.

## Operators

### operator ==(GuildAddress, GuildAddress)

```
public static bool operator ==(GuildAddress left, GuildAddress right)
```

Parameters

[left](#) [GuildAddress](#)

[right](#) [GuildAddress](#)

Returns

[bool](#)

## implicit operator Address(GuildAddress)

```
public static implicit operator Address(GuildAddress guildAddress)
```

Parameters

guildAddress [GuildAddress](#)

Returns

Address

## operator !=(GuildAddress, GuildAddress)

```
public static bool operator !=(GuildAddress left, GuildAddress right)
```

Parameters

left [GuildAddress](#)

right [GuildAddress](#)

Returns

[bool](#)

# Struct PledgeAddress

Namespace: [Nekoyume.TypedAddress](#)

Assembly: Lib9c.dll

```
public readonly struct PledgeAddress : IBencodable
```

## Implements

IBencodable

## Inherited Members

[object.Equals\(object, object\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### PledgeAddress(IValue)

```
public PledgeAddress(IValue value)
```

#### Parameters

**value** IValue

### PledgeAddress(Address)

```
public PledgeAddress(Address address)
```

#### Parameters

**address** Address

### PledgeAddress(byte[])

```
public PledgeAddress(byte[] bytes)
```

## Parameters

bytes [byte](#)[]

## PledgeAddress(string)

```
public PledgeAddress(string hex)
```

## Parameters

hex [string](#)

# Properties

## Bencoded

An Bencodex.Types.IValue representation of this object that can be decoded back to instantiate an equal object. The decoded object must be equal to the original in the sense that [Equals\(T\)](#) should be [true](#).

```
public IValue Bencoded { get; }
```

## Property Value

IValue

## Remarks

Note that the only requirement is that the produced Bencodex.Types.IValue can be decoded back to an equal object. This representation may not be canonical in the sense that additional junk data may be present in an Bencodex.Types.IValue that one may wish to decode and this may be discarded while decoding.

A specific implementation may decide to only allow the canonical representation to be decoded.

# Methods

## Equals(PledgeAddress)

```
public bool Equals(PledgeAddress other)
```

### Parameters

other [PledgeAddress](#)

### Returns

[bool](#)

## Equals(object)

Indicates whether this instance and a specified object are equal.

```
public override bool Equals(object obj)
```

### Parameters

obj [object](#)

The object to compare with the current instance.

### Returns

[bool](#)

[true](#) if [obj](#) and this instance are the same type and represent the same value; otherwise, [false](#).

## GetHashCode()

Returns the hash code for this instance.

```
public override int GetHashCode()
```

Returns

[int](#)

A 32-bit signed integer that is the hash code for this instance.

## ToString()

Returns the fully qualified type name of this instance.

```
public override string ToString()
```

Returns

[string](#)

The fully qualified type name.

## Operators

### operator ==(PledgeAddress, PledgeAddress)

```
public static bool operator ==(PledgeAddress left, PledgeAddress right)
```

Parameters

[left](#) [PledgeAddress](#)

[right](#) [PledgeAddress](#)

Returns

[bool](#)

### implicit operator Address(PledgeAddress)

```
public static implicit operator Address(PledgeAddress agentAddress)
```

Parameters

agentAddress [PledgeAddress](#)

Returns

Address

operator !=(PledgeAddress, PledgeAddress)

```
public static bool operator !=(PledgeAddress left, PledgeAddress right)
```

Parameters

left [PledgeAddress](#)

right [PledgeAddress](#)

Returns

[bool](#)

# Namespace Nekoyume.Validator Delegation

## Classes

[AbstainHistory](#)

[ProposerInfo](#)

[ValidatorComparer](#)

[ValidatorDelegatee](#)

[ValidatorDelegator](#)

[ValidatorList](#)

[ValidatorRepository](#)

# Class AbstainHistory

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class AbstainHistory
```

## Inheritance

[object](#) ← AbstainHistory

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### AbstainHistory()

```
public AbstainHistory()
```

### AbstainHistory(IValue)

```
public AbstainHistory(IValue bencoded)
```

#### Parameters

bencoded IValue

### AbstainHistory(List)

```
public AbstainHistory(List bencoded)
```

#### Parameters

bencoded List

## Properties

### Address

```
public static Address Address { get; }
```

Property Value

Address

### Bencoded

```
public IValue Bencoded { get; }
```

Property Value

IValue

### History

```
public SortedDictionary<PublicKey, List<long>> History { get; }
```

Property Value

[SortedDictionary](#)<PublicKey, [List](#)<[long](#)>>

### MaxAbstainAllowance

```
public static int MaxAbstainAllowance { get; }
```

Property Value

[int](#)

## WindowSize

```
public static int WindowSize { get; }
```

Property Value

[int](#)

## Methods

### FindToSlashAndAdd(IEnumerable<PublicKey>, long)

```
public List<PublicKey> FindToSlashAndAdd(IEnumerable<PublicKey> abstainList,  
long height)
```

Parameters

abstainList [IEnumerable](#)<PublicKey>

height [long](#)

Returns

[List](#)<PublicKey>

# Class ProposerInfo

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public class ProposerInfo
```

## Inheritance

[object](#) ← ProposerInfo

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Constructors

### ProposerInfo(IValue)

```
public ProposerInfo(IValue bencoded)
```

#### Parameters

bencoded IValue

### ProposerInfo(List)

```
public ProposerInfo(List bencoded)
```

#### Parameters

bencoded List

### ProposerInfo(long, Address)

```
public ProposerInfo(long blockIndex, Address proposer)
```

## Parameters

blockIndex [long](#)

proposer Address

## Properties

### Address

```
public static Address Address { get; }
```

#### Property Value

Address

### Bencoded

```
public IValue Bencoded { get; }
```

#### Property Value

IValue

### BlockIndex

```
public long BlockIndex { get; }
```

#### Property Value

[long](#)

# Proposer

```
public Address Proposer { get; }
```

Property Value

Address

# Class ValidatorComparer

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public class ValidatorComparer : IComparer<Validator>
```

## Inheritance

[object](#) ← ValidatorComparer

## Implements

[IComparer](#)<Validator>

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) , [object.ReferenceEquals\(object, object\)](#) ,  
[object.ToString\(\)](#)

## Methods

### Compare(Validator?, Validator?)

Compares two objects and returns a value indicating whether one is less than, equal to, or greater than the other.

```
public int Compare(Validator? x, Validator? y)
```

#### Parameters

x Validator

The first object to compare.

y Validator

The second object to compare.

#### Returns

A signed integer that indicates the relative values of  $x$  and  $y$ , as shown in the following table.

Value	Meaning
<b>Less than zero</b>	$x$ is less than $y$ .
<b>Zero</b>	$x$ equals $y$ .
<b>Greater than zero</b>	$x$ is greater than $y$ .

# Class ValidatorDelegatee

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class ValidatorDelegatee : Delegatee<ValidatorDelegator,
ValidatorDelegatee>, IDelegatee, IEquatable<ValidatorDelegatee>, IBencodable
```

## Inheritance

[object](#) ← [Delegatee<ValidatorDelegator, ValidatorDelegatee>](#) ← ValidatorDelegatee

## Implements

[IDelegatee](#), [IEquatable<ValidatorDelegatee>](#), [IBencodable](#)

## Inherited Members

[Delegatee<ValidatorDelegator, ValidatorDelegatee>.DelegationChanged](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Enjailed](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Unjailed](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Metadata](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Repository](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Address](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.AccountAddress](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.MetadataAddress](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.DelegationCurrency](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.RewardCurrencies](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.DelegationPoolAddress](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.RewardPoolAddress](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.RewardRemainderPoolAddress](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.SlashedPoolAddress](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.UnbondingPeriod](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.MaxUnbondLockInEntries](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.MaxRebondGraceEntries](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Delegators](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.TotalDelegated](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.TotalShares](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Jailed](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.JailedUntil](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Tombstoned](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.MetadataBencoded](#) ,

[Delegatee<ValidatorDelegator, ValidatorDelegatee>.ShareFromFAV\(FungibleAssetValue\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.FAVFromShare\(BigInteger\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Bond\(IDelegate, FungibleAssetValue, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Unbond\(IDelegate, BigInteger, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.DistributeReward\(IDelegate, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Jail\(long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Unjail\(long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Tombstone\(\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.BondAddress\(Address\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.UnbondLockInAddress\(Address\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.RebondGraceAddress\(Address\)](#) ,  
[Delegatee<ValidatorDelegator>](#).  
[ValidatorDelegatee>.CurrentLumpSumRewardsRecordAddress\(\)](#) ,  
[Delegatee<ValidatorDelegator>](#).  
[ValidatorDelegatee>.LumpSumRewardsRecordAddress\(long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Bond\(ValidatorDelegator, FungibleAssetValue, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Unbond\(ValidatorDelegator, BigInteger, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.DistributeReward\(ValidatorDelegator, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.CollectRewards\(long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.Slash\(BigInteger, long, long\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.AddUnbondingRef\(UnbondingRef\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.RemoveUnbondingRef\(UnbondingRef\)](#) ,  
[Delegatee<ValidatorDelegator, ValidatorDelegatee>.CalculateReward\(BigInteger, IEnumerable<LumpSumRewardsRecord>\)](#) ,  
[object.Equals\(object, object\) ↴](#) , [object.GetType\(\) ↴](#) ,  
[object.ReferenceEquals\(object, object\) ↴](#) , [object.ToString\(\) ↴](#)

## Constructors

**ValidatorDelegatee(Address, IValue, ValidatorRepository)**

```
public ValidatorDelegatee(Address address, IValue bencoded,
```

```
ValidatorRepository repository)
```

## Parameters

address Address

bencoded IValue

repository [ValidatorRepository](#)

## ValidatorDelegatee(Address, List, ValidatorRepository)

```
public ValidatorDelegatee(Address address, List bencoded, ValidatorRepository  
repository)
```

## Parameters

address Address

bencoded List

repository [ValidatorRepository](#)

## ValidatorDelegatee(Address, PublicKey, BigInteger, long, IEnumerable<Currency>, ValidatorRepository)

```
public ValidatorDelegatee(Address address, PublicKey publicKey, BigInteger  
commissionPercentage, long creationHeight, IEnumerable<Currency> rewardCurrencies,  
ValidatorRepository repository)
```

## Parameters

address Address

publicKey PublicKey

commissionPercentage [BigInteger](#)

creationHeight [long](#)

rewardCurrencies [IEnumerable](#)<Currency>

repository [ValidatorRepository](#)

## Properties

### ActiveDelegationPoolAddress

```
public static Address ActiveDelegationPoolAddress { get; }
```

Property Value

Address

### BaseProposerRewardPercentage

```
public static BigInteger BaseProposerRewardPercentage { get; }
```

Property Value

[BigInteger](#)

### Bencoded

```
public List Bencoded { get; }
```

Property Value

List

### BonusProposerRewardPercentage

```
public static BigInteger BonusProposerRewardPercentage { get; }
```

Property Value

[BigInteger](#)

## CommissionPercentage

```
public BigInteger CommissionPercentage { get; }
```

Property Value

[BigInteger](#)

## CommissionPercentageLastUpdateHeight

```
public long CommissionPercentageLastUpdateHeight { get; }
```

Property Value

[long](#)

## CommissionPercentageMaxChange

```
public static BigInteger CommissionPercentageMaxChange { get; }
```

Property Value

[BigInteger](#)

## CommissionPercentageUpdateCooldown

```
public static long CommissionPercentageUpdateCooldown { get; }
```

Property Value

[long](#) ↴

## DefaultCommissionPercentage

```
public static BigInteger DefaultCommissionPercentage { get; }
```

Property Value

[BigInteger](#) ↴

## InactiveDelegationPoolAddress

```
public static Address InactiveDelegationPoolAddress { get; }
```

Property Value

Address

## IsActive

```
public bool IsActive { get; }
```

Property Value

[bool](#) ↴

## MaxCommissionPercentage

```
public static BigInteger MaxCommissionPercentage { get; }
```

Property Value

[BigInteger](#) ↴

## MinCommissionPercentage

```
public static BigInteger MinCommissionPercentage { get; }
```

Property Value

[BigInteger](#)

## MinSelfDelegation

```
public FungibleAssetValue MinSelfDelegation { get; }
```

Property Value

FungibleAssetValue

## Power

```
public BigInteger Power { get; }
```

Property Value

[BigInteger](#)

## PublicKey

```
public PublicKey PublicKey { get; }
```

Property Value

PublicKey

## Validator

```
public Validator Validator { get; }
```

Property Value

Validator

## ValidatorDelegationCurrency

```
public static Currency ValidatorDelegationCurrency { get; }
```

Property Value

Currency

## ValidatorMaxRebondGraceEntries

```
public static int ValidatorMaxRebondGraceEntries { get; }
```

Property Value

[int](#)

## ValidatorMaxUnbondLockInEntries

```
public static int ValidatorMaxUnbondLockInEntries { get; }
```

Property Value

[int](#)

## ValidatorUnbondingPeriod

```
public static long ValidatorUnbondingPeriod { get; }
```

Property Value

[long](#)

## Methods

### Activate()

```
public void Activate()
```

### AllocateReward(FungibleAssetValue, BigInteger, BigInteger, Address, long)

```
public void AllocateReward(FungibleAssetValue rewardToAllocate, BigInteger
validatorPower, BigInteger validatorSetPower, Address RewardSource, long height)
```

Parameters

rewardToAllocate FungibleAssetValue

validatorPower [BigInteger](#)

validatorSetPower [BigInteger](#)

RewardSource Address

height [long](#)

### Deactivate()

```
public void Deactivate()
```

## Equals(IDelegatee?)

```
public bool Equals(IDelegatee? other)
```

Parameters

other [IDelegatee](#)

Returns

[bool](#)

## Equals(ValidatorDelegatee?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(ValidatorDelegatee? other)
```

Parameters

other [ValidatorDelegatee](#)

An object to compare with this object.

Returns

[bool](#)

[true](#) if the current object is equal to the other parameter; otherwise, [false](#).

## Equals(object?)

Determines whether the specified object is equal to the current object.

```
public override bool Equals(object? obj)
```

Parameters

`obj` [object](#)

The object to compare with the current object.

Returns

[bool](#)

[true](#) if the specified object is equal to the current object; otherwise, [false](#).

## GetHashCode()

Serves as the default hash function.

```
public override int GetHashCode()
```

Returns

[int](#)

A hash code for the current object.

## OnDelegationChanged(object?, long)

```
public void OnDelegationChanged(object? sender, long height)
```

Parameters

`sender` [object](#)

`height` [long](#)

## OnEnjailed(object?, EventArgs)

```
public void OnEnjailed(object? sender, EventArgs e)
```

Parameters

sender [object](#)

e [EventArgs](#)

## OnUnjailed(object?, EventArgs)

```
public void OnUnjailed(object? sender, EventArgs e)
```

### Parameters

sender [object](#)

e [EventArgs](#)

## SetCommissionPercentage(BigInteger, long)

```
public void SetCommissionPercentage(BigInteger percentage, long height)
```

### Parameters

percentage [BigInteger](#)

height [long](#)

## Unjail(long)

```
public void Unjail(long height)
```

### Parameters

height [long](#)

# Class ValidatorDelegate

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class ValidatorDelegate : Delegator<ValidatorDelegatee,  
ValidatorDelegate>, IDelegator, IEquatable<ValidatorDelegate>
```

## Inheritance

[object](#) ← [Delegator<ValidatorDelegatee, ValidatorDelegate>](#) ← ValidatorDelegate

## Implements

[IDelegator](#), [IEquatable<ValidatorDelegate>](#)

## Inherited Members

[Delegator<ValidatorDelegatee, ValidatorDelegate>.Metadata](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.Repository](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.Address](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.AccountAddress](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.MetadataAddress](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.DelegationPoolAddress](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.RewardAddress](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.Delegatees](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.MetadataBencoded](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.Delegate\(ValidatorDelegatee, FungibleAssetValue, long\)](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.Undelegate\(ValidatorDelegatee, BigInteger, long\)](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.Redelegate\(ValidatorDelegatee, ValidatorDelegatee, BigInteger, long\)](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.CancelUndelegate\(ValidatorDelegatee, FungibleAssetValue, long\)](#) ,  
[Delegator<ValidatorDelegatee, ValidatorDelegate>.ClaimReward\(ValidatorDelegatee, long\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

## ValidatorDelegator(Address, Address, ValidatorRepository)

```
public ValidatorDelegator(Address address, Address delegationPoolAddress,  
ValidatorRepository repository)
```

### Parameters

address Address

delegationPoolAddress Address

repository [ValidatorRepository](#)

## ValidatorDelegator(Address, ValidatorRepository)

```
public ValidatorDelegator(Address address, ValidatorRepository repository)
```

### Parameters

address Address

repository [ValidatorRepository](#)

## Methods

### Equals(ValidatorDelegator?)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(ValidatorDelegator? other)
```

### Parameters

other [ValidatorDelegator](#)

An object to compare with this object.

## Returns

[bool](#)

[true](#) if the current object is equal to the [other](#) parameter; otherwise, [false](#).

# Class ValidatorList

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class ValidatorList : IBencodable
```

## Inheritance

[object](#) ← ValidatorList

## Implements

IBencodable

## Inherited Members

[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Constructors

### ValidatorList()

```
public ValidatorList()
```

### ValidatorList(IValue)

```
public ValidatorList(IValue bencoded)
```

## Parameters

bencoded IValue

### ValidatorList(List)

```
public ValidatorList(List bencoded)
```

## Parameters

bencoded List

## Properties

### Address

```
public static Address Address { get; }
```

Property Value

Address

### Bencoded

```
public List Bencoded { get; }
```

Property Value

List

### MaxActiveSetSize

```
public static int MaxActiveSetSize { get; }
```

Property Value

[int ↗](#)

## Validators

```
public ImmutableList<Validator> Validators { get; }
```

Property Value

[ImmutableList](#)<Validator>

## Methods

### ActiveSet()

```
public List<Validator> ActiveSet()
```

Returns

[List](#)<Validator>

### InActiveSet()

```
public List<Validator> InActiveSet()
```

Returns

[List](#)<Validator>

### RemoveValidator(PublicKey)

```
public ValidatorList RemoveValidator(PublicKey publicKey)
```

Parameters

`publicKey` PublicKey

Returns

[ValidatorList](#)

## SetValidator(Validator)

```
public ValidatorList SetValidator(Validator validator)
```

### Parameters

**validator** Validator

### Returns

[ValidatorList](#)

# Class ValidatorRepository

Namespace: [Nekoyume.ValidatorDelegation](#)

Assembly: Lib9c.dll

```
public sealed class ValidatorRepository : DelegationRepository,  
IDelegationRepository
```

## Inheritance

[object](#) ↳ [DelegationRepository](#) ↳ ValidatorRepository

## Implements

[IDelegationRepository](#)

## Inherited Members

[DelegationRepository.ActionContext](#) , [DelegationRepository.DelegateeAccountAddress](#) ,  
[DelegationRepository.DelegatorAccountAddress](#) ,  
[DelegationRepository.GetDelegateeMetadata\(Address\)](#) ,  
[DelegationRepository.GetDelegatorMetadata\(Address\)](#) ,  
[DelegationRepository.GetBond\(IDelegatee, Address\)](#) ,  
[DelegationRepository.GetUnbondLockIn\(IDelegatee, Address\)](#) ,  
[DelegationRepository.GetUnlimitedUnbondLockIn\(Address\)](#) ,  
[DelegationRepository.GetRebondGrace\(IDelegatee, Address\)](#) ,  
[DelegationRepository.GetUnlimitedRebondGrace\(Address\)](#) ,  
[DelegationRepository.GetUnbondingSet\(\)](#) ,  
[DelegationRepository.GetLumpSumRewardsRecord\(IDelegatee, long\)](#) ,  
[DelegationRepository.GetCurrentLumpSumRewardsRecord\(IDelegatee\)](#) ,  
[DelegationRepository.GetBalance\(Address, Currency\)](#) ,  
[DelegationRepository.SetDelegateeMetadata\(DelegateeMetadata\)](#) ,  
[DelegationRepository.SetDelegatorMetadata\(DelegatorMetadata\)](#) ,  
[DelegationRepository.SetBond\(Bond\)](#) ,  
[DelegationRepository.SetUnbondLockIn\(UnbondLockIn\)](#) ,  
[DelegationRepository.SetRebondGrace\(RebondGrace\)](#) ,  
[DelegationRepository.SetUnbondingSet\(UnbondingSet\)](#) ,  
[DelegationRepository.SetLumpSumRewardsRecord\(LumpSumRewardsRecord\)](#) ,  
[DelegationRepository.TransferAsset\(Address, Address, FungibleAssetValue\)](#) ,  
[object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) ,  
[object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#) , [object.ToString\(\)](#)

## Extension Methods

[AbstainHistoryModule.GetAbstainHistory\(ValidatorRepository\)](#) ,  
[AbstainHistoryModule.SetAbstainHistory\(ValidatorRepository, AbstainHistory\)](#) ,  
[ValidatorDelegateeModule.CreateValidatorDelegatee\(ValidatorRepository, PublicKey, BigInteger\)](#) ,  
[ValidatorDelegateeModule.TryGetValidatorDelegatee\(ValidatorRepository, Address, out ValidatorDelegatee?\)](#) ,  
[ValidatorDelegatorModule.TryGetValidatorDelegator\(ValidatorRepository, Address, out ValidatorDelegator?\)](#) ,  
[ValidatorListModule.GetProposerInfo\(ValidatorRepository\)](#) ,  
[ValidatorListModule.GetValidatorList\(ValidatorRepository\)](#) ,  
[ValidatorListModule.SetProposerInfo\(ValidatorRepository, ProposerInfo\)](#) ,  
[ValidatorListModule.TryGetProposerInfo\(ValidatorRepository, Address, out ProposerInfo?\)](#) ,  
[ValidatorListModule.TryGetValidatorList\(ValidatorRepository, out ValidatorList?\)](#) ,  
[ValidatorUnbondingModule.ReleaseUnbondings\(ValidatorRepository\)](#).

## Constructors

### ValidatorRepository(IWorld, IActionContext)

```
public ValidatorRepository(IWorld world, IActionContext actionContext)
```

#### Parameters

`world` IWorld

`actionContext` IActionContext

### ValidatorRepository(IDelegationRepository)

```
public ValidatorRepository(IDelegationRepository repository)
```

#### Parameters

`repository` [IDelegationRepository](#)

## Properties

# World

```
public override IWorld World { get; }
```

Property Value

IWorld

## Methods

### GetDelegatee(Address)

```
public override IDelegatee GetDelegatee(Address address)
```

Parameters

address Address

Returns

[IDelegatee](#)

### GetDelegator(Address)

```
public override IDelegator GetDelegator(Address address)
```

Parameters

address Address

Returns

[IDelegator](#)

### GetValidatorDelegatee(Address)

```
public ValidatorDelegatee GetValidatorDelegatee(Address address)
```

Parameters

address Address

Returns

[ValidatorDelegatee](#)

## GetValidatorDelegator(Address)

```
public ValidatorDelegator GetValidatorDelegator(Address address)
```

Parameters

address Address

Returns

[ValidatorDelegator](#)

## GetValidatorList()

```
public ValidatorList GetValidatorList()
```

Returns

[ValidatorList](#)

## SetCommissionPercentage(Address, BigInteger, long)

```
public void SetCommissionPercentage(Address address, BigInteger  
commissionPercentage, long height)
```

Parameters

address Address

commissionPercentage [BigInteger](#)

height [long](#)

## SetDelegatee(IDelegatee)

```
public override void SetDelegatee(IDelegatee delegatee)
```

Parameters

delegatee [IDelegatee](#)

## SetDelegator(IDelegator)

```
public override void SetDelegator(IDelegator delegator)
```

Parameters

delegator [IDelegator](#)

## SetValidatorDelegatee(ValidatorDelegatee)

```
public void SetValidatorDelegatee(ValidatorDelegatee validatorDelegatee)
```

Parameters

validatorDelegatee [ValidatorDelegatee](#)

## SetValidatorDelegator(ValidatorDelegator)

```
public void SetValidatorDelegate(ValidatorDelegate validatorDelegate)
```

Parameters

validatorDelegate [ValidatorDelegate](#)

## SetValidatorList(ValidatorList)

```
public void SetValidatorList(ValidatorList validatorList)
```

Parameters

validatorList [ValidatorList](#)

## UpdateWorld(IWorld)

```
public override void UpdateWorld(IWorld world)
```

Parameters

world IWorld