

# Namespace BP.AdventureFramework. Assets

## Classes

### [ConditionalDescription](#)

Represents a conditional description of an object.

### [Description](#)

Represents a description of an object.

### [ExaminableObject](#)

Represents an object that can be examined.

### [ExaminationResult](#)

Represents the result of an examination.

### [Identifier](#)

Provides a class that can be used as an identifier.

### [Item](#)

Represents an item that can be used within the game.

## Structs

### [Size](#)

Represents a size.

## Interfaces

### [IExaminable](#)

Represents any object that is examinable.

### [IPlayerVisible](#)

Represents any object that is visible to a player.

## Delegates

### [ExaminationCallback](#)

Represents the callback for examinations.

# Class ConditionalDescription

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents a conditional description of an object.

```
public sealed class ConditionalDescription : Description
```

## Inheritance

[object](#) ← [Description](#) ← ConditionalDescription

## Inherited Members

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

## Constructors

### ConditionalDescription(string, string, Condition)

Initializes a new instance of the ConditionalDescription class.

```
public ConditionalDescription(string trueDescription, string falseDescription,  
Condition condition)
```

## Parameters

**trueDescription** [string](#)

The true description.

**falseDescription** [string](#)

The false description.

**condition** [Condition](#)

The condition.

# Properties

## Condition

Get or set the condition

```
public Condition Condition { get; set; }
```

Property Value

### [Condition](#)

Represents a conditional description of an object.

## Methods

### GetDescription()

Get the description.

```
public override string GetDescription()
```

Returns

### [string](#)

The description.

# Class Description

Namespace: [BP.AdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents a description of an object.

```
public class Description
```

## Inheritance

[object](#) ← Description

## Derived

[ConditionalDescription](#)

## Inherited Members

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

## Constructors

### Description(string)

Initializes a new instance of the Description class

```
public Description(string description)
```

## Parameters

description [string](#)

The description

## Properties

### DefaultDescription

Get or set the description.

```
protected string DefaultDescription { get; set; }
```

Property Value

[string](#)

Represents a description of an object.

## Empty

Get an empty description.

```
public static Description Empty { get; }
```

Property Value

[Description](#)

Represents a description of an object.

## Methods

### GetDescription()

Get the description.

```
public virtual string GetDescription()
```

Returns

[string](#)

The description.

# Class ExaminableObject

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents an object that can be examined.

```
public class ExaminableObject : IExaminable, IPlayerVisible
```

## Inheritance

[object](#) ← ExaminableObject

## Implements

[IExaminable](#), [IPlayerVisible](#)

## Derived

[Character](#), [Item](#), [Exit](#), [Overworld](#), [Region](#), [Room](#)

## Inherited Members

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

## Properties

### Commands

Get or set this objects commands.

```
public CustomCommand[] Commands { get; set; }
```

### Property Value

[CustomCommand\[\]](#)

Represents an object that can be examined.

## Description

Get or set a description of this object.

```
public Description Description { get; set; }
```

## Property Value

### [Description](#)

Represents an object that can be examined.

## Examination

Get or set the callback handling all examination of this object.

```
public ExaminationCallback Examination { get; set; }
```

## Property Value

### [ExaminationCallback](#)

Represents an object that can be examined.

## Identifier

Get this objects identifier.

```
public Identifier Identifier { get; protected set; }
```

## Property Value

### [Identifier](#)

Represents an object that can be examined.

## IsPlayerVisible

Get or set if this is visible to the player.

```
public bool IsPlayerVisible { get; set; }
```

Property Value

[bool](#)

Represents an object that can be examined.

## Methods

### Examine()

Examine this object.

```
public virtual ExaminationResult Examine()
```

Returns

[ExaminationResult](#)

A ExaminationResult detailing the examination of this object.

### ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

Returns

[string](#)

A string that represents the current object.

# Delegate ExaminationCallback

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents the callback for examinations.

```
public delegate ExaminationResult ExaminationCallback(IExaminable obj)
```

Parameters

**obj** [IExaminable](#)

The object to examine.

Returns

[ExaminationResult](#)

A string representing the result of the examination.

# Class ExaminationResult

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents the result of an examination.

```
public class ExaminationResult : Result
```

## Inheritance

[object](#) ← [Result](#) ← ExaminationResult

## Inherited Members

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

## Constructors

### ExaminationResult(string)

Initializes a new instance of the ExaminationResult class.

```
public ExaminationResult(string description)
```

## Parameters

**description** [string](#)

A description of the result.

# Interface IExaminable

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents any object that is examinable.

```
public interface IExaminable : IPlayerVisible
```

## Inherited Members

[IPlayerVisible.IsPlayerVisible](#)

## Properties

### Commands

Get or set this objects commands.

```
CustomCommand[] Commands { get; set; }
```

### Property Value

[CustomCommand\[\]](#)

Represents any object that is examinable.

## Description

Get or set a description of this object.

```
Description Description { get; set; }
```

### Property Value

[Description](#)

Represents any object that is examinable.

# Identifier

Get this objects identifier.

```
Identifier Identifier { get; }
```

Property Value

## [Identifier](#)

Represents any object that is examinable.

# Methods

## Examine()

Examine this object.

```
ExaminationResult Examine()
```

Returns

## [ExaminationResult](#)

A ExaminationResult detailing the examination of this object.

# Interface IPlayerVisible

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents any object that is visible to a player.

```
public interface IPlayerVisible
```

## Properties

### IsPlayerVisible

Get or set if this is visible to the player.

```
bool IsPlayerVisible { get; set; }
```

Property Value

[bool](#) ↗

Represents any object that is visible to a player.

# Class Identifier

Namespace: [BP.AdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Provides a class that can be used as an identifier.

```
public class Identifier : IEquatable<string>, IEquatable<Identifier>
```

## Inheritance

[object](#) ← Identifier

## Implements

[IEquatable](#)<[string](#)>, [IEquatable](#)<[Identifier](#)>

## Inherited Members

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

# Constructors

## Identifier(string)

Creates a new instance of the Identifier class.

```
public Identifier(string name)
```

## Parameters

name [string](#)

The name.

# Properties

## Empty

Get an empty identifier.

```
public static Identifier Empty { get; }
```

Property Value

#### [Identifier](#)

Provides a class that can be used as an identifier.

**IdentifiableName**

Get the name as a case insensitive identifier.

```
public string IdentifiableName { get; }
```

Property Value

#### [string](#)

Provides a class that can be used as an identifier.

**Name**

Get the name.

```
public string Name { get; }
```

Property Value

#### [string](#)

Provides a class that can be used as an identifier.

**Methods**

## Equals(Identifier)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(Identifier other)
```

Parameters

`other` [Identifier](#)

An object to compare with this object.

Returns

[bool](#)

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

## Equals(string)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(string other)
```

Parameters

`other` [string](#)

An object to compare with this object.

Returns

[bool](#)

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

## ToIdentifiableString(string)

Convert a string to an identifiable string.

```
protected string ToIdentifiableString(string value)
```

## Parameters

**value** [string](#)

The value to convert.

## Returns

[string](#)

The identifiable string.

## ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

## Returns

[string](#)

A string that represents the current object.

# Class Item

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents an item that can be used within the game.

```
public sealed class Item : ExaminableObject, IExaminable,  
IPlayerVisible, IInteractWithItem
```

## Inheritance

[object](#) ← [ExaminableObject](#) ← Item

## Implements

[IExaminable](#), [IPlayerVisible](#), [IInteractWithItem](#)

## Inherited Members

[ExaminableObject.Examination](#) , [ExaminableObject.ToString\(\)](#) ,  
[ExaminableObject.Identifier](#) , [ExaminableObject.Description](#) ,  
[ExaminableObject.Commands](#) , [ExaminableObject.Examine\(\)](#) ,  
[ExaminableObject.IsPlayerVisible](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.ReferenceEquals\(object, object\)](#)

## Constructors

### Item(Identifier, Description, bool)

Initializes a new instance of the Item class.

```
public Item(Identifier identifier, Description description, bool isTakeable = false)
```

## Parameters

### identifier [Identifier](#)

This Items identifier.

### description [Description](#)

A description of this Item.

### [isTakeable](#) [bool](#)

Specify if this item is takeable.

## Item(string, string, bool)

Initializes a new instance of the Item class.

```
public Item(string identifier, string description, bool isTakeable = false)
```

### Parameters

#### [identifier](#) [string](#)

This Items identifier.

#### [description](#) [string](#)

A description of this Item.

#### [isTakeable](#) [bool](#)

Specify if this item is takeable.

## Properties

### Interaction

Get or set the interaction.

```
public InteractionCallback Interaction { get; set; }
```

### Property Value

#### [InteractionCallback](#)

Represents an item that can be used within the game.

# IsTakeable

Get or set if this is takeable.

```
public bool IsTakeable { get; }
```

Property Value

[bool](#)

Represents an item that can be used within the game.

## Methods

### Interact(Item)

Interact with an item.

```
public InteractionResult Interact(Item item)
```

Parameters

[item](#) [Item](#)

The item to interact with.

Returns

[InteractionResult](#)

The result of the interaction.

### Morph(Item)

Handle item morphing.

```
public void Morph(Item item)
```

## Parameters

### **item** [Item](#)

The item to morph into.

# Struct Size

Namespace: [BPAdventureFramework.Assets](#)

Assembly: BP.AdventureFramework.dll

Represents a size.

```
public struct Size
```

## Inherited Members

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

## Constructors

### Size(int, int)

Initializes a new instance of the Size struct.

```
public Size(int width, int height)
```

#### Parameters

**width** [int](#)

The width.

**height** [int](#)

The height.

## Properties

### Height

Get the height.

```
public int Height { get; }
```

Property Value

[int](#)

Represents a size.

## Width

Get the width.

```
public int Width { get; }
```

Property Value

[int](#)

Represents a size.

# Namespace BP.AdventureFramework. Assets.Characters

## Classes

### [Character](#)

Represents a generic in game character.

### [NonPlayableCharacter](#)

Represents a non-playable character.

### [PlayableCharacter](#)

Represents a playable character.

## Interfaces

### [IConverser](#)

Represents an object that can converse.

# Class Character

Namespace: [BPAdventureFramework.Assets.Characters](#)

Assembly: BP.AdventureFramework.dll

Represents a generic in game character.

```
public abstract class Character : ExaminableObject, IExaminable,  
IPlayerVisible, IInteractWithItem
```

## Inheritance

[object](#) ← [ExaminableObject](#) ← Character

## Implements

[IExaminable](#), [IPlayerVisible](#), [IInteractWithItem](#)

## Derived

[NonPlayableCharacter](#), [PlayableCharacter](#)

## Inherited Members

[ExaminableObject.Examination](#) , [ExaminableObject.ToString\(\)](#) ,  
[ExaminableObject.Identifier](#) , [ExaminableObject.Description](#) ,  
[ExaminableObject.Commands](#) , [ExaminableObject.Examine\(\)](#) ,  
[ExaminableObject.IsPlayerVisible](#) , [object.Equals\(object\)](#) , [object.Equals\(object, object\)](#) ,  
[object.GetHashCode\(\)](#) , [object.GetType\(\)](#) , [object.MemberwiseClone\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

# Properties

## Interaction

Get or set the interaction.

```
public InteractionCallback Interaction { get; set; }
```

## Property Value

[InteractionCallback](#)

Represents a generic in game character.

## IsAlive

Get if this character is alive.

```
public bool IsAlive { get; protected set; }
```

Property Value

[bool](#) ↗

Represents a generic in game character.

## Items

Get the items this Character holds.

```
public Item[] Items { get; protected set; }
```

Property Value

[Item](#)[]

Represents a generic in game character.

## Methods

### AcquireItem(Item)

Acquire an item.

```
public virtual void AcquireItem(Item item)
```

Parameters

[item](#) [Item](#)

The item to acquire.

## DequireItem(Item)

De-acquire an item.

```
public virtual void DequireItem(Item item)
```

Parameters

`item` [Item](#)

The item to de-acquire.

## FindItem(string, out Item, bool)

Find an item.

```
public virtual bool FindItem(string itemName, out Item item, bool includeInvisibleItems = false)
```

Parameters

`itemName` [string](#)

The items name.

`item` [Item](#)

The item.

`includeInvisibleItems` [bool](#)

Specify if invisible items should be included.

Returns

[bool](#)

True if the item was found.

# Give(Item, Character)

Give an item to another in game Character.

```
public virtual bool Give(Item item, Character character)
```

Parameters

[item](#) [Item](#)

The item to give.

[character](#) [Character](#)

The Character to give the item to.

Returns

[bool](#) ↗

True if the transaction completed OK, else false.

# HasItem(Item, bool)

Determine if this PlayableCharacter has an item.

```
public virtual bool HasItem(Item item, bool includeInvisibleItems = false)
```

Parameters

[item](#) [Item](#)

The item.

[includeInvisibleItems](#) [bool](#) ↗

Specify if invisible items should be included.

Returns

[bool](#) ↗

True if the item is found, else false.

## Interact(Item)

Interact with an item.

```
public InteractionResult Interact(Item item)
```

Parameters

`item` [Item](#)

The item to interact with.

Returns

[InteractionResult](#)

The result of the interaction.

## InteractWithItem(Item)

Interact with a specified item.

```
protected virtual InteractionResult InteractWithItem(Item item)
```

Parameters

`item` [Item](#)

The item to interact with.

Returns

[InteractionResult](#)

The result of the interaction.

## Kill()

Kill the character.

```
public virtual void Kill()
```

# Interface IConverser

Namespace: [BPAdventureFramework.Assets.Characters](#)

Assembly: BP.AdventureFramework.dll

Represents an object that can converse.

```
public interface IConverser : IExaminable, IPlayerVisible
```

## Inherited Members

[IExaminable.Identifier](#) , [IExaminable.Description](#) , [IExaminable.Commands](#) ,  
[IExaminable.Examine\(\)](#) , [IPlayerVisible.IsPlayerVisible](#)

## Properties

### Conversation

Get or set the conversation.

```
Conversation Conversation { get; set; }
```

### Property Value

#### [Conversation](#)

Represents an object that can converse.

# Class NonPlayableCharacter

Namespace: [BPAdventureFramework.Assets.Characters](#)

Assembly: BP.AdventureFramework.dll

Represents a non-playable character.

```
public sealed class NonPlayableCharacter : Character, IInteractWithItem, IConverser, IExaminable, IPlayerVisible
```

## Inheritance

[object](#) ← [ExaminableObject](#) ← [Character](#) ← NonPlayableCharacter

## Implements

[IInteractWithItem](#), [IConverser](#), [IExaminable](#), [IPlayerVisible](#)

## Inherited Members

[Character.IsAlive](#), [Character.Interaction](#), [Character.Items](#), [Character.Kill\(\)](#),  
[Character.AcquireItem\(Item\)](#), [Character.DequireItem\(Item\)](#),  
[Character.HasItem\(Item, bool\)](#), [Character.FindItem\(string, out Item, bool\)](#),  
[Character.Give\(Item, Character\)](#), [Character.Interact\(Item\)](#),  
[ExaminableObject.Examination](#), [ExaminableObject.ToString\(\)](#),  
[ExaminableObject.Identifier](#), [ExaminableObject.Description](#),  
[ExaminableObject.Commands](#), [ExaminableObject.Examine\(\)](#),  
[ExaminableObject.IsPlayerVisible](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Constructors

### NonPlayableCharacter(Identifier, Description, Conversation)

Initializes a new instance of the NonPlayableCharacter class.

```
public NonPlayableCharacter(Identifier identifier, Description description, Conversation conversation = null)
```

## Parameters

### **identifier** [Identifier](#)

This NonPlayableCharacter's identifier.

### **description** [Description](#)

The description of this NonPlayableCharacter.

### **conversation** [Conversation](#)

The conversation.

## NonPlayableCharacter(Identifier, Description, Conversation, bool, InteractionCallback)

Initializes a new instance of the NonPlayableCharacter class.

```
public NonPlayableCharacter(Identifier identifier, Description description,
Conversation conversation, bool isAlive, InteractionCallback interaction)
```

## Parameters

### **identifier** [Identifier](#)

This NonPlayableCharacter's identifier.

### **description** [Description](#)

The description of this NonPlayableCharacter.

### **conversation** [Conversation](#)

The conversation.

### **isAlive** [bool](#)

Set if this NonPlayableCharacter is alive.

### **interaction** [InteractionCallback](#)

Set this NonPlayableCharacter's interaction.

## NonPlayableCharacter(Identifier, Description, Conversation, bool, InteractionCallback, ExaminationCallback)

Initializes a new instance of the NonPlayableCharacter class.

```
public NonPlayableCharacter(Identifier identifier, Description description,
Conversation conversation, bool isAlive, InteractionCallback interaction,
ExaminationCallback examination)
```

### Parameters

`identifier` [Identifier](#)

This NonPlayableCharacter's identifier.

`description` [Description](#)

The description of this NonPlayableCharacter.

`conversation` [Conversation](#)

The conversation.

`isAlive` [bool](#)

Set if this NonPlayableCharacter is alive.

`interaction` [InteractionCallback](#)

Set this NonPlayableCharacter's interaction.

`examination` [ExaminationCallback](#)

Set this NonPlayableCharacter's examination.

## NonPlayableCharacter(string, string, Conversation)

Initializes a new instance of the NonPlayableCharacter class.

```
public NonPlayableCharacter(string identifier, string description, Conversation
conversation = null)
```

## Parameters

**identifier** [string](#)

This NonPlayableCharacter's identifier.

**description** [string](#)

The description of this NonPlayableCharacter.

**conversation** [Conversation](#)

The conversation.

## Properties

### Conversation

Get or set the conversation.

```
public Conversation Conversation { get; set; }
```

Property Value

[Conversation](#)

Represents a non-playable character.

# Class PlayableCharacter

Namespace: [BPAdventureFramework.Assets.Characters](#)

Assembly: BP.AdventureFramework.dll

Represents a playable character.

```
public sealed class PlayableCharacter : Character, IExaminable,  
IPlayerVisible, IInteractWithItem
```

## Inheritance

[object](#) ← [ExaminableObject](#) ← [Character](#) ← PlayableCharacter

## Implements

[IExaminable](#), [IPlayerVisible](#), [IInteractWithItem](#)

## Inherited Members

[Character.IsAlive](#), [Character.Interaction](#), [Character.Items](#), [Character.Kill\(\)](#),  
[Character.AcquireItem\(Item\)](#), [Character.DequireItem\(Item\)](#),  
[Character.HasItem\(Item, bool\)](#), [Character.FindItem\(string, out Item, bool\)](#),  
[Character.Give\(Item, Character\)](#), [Character.Interact\(Item\)](#),  
[ExaminableObject.Examination](#), [ExaminableObject.ToString\(\)](#),  
[ExaminableObject.Identifier](#), [ExaminableObject.Description](#),  
[ExaminableObject.Commands](#), [ExaminableObject.Examine\(\)](#),  
[ExaminableObject.IsPlayerVisible](#), [object.Equals\(object\)](#), [object.Equals\(object, object\)](#),  
[object.GetHashCode\(\)](#), [object.GetType\(\)](#), [object.ReferenceEquals\(object, object\)](#)

## Constructors

### PlayableCharacter(Identifier, Description, params Item[])

Initializes a new instance of the PlayableCharacter class.

```
public PlayableCharacter(Identifier identifier, Description description, params  
Item[] items)
```

## Parameters

**identifier** [Identifier](#)

This PlayableCharacter's identifier.

**description** [Description](#)

The description of the player.

**items** [Item\[\]](#)

The players items.

## PlayableCharacter(string, string, params Item[])

Initializes a new instance of the PlayableCharacter class.

```
public PlayableCharacter(string identifier, string description, params Item[] items)
```

### Parameters

**identifier** [string](#)

This PlayableCharacter's identifier.

**description** [string](#)

The description of the player.

**items** [Item\[\]](#)

The players items.

## Methods

### UseItem(IInteractWithItem, Item)

Use an item.

```
public InteractionResult UseItem(IInteractWithItem targetObject, Item item)
```

## Parameters

**targetObject** [IInteractWithItem](#)

A target object to use the item on.

**item** [Item](#)

The item to use.

## Returns

[InteractionResult](#)

The result of the items usage.

# Namespace BP.AdventureFramework. Assets.Interaction

## Classes

### [InteractionResult](#)

Represents a result of an interaction.

### [Reaction](#)

Represents a reaction.

### [Result](#)

Represents a result.

## Interfaces

### [IInteractWithItem](#)

Represents any object that can interact with an item.

## Enums

### [InteractionEffect](#)

Enumeration of interaction effects.

### [ReactionResult](#)

Enumeration of reaction results.

## Delegates

### [Condition](#)

Represents a callback for conditions.

### [InteractionCallback](#)

Represents the callback for interacting with objects.

# Delegate Condition

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Represents a callback for conditions.

```
public delegate bool Condition()
```

Returns

[bool](#)

The result of the condition.

# Interface IInteractWithItem

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can interact with an item.

```
public interface IInteractWithItem
```

## Methods

### Interact(Item)

Interact with an item.

```
InteractionResult Interact(Item item)
```

#### Parameters

`item` [Item](#)

The item to interact with.

#### Returns

[InteractionResult](#)

The result of the interaction.

# Delegate InteractionCallback

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Represents the callback for interacting with objects.

```
public delegate InteractionResult InteractionCallback(Item item)
```

Parameters

**item** [Item](#)

The item to interact with.

Returns

[InteractionResult](#)

The result of the interaction.

# Enum InteractionEffect

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Enumeration of interaction effects.

```
public enum InteractionEffect
```

## Fields

**FatalEffect = 3**

A fatal effect to the interaction.

**ItemMorphed = 2**

Item morphed into another object.

**ItemUsedUp = 1**

Item was used up.

**NoEffect = 0**

No effect to the interaction on either the item or the target.

**SelfContained = 5**

Any other self contained effect.

**TargetUsedUp = 4**

The target was used up.

# Class InteractionResult

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Represents a result of an interaction.

```
public sealed class InteractionResult : Result
```

## Inheritance

[object](#) ← [Result](#) ← InteractionResult

## Inherited Members

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

## Constructors

### InteractionResult(InteractionEffect, Item)

Initializes a new instance of the InteractionResult class.

```
public InteractionResult(InteractionEffect effect, Item item)
```

#### Parameters

**effect** [InteractionEffect](#)

The effect of this interaction.

**item** [Item](#)

The item used in this interaction.

### InteractionResult(InteractionEffect, Item, string)

Initializes a new instance of the InteractionResult class.

```
public InteractionResult(InteractionEffect effect, Item item,  
string descriptionOfEffect)
```

## Parameters

**effect** [InteractionEffect](#)

The effect of this interaction.

**item** [Item](#)

The item used in this interaction.

**descriptionOfEffect** [string](#)

A description of the effect.

## Properties

### Effect

Get the effect.

```
public InteractionEffect Effect { get; }
```

### Property Value

[InteractionEffect](#)

Represents a result of an interaction.

### Item

Get the item used in the interaction.

```
public Item Item { get; }
```

### Property Value

## Item

Represents a result of an interaction.

# Class Reaction

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Represents a reaction.

```
public sealed class Reaction
```

## Inheritance

[object](#) ← Reaction

## Inherited Members

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

## Constructors

### Reaction(ReactionResult, string)

Initializes a new instance of the Reaction class.

```
public Reaction(ReactionResult result, string description)
```

#### Parameters

**result** [ReactionResult](#)

The result.

**description** [string](#)

A description of the result.

## Properties

### Description

Get a description of the result.

```
public string Description { get; }
```

Property Value

[string](#)

Represents a reaction.

## Result

Get the result.

```
public ReactionResult Result { get; }
```

Property Value

[ReactionResult](#)

Represents a reaction.

# Enum ReactionResult

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Enumeration of reaction results.

```
public enum ReactionResult
```

## Fields

**Error** = 0

Error.

**Fatal** = 3

A reaction that has a fatal effect on the player.

**Internal** = 2

An internal reaction.

**OK** = 1

OK.

# Class Result

Namespace: [BPAdventureFramework.Assets.Interaction](#)

Assembly: BP.AdventureFramework.dll

Represents a result.

```
public abstract class Result
```

## Inheritance

[object](#) ← Result

## Derived

[ExaminationResult](#), [InteractionResult](#)

## Inherited Members

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

# Constructors

## Result()

Initializes a new instance of the Result class.

```
protected Result()
```

# Properties

## Description

Get the description.

```
public string Description { get; protected set; }
```

## Property Value

[string](#) ↗

Represents a result.

# Namespace BP.AdventureFramework. Assets.Locations

## Classes

### [Exit](#)

Represents an exit from a GameLocation.

### [Matrix](#)

Provides a 3D matrix of rooms.

### [Overworld](#)

Represents an entire overworld.

### [Region](#)

Represents a region.

### [Room](#)

Represents a room

### [RoomPosition](#)

Represents a room position.

### [ViewPoint](#)

Represents a view point from a room.

## Enums

### [Direction](#)

Enumeration of directions.

# Enum Direction

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Enumeration of directions.

```
public enum Direction
```

## Extension Methods

[DirectionExtensions.Inverse\(Direction\)](#)

## Fields

Down = 5

Down.

East = 1

East.

North = 0

North.

South = 2

South.

Up = 4

Up.

West = 3

West.

# Class Exit

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Represents an exit from a GameLocation.

```
public sealed class Exit : ExaminableObject, IExaminable, IPlayerVisible
```

## Inheritance

[object](#) ↗ ← [ExaminableObject](#) ← Exit

## Implements

[IExaminable](#), [IPlayerVisible](#)

## Inherited Members

[ExaminableObject.Examination](#) , [ExaminableObject.ToString\(\)](#) ,  
[ExaminableObject.Identifier](#) , [ExaminableObject.Description](#) ,  
[ExaminableObject.Commands](#) , [ExaminableObject.Examine\(\)](#) ,  
[ExaminableObject.IsPlayerVisible](#) , [object.Equals\(object\)](#) ↗ , [object.Equals\(object, object\)](#) ↗ ,  
[object.GetHashCode\(\)](#) ↗ , [object.GetType\(\)](#) ↗ , [object.ReferenceEquals\(object, object\)](#) ↗

## Constructors

### Exit(Direction, bool, Description)

Initializes a new instance of the Exit class.

```
public Exit(Direction direction, bool isLocked = false, Description description  
= null)
```

## Parameters

### direction [Direction](#)

The direction of the exit.

### isLocked [bool](#) ↗

If this exit is locked.

### [description](#) [Description](#)

A description of the exit.

## Properties

### Direction

Get the direction of the exit.

```
public Direction Direction { get; }
```

#### Property Value

##### [Direction](#)

Represents an exit from a GameLocation.

### IsLocked

Get if this Exit is locked.

```
public bool IsLocked { get; }
```

#### Property Value

##### [bool](#) ↗

Represents an exit from a GameLocation.

## Methods

### Lock()

Lock this exit.

```
public void Lock()
```

## Unlock()

Unlock this exit.

```
public void Unlock()
```

# Class Matrix

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Provides a 3D matrix of rooms.

```
public sealed class Matrix
```

## Inheritance

[object](#) ← Matrix

## Inherited Members

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

## Constructors

### Matrix(Room[, ,])

Initializes a new instance of the Matrix class.

```
public Matrix(Room[, ,] rooms)
```

#### Parameters

rooms [Room\[, ,\]](#)

The rooms to be represented.

## Properties

### Depth

Get the depth of the matrix.

```
public int Depth { get; }
```

Property Value

[int](#)

Provides a 3D matrix of rooms.

## Height

Get the height of the matrix.

```
public int Height { get; }
```

Property Value

[int](#)

Provides a 3D matrix of rooms.

## this[int, int, int]

Get a room in this matrix.

```
public Room this[int x, int y, int z] { get; }
```

Parameters

x [int](#)

The x position.

y [int](#)

The y position.

z [int](#)

The z position.

## Property Value

### [Room](#)

The room.

## Width

Get the width of the matrix.

```
public int Width { get; }
```

## Property Value

### [int](#)

Provides a 3D matrix of rooms.

## Methods

### ToRooms()

Return this matrix as a one dimensional array of rooms.

```
public Room[] ToRooms()
```

## Returns

### [Room\[\]](#)

The rooms, as a one dimensional array.

# Class Overworld

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Represents an entire overworld.

```
public sealed class Overworld : ExaminableObject, IExaminable, IPlayerVisible
```

## Inheritance

[object](#) ← [ExaminableObject](#) ← Overworld

## Implements

[IExaminable](#), [IPlayerVisible](#)

## Inherited Members

[ExaminableObject.Examination](#) , [ExaminableObject.ToString\(\)](#) ,  
[ExaminableObject.Identifier](#) , [ExaminableObject.Description](#) ,  
[ExaminableObject.Commands](#) , [ExaminableObject.IsPlayerVisible](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### Overworld(Identifier, Description)

Initializes a new instance of the overworld class.

```
public Overworld(Identifier identifier, Description description)
```

## Parameters

### identifier [Identifier](#)

The identifier for this overworld.

### description [Description](#)

A description of this overworld.

# Overworld(string, string)

Initializes a new instance of the overworld class.

```
public Overworld(string identifier, string description)
```

## Parameters

**identifier** [string](#)

The identifier for this overworld.

**description** [string](#)

A description of this overworld.

## Properties

### CurrentRegion

Get the current region.

```
public Region CurrentRegion { get; }
```

## Property Value

[Region](#)

Represents an entire overworld.

### Regions

Get the regions in this overworld.

```
public Region[] Regions { get; }
```

## Property Value

## [Region\[\]](#)

Represents an entire overworld.

## Methods

### AddRegion(Region)

Add a region to this overworld.

```
public void AddRegion(Region region)
```

Parameters

`region` [Region](#)

The region to add.

### Examine()

Examine this object.

```
public override ExaminationResult Examine()
```

Returns

[ExaminationResult](#)

A `ExaminationResult` detailing the examination of this object.

### FindRegion(string, out Region)

Find a region.

```
public bool FindRegion(string regionName, out Region region)
```

Parameters

`regionName` [string](#)

The regions name.

`region` [Region](#)

The region.

Returns

[bool](#)

True if the region was found.

## Move(Region)

Move to a region.

```
public bool Move(Region region)
```

Parameters

`region` [Region](#)

The region to move to.

Returns

[bool](#)

True if the region could be moved to, else false.

## RemoveRegion(Region)

Remove a region from this overworld.

```
public void RemoveRegion(Region region)
```

Parameters

## region [Region](#)

The region to remove.

# Class Region

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Represents a region.

```
public sealed class Region : ExaminableObject, IExaminable, IPlayerVisible
```

## Inheritance

[object](#) ↳ [ExaminableObject](#) ↳ Region

## Implements

[IExaminable](#), [IPlayerVisible](#)

## Inherited Members

[ExaminableObject.Examination](#) , [ExaminableObject.ToString\(\)](#) ,  
[ExaminableObject.Identifier](#) , [ExaminableObject.Description](#) ,  
[ExaminableObject.Commands](#) , [ExaminableObject.IsPlayerVisible](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### Region(Identifier, Description)

Initializes a new instance of the Region class.

```
public Region(Identifier identifier, Description description)
```

## Parameters

**identifier** [Identifier](#)

This Regions identifier.

**description** [Description](#)

The description of this Region.

## Region(string, string)

Initializes a new instance of the Region class.

```
public Region(string identifier, string description)
```

### Parameters

**identifier** [string](#)

This Regions identifier.

**description** [string](#)

The description of this Region.

## Properties

### CurrentRoom

Get the current room.

```
public Room CurrentRoom { get; }
```

### Property Value

[Room](#)

Represents a region.

### this[int, int, int]

Get a room at a specified location.

```
public Room this[int x, int y, int z] { get; }
```

### Parameters

x [int](#)

The x position.

y [int](#)

The y position.

z [int](#)

The z position.

Property Value

[Room](#)

The room.

## Rooms

Get the number of rooms region contains.

```
public int Rooms { get; }
```

Property Value

[int](#)

Represents a region.

## VisibleWithoutDiscovery

Get if the current region is visible without discovery.

```
public bool VisibleWithoutDiscovery { get; set; }
```

Property Value

[bool](#)

Represents a region.

## Methods

### AddRoom(Room, int, int, int)

Add a Room to this region.

```
public bool AddRoom(Room room, int x, int y, int z)
```

#### Parameters

room [Room](#)

The room to add.

x [int](#)

The x position within the region.

y [int](#)

The y position within the region.

z [int](#)

The z position within the region.

#### Returns

[bool](#)

Represents a region.

### Examine()

Examine this object.

```
public override ExaminationResult Examine()
```

Returns

## [ExaminationResult](#)

A ExaminationResult detailing the examination of this object.

## GetAdjoiningRoom(Direction)

Get an adjoining room to the Region.CurrentRoom property.

```
public Room GetAdjoiningRoom(Direction direction)
```

Parameters

### [direction](#) [Direction](#)

The direction of the adjoining Room.

Returns

## [Room](#)

The adjoining Room.

## GetAdjoiningRoom(Direction, Room)

Get an adjoining room to a room.

```
public Room GetAdjoiningRoom(Direction direction, Room room)
```

Parameters

### [direction](#) [Direction](#)

The direction of the adjoining room.

### [room](#) [Room](#)

The room to use as the reference.

Returns

[Room](#)

The adjoining room.

## GetPositionOfRoom(Room)

Get the position of a room.

```
public RoomPosition GetPositionOfRoom(Room room)
```

Parameters

[room](#) [Room](#)

The room.

Returns

[RoomPosition](#)

The position of the room.

## JumpToRoom(int, int, int)

Jump to a room.

```
public bool JumpToRoom(int x, int y, int z)
```

Parameters

[x](#) [int](#)

The x location of the room.

[y](#) [int](#)

The y location of the room.

`z` [int ↗](#)

The z location of the room.

Returns

[bool ↗](#)

True if the room could be jumped to, else false.

## Move(Direction)

Move in a direction.

```
public bool Move(Direction direction)
```

Parameters

`direction` [Direction](#)

The direction to move in.

Returns

[bool ↗](#)

True if the move was successful, else false.

## SetStartRoom(Room)

Set the room to start in.

```
public void SetStartRoom(Room room)
```

Parameters

`room` [Room](#)

The Room to start in.

## SetStartRoom(int, int, int)

Set the room to start in.

```
public void SetStartRoom(int x, int y, int z)
```

Parameters

x [int](#)

The x position.

y [int](#)

The y position.

z [int](#)

The z position.

## ToMatrix()

Get this region as a 3D matrix of rooms.

```
public Matrix ToMatrix()
```

Returns

[Matrix](#)

This region, as a 3D matrix.

## UnlockDoorPair(Direction)

Unlock a pair of doors in a specified direction in the CurrentRoom.

```
public bool UnlockDoorPair(Direction direction)
```

Parameters

**direction** [Direction](#)

The direction to unlock in.

Returns

[bool](#) ↗

True if the door pair could be unlocked, else false.

# Class Room

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Represents a room

```
public sealed class Room : ExaminableObject, IExaminable,  
IPlayerVisible, IInteractWithItem
```

## Inheritance

[object](#) ← [ExaminableObject](#) ← Room

## Implements

[IExaminable](#), [IPlayerVisible](#), [IInteractWithItem](#)

## Inherited Members

[ExaminableObject.Examination](#) , [ExaminableObject.ToString\(\)](#) ,  
[ExaminableObject.Identifier](#) , [ExaminableObject.Description](#) ,  
[ExaminableObject.Commands](#) , [ExaminableObject.IsPlayerVisible](#) , [object.Equals\(object\)](#) ,  
[object.Equals\(object, object\)](#) , [object.GetHashCode\(\)](#) , [object.GetType\(\)](#) ,  
[object.ReferenceEquals\(object, object\)](#)

## Constructors

### Room(Identifier, Description, params Exit[])

Initializes a new instance of the Room class.

```
public Room(Identifier identifier, Description description, params Exit[] exits)
```

## Parameters

### identifier [Identifier](#)

This rooms identifier.

### description [Description](#)

This rooms description.

`exits` [Exit\[\]](#)

The exits from this room.

## Room(Identifier, Description, Exit[], params Item[])

Initializes a new instance of the Room class.

```
public Room(Identifier identifier, Description description, Exit[] exits = null,  
params Item[] items)
```

### Parameters

`identifier` [Identifier](#)

This rooms identifier.

`description` [Description](#)

This rooms description.

`exits` [Exit\[\]](#)

The exits from this room.

`items` [Item\[\]](#)

The items in this room.

## Room(string, string, params Exit[])

Initializes a new instance of the Room class.

```
public Room(string identifier, string description, params Exit[] exits)
```

### Parameters

`identifier` [string](#) ↗

This rooms identifier.

**description** [string](#)

This rooms description.

**exits** [Exit\[\]](#)

The exits from this room.

## Room(string, string, Exit[], params Item[])

Initializes a new instance of the Room class.

```
public Room(string identifier, string description, Exit[] exits = null, params Item[] items)
```

### Parameters

**identifier** [string](#)

This rooms identifier.

**description** [string](#)

This rooms description.

**exits** [Exit\[\]](#)

The exits from this room.

**items** [Item\[\]](#)

The items in this room.

## Properties

### Characters

Get the characters in this Room.

```
public NonPlayableCharacter[] Characters { get; }
```

Property Value

[NonPlayableCharacter\[\]](#)

Represents a room

## EnteredFrom

Get which direction this Room was entered from.

```
public Direction? EnteredFrom { get; }
```

Property Value

[Direction?](#)

Represents a room

## Exits

Get the exits.

```
public Exit[] Exits { get; }
```

Property Value

[Exit\[\]](#)

Represents a room

## HasBeenVisited

Get if this location has been visited.

```
public bool HasBeenVisited { get; }
```

Property Value

[bool](#)

Represents a room

## Interaction

Get or set the interaction.

```
public InteractionCallback Interaction { get; set; }
```

Property Value

[InteractionCallback](#)

Represents a room

## this[Direction]

Get an exit.

```
public Exit this[Direction direction] { get; }
```

Parameters

**direction** [Direction](#)

The direction of an exit.

Property Value

[Exit](#)

The exit.

## Items

Get the items in this Room.

```
public Item[] Items { get; }
```

Property Value

[Item\[\]](#)

Represents a room

## UnlockedExits

Get all unlocked exits.

```
public Exit[] UnlockedExits { get; }
```

Property Value

[Exit\[\]](#)

Represents a room

## Methods

### AddCharacter(NonPlayableCharacter)

Add a character to this room.

```
public void AddCharacter(NonPlayableCharacter character)
```

Parameters

character [NonPlayableCharacter](#)

The character to add.

## AddExit(Exit)

Add an exit to this room.

```
public void AddExit(Exit exit)
```

Parameters

exit [Exit](#)

The exit to add.

## AddItem(Item)

Add an item to this room.

```
public void AddItem(Item item)
```

Parameters

item [Item](#)

The item to add.

## CanMove(Direction)

Test if a move is possible.

```
public bool CanMove(Direction direction)
```

Parameters

direction [Direction](#)

The direction to test.

Returns

[bool](#)

If a move in the specified direction is possible.

## ContainsCharacter(NonPlayableCharacter, bool)

Get if this Room contains a character.

```
public bool ContainsCharacter(NonPlayableCharacter character, bool  
includeInvisibleCharacters = false)
```

Parameters

**character** [NonPlayableCharacter](#)

The character.

**includeInvisibleCharacters** [bool](#)

Specify if invisible characters should be included.

Returns

[bool](#)

True if the item is in this room, else false.

## ContainsCharacter(string, bool)

Get if this Room contains a character.

```
public bool ContainsCharacter(string characterName, bool includeInvisibleCharacters  
= false)
```

Parameters

**characterName** [string](#)

The character name to check for.

`includeInvisibleCharacters` [bool](#)

Specify if invisible characters should be included.

Returns

[bool](#)

True if the item is in this room, else false.

## ContainsExit(Direction, bool)

Get if this Room contains an exit.

```
public bool ContainsExit(Direction direction, bool includeInvisibleExits = false)
```

Parameters

`direction` [Direction](#)

The direction of the exit to check for.

`includeInvisibleExits` [bool](#)

Specify if invisible exits should be included.

Returns

[bool](#)

True if the exit exists, else false.

## ContainsExit(bool)

Get if this Room contains an exit.

```
public bool ContainsExit(bool includeInvisibleExits = false)
```

Parameters

`includeInvisibleExits` [bool](#)

Specify if invisible exits should be included.

Returns

[bool](#)

True if the exit exists, else false.

## ContainsInteractionTarget(string)

Get if this Room contains an interaction target.

```
public bool ContainsInteractionTarget(string targetName)
```

Parameters

`targetName` [string](#)

The name of the target to check for.

Returns

[bool](#)

True if the target is in this room, else false.

## ContainsItem(Item)

Get if this Room contains an item. This will not include items whose ExaminableObject.IsPlayerVisible property is set to false.

```
public bool ContainsItem(Item item)
```

Parameters

`item` [Item](#)

The item to check for.

Returns

[bool](#)

True if the item is in this room, else false.

## ContainsItem(string, bool)

Get if this Room contains an item.

```
public bool ContainsItem(string itemName, bool includeInvisibleItems = false)
```

Parameters

[itemName](#) [string](#)

The item name to check for.

[includeInvisibleItems](#) [bool](#)

Specify if invisible items should be included.

Returns

[bool](#)

True if the item is in this room, else false.

## Examine()

Handle examination this Room.

```
public override ExaminationResult Examine()
```

Returns

[ExaminationResult](#)

The result of this examination.

## FindCharacter(string, out NonPlayableCharacter)

Find a character. This will not include characters whose ExaminableObject.IsPlayerVisible property is set to false.

```
public bool FindCharacter(string characterName, out NonPlayableCharacter character)
```

### Parameters

**characterName** [string](#)

The character name.

**character** [NonPlayableCharacter](#)

The character name.

### Returns

[bool](#)

True if the character was found.

## FindCharacter(string, out NonPlayableCharacter, bool)

Find a character.

```
public bool FindCharacter(string characterName, out NonPlayableCharacter character,  
bool includeInvisibleCharacters)
```

### Parameters

**characterName** [string](#)

The character name.

**character** [NonPlayableCharacter](#)

The character.

`includeInvisibleCharacters` [bool](#)

Specify if invisible characters should be included.

Returns

[bool](#)

True if the character was found.

## FindExit(Direction, bool, out Exit)

Find an exit.

```
public bool FindExit(Direction direction, bool includeInvisibleExits, out Exit exit)
```

Parameters

`direction` [Direction](#)

The exits direction.

`includeInvisibleExits` [bool](#)

Specify if invisible exists should be included.

`exit` [Exit](#)

The exit.

Returns

[bool](#)

True if the exit was found.

## FindInteractionTarget(string, out IInteractWithItem)

Find an interaction target.

```
public bool FindInteractionTarget(string targetName, out IInteractWithItem target)
```

## Parameters

**targetName** [string](#)

The targets name.

**target** [IInteractWithItem](#)

The target.

## Returns

[bool](#)

True if the target was found.

## FindItem(string, out Item)

Find an item. This will not include items whose ExaminableObject.IsPlayerVisible property is set to false

```
public bool FindItem(string itemName, out Item item)
```

## Parameters

**itemName** [string](#)

The items name. This is case insensitive

**item** [Item](#)

The item

## Returns

[bool](#)

True if the item was found

## FindItem(string, out Item, bool)

Find an item.

```
public bool FindItem(string itemName, out Item item, bool includeInvisibleItems)
```

Parameters

`itemName` [string](#)

The items name.

`item` [Item](#)

The item.

`includeInvisibleItems` [bool](#)

Specify is invisible items should be included.

Returns

[bool](#)

True if the item was found.

## HasLockedExitInDirection(Direction, bool)

Get if this room has a visible locked exit in a specified direction.

```
public bool HasLockedExitInDirection(Direction direction, bool includeInvisibleExits = false)
```

Parameters

`direction` [Direction](#)

The direction to check.

`includeInvisibleExits` [bool](#)

Specify if invisible exits should be included.

Returns

[bool](#)

If there is a locked exit in the specified direction.

## HasUnlockedExitInDirection(Direction, bool)

Get if this room has a visible unlocked exit in a specified direction.

```
public bool HasUnlockedExitInDirection(Direction direction, bool  
includeInvisibleExits = false)
```

Parameters

[direction](#) [Direction](#)

The direction to check.

[includeInvisibleExits](#) [bool](#)

Specify if invisible exits should be included.

Returns

[bool](#)

If there is a unlocked exit in the specified direction.

## Interact(Item)

Interact with an item.

```
public InteractionResult Interact(Item item)
```

Parameters

[item](#) [Item](#)

The item to interact with.

Returns

## [InteractionResult](#)

The result of the interaction.

## MovedInto(Direction?)

Handle movement into this GameLocation.

```
public void MovedInto(Direction? fromDirection)
```

Parameters

### [fromDirection Direction?](#)

The direction movement into this Room is from. Use null if there is no direction.

## RemoveCharacter(NonPlayableCharacter)

Remove a character from the room.

```
public void RemoveCharacter(NonPlayableCharacter character)
```

Parameters

### [character NonPlayableCharacter](#)

The character to remove.

## RemoveExit(Exit)

Remove an exit from the room.

```
public void RemoveExit(Exit exit)
```

Parameters

exit [Exit](#)

The exit to remove.

## RemoveInteractionTarget(IInteractWithItem)

Remove an interaction target from the room.

```
public IInteractWithItem RemoveInteractionTarget(IInteractWithItem target)
```

Parameters

target [IInteractWithItem](#)

The target to remove.

Returns

[IInteractWithItem](#)

The target removed from this room.

## RemoveItem(Item)

Remove an item from the room.

```
public void RemoveItem(Item item)
```

Parameters

item [Item](#)

The item to remove.

## SpecifyConditionalDescription(ConditionalDescription)

Specify a conditional description of this room.

```
public void SpecifyConditionalDescription(ConditionalDescription description)
```

## Parameters

**description** [ConditionalDescription](#)

The description of this room.

# Class RoomPosition

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Represents a room position.

```
public class RoomPosition
```

## Inheritance

[object](#) ← RoomPosition

## Inherited Members

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

## Constructors

### RoomPosition(Room, int, int, int)

Initializes a new instance of the RoomPosition class.

```
public RoomPosition(Room room, int x, int y, int z)
```

## Parameters

room [Room](#)

The room/

x [int](#)

The x position of the room.

y [int](#)

The y position of the room.

z [int](#)

The z position of the room.

## Properties

### Room

Get the room.

```
public Room Room { get; }
```

Property Value

[Room](#)

Represents a room position.

### X

Get the X position of the room.

```
public int X { get; }
```

Property Value

[int](#)

Represents a room position.

### Y

Get the Y position of the room.

```
public int Y { get; }
```

Property Value

[int](#)

Represents a room position.

Z

Get the Z position of the room.

```
public int Z { get; }
```

Property Value

[int](#)

Represents a room position.

## Methods

**IsAtPosition(int, int, int)**

Get if this RoomPosition is at a position.

```
public bool IsAtPosition(int x, int y, int z)
```

Parameters

x [int](#)

The X position.

y [int](#)

The Y position.

z [int](#)

The Z position.

Returns

## bool ↴

True if this is at the position, else false.

# Class ViewPoint

Namespace: [BPAdventureFramework.Assets.Locations](#)

Assembly: BP.AdventureFramework.dll

Represents a view point from a room.

```
public sealed class ViewPoint
```

## Inheritance

[object](#) ← ViewPoint

## Inherited Members

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

## Properties

### Any

Get if there is a view in any direction.

```
public bool Any { get; }
```

Property Value

[bool](#)

Represents a view point from a room.

### AnyNotVisited

Get if there is a view in any direction.

```
public bool AnyNotVisited { get; }
```

Property Value

[bool](#) ↗

Represents a view point from a room.

## AnyVisited

Get if there is a view in any direction.

```
public bool AnyVisited { get; }
```

Property Value

[bool](#) ↗

Represents a view point from a room.

## this[Direction]

Get the room that lies in a specified direction.

```
public Room this[Direction direction] { get; }
```

Parameters

[direction](#) [Direction](#)

The direction to check.

Property Value

[Room](#)

The room.

## NoView

Get a view point representing no view.

```
public static ViewPoint NoView { get; }
```

Property Value

#### [ViewPoint](#)

Represents a view point from a room.

## Methods

### Create(Region)

Create a new ViewPoint.

```
public static ViewPoint Create(Region region)
```

Parameters

#### [region](#) [Region](#)

The region to create the view point from.

Returns

#### [ViewPoint](#)

The view point.

# Namespace BP.AdventureFramework.Commands

## Classes

### [CustomCommand](#)

Provides a custom command.

## Interfaces

### [ICommand](#)

Represents a command.

## Delegates

### [CustomCommandCallback](#)

Provides a callback for custom commands.

# Class CustomCommand

Namespace: [BPAdventureFramework.Commands](#)

Assembly: BP.AdventureFramework.dll

Provides a custom command.

```
public class CustomCommand : ICommand, IPlayerVisible
```

## Inheritance

[object](#) ← CustomCommand

## Implements

[ICommand](#), [IPlayerVisible](#)

## Inherited Members

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

## Constructors

CustomCommand(CommandHelp, bool,  
CustomCommandCallback)

Initializes a new instance of the CustomCommand class.

```
public CustomCommand(CommandHelp help, bool isPlayerVisible,  
CustomCommandCallback callback)
```

## Parameters

help [CommandHelp](#)

The help for this command.

isPlayerVisible [bool](#)

If this is visible to the player.

## callback [CustomCommandCallback](#)

The callback to invoke when this command is invoked.

## Properties

### Arguments

Get or set the arguments.

```
public string[] Arguments { get; set; }
```

### Property Value

[string](#)[]

Provides a custom command.

## Help

Get the help for this command.

```
public CommandHelp Help { get; }
```

### Property Value

[CommandHelp](#)

Provides a custom command.

## IsPlayerVisible

Get or set if this is visible to the player.

```
public bool IsPlayerVisible { get; set; }
```

### Property Value

[bool](#)

Provides a custom command.

## Methods

### Invoke(Game)

Invoke the command.

```
public Reaction Invoke(Game game)
```

#### Parameters

game [Game](#)

The game to invoke the command on.

#### Returns

[Reaction](#)

The reaction.

# Delegate CustomCommandCallback

Namespace: [BPAdventureFramework.Commands](#)

Assembly: BP.AdventureFramework.dll

Provides a callback for custom commands.

```
public delegate Reaction CustomCommandCallback(Game game, string[] arguments)
```

Parameters

game [Game](#)

The game to invoke the command on.

arguments [string](#)[]

The arguments to invoke the command with.

Returns

[Reaction](#)

The reaction to the command.

# Interface ICommand

Namespace: [BPAdventureFramework.Commands](#)

Assembly: BP.AdventureFramework.dll

Represents a command.

```
public interface ICommand
```

## Methods

### Invoke(Game)

Invoke the command.

```
Reaction Invoke(Game game)
```

#### Parameters

game [Game](#)

The game to invoke the command on.

#### Returns

[Reaction](#)

The reaction.

# Namespace BP.AdventureFramework. Conversations

## Classes

### [Conversation](#)

Represents a conversation.

### [LogItem](#)

Provides a container for log items.

### [Paragraph](#)

Represents a paragraph in a Conversation.

### [Response](#)

Provides a response to a conversation.

## Enums

### [Participant](#)

Enumeration of participants in a conversation.

## Delegates

### [ConversationActionCallback](#)

Provides a callback that can be used in conversations invoking actions.

# Class Conversation

Namespace: [BPAdventureFramework.Conversations](#)

Assembly: BP.AdventureFramework.dll

Represents a conversation.

```
public sealed class Conversation
```

## Inheritance

[object](#) ← Conversation

## Inherited Members

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

## Constructors

### Conversation(params Paragraph[])

Initializes a new instance of the Conversation class.

```
public Conversation(params Paragraph[] paragraphs)
```

#### Parameters

paragraphs [Paragraph\[\]](#)

The paragraphs.

## Properties

### CurrentParagraph

Get the current paragraph in the conversation.

```
public Paragraph CurrentParagraph { get; }
```

Property Value

### Paragraph

Represents a conversation.

## Log

Get the log.

```
public LogItem[] Log { get; }
```

Property Value

### LogItem[]

Represents a conversation.

## Paragraphs

Get the current paragraph in the conversation.

```
public Paragraph[] Paragraphs { get; }
```

Property Value

### Paragraph[]

Represents a conversation.

## Methods

### Next(Game)

Trigger the next line in this conversation.

```
public Reaction Next(Game game)
```

## Parameters

game [Game](#)

The game.

## Returns

[Reaction](#)

The reaction to the line.

## Respond(Response, Game)

Respond to the conversation.

```
public Reaction Respond(Response response, Game game)
```

## Parameters

response [Response](#)

The response.

game [Game](#)

The game.

## Returns

[Reaction](#)

The reaction to the response.

# Delegate ConversationActionCallback

Namespace: [BPAdventureFramework.Conversations](#)

Assembly: BP.AdventureFramework.dll

Provides a callback that can be used in conversations invoking actions.

```
public delegate void ConversationActionCallback(Game game)
```

## Parameters

game [Game](#)

The game to invoke the callback on.

# Class LogItem

Namespace: [BPAdventureFramework.Conversations](#)

Assembly: BP.AdventureFramework.dll

Provides a container for log items.

```
public sealed class LogItem
```

## Inheritance

[object](#) ← LogItem

## Inherited Members

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

## Constructors

### LogItem(Participant, string)

Initializes a new instance of the LogItem class.

```
public LogItem(Participant participant, string line)
```

#### Parameters

participant [Participant](#)

The participant.

line [string](#)

The line.

## Properties

### Line

Get the line.

```
public string Line { get; }
```

Property Value

[string](#)

Provides a container for log items.

## Participant

Get the participant.

```
public Participant Participant { get; }
```

Property Value

[Participant](#)

Provides a container for log items.

# Class Paragraph

Namespace: [BP.AdventureFramework.Conversations](#)

Assembly: BP.AdventureFramework.dll

Represents a paragraph in a Conversation.

```
public sealed class Paragraph
```

## Inheritance

[object](#) ← Paragraph

## Inherited Members

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

## Constructors

### Paragraph(string)

Initializes a new instance of the Paragraph class.

```
public Paragraph(string line)
```

#### Parameters

line [string](#)

Specify the line.

### Paragraph(string, ConversationActionCallback, int)

Initializes a new instance of the Paragraph class.

```
public Paragraph(string line, ConversationActionCallback action, int delta = 1)
```

## Parameters

**line** [string](#)

Specify the line.

**action** [ConversationActionCallback](#)

Specify any action to be carried out with this line.

**delta** [int](#)

Specify the delta. This can be applied to a conversation to direct the conversation after this paragraph.

## Paragraph(string, int)

Initializes a new instance of the Paragraph class.

```
public Paragraph(string line, int delta = 1)
```

## Parameters

**line** [string](#)

Specify the line.

**delta** [int](#)

Specify the delta. This can be applied to a conversation to direct the conversation after this paragraph.

## Properties

### Action

Get or set any action to carry out on this line.

```
public ConversationActionCallback Action { get; set; }
```

## Property Value

### [ConversationActionCallback](#)

Represents a paragraph in a Conversation.

## CanRespond

Get if a response is possible.

```
public bool CanRespond { get; }
```

## Property Value

### [bool](#)

Represents a paragraph in a Conversation.

## Delta

Get the delta. This can be applied to a conversation to direct the conversation after this paragraph.

```
public int Delta { get; }
```

## Property Value

### [int](#)

Represents a paragraph in a Conversation.

## Line

Get or set the line.

```
public string Line { get; set; }
```

Property Value

[string](#) ↗

Represents a paragraph in a Conversation.

## Responses

Get or set the responses, applicable to the last line.

```
public Response[] Responses { get; set; }
```

Property Value

[Response\[\]](#)

Represents a paragraph in a Conversation.

# Enum Participant

Namespace: [BPAdventureFramework.Conversations](#)

Assembly: BP.AdventureFramework.dll

Enumeration of participants in a conversation.

```
public enum Participant
```

## Fields

Other = 1

Any other participant.

Player = 0

The player.

# Class Response

Namespace: [BP.AdventureFramework.Conversations](#)

Assembly: BP.AdventureFramework.dll

Provides a response to a conversation.

```
public sealed class Response
```

## Inheritance

[object](#) ← Response

## Inherited Members

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

## Constructors

### Response(string, int)

Initializes a new instance of the Response class.

```
public Response(string line, int delta = 1)
```

#### Parameters

**line** [string](#)

The line to trigger this response.

**delta** [int](#)

Specify the delta. This can be applied to a conversation to direct the conversation after this paragraph.

## Properties

## Delta

Get the delta. This can be applied to a conversation to direct the conversation after this paragraph.

```
public int Delta { get; }
```

Property Value

[int](#)

Provides a response to a conversation.

## Line

Get the line.

```
public string Line { get; }
```

Property Value

[string](#)

Provides a response to a conversation.

# Namespace BP.AdventureFramework.Extensions

## Classes

### [DirectionExtensions](#)

Provides extension versions for Directions.

### [StringExtensions](#)

Provides extension methods for strings.

# Class DirectionExtensions

Namespace: [BP.AdventureFramework.Extensions](#)

Assembly: BP.AdventureFramework.dll

Provides extension versions for Directions.

```
public static class DirectionExtensions
```

## Inheritance

[object](#) ← DirectionExtensions

## Inherited Members

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

## Methods

### Inverse(Direction)

Get an inverse direction.

```
public static Direction Inverse(this Direction value)
```

#### Parameters

**value** [Direction](#)

The direction.

#### Returns

[Direction](#)

The inverse direction.

# Class StringExtensions

Namespace: [BP.AdventureFramework.Extensions](#)

Assembly: BP.AdventureFramework.dll

Provides extension methods for strings.

```
public static class StringExtensions
```

## Inheritance

[object](#) ← StringExtensions

## Inherited Members

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

## Methods

### CASEINSENSITIVECONTAINS(string, string)

Returns a value indicating whether a specified substring occurs within this string. This is not case sensitive.

```
public static bool CaseInsensitiveContains(this string value, string subString)
```

#### Parameters

**value** [string](#)

The value.

**subString** [string](#)

The string to seek.

#### Returns

[bool](#)

True if the value parameter occurs within this string, or if value is the empty string (""); otherwise, false.

## EnsureFinishedSentence(string)

Ensure this string is a finished sentence, ending in either ?, ! or .

```
public static string EnsureFinishedSentence(this string value)
```

### Parameters

**value** [string](#)

The string to finish.

### Returns

[string](#)

The finished string.

## EqualsExaminable(string, IExaminable)

Determine if this string equals an IExaminable.

```
public static bool EqualsExaminable(this string value, IExaminable examinable)
```

### Parameters

**value** [string](#)

The value.

**examinable** [IExaminable](#)

The examinable.

### Returns

## [bool](#)

True if this string equals the identifier, else false.

## EqualsIdentifier(string, Identifier)

Determine if this string equals an identifier.

```
public static bool EqualsIdentifier(this string value, Identifier identifier)
```

Parameters

### [value](#) [string](#)

The value.

### [identifier](#) [Identifier](#)

The identifier.

Returns

## [bool](#)

True if this string equals the identifier, else false.

## GetObjectifier(string)

Get an objectifier for a word.

```
public static string GetObjectifier(this string word)
```

Parameters

### [word](#) [string](#)

The word.

Returns

## [string](#)

The objectifier.

## IsPlural(string)

Get if a word is plural.

```
public static bool IsPlural(this string word)
```

Parameters

### [word](#) [string](#)

The word to check.

Returns

### [bool](#)

True if the word is plural.

## IsVowel(string)

Get if a character is a vowel.

```
public static bool IsVowel(this string value)
```

Parameters

### [value](#) [string](#)

The value to check.

Returns

### [bool](#)

True if the character is a vowel.

## LineCount(string)

Determine the number of lines in this string.

```
public static int LineCount(this string value)
```

Parameters

**value** [string](#)

The value.

Returns

[int](#)

The number of lines in the string.

## RemoveSentenceEnd(string)

Ensure this string is not a finished sentence, ending in either ?, ! or .

```
public static string RemoveSentenceEnd(this string value)
```

Parameters

**value** [string](#)

The string to ensure isn't finished finish.

Returns

[string](#)

The unfinished string.

## ToDescription(string)

Returns this string as a Description.

```
public static Description ToDescription(this string value)
```

## Parameters

**value** [string](#)

The value.

## Returns

[Description](#)

This string as a description.

## ToIdentifier(string)

Returns this string as an Identifier.

```
public static Identifier ToIdentifier(this string value)
```

## Parameters

**value** [string](#)

The value.

## Returns

[Identifier](#)

This string as an identifier.

## ToSentenceCase(string)

Convert a string to sentence case.

```
public static string ToSentenceCase(this string value)
```

## Parameters

**value** [string](#)

The value.

## Returns

[string](#)

The word in sentence case.

## ToSpeech(string)

Convert a string to speech.

```
public static string ToSpeech(this string value)
```

## Parameters

**value** [string](#)

The value.

## Returns

[string](#)

The value in sentence case.

# Namespace BP.AdventureFramework. Interpretation

## Classes

### [CommandHelp](#)

Provides help for a command.

### [CustomCommandInterpreter](#)

Provides an object that can be used for interpreting custom commands.

### [InterpretationResult](#)

Represents the result of an interpretation.

## Interfaces

### [IInterpreter](#)

Represents any object that can act as an interpreter for input.

# Class CommandHelp

Namespace: [BP.AdventureFramework.Interpretation](#)

Assembly: BP.AdventureFramework.dll

Provides help for a command.

```
public class CommandHelp : IEquatable<CommandHelp>
```

## Inheritance

[object](#) ← CommandHelp

## Implements

[IEquatable](#)<[CommandHelp](#)>

## Inherited Members

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

## Constructors

### CommandHelp(string, string)

Initializes a new instance of the CommandHelp class.

```
public CommandHelp(string command, string description)
```

## Parameters

**command** [string](#)

The command.

**description** [string](#)

The help.

# Properties

## Command

Get the command.

```
public string Command { get; }
```

### Property Value

[string](#)

Provides help for a command.

## Description

Get the description of the command.

```
public string Description { get; }
```

### Property Value

[string](#)

Provides help for a command.

## Methods

### Equals(CommandHelp)

Indicates whether the current object is equal to another object of the same type.

```
public bool Equals(CommandHelp other)
```

### Parameters

other [CommandHelp](#)

An object to compare with this object.

Returns

[bool](#)

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

# Class CustomCommandInterpreter

Namespace: [BP.AdventureFramework.Interpretation](#)

Assembly: BP.AdventureFramework.dll

Provides an object that can be used for interpreting custom commands.

```
public class CustomCommandInterpreter : IInterpreter
```

## Inheritance

[object](#) ← CustomCommandInterpreter

## Implements

[IInterpreter](#)

## Inherited Members

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

## Properties

### SupportedCommands

Get an array of all supported commands.

```
public CommandHelp[] SupportedCommands { get; }
```

## Property Value

[CommandHelp\[\]](#)

Provides an object that can be used for interpreting custom commands.

## Methods

### GetContextualCommandHelp(Game)

Get contextual command help for a game, based on its current state.

```
public CommandHelp[] GetContextualCommandHelp(Game game)
```

Parameters

game [Game](#)

The game.

Returns

[CommandHelp\[\]](#)

The contextual help.

## Interpret(string, Game)

Interpret a string.

```
public InterpretationResult Interpret(string input, Game game)
```

Parameters

input [string](#)

The string to interpret.

game [Game](#)

The game.

Returns

[InterpretationResult](#)

The result of the interpretation.

# Interface IInterpreter

Namespace: [BPAdventureFramework.Interpretation](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can act as an interpreter for input.

```
public interface IInterpreter
```

## Properties

### SupportedCommands

Get an array of all supported commands.

```
CommandHelp[] SupportedCommands { get; }
```

Property Value

#### [CommandHelp\[\]](#)

Represents any object that can act as an interpreter for input.

## Methods

### GetContextualCommandHelp(Game)

Get contextual command help for a game, based on its current state.

```
CommandHelp[] GetContextualCommandHelp(Game game)
```

Parameters

#### game [Game](#)

The game.

Returns

[CommandHelp\[\]](#)

The contextual help.

## Interpret(string, Game)

Interpret a string.

```
InterpretationResult Interpret(string input, Game game)
```

Parameters

[input string](#)

The string to interpret.

[game Game](#)

The game.

Returns

[InterpretationResult](#)

The result of the interpretation.

# Class InterpretationResult

Namespace: [BP.AdventureFramework.Interpretation](#)

Assembly: BP.AdventureFramework.dll

Represents the result of an interpretation.

```
public class InterpretationResult
```

## Inheritance

[object](#) ← InterpretationResult

## Inherited Members

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

## Constructors

### InterpretationResult(bool, ICommand)

Initializes a new instance of the InterpretationResult class.

```
public InterpretationResult(bool wasInterpretedSuccessfully, ICommand command)
```

#### Parameters

`wasInterpretedSuccessfully` [bool](#)

If interpretation was successful.

`command` [ICommand](#)

The command.

## Properties

# Command

Get the command.

```
public ICommand Command { get; }
```

Property Value

## [ICommand](#)

Represents the result of an interpretation.

# Fail

Get a default result for failure.

```
public static InterpretationResult Fail { get; }
```

Property Value

## [InterpretationResult](#)

Represents the result of an interpretation.

# WasInterpretedSuccessfully

Get if interpretation was successful.

```
public bool WasInterpretedSuccessfully { get; }
```

Property Value

## [bool](#)

Represents the result of an interpretation.

# Namespace BP.AdventureFramework.Logic

## Classes

### [EndCheckResult](#)

Represents the result of an end check.

### [Game](#)

Represents the structure of the game

## Enums

### [ExitMode](#)

Enumeration of exit modes.

### [GameState](#)

Enumeration of game states.

## Delegates

### [EndCheck](#)

Represents the callback used for end checks.

### [GameCreationCallback](#)

Represents the callback used for Game creation.

### [OverworldCreationCallback](#)

Represents a callback for Overworld creation.

### [PlayerCreationCallback](#)

Represents a callback for Player creation.

# Delegate EndCheck

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Represents the callback used for end checks.

```
public delegate EndCheckResult EndCheck(Game game)
```

Parameters

game [Game](#)

The game to check for end.

Returns

[EndCheckResult](#)

Returns a result from the check.

# Class EndCheckResult

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Represents the result of an end check.

```
public class EndCheckResult
```

## Inheritance

[object](#) ← EndCheckResult

## Inherited Members

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

## Constructors

### EndCheckResult(bool, string, string)

Initializes a new instance of the EndCheckResult class.

```
public EndCheckResult(bool isCompleted, string title, string description)
```

## Parameters

**isCompleted** [bool](#)

If the game has ended.

**title** [string](#)

A title to describe the end.

**description** [string](#)

A description of the end.

# Properties

## Description

Get a description of the end.

```
public string Description { get; }
```

### Property Value

[string](#)

Represents the result of an end check.

## HasEnded

Get if the game has come to an end.

```
public bool HasEnded { get; }
```

### Property Value

[bool](#)

Represents the result of an end check.

## NotEnded

Get a default result for not ended.

```
public static EndCheckResult NotEnded { get; }
```

### Property Value

[EndCheckResult](#)

Represents the result of an end check.

## Title

Get a title to describe the end.

```
public string Title { get; }
```

Property Value

[string](#)

Represents the result of an end check.

# Enum ExitMode

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Enumeration of exit modes.

```
public enum ExitMode
```

## Fields

**ExitApplication** = 0

Exit the application.

**ReturnToTitleScreen** = 1

Return to the title screen.

# Class Game

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Represents the structure of the game

```
public sealed class Game
```

## Inheritance

[object](#) ← Game

## Inherited Members

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

## Fields

### DefaultErrorPrefix

Get the default error prefix.

```
public const string DefaultErrorPrefix = "Oops"
```

Field Value

[string](#)

Represents the structure of the game

## Properties

### ActiveConverser

Get the active converser.

```
public IConverser ActiveConverser { get; }
```

## Property Value

### [IConverser](#)

Represents the structure of the game

## Author

Get or set the name of the author.

```
public string Author { get; set; }
```

## Property Value

### [string](#)

Represents the structure of the game

## DefaultInterpreter

Get the default interpreter.

```
public static IInterpreter DefaultInterpreter { get; }
```

## Property Value

### [IInterpreter](#)

Represents the structure of the game

## DefaultSize

Get the default size.

```
public static Size DefaultSize { get; }
```

## Property Value

### Size

Represents the structure of the game

## Description

Get the description.

```
public string Description { get; }
```

## Property Value

### string ↗

Represents the structure of the game

## DisplayCommandListInSceneFrames

Get or set if the command list is displayed in scene frames.

```
public bool DisplayCommandListInSceneFrames { get; set; }
```

## Property Value

### bool ↗

Represents the structure of the game

## DisplaySize

Get the size of the display area.

```
public Size DisplaySize { get; }
```

## Property Value

### [Size](#)

Represents the structure of the game

## ErrorPrefix

Get or set the error prefix.

```
public string ErrorPrefix { get; set; }
```

## Property Value

### [string](#)

Represents the structure of the game

## FrameBuilders

Get or set the collection of frame builders used to render this game.

```
public FrameBuilderCollection FrameBuilders { get; set; }
```

## Property Value

### [FrameBuilderCollection](#)

Represents the structure of the game

## Introduction

Get the introduction.

```
public string Introduction { get; }
```

Property Value

[string](#)

Represents the structure of the game

## IsExecuting

Get if this is executing.

```
public bool IsExecuting { get; }
```

Property Value

[bool](#)

Represents the structure of the game

## Name

Get the name.

```
public string Name { get; }
```

Property Value

[string](#)

Represents the structure of the game

## Overworld

Get the overworld.

```
public Overworld Overworld { get; }
```

Property Value

### [Overworld](#)

Represents the structure of the game

## Player

Get the player.

```
public PlayableCharacter Player { get; }
```

Property Value

### [PlayableCharacter](#)

Represents the structure of the game

## SceneMapKeyType

Get or set the type of key to use on the scene map.

```
public KeyType SceneMapKeyType { get; set; }
```

Property Value

### [KeyType](#)

Represents the structure of the game

## Methods

Create(string, string, string, OverworldCreationCallback,  
PlayerCreationCallback, EndCheck, EndCheck)

Create a new callback for generating instances of a game.

```
public static GameCreationCallback Create(string name, string introduction, string description, OverworldCreationCallback overworldGenerator, PlayerCreationCallback playerGenerator, EndCheck completionCondition, EndCheck gameOverCondition)
```

## Parameters

**name** [string](#)

The name of the game.

**introduction** [string](#)

An introduction to the game.

**description** [string](#)

A description of the game.

**overworldGenerator** [OverworldCreationCallback](#)

A function to generate the overworld with.

**playerGenerator** [PlayerCreationCallback](#)

The function to generate the player with.

**completionCondition** [EndCheck](#)

The callback used to check game completion.

**gameOverCondition** [EndCheck](#)

The callback used to check game over.

## Returns

[GameCreationCallback](#)

A new GameCreationHelper that will create a GameCreator with the parameters specified.

`Create(string name, string introduction, string description, OverworldCreationCallback overworldGenerator, PlayerCreationCallback playerGenerator, EndCheck completionCondition, EndCheck gameOverCondition, Size displaySize, FrameBuilderCollection frameBuilders, ExitMode exitMode, string errorPrefix, IInterpreter interpreter)`

Create a new callback for generating instances of a game.

```
public static GameCreationCallback Create(string name, string introduction, string description, OverworldCreationCallback overworldGenerator, PlayerCreationCallback playerGenerator, EndCheck completionCondition, EndCheck gameOverCondition, Size displaySize, FrameBuilderCollection frameBuilders, ExitMode exitMode, string errorPrefix, IInterpreter interpreter)
```

## Parameters

`name` [string](#)

The name of the game.

`introduction` [string](#)

An introduction to the game.

`description` [string](#)

A description of the game.

`overworldGenerator` [OverworldCreationCallback](#)

A function to generate the overworld with.

`playerGenerator` [PlayerCreationCallback](#)

The function to generate the player with.

`completionCondition` [EndCheck](#)

The callback used to check game completion.

`gameOverCondition` [EndCheck](#)

The callback used to check game over.

`displaySize` [Size](#)

The display size.

`frameBuilders` [FrameBuilderCollection](#)

The collection of frame builders to use to render the game.

`exitMode` [ExitMode](#)

The exit mode.

`errorPrefix` [string](#) ↴

A prefix to use when displaying errors.

`interpreter` [IInterpreter](#)

The interpreter.

Returns

[GameCreationCallback](#)

A new GameCreationHelper that will create a GameCreator with the parameters specified.

## DisplayAbout()

Display the about frame.

```
public void DisplayAbout()
```

## DisplayHelp()

Display the help frame.

```
public void DisplayHelp()
```

## DisplayMap()

Display the map frame.

```
public void DisplayMap()
```

## DisplayTransition(string, string)

Display a transition frame.

```
public void DisplayTransition(string title, string message)
```

### Parameters

**title** [string](#)

The title.

**message** [string](#)

The message.

## Execute(GameCreationCallback)

Execute a game.

```
public static void Execute(GameCreationCallback creator)
```

### Parameters

**creator** [GameCreationCallback](#)

The creator to use to create the game.

## FindInteractionTarget(string)

Find an interaction target within the current scope for this Game.

```
public IInteractWithItem FindInteractionTarget(string name)
```

## Parameters

`name` [string](#)

The targets name.

## Returns

[IInteractWithItem](#)

The first IInteractWithItem object which has a name that matches the name parameter.

## GetAllPlayerVisibleExaminables()

Get all examinables that are currently visible to the player.

`public IExaminable[] GetAllPlayerVisibleExaminables()`

## Returns

[IExaminable\[\]](#)

An array of all examinables that are currently visible to the player.

# Delegate GameCreationCallback

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Represents the callback used for Game creation.

```
public delegate Game GameCreationCallback()
```

Returns

[Game](#)

A game created by the callback.

# Enum GameState

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Enumeration of game states.

```
public enum GameState
```

## Fields

**Active = 1**

Active.

**Finished = 2**

Finished.

**NotStarted = 0**

Not started.

# Delegate OverworldCreationCallback

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Represents a callback for Overworld creation.

```
public delegate Overworld OverworldCreationCallback(PlayableCharacter pC)
```

Parameters

pC [PlayableCharacter](#)

The playable character that will appear in the Overworld.

Returns

[Overworld](#)

A generated Overworld.

# Delegate PlayerCreationCallback

Namespace: [BPAdventureFramework.Logic](#)

Assembly: BP.AdventureFramework.dll

Represents a callback for Player creation.

```
public delegate PlayableCharacter PlayerCreationCallback()
```

Returns

[PlayableCharacter](#)

A generated Player.

# Namespace BP.AdventureFramework.Rendering

## Enums

### [KeyType](#)

Enumeration of key types.

### [RegionMapMode](#)

Enumeration of region map modes.

# Enum KeyType

Namespace: [BPAdventureFramework.Rendering](#)

Assembly: BP.AdventureFramework.dll

Enumeration of key types.

```
public enum KeyType
```

## Fields

**Dynamic** = 2

Dynamic key, only show relevant key items.

**Full** = 1

Full key.

**None** = 0

No key.

# Enum RegionMapMode

Namespace: [BPAdventureFramework.Rendering](#)

Assembly: BP.AdventureFramework.dll

Enumeration of region map modes.

```
public enum RegionMapMode
```

## Fields

**Detailed = 0**

Shows rooms at a detailed level.

**Dynamic = 2**

Dynamic region map - uses detailed if there is room, else map will be undetailed.

**Undetailed = 1**

Shows rooms as one character, which allows larger maps to be displayed in a limited area.

# Namespace BP.AdventureFramework.Rendering.FrameBuilders

## Classes

### [FrameBuilderCollection](#)

Provides a collection of all of the frame builders required to run a game.

### [FrameBuilderCollections](#)

Provides a container from frame builder collections.

### [GridStringBuilder](#)

Provides a class for building strings as part of a grid.

## Interfaces

### [IAboutFrameBuilder](#)

Represents any object that can build about frames.

### [ICompletionFrameBuilder](#)

Represents any object that can build completion frames.

### [IConversationFrameBuilder](#)

Represents any object that can build conversation frames.

### [IGameOverFrameBuilder](#)

Represents any object that can build game over frames.

### [IHelpFrameBuilder](#)

Represents any object that can build help frames.

### [IRegionMapBuilder](#)

Represents any object that can build region maps.

### [IRegionMapFrameBuilder](#)

Represents any object that can build region map frames.

### [IRoomMapBuilder](#)

Represents any object that can build room maps.

### [ISceneFrameBuilder](#)

Represents any object that can build scene frames.

### [ITitleFrameBuilder](#)

Represents any object that can build title frames.

## [ITransitionFrameBuilder](#)

Represents any object that can build transition frames.

# Class FrameBuilderCollection

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Provides a collection of all of the frame builders required to run a game.

```
public class FrameBuilderCollection
```

## Inheritance

[object](#) ← FrameBuilderCollection

## Inherited Members

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

## Constructors

FrameBuilderCollection(ITitleFrameBuilder,  
ISceneFrameBuilder, IRegionMapFrameBuilder,  
IHelpFrameBuilder, ICompletionFrameBuilder,  
IGameOverFrameBuilder, IAboutFrameBuilder,  
ITransitionFrameBuilder, IConversationFrameBuilder)

Initializes a new instance of the FrameBuilderCollection class.

```
public FrameBuilderCollection(ITitleFrameBuilder titleFrameBuilder,  
ISceneFrameBuilder sceneFrameBuilder, IRegionMapFrameBuilder regionMapFrameBuilder,  
IHelpFrameBuilder helpFrameBuilder, ICompletionFrameBuilder completionFrameBuilder,  
IGameOverFrameBuilder gameOverFrameBuilder, IAboutFrameBuilder aboutFrameBuilder,  
ITransitionFrameBuilder transitionFrameBuilder, IConversationFrameBuilder  
conversationFrameBuilder)
```

## Parameters

titleFrameBuilder [ITitleFrameBuilder](#)

The builder to use for building title frames.

`sceneFrameBuilder` [ISceneFrameBuilder](#)

The builder to use for building scene frames.

`regionMapFrameBuilder` [IRegionMapFrameBuilder](#)

The builder to use for building region map frames.

`helpFrameBuilder` [IHelpFrameBuilder](#)

The builder to use for building help frames.

`completionFrameBuilder` [ICompletionFrameBuilder](#)

The builder to use for building completion frames.

`gameOverFrameBuilder` [IGameOverFrameBuilder](#)

The builder to use for building game over frames.

`aboutFrameBuilder` [IAboutFrameBuilder](#)

The builder to use for building about frames.

`transitionFrameBuilder` [ITransitionFrameBuilder](#)

The builder to use for building transition frames.

`conversationFrameBuilder` [IConversationFrameBuilder](#)

The builder to use for building conversation frames.

## Properties

### AboutFrameBuilder

Get the builder to use for about frames.

```
public IAboutFrameBuilder AboutFrameBuilder { get; }
```

Property Value

## [IAboutFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## CompletionFrameBuilder

Get the builder to use for completion frames.

```
public ICompletionFrameBuilder CompletionFrameBuilder { get; }
```

Property Value

### [ICompletionFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## ConversationFrameBuilder

Get the builder to use for conversation frames.

```
public IConversationFrameBuilder ConversationFrameBuilder { get; }
```

Property Value

### [IConversationFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## GameOverFrameBuilder

Get the builder to use for game over frames.

```
public IGameOverFrameBuilder GameOverFrameBuilder { get; }
```

Property Value

### [IGameOverFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## HelpFrameBuilder

Get the builder to use for help frames.

```
public IHelpFrameBuilder HelpFrameBuilder { get; }
```

Property Value

[IHelpFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## RegionMapFrameBuilder

Get the builder to use for region map frames.

```
public IRegionMapFrameBuilder RegionMapFrameBuilder { get; }
```

Property Value

[IRegionMapFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## SceneFrameBuilder

Get the builder to use for scene frames.

```
public ISceneFrameBuilder SceneFrameBuilder { get; }
```

Property Value

[ISceneFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## TitleFrameBuilder

Get the builder to use for title frames.

```
public ITitleFrameBuilder TitleFrameBuilder { get; }
```

Property Value

[ITitleFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

## TransitionFrameBuilder

Get the builder to use for transition frames.

```
public ITransitionFrameBuilder TransitionFrameBuilder { get; }
```

Property Value

[ITransitionFrameBuilder](#)

Provides a collection of all of the frame builders required to run a game.

# Class FrameBuilderCollections

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Provides a container from frame builder collections.

```
public static class FrameBuilderCollections
```

## Inheritance

[object](#) ← FrameBuilderCollections

## Inherited Members

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

## Properties

### Default

Get the default frame builder collection.

```
public static FrameBuilderCollection Default { get; }
```

## Property Value

[FrameBuilderCollection](#)

Provides a container from frame builder collections.

# Class GridStringBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Provides a class for building strings as part of a grid.

```
public class GridStringBuilder
```

## Inheritance

[object](#) ← GridStringBuilder

## Inherited Members

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

## Constructors

### GridStringBuilder(char, char, char)

Initializes a new instance of the GridStringBuilder class.

```
public GridStringBuilder(char leftBoundaryCharacter = '|', char  
rightBoundaryCharacter = '|', char horizontalDividerCharacter = '-')
```

## Parameters

**leftBoundaryCharacter** [char](#)

The character to use for left boundaries.

**rightBoundaryCharacter** [char](#)

The character to use for right boundaries.

**horizontalDividerCharacter** [char](#)

The character to use for horizontal dividers.

# Properties

## DisplaySize

Get the display size.

```
public Size DisplaySize { get; }
```

### Property Value

#### [Size](#)

Provides a class for building strings as part of a grid.

## HorizontalDividerCharacter

Get or set the character used for horizontal dividers.

```
public char HorizontalDividerCharacter { get; set; }
```

### Property Value

#### [char](#)

Provides a class for building strings as part of a grid.

## LeftBoundaryCharacter

Get or set the character used for left boundary.

```
public char LeftBoundaryCharacter { get; set; }
```

### Property Value

#### [char](#)

Provides a class for building strings as part of a grid.

# LineTerminator

Get or set the line terminator.

```
public string LineTerminator { get; set; }
```

Property Value

[string](#)

Provides a class for building strings as part of a grid.

# RightBoundaryCharacter

Get or set the character used for right boundary.

```
public char RightBoundaryCharacter { get; set; }
```

Property Value

[char](#)

Provides a class for building strings as part of a grid.

# Methods

## DrawBoundary(AnsiColor)

Draw the boundary.

```
public void DrawBoundary(AnsiColor color)
```

Parameters

color [AnsiColor](#)

The color to draw the boundary.

## DrawCentralisedWrapped(string, int, int, AnsiColor, out int, out int)

Draw a wrapped string.

```
public void DrawCentralisedWrapped(string value, int startY, int maxWidth, AnsiColor color, out int endX, out int endY)
```

Parameters

**value** [string](#)

The string.

**startY** [int](#)

The start y position.

**maxWidth** [int](#)

The max width of the string.

**color** [AnsiColor](#)

The color to draw the text.

**endX** [int](#)

The end x position.

**endY** [int](#)

The end y position.

## DrawHorizontalDivider(int, AnsiColor)

Draw a horizontal divider.

```
public void DrawHorizontalDivider(int y, AnsiColor color)
```

Parameters

y [int](#)

The y position of the divider.

color [AnsiColor](#)

The color to draw the boundary.

## DrawUnderline(int, int, int, AnsiColor)

Draw an underline.

```
public void DrawUnderline(int x, int y, int length, AnsiColor color)
```

### Parameters

x [int](#)

The position of the underline, in x.

y [int](#)

The position of the underline, in y.

length [int](#)

The length of the underline.

color [AnsiColor](#)

The color of the underline.

## DrawWrapped(string, int, int, int, AnsiColor, out int, out int)

Draw a wrapped string.

```
public void DrawWrapped(string value, int startX, int startY, int maxWidth,
AnsiColor color, out int endX, out int endY)
```

## Parameters

`value string`

The string.

`startX int`

The start x position.

`startY int`

The start y position.

`maxWidth int`

The max width of the string.

`color AnsiColor`

The color to draw the text.

`endX int`

The end x position.

`endY int`

The end y position.

## Flush()

Flush the buffer.

```
public void Flush()
```

## GetCellColor(int, int)

Get a color for a cell.

```
public AnsiColor GetCellColor(int x, int y)
```

## Parameters

x [int](#)

The x position of the cell.

y [int](#)

The y position of the cell.

## Returns

[AnsiColor](#)

The cell color.

## GetCharacter(int, int)

Get a character from the buffer.

```
public char GetCharacter(int x, int y)
```

## Parameters

x [int](#)

The x position of the character.

y [int](#)

The y position of the character.

## Returns

[char](#)

The character.

## GetNumberOfLines(string, int, int, int)

Get the number of lines a string will take up.

```
public int GetNumberOfLines(string value, int startX, int startY, int maxWidth)
```

## Parameters

**value** [string](#)

The string.

**startX** [int](#)

The start x position.

**startY** [int](#)

The start y position.

**maxWidth** [int](#)

The max width of the string.

## Returns

[int](#)

The number of lines the string will take up.

## Resize(Size)

Resize this builder.

```
public void Resize(Size displaySize)
```

## Parameters

**displaySize** [Size](#)

The new size.

## SetCell(int, int, char, AnsiColor)

Set a cell.

```
public void SetCell(int x, int y, char character, AnsiColor color)
```

## Parameters

x [int](#)

The x position of the cell.

y [int](#)

The y position of the cell.

character [char](#)

The character.

color [AnsiColor](#)

The color of the character.

# Interface IAboutFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build about frames.

```
public interface IAboutFrameBuilder
```

## Methods

### Build(string, Game, int, int)

Build a frame.

```
IFrame Build(string title, Game game, int width, int height)
```

#### Parameters

**title** [string](#)

The title.

**game** [Game](#)

The game.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

#### Returns

[IFrame](#)

Represents any object that can build about frames.



# Interface ICompletionFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build completion frames.

```
public interface ICompletionFrameBuilder
```

## Methods

### Build(string, string, int, int)

Build a frame.

```
IFrame Build(string message, string reason, int width, int height)
```

#### Parameters

`message` [string](#)

The message to display to the user.

`reason` [string](#)

The reason the game ended.

`width` [int](#)

The width of the frame.

`height` [int](#)

The height of the frame.

#### Returns

[IFrame](#)

Represents any object that can build completion frames.



# Interface IConversationFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build conversation frames.

```
public interface IConversationFrameBuilder
```

## Methods

### Build(string, IConverser, CommandHelp[], int, int)

Build a frame.

```
IFrame Build(string title, IConverser converser, CommandHelp[] contextualCommands,  
int width, int height)
```

## Parameters

**title** [string](#)

The title to display to the user.

**converser** [IConverser](#)

The converser.

**contextualCommands** [CommandHelp\[\]](#)

The contextual commands to display.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

## Returns

### [IFrame](#)

Represents any object that can build conversation frames.

# Interface IGameOverFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build game over frames.

```
public interface IGameOverFrameBuilder
```

## Methods

### Build(string, string, int, int)

Build a frame.

```
IFrame Build(string message, string reason, int width, int height)
```

#### Parameters

**message** [string](#)

The message to display to the user.

**reason** [string](#)

The reason the game ended.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

#### Returns

[IFrame](#)

Represents any object that can build game over frames.



# Interface IHelpFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build help frames.

```
public interface IHelpFrameBuilder
```

## Methods

### Build(string, string, CommandHelp[], int, int)

Build a frame.

```
IFrame Build(string title, string description, CommandHelp[] commandHelp, int width, int height)
```

#### Parameters

**title** [string](#)

The title.

**description** [string](#)

The description.

**commandHelp** [CommandHelp\[\]](#)

The command help.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

## Returns

### [IFrame](#)

Represents any object that can build help frames.

# Interface IRegionMapBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build region maps.

```
public interface IRegionMapBuilder
```

## Methods

**BuildRegionMap(GridStringBuilder, Region, int, int, int, int)**

Build a map of a region.

```
void BuildRegionMap(GridStringBuilder gridStringBuilder, Region region, int x, int y, int maxWidth, int maxHeight)
```

### Parameters

**gridStringBuilder** [GridStringBuilder](#)

The string builder to use.

**region** [Region](#)

The region.

**x** [int](#)

The x position to start building at.

**y** [int](#)

The y position to start building at.

**maxWidth** [int](#)

The maximum horizontal space available in which to build the map.

`maxHeight` [int](#)

The maximum vertical space available in which to build the map.

# Interface IRegionMapFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build region map frames.

```
public interface IRegionMapFrameBuilder
```

## Methods

### Build(Region, int, int)

Build a frame.

```
IFrame Build(Region region, int width, int height)
```

#### Parameters

**region** [Region](#)

The region.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

#### Returns

[IFrame](#)

Represents any object that can build region map frames.

# Interface IRoomMapBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build room maps.

```
public interface IRoomMapBuilder
```

## Methods

**BuildRoomMap(GridStringBuilder, Room, ViewPoint, KeyType, int, int, out int, out int)**

Build a map for a room.

```
void BuildRoomMap(GridStringBuilder gridStringBuilder, Room room, ViewPoint  
viewPoint, KeyType key, int startX, int startY, out int endX, out int endY)
```

### Parameters

**gridStringBuilder** [GridStringBuilder](#)

The string builder to use.

**room** [Room](#)

The room.

**viewPoint** [ViewPoint](#)

The viewpoint from the room.

**key** [KeyType](#)

The key type.

**startX** [int](#)

The start position, x.

**startY** [int](#)

The start position, x.

**endX** [int](#)

The end position, x.

**endY** [int](#)

The end position, x.

# Interface ISceneFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build scene frames.

```
public interface ISceneFrameBuilder
```

## Methods

**Build(Room, ViewPoint, PlayableCharacter, string, CommandHelp[], KeyType, int, int)**

Build a frame.

```
IFrame Build(Room room, ViewPoint viewPoint, PlayableCharacter player, string message, CommandHelp[] contextualCommands, KeyType keyType, int width, int height)
```

## Parameters

**room** [Room](#)

Specify the Room.

**viewPoint** [ViewPoint](#)

Specify the viewpoint from the room.

**player** [PlayableCharacter](#)

Specify the player.

**message** [string](#) ↴

Any additional message.

**contextualCommands** [CommandHelp\[\]](#)

The contextual commands to display.

**keyType** [KeyType](#)

The type of key to use.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

Returns

[IFrame](#)

Represents any object that can build scene frames.

# Interface ITitleFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build title frames.

```
public interface ITitleFrameBuilder
```

## Methods

### Build(string, string, int, int)

Build a frame.

```
IFrame Build(string title, string description, int width, int height)
```

#### Parameters

**title** [string](#)

The title.

**description** [string](#)

The description.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

#### Returns

[IFrame](#)

Represents any object that can build title frames.



# Interface ITransitionFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can build transition frames.

```
public interface ITransitionFrameBuilder
```

## Methods

### Build(string, string, int, int)

Build a frame.

```
IFrame Build(string title, string message, int width, int height)
```

#### Parameters

**title** [string](#)

The title to display to the user.

**message** [string](#)

The message to display to the user.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

#### Returns

[IFrame](#)

Represents any object that can build transition frames.



# Namespace BP.AdventureFramework.Rendering.FrameBuilders.Color Classes

## [ColorAboutFrameBuilder](#)

Provides a builder of color about frames.

## [ColorCompletionFrameBuilder](#)

Provides a builder of color completion frames.

## [ColorConversationFrameBuilder](#)

Provides a builder of color conversation frames.

## [ColorGameOverFrameBuilder](#)

Provides a builder of color game over frames.

## [ColorHelpFrameBuilder](#)

Provides a builder of color help frames.

## [ColorRegionMapBuilder](#)

Provides a color builder for region maps.

## [ColorRegionMapFrameBuilder](#)

Provides a builder of color region map frames.

## [ColorRoomMapBuilder](#)

Provides a color room map builder.

## [ColorSceneFrameBuilder](#)

Provides a builder for color scene frames.

## [ColorTitleFrameBuilder](#)

Provides a builder of color title frames.

## [ColorTransitionFrameBuilder](#)

Provides a builder of color transition frames.

## Enums

### [AnsiColor](#)

Enumeration of ANSI colors.

# Enum AnsiColor

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Enumeration of ANSI colors.

```
public enum AnsiColor
```

## Fields

**Black** = 30

Black (30).

**Blue** = 34

Blue (34).

**BrightBlack** = 90

Bright black (90).

**BrightBlue** = 94

Bright blue (94).

**BrightCyan** = 96

Bright cyan (96).

**BrightGreen** = 92

Bright green (92).

**BrightMagenta** = 95

Bright magenta (95).

**BrightRed** = 91

Bright red (91).

**BrightWhite** = 97

Bright white (97).

**BrightYellow** = 93

Bright yellow (93).

**Cyan** = 36

Cyan (36).

**Green** = 32

Green (32).

**Magenta** = 35

Magenta (35).

**Red** = 31

Red (31).

**Reset** = 0

Reset (0).

**White** = 37

White (37).

**Yellow** = 33

Yellow (33).

# Class ColorAboutFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color about frames.

```
public sealed class ColorAboutFrameBuilder : IAboutFrameBuilder
```

## Inheritance

[object](#) ← ColorAboutFrameBuilder

## Implements

[IAboutFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorAboutFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorAboutFrameBuilder class.

```
public ColorAboutFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### AuthorColor

Get or set the author color.

```
public AnsiColor AuthorColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color about frames.

## BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color about frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color about frames.

## DescriptionColor

Get or set the description color.

```
public AnsiColor DescriptionColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color about frames.

## NameColor

Get or set the name color.

```
public AnsiColor NameColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color about frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color about frames.

## Methods

### Build(string, Game, int, int)

Build a frame.

```
public IFrame Build(string title, Game game, int width, int height)
```

## Parameters

**title** [string](#)

The title.

**game** [Game](#)

The game.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

## Returns

[IFrame](#)

Provides a builder of color about frames.

# Class ColorCompletionFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color completion frames.

```
public sealed class ColorCompletionFrameBuilder : ICompletionFrameBuilder
```

## Inheritance

[object](#) ← ColorCompletionFrameBuilder

## Implements

[ICompletionFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorCompletionFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorCompletionFrameBuilder class.

```
public ColorCompletionFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color completion frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color completion frames.

## DescriptionColor

Get or set the description color.

```
public AnsiColor DescriptionColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color completion frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color completion frames.

## Methods

### Build(string, string, int, int)

Build a frame.

```
public IFrame Build(string message, string reason, int width, int height)
```

## Parameters

### `message` [string](#) ↗

The message to display to the user.

### `reason` [string](#) ↗

The reason the game ended.

### `width` [int](#) ↗

The width of the frame.

### `height` [int](#) ↗

The height of the frame.

## Returns

### [IFrame](#)

Provides a builder of color completion frames.

# Class ColorConversationFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color conversation frames.

```
public sealed class ColorConversationFrameBuilder : IConversationFrameBuilder
```

## Inheritance

[object](#) ← ColorConversationFrameBuilder

## Implements

[IConversationFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorConversationFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorConversationFrameBuilder class.

```
public ColorConversationFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color conversation frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color conversation frames.

## InputColor

Get or set the input color.

```
public AnsiColor InputColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color conversation frames.

## NonPlayerMessageColor

Get or set the player message color.

```
public AnsiColor NonPlayerMessageColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color conversation frames.

## PlayerMessageColor

Get or set the player message color.

```
public AnsiColor PlayerMessageColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color conversation frames.

## ResponseColor

Get or set the response color.

```
public AnsiColor ResponseColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color conversation frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color conversation frames.

## Methods

### Build(string, IConverser, CommandHelp[], int, int)

Build a frame.

```
public IFrame Build(string title, IConverser converser, CommandHelp[] contextualCommands, int width, int height)
```

## Parameters

### `title` [string](#)

The title to display to the user.

### `converser` [IConverser](#)

The converser.

### `contextualCommands` [CommandHelp\[\]](#)

The contextual commands to display.

### `width` [int](#)

The width of the frame.

### `height` [int](#)

The height of the frame.

## Returns

## IFrame

Provides a builder of color conversion frames.

# Class ColorGameOverFrameBuilder

Namespace: [BPAdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color game over frames.

```
public sealed class ColorGameOverFrameBuilder : IGameOverFrameBuilder
```

## Inheritance

[object](#) ← ColorGameOverFrameBuilder

## Implements

[IGameOverFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorGameOverFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorGameOverFrameBuilder class.

```
public ColorGameOverFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color game over frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color game over frames.

## DescriptionColor

Get or set the description color.

```
public AnsiColor DescriptionColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color game over frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color game over frames.

## Methods

### Build(string, string, int, int)

Build a frame.

```
public IFrame Build(string message, string reason, int width, int height)
```

## Parameters

### `message` [string](#) ↗

The message to display to the user.

### `reason` [string](#) ↗

The reason the game ended.

### `width` [int](#) ↗

The width of the frame.

### `height` [int](#) ↗

The height of the frame.

## Returns

### [IFrame](#)

Provides a builder of color game over frames.

# Class ColorHelpFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color help frames.

```
public sealed class ColorHelpFrameBuilder : IHelpFrameBuilder
```

## Inheritance

[object](#) ← ColorHelpFrameBuilder

## Implements

[IHelpFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorHelpFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorHelpFrameBuilder class.

```
public ColorHelpFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color help frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color help frames.

## CommandColor

Get or set the command color.

```
public AnsiColor CommandColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color help frames.

## CommandDescriptionColor

Get or set the description color.

```
public AnsiColor CommandDescriptionColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color help frames.

## DescriptionColor

Get or set the description color.

```
public AnsiColor DescriptionColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color help frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color help frames.

## Methods

### Build(string, string, CommandHelp[], int, int)

Build a frame.

```
public IFrame Build(string title, string description, CommandHelp[] commandHelp, int width, int height)
```

## Parameters

**title** [string](#)

The title.

**description** [string](#)

The description.

**commandHelp** [CommandHelp\[\]](#)

The command help.

**width** [int](#)

The width of the frame.

**height** [int](#)

The height of the frame.

## Returns

[IFrame](#)

Provides a builder of color help frames.

# Class ColorRegionMapBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a color builder for region maps.

```
public sealed class ColorRegionMapBuilder : IRegionMapBuilder
```

## Inheritance

[object](#) ← ColorRegionMapBuilder

## Implements

[IRegionMapBuilder](#)

## Inherited Members

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

## Properties

### CurrentFloorIndicator

Get or set the character to use for the current floor.

```
public char CurrentFloorIndicator { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

### EmptySpace

Get or set the character used for representing an empty space.

```
public char EmptySpace { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

## HorizontalBoundary

Get or set the character to use for horizontal boundaries.

```
public char HorizontalBoundary { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

## LockedExit

Get or set the character used for representing a locked exit.

```
public char LockedExit { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

## LockedExitColor

Get or set the locked exit color.

```
public AnsiColor LockedExitColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a color builder for region maps.

## LowerLevel

Get or set the character to use for lower levels.

```
public char LowerLevel { get; set; }
```

## Property Value

### [char](#)

Provides a color builder for region maps.

## LowerLevelColor

Get or set the lower level color.

```
public AnsiColor LowerLevelColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a color builder for region maps.

## Player

Get or set the character to use for indicating the player.

```
public char Player { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

## PlayerColor

Get or set the player color.

```
public AnsiColor PlayerColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a color builder for region maps.

## ShowLowerFloors

Get or set if lower floors should be shown.

```
public bool ShowLowerFloors { get; set; }
```

Property Value

[bool](#)

Provides a color builder for region maps.

## UnLockedExit

Get or set the character used for representing an unlocked exit.

```
public char UnLockedExit { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

## UnvisitedBoundaryColor

Get or set the unvisited room boundary color.

```
public AnsiColor UnvisitedBoundaryColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a color builder for region maps.

## VerticalBoundary

Get or set the character to use for vertical boundaries.

```
public char VerticalBoundary { get; set; }
```

Property Value

[char](#)

Provides a color builder for region maps.

## VisitedBoundaryColor

Get or set the visited room boundary color.

```
public AnsiColor VisitedBoundaryColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a color builder for region maps.

## Methods

### BuildRegionMap(GridStringBuilder, Region, int, int, int, int)

Build a map of a region.

```
public void BuildRegionMap(GridStringBuilder gridStringBuilder, Region region, int  
x, int y, int maxWidth, int maxHeight)
```

## Parameters

### `gridStringBuilder` [GridStringBuilder](#)

The string builder to use.

### `region` [Region](#)

The region.

### `x` [int](#)

The x position to start building at.

### `y` [int](#)

The y position to start building at.

### `maxWidth` [int](#)

The maximum horizontal space available in which to build the map.

### `maxHeight` [int](#)

The maximum vertical space available in which to build the map.

# Class ColorRegionMapFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color region map frames.

```
public sealed class ColorRegionMapFrameBuilder : IRegionMapFrameBuilder
```

## Inheritance

[object](#) ← ColorRegionMapFrameBuilder

## Implements

[IRegionMapFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorRegionMapFrameBuilder(GridStringBuilder, IRegionMapBuilder)

Initializes a new instance of the ColorRegionMapFrameBuilder class.

```
public ColorRegionMapFrameBuilder(GridStringBuilder gridStringBuilder,  
IRegionMapBuilder regionMapBuilder)
```

## Parameters

**gridStringBuilder** [GridStringBuilder](#)

A builder to use for the string layout.

**regionMapBuilder** [IRegionMapBuilder](#)

A builder for region maps.

# Properties

## BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

### Property Value

[AnsiColor](#)

Provides a builder of color region map frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

### Property Value

[AnsiColor](#)

Provides a builder of color region map frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

### Property Value

[AnsiColor](#)

Provides a builder of color region map frames.

# Methods

## Build(Region, int, int)

Build a frame.

```
public IFrame Build(Region region, int width, int height)
```

Parameters

`region` [Region](#)

The region.

`width` [int](#)

The width of the frame.

`height` [int](#)

The height of the frame.

Returns

[IFrame](#)

Provides a builder of color region map frames.

# Class ColorRoomMapBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a color room map builder.

```
public sealed class ColorRoomMapBuilder : IRoomMapBuilder
```

## Inheritance

[object](#) ← ColorRoomMapBuilder

## Implements

[IRoomMapBuilder](#)

## Inherited Members

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

## Properties

### BoundaryColor

Get or set the room boundary color.

```
public AnsiColor BoundaryColor { get; set; }
```

#### Property Value

[AnsiColor](#)

Provides a color room map builder.

### Corner

Get or set the character to use for corners.

```
public char Corner { get; set; }
```

Property Value

[char](#)

Provides a color room map builder.

## HorizontalBoundary

Get or set the character to use for horizontal boundaries.

```
public char HorizontalBoundary { get; set; }
```

Property Value

[char](#)

Provides a color room map builder.

## HorizontalExitBorder

Get or set the character to use for horizontal exit borders.

```
public char HorizontalExitBorder { get; set; }
```

Property Value

[char](#)

Provides a color room map builder.

## ItemOrCharacterColor

Get or set the item or character color.

```
public AnsiColor ItemOrCharacterColor { get; set; }
```

Property Value

#### [AnsiColor](#)

Provides a color room map builder.

## ItemOrCharacterInRoom

Get or set the character used for representing there is an item or a character in the room.

```
public char ItemOrCharacterInRoom { get; set; }
```

Property Value

#### [char](#)

Provides a color room map builder.

## KeyPadding

Get or set the padding between the key and the map.

```
public int KeyPadding { get; set; }
```

Property Value

#### [int](#)

Provides a color room map builder.

## LockedExit

Get or set the character used for representing a locked exit.

```
public char LockedExit { get; set; }
```

Property Value

[char](#)

Provides a color room map builder.

## LockedExitColor

Get or set the locked exit color.

```
public AnsiColor LockedExitColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a color room map builder.

## UnvisitedExitColor

Get or set the unvisited exit color.

```
public AnsiColor UnvisitedExitColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a color room map builder.

## VerticalBoundary

Get or set the character to use for vertical boundaries.

```
public char VerticalBoundary { get; set; }
```

Property Value

[char](#)

Provides a color room map builder.

## VerticalExitBorder

Get or set the character to use for vertical exit borders.

```
public char VerticalExitBorder { get; set; }
```

Property Value

[char](#)

Provides a color room map builder.

## VisitedExitColor

Get or set the visited exit color.

```
public AnsiColor VisitedExitColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a color room map builder.

## Methods

BuildRoomMap(GridStringBuilder, Room, ViewPoint, KeyType, int, int, out int, out int)

Build a map for a room.

```
public void BuildRoomMap(GridStringBuilder gridStringBuilder, Room room, ViewPoint  
viewPoint, KeyType key, int startX, int startY, out int endX, out int endY)
```

## Parameters

`gridStringBuilder` [GridStringBuilder](#)

The string builder to use.

`room` [Room](#)

The room.

`viewPoint` [ViewPoint](#)

The viewpoint from the room.

`key` [KeyType](#)

The key type.

`startX` [int](#)

The start position, x.

`startY` [int](#)

The start position, x.

`endX` [int](#)

The end position, x.

`endY` [int](#)

The end position, x.

# Class ColorSceneFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder for color scene frames.

```
public sealed class ColorSceneFrameBuilder : ISceneFrameBuilder
```

## Inheritance

[object](#) ← ColorSceneFrameBuilder

## Implements

[ISceneFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorSceneFrameBuilder(GridStringBuilder, IRoomMapBuilder)

Initializes a new instance of the ColorSceneFrameBuilder class.

```
public ColorSceneFrameBuilder(GridStringBuilder gridStringBuilder,  
IRoomMapBuilder roomMapBuilder)
```

## Parameters

**gridStringBuilder** [GridStringBuilder](#)

A builder to use for the string layout.

**roomMapBuilder** [IRoomMapBuilder](#)

A builder to use for room maps.

# Properties

## BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

### Property Value

[AnsiColor](#)

Provides a builder for color scene frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

### Property Value

[AnsiColor](#)

Provides a builder for color scene frames.

## CommandsColor

Get or set the commands color.

```
public AnsiColor CommandsColor { get; set; }
```

### Property Value

[AnsiColor](#)

Provides a builder for color scene frames.

## DisplayMessagesInIsolation

Get or set if messages should be displayed in isolation.

```
public bool DisplayMessagesInIsolation { get; set; }
```

Property Value

[bool](#)

Provides a builder for color scene frames.

## InputColor

Get or set the input color.

```
public AnsiColor InputColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder for color scene frames.

## SupressMovementMessages

Get or set if movement messages should be suppressed.

```
public bool SupressMovementMessages { get; set; }
```

Property Value

[bool](#)

Provides a builder for color scene frames.

## TextColor

Get or set the text color.

```
public AnsiColor TextColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder for color scene frames.

## Methods

**Build(Room, ViewPoint, PlayableCharacter, string, CommandHelp[], KeyType, int, int)**

Build a frame.

```
public IFrame Build(Room room, ViewPoint viewPoint, PlayableCharacter player, string message, CommandHelp[] contextualCommands, KeyType keyType, int width, int height)
```

Parameters

**room** [Room](#)

Specify the Room.

**viewPoint** [ViewPoint](#)

Specify the viewpoint from the room.

**player** [PlayableCharacter](#)

Specify the player.

**message** [string](#)

Any additional message.

**contextualCommands** [CommandHelp\[\]](#)

The contextual commands to display.

`keyType` [KeyType](#)

The type of key to use.

`width` [int](#)

The width of the frame.

`height` [int](#)

The height of the frame.

Returns

[IFrame](#)

Provides a builder for color scene frames.

# Class ColorTitleFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color title frames.

```
public sealed class ColorTitleFrameBuilder : ITitleFrameBuilder
```

## Inheritance

[object](#) ← ColorTitleFrameBuilder

## Implements

[ITitleFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorTitleFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorTitleFrameBuilder class.

```
public ColorTitleFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color title frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color title frames.

## DescriptionColor

Get or set the description color.

```
public AnsiColor DescriptionColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color title frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color title frames.

## Methods

### Build(string, string, int, int)

Build a frame.

```
public IFrame Build(string title, string description, int width, int height)
```

## Parameters

### [title](#) [string](#)

The title.

### [description](#) [string](#)

The description.

### [width](#) [int](#)

The width of the frame.

### [height](#) [int](#)

The height of the frame.

## Returns

### [IFrame](#)

Provides a builder of color title frames.

# Class ColorTransitionFrameBuilder

Namespace: [BP.AdventureFramework.Rendering.FrameBuilders.Color](#)

Assembly: BP.AdventureFramework.dll

Provides a builder of color transition frames.

```
public sealed class ColorTransitionFrameBuilder : ITransitionFrameBuilder
```

## Inheritance

[object](#) ← ColorTransitionFrameBuilder

## Implements

[ITransitionFrameBuilder](#)

## Inherited Members

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

## Constructors

### ColorTransitionFrameBuilder(GridStringBuilder)

Initializes a new instance of the ColorTransitionFrameBuilder class.

```
public ColorTransitionFrameBuilder(GridStringBuilder gridStringBuilder)
```

## Parameters

gridStringBuilder [GridStringBuilder](#)

A builder to use for the string layout.

## Properties

### BackgroundColor

Get or set the background color.

```
public AnsiColor BackgroundColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color transition frames.

## BorderColor

Get or set the border color.

```
public AnsiColor BorderColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color transition frames.

## MessageColor

Get or set the message color.

```
public AnsiColor MessageColor { get; set; }
```

Property Value

[AnsiColor](#)

Provides a builder of color transition frames.

## TitleColor

Get or set the title color.

```
public AnsiColor TitleColor { get; set; }
```

## Property Value

### [AnsiColor](#)

Provides a builder of color transition frames.

## Methods

### Build(string, string, int, int)

Build a frame.

```
public IFrame Build(string title, string message, int width, int height)
```

## Parameters

### `title` [string](#)

The title to display to the user.

### `message` [string](#)

The message to display to the user.

### `width` [int](#)

The width of the frame.

### `height` [int](#)

The height of the frame.

## Returns

### [IFrame](#)

Provides a builder of color transition frames.

# Namespace BP.AdventureFramework.Rendering.Frames

## Classes

### [GridTextFrame](#)

Provides a grid based frame for displaying a command based interface.

### [TextFrame](#)

Provides a simple text based frame for displaying a command based interface.

## Interfaces

### [IFrame](#)

Represents any object that is a frame that can display a command based interface.

# Class GridTextFrame

Namespace: [BP.AdventureFramework.Rendering.Frames](#)

Assembly: BP.AdventureFramework.dll

Provides a grid based frame for displaying a command based interface.

```
public sealed class GridTextFrame : IFrame
```

## Inheritance

[object](#) ← GridTextFrame

## Implements

[IFrame](#)

## Inherited Members

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

## Constructors

### GridTextFrame(GridStringBuilder, int, int, AnsiColor)

Initializes a new instance of the GridTextFrame class.

```
public GridTextFrame(GridStringBuilder builder, int cursorLeft, int cursorTop,  
AnsiColor backgroundColor)
```

## Parameters

**builder** [GridStringBuilder](#)

The builder that creates the frame.

**cursorLeft** [int](#)

The cursor left position.

**cursorTop** [int](#)

The cursor top position.

### **backgroundColor** [AnsiColor](#)

The background color.

## Properties

### AcceptsInput

Get or set if this Frame accepts input.

```
public bool AcceptsInput { get; set; }
```

Property Value

[bool](#) ↗

Provides a grid based frame for displaying a command based interface.

### BackgroundColor

Get the background color.

```
public AnsiColor BackgroundColor { get; }
```

Property Value

[AnsiColor](#)

Provides a grid based frame for displaying a command based interface.

### CursorLeft

Get the cursor left position.

```
public int CursorLeft { get; }
```

## Property Value

[int](#) ↗

Provides a grid based frame for displaying a command based interface.

## CursorTop

Get the cursor top position.

```
public int CursorTop { get; }
```

## Property Value

[int](#) ↗

Provides a grid based frame for displaying a command based interface.

## ShowCursor

Get or set if the cursor should be shown.

```
public bool ShowCursor { get; set; }
```

## Property Value

[bool](#) ↗

Provides a grid based frame for displaying a command based interface.

## Methods

### Render(TextWriter)

Render this frame on a writer.

```
public void Render(TextWriter writer)
```

## Parameters

writer [TextWriter](#)

The writer.

## ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

Returns

[string](#)

A string that represents the current object.

# Interface IFrame

Namespace: [BPAdventureFramework.Rendering.Frames](#)

Assembly: BP.AdventureFramework.dll

Represents any object that is a frame that can display a command based interface.

```
public interface IFrame
```

## Properties

### AcceptsInput

Get or set if this Frame accepts input.

```
bool AcceptsInput { get; set; }
```

Property Value

[bool](#) ↗

Represents any object that is a frame that can display a command based interface.

### CursorLeft

Get the cursor left position.

```
int CursorLeft { get; }
```

Property Value

[int](#) ↗

Represents any object that is a frame that can display a command based interface.

## CursorTop

Get the cursor top position.

```
int CursorTop { get; }
```

Property Value

[int](#)

Represents any object that is a frame that can display a command based interface.

## ShowCursor

Get or set if the cursor should be shown.

```
bool ShowCursor { get; set; }
```

Property Value

[bool](#)

Represents any object that is a frame that can display a command based interface.

## Methods

### Render(TextWriter)

Render this frame on a writer.

```
void Render(TextWriter writer)
```

Parameters

[writer](#) [TextWriter](#)

The writer.

# Class TextFrame

Namespace: [BP.AdventureFramework.Rendering.Frames](#)

Assembly: BP.AdventureFramework.dll

Provides a simple text based frame for displaying a command based interface.

```
public sealed class TextFrame : IFrame
```

## Inheritance

[object](#) ← TextFrame

## Implements

[IFrame](#)

## Inherited Members

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

## Constructors

### TextFrame(string, int, int)

Initializes a new instance of the TextFrame class.

```
public TextFrame(string frameData, int cursorLeft, int cursorTop)
```

## Parameters

**frameData** [string](#)

The data the frame provides.

**cursorLeft** [int](#)

The cursor left position.

**cursorTop** [int](#)

The cursor top position.

# Properties

## AcceptsInput

Get or set if this Frame accepts input.

```
public bool AcceptsInput { get; set; }
```

### Property Value

[bool](#)

Provides a simple text based frame for displaying a command based interface.

## CursorLeft

Get the cursor left position.

```
public int CursorLeft { get; }
```

### Property Value

[int](#)

Provides a simple text based frame for displaying a command based interface.

## CursorTop

Get the cursor top position.

```
public int CursorTop { get; }
```

### Property Value

[int](#)

Provides a simple text based frame for displaying a command based interface.

# ShowCursor

Get or set if the cursor should be shown.

```
public bool ShowCursor { get; set; }
```

Property Value

[bool](#)

Provides a simple text based frame for displaying a command based interface.

## Methods

### Render(TextWriter)

Render this frame on a writer.

```
public void Render(TextWriter writer)
```

Parameters

[writer](#) [TextWriter](#)

The writer.

### ToString()

Returns a string that represents the current object.

```
public override string ToString()
```

Returns

[string](#)

A string that represents the current object.



# Namespace BP.AdventureFramework.Utilities

## Classes

### [OverworldMaker](#)

Provides a class for helping to make Regions.

### [RegionMaker](#)

Provides a class for helping to make Regions.

## Interfaces

### [IAssetTemplate<T>](#)

Represents any object that is a template for an asset.

# Interface IAssetTemplate<T>

Namespace: [BPAdventureFramework.Utilities](#)

Assembly: BP.AdventureFramework.dll

Represents any object that is a template for an asset.

```
public interface IAssetTemplate<out T>
```

## Type Parameters

T

The type of asset being templated.

## Methods

### Instantiate()

Instantiate a new instance of the templated asset.

```
T Instantiate()
```

## Returns

T

The asset.

# Class OverworldMaker

Namespace: [BPAdventureFramework.Utilities](#)

Assembly: BP.AdventureFramework.dll

Provides a class for helping to make Regions.

```
public sealed class OverworldMaker
```

## Inheritance

[object](#) ← OverworldMaker

## Inherited Members

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

## Constructors

**OverworldMaker(Identifier, Description, params  
RegionMaker[])**

Initializes a new instance of the OverworldMaker class.

```
public OverworldMaker(Identifier identifier, Description description, params  
RegionMaker[] regionMakers)
```

### Parameters

**identifier** [Identifier](#)

An identifier for the region.

**description** [Description](#)

A description for the region.

**regionMakers** [RegionMaker\[\]](#)

The region makes to use to construct regions.

# OverworldMaker(string, string, params RegionMaker[])

Initializes a new instance of the OverworldMaker class.

```
public OverworldMaker(string identifier, string description, params  
RegionMaker[] regionMakers)
```

## Parameters

**identifier** [string](#)

An identifier for the region.

**description** [string](#)

A description for the region.

**regionMakers** [RegionMaker](#)[]

The region makes to use to construct regions.

## Methods

**Make()**

Make an overworld.

```
public Overworld Make()
```

## Returns

[Overworld](#)

The created overworld.

# Class RegionMaker

Namespace: [BP.AdventureFramework.Utilities](#)

Assembly: BP.AdventureFramework.dll

Provides a class for helping to make Regions.

```
public sealed class RegionMaker
```

## Inheritance

[object](#) ← RegionMaker

## Inherited Members

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

## Constructors

### RegionMaker(Identifier, Description)

Initializes a new instance of the RegionMaker class.

```
public RegionMaker(Identifier identifier, Description description)
```

#### Parameters

**identifier** [Identifier](#)

An identifier for the region.

**description** [Description](#)

A description for the region.

### RegionMaker(string, string)

Initializes a new instance of the RegionMaker class.

```
public RegionMaker(string identifier, string description)
```

## Parameters

**identifier** [string](#)

An identifier for the region.

**description** [string](#)

A description for the region.

## Properties

**this[int, int, int]**

Get or set the room at a location.

```
public Room this[int x, int y, int z] { get; set; }
```

## Parameters

**x** [int](#)

The x position.

**y** [int](#)

The y position.

**z** [int](#)

The z position.

## Property Value

[Room](#)

The room.

# Methods

## CanPlaceRoom(int, int, int)

Determine if a room can be placed at a location

```
public bool CanPlaceRoom(int x, int y, int z)
```

Parameters

x [int](#)

The X position.

y [int](#)

The Y position.

z [int](#)

The Z position.

Returns

[bool](#)

True if the room can be placed, else false.

## GetRoomPositions()

Get all current room positions.

```
public RoomPosition[] GetRoomPositions()
```

Returns

[RoomPosition\[\]](#)

The room positions.

## Make()

Make a region.

```
public Region Make()
```

Returns

[Region](#)

The created region.

## Make(RoomPosition)

Make a region.

```
public Region Make(RoomPosition startPosition)
```

Parameters

[startPosition RoomPosition](#)

The start position.

Returns

[Region](#)

The created region.

## Make(int, int, int)

Make a region.

```
public Region Make(int x, int y, int z)
```

Parameters

x [int](#)

The start x position.

y [int](#)

The start y position.

z [int](#)

The start z position.

Returns

[Region](#)

The created region.

# Namespace BP.AdventureFramework. Utilities.Generation Classes

## [GameGenerationOptions](#)

Provides options for generating games.

## [GameGenerator](#)

Represents a class for generating games.

## Interfaces

### [IDescriptionGenerator](#)

Represents a generator for descriptions.

### [IExaminableGenerator](#)

Represents any object that provides examinable generation.

### [IItemGenerator](#)

Represents any object that can generate items.

### [IRegionGenerator](#)

Represents any object that can generate a region.

### [IRoomGenerator](#)

Represents any object that is a room generator.

# Class GameGenerationOptions

Namespace: [BPAdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Provides options for generating games.

```
public sealed class GameGenerationOptions
```

## Inheritance

[object](#) ← GameGenerationOptions

## Inherited Members

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

## Properties

### MaximumRegions

Get the maximum regions.

```
public uint MaximumRegions { get; set; }
```

Property Value

[uint](#)

Provides options for generating games.

### MaximumRooms

Get the maximum rooms.

```
public uint MaximumRooms { get; set; }
```

## Property Value

[uint](#)

Provides options for generating games.

## MinimumRegions

Get the minimum regions.

```
public uint MinimumRegions { get; set; }
```

## Property Value

[uint](#)

Provides options for generating games.

## MinimumRooms

Get the minimum rooms.

```
public uint MinimumRooms { get; set; }
```

## Property Value

[uint](#)

Provides options for generating games.

## RegionComplexity

Get or set the complexity of the regions - higher numbers are more complex, lower numbers are less complex.

```
public uint RegionComplexity { get; set; }
```

Property Value

[uint](#)

Provides options for generating games.

## RoomToItemRatio

Get the ratio of rooms to items.

```
public double RoomToItemRatio { get; set; }
```

Property Value

[double](#)

Provides options for generating games.

# Class GameGenerator

Namespace: [BPAdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Represents a class for generating games.

```
public sealed class GameGenerator
```

## Inheritance

[object](#) ← GameGenerator

## Inherited Members

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

## Constructors

### GameGenerator(Identifier, Description)

Initializes a new instance of the OverworldMaker class.

```
public GameGenerator(Identifier identifier, Description description)
```

#### Parameters

`identifier` [Identifier](#)

An identifier for the region.

`description` [Description](#)

A description for the region.

### GameGenerator(string, string)

Initializes a new instance of the GameGenerator class.

```
public GameGenerator(string identifier, string description)
```

## Parameters

`identifier` [string](#)

An identifier for the region.

`description` [string](#)

A description for the region.

## Methods

`Generate(GameGenerationOptions, ITheme, out int)`

Generate an OverworldMaker.

```
public OverworldMaker Generate(GameGenerationOptions options, ITheme theme, out int seed)
```

## Parameters

`options` [GameGenerationOptions](#)

The generation options.

`theme` [ITheme](#)

The theme.

`seed` [int](#)

The seed used for generation.

## Returns

[OverworldMaker](#)

The generated overworld maker.

# Generate(int, GameGenerationOptions, ITheme)

Generate an OverworldMaker.

```
public OverworldMaker Generate(int seed, GameGenerationOptions options,  
ITheme theme)
```

## Parameters

**seed** [int](#)

The see to use for generation.

**options** [GameGenerationOptions](#)

The generation options.

**theme** [ITheme](#)

The theme.

## Returns

[OverworldMaker](#)

The created overworld maker.

# Interface IDescriptionGenerator

Namespace: [BPAdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Represents a generator for descriptions.

```
public interface IDescriptionGenerator
```

## Methods

### Generate(Identifier)

Generate a description.

```
Description Generate(Identifier identifier)
```

#### Parameters

`identifier` [Identifier](#)

The identifier to generate the description for.

#### Returns

[Description](#)

The description.

# Interface IExaminableGenerator

Namespace: [BP.AdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Represents any object that provides examinable generation.

```
public interface IExaminableGenerator
```

## Methods

### Generate(Random)

Generate an examinable.

```
IExaminable Generate(Random generator)
```

#### Parameters

generator [Random](#)

The generator.

#### Returns

[IExaminable](#)

The generated examinable.

# Interface IItemGenerator

Namespace: [BPAdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can generate items.

```
public interface IItemGenerator
```

## Methods

### Generate(Random)

Generate an item.

```
Item Generate(Random generator)
```

#### Parameters

generator [Random](#)

The generator.

#### Returns

[Item](#)

The generated item.

# Interface IRegionGenerator

Namespace: [BPAdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Represents any object that can generate a region.

```
public interface IRegionGenerator
```

## Methods

**GenerateRegion(Identifier, Description, Random, IRoomGenerator, IItemGenerator, IItemGenerator, GameGenerationOptions)**

Generate a region.

```
RegionMaker GenerateRegion(Identifier identifier, Description description, Random generator, IRoomGenerator roomGenerator, IItemGenerator takeableItemGenerator, IItemGenerator nonTakeableItemGenerator, GameGenerationOptions options)
```

## Parameters

**identifier** [Identifier](#)

The region identifier.

**description** [Description](#)

The region description.

**generator** [Random](#) ↴

The generator.

**roomGenerator** [IRoomGenerator](#)

The room generator.

`takeableItemGenerator` [IItemGenerator](#)

The item generator for takeable items.

`nonTakeableItemGenerator` [IItemGenerator](#)

The item generator for non-takeable items.

`options` [GameGenerationOptions](#)

The generation options.

Returns

[RegionMaker](#)

The generated region maker.

# Interface IRoomGenerator

Namespace: [BPAdventureFramework.Utilities.Generation](#)

Assembly: BP.AdventureFramework.dll

Represents any object that is a room generator.

```
public interface IRoomGenerator
```

## Methods

### GenerateRooms(RegionMaker, Random, GameGenerationOptions)

Generate the rooms.

```
void GenerateRooms(RegionMaker regionMaker, Random generator,  
GameGenerationOptions options)
```

#### Parameters

**regionMaker** [RegionMaker](#)

The region maker.

**generator** [Random](#)

The generator.

**options** [GameGenerationOptions](#)

The game generation options.

# Namespace BP.AdventureFramework. Utilities.Generation.Simple Classes

## [ExaminableGenerator](#)

Provides a examinable generator.

## [ItemGenerator](#)

Provides an item generator.

## [RegionGenerator](#)

Provides a region generator.

# Interfaces

## [ITheme](#)

Represents a theme that can be used for simple generation.

# Class ExaminableGenerator

Namespace: [BP.AdventureFramework.Utilities.Generation.Simple](#)

Assembly: BP.AdventureFramework.dll

Provides a examinable generator.

```
public sealed class ExaminableGenerator : IExaminableGenerator
```

## Inheritance

[object](#) ← ExaminableGenerator

## Implements

[IExaminableGenerator](#)

## Inherited Members

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

## Constructors

ExaminableGenerator(IEnumerable<string>,  
IEnumerable<string>, IDescriptionGenerator, bool)

Initializes a new instance of the ExaminableGenerator class.

```
public ExaminableGenerator(IEnumerable<string> nouns, IEnumerable<string>  
adjectives, IDescriptionGenerator descriptionGenerator, bool allowReuse)
```

## Parameters

nouns [IEnumerable](#)<string>

The nouns.

adjectives [IEnumerable](#)<string>

The adjectives.

`descriptionGenerator` [IDescriptionGenerator](#)

A generator to use for generating descriptions.

`allowReuse` [bool](#)

If reuse of nouns or adjectives are used.

## Methods

### Generate(Random)

Generate an examinable.

```
public IExaminable Generate(Random generator)
```

#### Parameters

`generator` [Random](#)

The generator.

#### Returns

[IExaminable](#)

The generated examinable.

# Interface ITheme

Namespace: [BP.AdventureFramework.Utilities.Generation.Simple](#)

Assembly: BP.AdventureFramework.dll

Represents a theme that can be used for simple generation.

```
public interface ITheme
```

## Properties

### Name

Get the name.

```
string Name { get; }
```

### Property Value

[string](#)

Represents a theme that can be used for simple generation.

### NonTakeableItemAdjectives

Get the non-takeable item adjectives.

```
string[] NonTakeableItemAdjectives { get; }
```

### Property Value

[string](#)[]

Represents a theme that can be used for simple generation.

## NonTakeableItemNouns

Get the non-takeable item nouns.

```
string[] NonTakeableItemNouns { get; }
```

Property Value

[string](#)[]

Represents a theme that can be used for simple generation.

## RoomAdjectives

Get the room adjectives.

```
string[] RoomAdjectives { get; }
```

Property Value

[string](#)[]

Represents a theme that can be used for simple generation.

## RoomNouns

Get the room nouns.

```
string[] RoomNouns { get; }
```

Property Value

[string](#)[]

Represents a theme that can be used for simple generation.

## TakeableItemAdjectives

Get the takeable item adjectives.

```
string[] TakeableItemAdjectives { get; }
```

Property Value

[string](#)[]

Represents a theme that can be used for simple generation.

## TakeableItemNouns

Get the takeable item nouns.

```
string[] TakeableItemNouns { get; }
```

Property Value

[string](#)[]

Represents a theme that can be used for simple generation.

# Class ItemGenerator

Namespace: [BP.AdventureFramework.Utilities.Generation.Simple](#)

Assembly: BP.AdventureFramework.dll

Provides an item generator.

```
public sealed class ItemGenerator : IItemGenerator
```

## Inheritance

[object](#) ← ItemGenerator

## Implements

[IItemGenerator](#)

## Inherited Members

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

## Constructors

### ItemGenerator(IExaminableGenerator, bool)

Initializes a new instance of the ItemGenerator class.

```
public ItemGenerator(IExaminableGenerator examinableGenerator, bool isTakeable)
```

## Parameters

examinableGenerator [IExaminableGenerator](#)

An examinable generator.

isTakeable [bool](#)

True if the generated items are takeable, else false.

## Methods

# Generate(Random)

Generate an item.

```
public Item Generate(Random generator)
```

Parameters

generator [Random](#)

The generator.

Returns

[Item](#)

The generated item.

# Class RegionGenerator

Namespace: [BP.AdventureFramework.Utilities.Generation.Simple](#)

Assembly: BP.AdventureFramework.dll

Provides a region generator.

```
public sealed class RegionGenerator : IRegionGenerator
```

## Inheritance

[object](#) ← RegionGenerator

## Implements

[IRegionGenerator](#)

## Inherited Members

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

## Methods

**GenerateRegion(Identifier, Description, Random, IRoomGenerator, IItemGenerator, IItemGenerator, GameGenerationOptions)**

Generate a region.

```
public RegionMaker GenerateRegion(Identifier identifier, Description description,  
Random generator, IRoomGenerator roomGenerator, IItemGenerator  
takeableItemGenerator, IItemGenerator nonTakeableItemGenerator,  
GameGenerationOptions options)
```

## Parameters

**identifier** [Identifier](#)

The region identifier.

`description` [Description](#)

The region description.

`generator` [Random](#) ↗

The generator.

`roomGenerator` [IRoomGenerator](#)

The room generator.

`takeableItemGenerator` [IItemGenerator](#)

The item generator for takeable items.

`nonTakeableItemGenerator` [IItemGenerator](#)

The item generator for non-takeable items.

`options` [GameGenerationOptions](#)

The generation options.

Returns

[RegionMaker](#)

The generated region maker.

# Namespace BP.AdventureFramework. Utilities.Generation.Themes

## Classes

### [Castle](#)

Provides the castle theme.

# Class Castle

Namespace: [BP.AdventureFramework.Utilities.Generation.Themes](#)

Assembly: BP.AdventureFramework.dll

Provides the castle theme.

```
public sealed class Castle : ITheme
```

## Inheritance

[object](#) ← Castle

## Implements

[ITheme](#)

## Inherited Members

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

# Properties

## Name

Get the name.

```
public string Name { get; }
```

## Property Value

[string](#)

Provides the castle theme.

# NonTakeableItemAdjectives

Get the non-takeable item adjectives.

```
public string[] NonTakeableItemAdjectives { get; }
```

Property Value

[string](#)[]

Provides the castle theme.

## NonTakeableItemNouns

Get the non-takeable item nouns.

```
public string[] NonTakeableItemNouns { get; }
```

Property Value

[string](#)[]

Provides the castle theme.

## RoomAdjectives

Get the room adjectives.

```
public string[] RoomAdjectives { get; }
```

Property Value

[string](#)[]

Provides the castle theme.

## RoomNouns

Get the room nouns.

```
public string[] RoomNouns { get; }
```

Property Value

[string](#)[]

Provides the castle theme.

## TakeableItemAdjectives

Get the takeable item adjectives.

```
public string[] TakeableItemAdjectives { get; }
```

Property Value

[string](#)[]

Provides the castle theme.

## TakeableItemNouns

Get the takeable item nouns.

```
public string[] TakeableItemNouns { get; }
```

Property Value

[string](#)[]

Provides the castle theme.