Monster RPG Game Kit Reference

PiRho Soft

Ability

PiRhoSoft.MonsterRpgEngine.Ability : Resource, IReloadable, IVariableStore

Description

Constructors

Ability()

# Public Fields

Ecosystem *Ecosystem*

string *Name*

VariableList *Traits*

Expression *UseInWorldCondition*

InstructionCaller *UseInWorldInstruction*

Expression *UseInBattleCondition*

InstructionCaller *UseInBattleInstruction*

# Public Methods

void OnEnable() *(virtual)*

void OnDisable() *(virtual)*

bool IsUsableInWorld(IVariableStore *variables***)** *(virtual)*

IEnumerator UseInWorld(InstructionStore *variables***)** *(virtual)*

bool IsUsableInBattle(IVariableStore *variables***)** *(virtual)*

IEnumerator UseInBattle(InstructionStore *variables***)** *(virtual)*

Move CreateMove(Creature *creature***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

# Protected Methods

void AddPropertiesToMap<AbilityType>(PropertyMap<AbilityType> *map***)**

PropertyMap GetPropertyMap() *(virtual)*

<<<

AbilityUseLocation

PiRhoSoft.MonsterRpgEngine.AbilityUseLocation

Description

Values

AbilityUseLocation *World*

AbilityUseLocation *Battle*

<<<

AbilityVariableSource

PiRhoSoft.MonsterRpgEngine.AbilityVariableSource : ObjectVariableSource<Ability>

Description

Constructors

AbilityVariableSource()

<<<

Accessory

PiRhoSoft.MonsterRpgEngine.Accessory : Part

Description

Constructors

Accessory()

# Public Fields

AudioSource *Audio*

Animator *Animator*

SimpleAnimationPlayer *Animation*

<<<

AiSelection

PiRhoSoft.MonsterRpgEngine.AiSelection : InstructionGraphNode

Description

Constructors

AiSelection()

# Public Fields

string *Tag*

AiSelectionItemList *Items*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

void GetConnections(NodeData *data***)** *(virtual)*

void SetConnection(ConnectionData *connection*, **InstructionGraphNode** *target***)** *(virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

AiSelectionItem

PiRhoSoft.MonsterRpgEngine.AiSelectionItem

Description

Constructors

AiSelectionItem()

# Public Fields

InstructionGraphNode *OnSelected*

string *Label*

VariableReference *Variables*

<<<

AiSelectionItemList

PiRhoSoft.MonsterRpgEngine.AiSelectionItemList : SerializedList<AiSelectionItem>

Description

Constructors

AiSelectionItemList()

<<<

AnimatedTile

PiRhoSoft.MonsterRpgEngine.AnimatedTile : TileBase

Description

Constructors

AnimatedTile()

# Public Fields

bool *RandomizeStart*

float *AnimationStartTime*

float *AnimationSpeed*

float *AnimationSpeedMaximum*

List<TileTransformInfo> *Tiles*

# Public Methods

void GetTileData(Vector3Int *location*, **ITilemap** *tileMap*, **TileData** *tileData (ref)***)** *(virtual)*

bool GetTileAnimationData(Vector3Int *location*, **ITilemap** *tileMap*, **TileAnimationData** *tileAnimationData (ref)***)** *(virtual)*

<<<

AnimationClipDictionary

PiRhoSoft.MonsterRpgEngine.AnimationClipDictionary : SerializedDictionary<string, string>

Description

Constructors

AnimationClipDictionary()

<<<

ApproachMover

PiRhoSoft.MonsterRpgEngine.ApproachMover : ApproachNode

Description

Constructors

ApproachMover()

# Public Fields

VariableReference *Target*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

# Protected Methods

Vector2Int GetTargetPosition(InstructionStore *variables***)** *(virtual)*

<<<

ApproachNode

PiRhoSoft.MonsterRpgEngine.ApproachNode : InstructionGraphNode

Description

# Public Fields

InstructionGraphNode *Next*

bool *WaitForCompletion*

bool *UsePathfinding*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Constructors

ApproachNode()

# Protected Methods

Vector2Int GetTargetPosition(InstructionStore *variables***)** *(abstract)*

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ApproachPosition

PiRhoSoft.MonsterRpgEngine.ApproachPosition : ApproachNode

Description

Constructors

ApproachPosition()

# Public Fields

Vector2Int *Position*

# Protected Methods

Vector2Int GetTargetPosition(InstructionStore *variables***)** *(virtual)*

<<<

AreaController

PiRhoSoft.MonsterRpgEngine.AreaController : Controller

Description

Constructors

AreaController()

 # Public Fields

int *LeftDistance*

int *RightDistance*

int *UpDistance*

int *DownDistance*

float *MovementDelay*

 <<<

AudioClipDictionary

PiRhoSoft.MonsterRpgEngine.AudioClipDictionary : SerializedDictionary<string, string>

Description

Constructors

AudioClipDictionary()

 <<<

AudioManager

PiRhoSoft.MonsterRpgEngine.AudioManager : SingletonBehaviour<AudioManager>

Description

Constructors

AudioManager()

 # Public Fields

AudioMixerGroup *MasterMixer*

 # Public Methods

void Push(AudioClip *clip*, **float** *fadeOut*, **float** *fadeIn*, **float** *crossFade***)**

void Pop(float *fadeOut*, **float** *fadeIn*, **float** *crossFade***)**

<<<

BattleAi

PiRhoSoft.MonsterRpgEngine.BattleAi : MonoBehaviour

Description

Constructors

BattleAi()

# Public Methods

int MakeSelection(InstructionStore *variables*, **List<VariableValue>** *values*, **string** *tag***)**

# Protected Methods

int SelectFirst(List<VariableValue> *values*, **string** *tag***)**

int SelectRandom(List<VariableValue> *values*, **string** *tag***)**

<<<

BattleAnimationClip

PiRhoSoft.MonsterRpgEngine.BattleAnimationClip

Description

Constructors

BattleAnimationClip()

# Public Fields

bool *WaitForCompletion*

CreatureDisplayTarget *AnimationTarget*

BattleAnimationSource *AnimationSource*

string *Animation*

string *Parameter*

<<<

BattleAnimationClipDictionary

PiRhoSoft.MonsterRpgEngine.BattleAnimationClipDictionary : SerializedDictionary<string, string>

Description

Constructors

BattleAnimationClipDictionary()

<<<

BattleAnimationEffect

PiRhoSoft.MonsterRpgEngine.BattleAnimationEffect

Description

Constructors

BattleAnimationEffect()

# Public Fields

bool *WaitForCompletion*

BattleAnimationSource *EffectSource*

string *Effect*

string *MountPoint*

CreatureDisplayTarget *EffectTarget*

bool *FollowParent*

<<<

BattleAnimationEffectDictionary

PiRhoSoft.MonsterRpgEngine.BattleAnimationEffectDictionary : SerializedDictionary<string, string>

Description

Constructors

BattleAnimationEffectDictionary()

<<<

BattleAnimationEvents

PiRhoSoft.MonsterRpgEngine.BattleAnimationEvents : MonoBehaviour

Description

Constructors

BattleAnimationEvents()

# Public Fields

BattleAnimationClipDictionary *Clips*

BattleAnimationSoundDictionary *Sounds*

BattleAnimationEffectDictionary *Effects*

# Public Methods

void StartClip(string *name***)**

void PlayClip(string *name***)**

void StartSound(string *name***)**

void PlaySound(string *name***)**

void StartEffect(string *name***)**

void PlayEffect(string *name***)**

# Protected Methods

void Pause(bool *dummy1*, **bool** *dummy2***)**

void Unpause(bool *dummy1*, **bool** *dummy2***)**

<<<

BattleAnimationSound

PiRhoSoft.MonsterRpgEngine.BattleAnimationSound

Description

Constructors

BattleAnimationSound()

# Public Fields

bool *WaitForCompletion*

BattleAnimationSource *SoundSource*

string *Sound*

<<<

BattleAnimationSoundDictionary

PiRhoSoft.MonsterRpgEngine.BattleAnimationSoundDictionary : SerializedDictionary<string, string>

Description

Constructors

BattleAnimationSoundDictionary()

<<<

BattleAnimationSource

PiRhoSoft.MonsterRpgEngine.BattleAnimationSource

Description

Values

BattleAnimationSource *Display*

BattleAnimationSource *Variables*

<<<

BattleContext

PiRhoSoft.MonsterRpgEngine.BattleContext : VariableStore

Description

Constructors

BattleContext(IList<**ITrainer**> *trainers***)**

# Public Properties

TrainerContextList *Trainers (read only)*

TrainerContextList *ActiveTrainers (read only)*

# Public Methods

TrainerContext CreateTrainer(ITrainer *trainer***)** *(virtual)*

CreatureContext CreateCreature(TrainerContext *trainer*, **Creature** *creature***)** *(virtual)*

MoveContext CreateMove(CreatureContext *creature*, **Move** *move***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

<<<

BattleGraph

PiRhoSoft.MonsterRpgEngine.BattleGraph : InstructionGraph

Description

Constructors

BattleGraph()

# Public Fields

InstructionGraphNode *Enter*

InstructionGraphNode *Start*

InstructionGraphNode *Process*

InstructionGraphNode *Finish*

InstructionGraphNode *Exit*

SceneReference *UiScene*

TrainerReferenceList *Trainers*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

IEnumerator Setup(InstructionStore *variables*, **IList<ITrainer>** *trainers***)** *(virtual)*

IEnumerator Teardown() *(virtual)*

# Protected Methods

IEnumerator Run(InstructionStore *variables***)** *(virtual)*

IEnumerator Run(InstructionStore *variables*, **IList<ITrainer>** *trainers***)** *(virtual)*

void OnEnable() *(virtual)*

void OnDisable() *(virtual)*

IEnumerator LoadScenes()

IEnumerator UnloadScenes()

<<<

BattleInterface

PiRhoSoft.MonsterRpgEngine.BattleInterface : Interface

Description

Constructors

BattleInterface()

# Public Fields

bool *HideWorld*

CreatureDisplayList *CreatureDisplays*

# Public Methods

CreatureDisplay GetCreatureDisplay(int *index*)

# Protected Methods

void Setup() *(virtual)*

void Teardown() *(virtual)*

<<<

Building

PiRhoSoft.MonsterRpgEngine.Building : MonoBehaviour

Description

Constructors

Building()

# Public Fields

CollisionLayer *CollisionLayer*

Rect *Bounds*

float *SortPoint*

int *RoofHeight*

Part *Roof*

Part *Facade*

List<DoorPart> *Doors*

List<Accessory> *Accessories*

<<<

CanPurchaseItem

PiRhoSoft.MonsterRpgEngine.CanPurchaseItem : InstructionGraphNode

Description

Constructors

CanPurchaseItem()

# Public Fields

InstructionGraphNode *OnTrue*

InstructionGraphNode *OnFalse*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

CanTossItem

PiRhoSoft.MonsterRpgEngine.CanTossItem : InstructionGraphNode

Description

Constructors

CanTossItem()

# Public Fields

InstructionGraphNode *OnTrue*

InstructionGraphNode *OnFalse*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

CanUseItem

PiRhoSoft.MonsterRpgEngine.CanUseItem : InstructionGraphNode

Description

Constructors

CanUseItem()

# Public Fields

ItemUseLocation *Location*

InstructionGraphNode *OnTrue*

InstructionGraphNode *OnFalse*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

CanUseMove

PiRhoSoft.MonsterRpgEngine.CanUseMove : InstructionGraphNode

Description

Constructors

CanUseMove()

# Public Fields

AbilityUseLocation *Location*

InstructionGraphNode *OnTrue*

InstructionGraphNode *OnFalse*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

CheckerboardTile

PiRhoSoft.MonsterRpgEngine.CheckerboardTile : TileBase

Description

Constructors

CheckerboardTile()

# Public Fields

TileTransformInfo *First*

TileTransformInfo *Second*

# Public Methods

void GetTileData(Vector3Int *position*, **ITilemap** *tileMap*, **TileData** *tileData (ref)*) *(virtual)*

TileTransformInfo GetInfo(Vector3Int *position*)

<<<

CollisionLayer

PiRhoSoft.MonsterRpgEngine.CollisionLayer

Description

Values

CollisionLayer *None*

CollisionLayer *One*

CollisionLayer *Two*

CollisionLayer *Three*

CollisionLayer *Four*

CollisionLayer *Five*

CollisionLayer *All*

<<<

Controller

PiRhoSoft.MonsterRpgEngine.Controller : MonoBehaviour

Description

Public Properties

Mover *Mover (read only)*

bool *IsFrozen (read only)*

# Public Methods

void Freeze()

void Thaw()

# Protected Constructors

Controller()

# Protected Methods

void Awake() *(virtual)*

void UpdateMover(float *horizontal*, **float** *vertical***)**

<<<

CreateCreature

PiRhoSoft.MonsterRpgEngine.CreateCreature : InstructionGraphNode

Description

# Constructors

CreateCreature()

# Public Fields

InstructionGraphNode *ProcessCreature*
VariableSchema *Schema*
Expression *Initializer*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

# Protected Methods

IEnumerator **Run_(**InstructionGraph *graph*, InstructionStore *variables*, **int** *iteration***)** *(virtual)*

<<<

Creature

PiRhoSoft.MonsterRpgEngine.Creature : ScriptableObject, IVariableStore, IVariableListener

Description

Static Methods

Creature **Create(**CreatureSaveData *data*, ITrainer *trainer***)**
CreatureSaveData **Save(**Creature *creature***)**

# Constructors

Creature()

# Public Fields

Species *Species*
string *Name*
VariableList *Traits*
MoveList *Moves*

# Public Properties

ITrainer *Trainer (read only)*

# Public Methods

void Setup(ITrainer *trainer***)**

void Teardown()

Creature Clone(ITrainer *trainer***)**

List<int> TakePendingTraits()

void UpdatePendingTraits()

void UpdateTrait(int *index***)**

void ResetTraits(string *availability***)** *(virtual)*

void ResetTraits(IList<string> *traits***)** *(virtual)*

bool HasPendingSkill()

Skill TakePendingSkill()

List<int> TakePendingSkills()

void TeachPendingSkills()

void TeachSkill(int *index***)**

bool CanLearnSkill(Skill *skill***)**

void TeachSkill(Skill *skill***)**

IEnumerator TeachSkill(Skill *skill*, **InstructionContext** *context***)**

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

void VariableChanged(int *index*, **VariableValue** *value***)** *(virtual)*

# Protected Methods

void AddPropertiesToMap<CreatureType>(PropertyMap<CreatureType> *map***)**

PropertyMap GetPropertyMap() *(virtual)*

void TraitChanged(int *index***)** *(virtual)*

void Load(CreatureSaveData *data***)** *(virtual)*

void Save(CreatureSaveData *data***)** *(virtual)*

<<<

CreatureContext

PiRhoSoft.MonsterRpgEngine.CreatureContext : LocalVariableStore<Creature>

Description

Constructors

CreatureContext(BattleContext *battle*, **TrainerContext** *trainer*, **Creature** *creature***)**

# Public Properties

Creature *Creature (read only)*

MoveContextList *Moves (read only)*

TrainerContext *Trainer (read only)*

# Public Methods

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

<<<

CreatureContextList

PiRhoSoft.MonsterRpgEngine.CreatureContextList : IndexedVariableStore<CreatureContext>

Description

Constructors

CreatureContextList()

<<<

CreatureDisplay

PiRhoSoft.MonsterRpgEngine.CreatureDisplay : InterfaceControl

Description

Constructors

CreatureDisplay()

# Public Fields

AnimationPlayer *BattlerPlayer*

AudioPlayer *BattlerAudio*

Animator *CreatureAnimator*

AnimationClipDictionary *Animations*

AudioClipDictionary *Sounds*

GameObjectDictionary *Effects*

# Public Properties

CreatureContext *Creature (read only)*

# Public Methods

void SetCreature(CreatureContext *creature***)**

void UpdateCreature(string *group*, **BindingAnimationStatus** *status***)**

IEnumerator PlayAndWait(BattleAnimationClip *animation*, **IVariableStore** *variables***)**

void Play(BattleAnimationClip *animation*, **IVariableStore** *variables***)**

IEnumerator PlayAndWait(BattleAnimationSound *animation*, **IVariableStore** *variables***)**

void Play(BattleAnimationSound *animation*, **IVariableStore** *variables***)**

IEnumerator PlayAndWait(BattleAnimationEffect *animation*, **IVariableStore** *variables***)**

void Play(BattleAnimationEffect *animation*, **IVariableStore** *variables***)**

<<<

CreatureDisplayList

PiRhoSoft.MonsterRpgEngine.CreatureDisplayList : SerializedList<CreatureDisplay>

Description

Constructors

CreatureDisplayList()

<<<

CreatureDisplayNode

PiRhoSoft.MonsterRpgEngine.CreatureDisplayNode : InstructionGraphNode

Description

Public Fields

InstructionGraphNode *Next*

string *InterfaceName*

VariableReference *IndexVariable*

# Protected Constructors

CreatureDisplayNode()

# Protected Methods

CreatureDisplay GetDisplay(IVariableStore *variables***)**

<<<

CreatureDisplayTarget

PiRhoSoft.MonsterRpgEngine.CreatureDisplayTarget

Description

Values

CreatureDisplayTarget *Battler*

CreatureDisplayTarget *Creature*

<<<

CreatureReference

PiRhoSoft.MonsterRpgEngine.CreatureReference : IVariableStore

Description

Constructors

CreatureReference()

# Public Fields

Creature *Creature*

Species *Species*

InstructionCaller *Generator*

# Public Methods

void Setup()

Creature CreateCreature(ITrainer *trainer***)**

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

CreatureSaveData

PiRhoSoft.MonsterRpgEngine.CreatureSaveData

Description

Constructors

CreatureSaveData()

# Public Fields

string *SpeciesPath*

string *Name*

VariableList *Traits*

SkillsDictionary *LearnedSkills*

List<**MoveSaveData**> *Moves*

<<<

DamageModifier

PiRhoSoft.MonsterRpgEngine.DamageModifier

Description

Constructors

DamageModifier()

# Public Fields

string *Name*

Expression *Condition*

Expression *Modifier*

<<<

DamageModifierList

PiRhoSoft.MonsterRpgEngine.DamageModifierList : SerializedList<DamageModifier>

Description

Constructors

DamageModifierList()

<<<

DealDamage

PiRhoSoft.MonsterRpgEngine.DealDamage : InstructionGraphNode

Description

Constructors

DealDamage()

# Public Fields

InstructionGraphNode *Next*

string *VariableName*

string *HpTrait*

Expression *BaseDamage*

DamageModifierList *Modifiers*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

Direction

PiRhoSoft.MonsterRpgEngine.Direction

Description

Static Methods

bool Contains(InteractionDirection *interactionDirection*, **MovementDirection** *movementDirection***)**

MovementDirection Opposite(MovementDirection *direction***)**

Vector2Int GetVector(MovementDirection *direction***)**

MovementDirection GetDirection(float *horizontal*, **float** *vertical***)**

MovementDirection GetDirection(Vector2Int *direction***)**

void GetMovement(MovementDirection *direction*, **float** *horizontal (out)*, **float** *vertical (out)***)**

<<<

Door

PiRhoSoft.MonsterRpgEngine.Door : AnimationPlayer

Description

Constructors

Door()

# Public Fields

AnimationClip *OpenAnimation*

AnimationClip *CloseAnimation*

AudioClip *OpenSound*

AudioClip *CloseSound*

# Protected Methods

void Awake() *(virtual)*

<<<

DoorPart

PiRhoSoft.MonsterRpgEngine.DoorPart : Part

Description

## Constructors

DoorPart()

# Public Fields

AudioSource *Audio*

Animator *Animator*

Door *Door*

<<<

Ecosystem

PiRhoSoft.MonsterRpgEngine.Ecosystem : ScriptableObject

Description

Static Fields

string *InBattle*

string *ActiveInBattle*

# Constructors

Ecosystem()

# Public Fields

VariableSchema *TrainerSchema*

VariableSchema *SpeciesSchema*

VariableSchema *CreatureSchema*

VariableSchema *AbilitySchema*

VariableSchema *MoveSchema*

# Public Methods

VariableMap **GetTrainerMap(**PropertyMap *propertyMap***)**

VariableMap **GetSpeciesMap(**PropertyMap *propertyMap***)**

VariableMap **GetCreatureMap(**PropertyMap *propertyMap***)**

VariableMap **GetAbilityMap(**PropertyMap *propertyMap***)**

VariableMap **GetMoveMap(**PropertyMap *propertyMap***)**

<<<

Encounter

PiRhoSoft.MonsterRpgEngine.Encounter : MonoBehaviour, IVariableStore, ITrainer

Description

Constructors

Encounter()

# Public Fields

InstructionCaller *Instructions*

float *EncounterChance*

EncounterTable *Encounters*

# Public Properties

Roster *Roster (read only) (virtual)*

BattleAi *Ai (read only) (virtual)*

bool *IsPlayer (read only) (virtual)*

# Public Methods

void Enter() *(virtual)*

void ResetTraits(string *availability***)** *(virtual)*

void ResetTraits(IList<**string**> *traits***)** *(virtual)*

VariableValue **GetVariable(string** *name***)** *(virtual)*

SetVariableResult **SetVariable(string** *name*, VariableValue *value***)** *(virtual)*

IEnumerable<**string**> **GetVariableNames()** *(virtual)*

# Protected Methods

void DoEncounter() *(virtual)*

Creature **PickCreature()** *(virtual)*

IEnumerator **RunEncounter(**Creature *creature***)** *(virtual)*

<<<

EncounterTable

PiRhoSoft.MonsterRpgEngine.EncounterTable : DropTable<CreatureReference>

Description

Constructors

EncounterTable()

<<<

FaceDirection

PiRhoSoft.MonsterRpgEngine.FaceDirection : InstructionGraphNode

Description

Constructors

FaceDirection()

# Public Fields

InstructionGraphNode *Next*

MovementDirection *Direction*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

FaceMover

PiRhoSoft.MonsterRpgEngine.FaceMover : InstructionGraphNode

Description

Constructors

FaceMover()

# Public Fields

InstructionGraphNode *Next*

VariableReference *Target*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

# Protected Methods

IEnumerator **Run_(**InstructionGraph *graph*, InstructionStore *variables*, **int** *iteration***)** *(virtual)*

<<<

FollowCamera

PiRhoSoft.MonsterRpgEngine.FollowCamera : MonoBehaviour

Description

Constructors

FollowCamera()

# Public Fields

float *MoveSpeed*

float *ZoomSpeed*

GameObject *TargetObject*

# Public Methods

void SetBounds(Rect *bounds***)**

void ClearBounds()

void StartFollowing(GameObject *gameObject*, **bool** *warp***)**

void StopFollowing()

void WarpToZoom(float *zoom***)**

void MoveTowardZoom(float *zoom***)**

void WarpToPosition(Vector2 *position***)**

void MoveTowardsPosition(Vector2 *position***)**

# Protected Methods

void Awake() *(virtual)*

void Update() *(virtual)*

void LateUpdate() *(virtual)*

Vector2 **ClampBounds(**Vector2 *position*, **float** *size***)**

Vector2 **ClampToBounds(**Vector2 *position*, **float** *size*, Rect *clampArea*, **bool** *left*, **bool** *right*, **bool** *bottom*, **bool** *top***)**

Vector2 **ClampToZone(**Vector2 *position*, **float** *size*, MapProperties *map***)**

<<<

FollowPath

PiRhoSoft.MonsterRpgEngine.FollowPath : PathNode

Description

Constructors

FollowPath()

# Public Fields

Path *Path*

# Protected Methods

Path GetPath(Mover *mover***)** *(virtual)*

<<<

FreezeController

PiRhoSoft.MonsterRpgEngine.FreezeController : InstructionGraphNode

Description

Constructors

FreezeController()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

GameObjectDictionary

PiRhoSoft.MonsterRpgEngine.GameObjectDictionary : SerializedDictionary<string, string>

Description

Constructors

GameObjectDictionary()

<<<

GameSaveData

PiRhoSoft.MonsterRpgEngine.GameSaveData

Description

Constructors

GameSaveData()

# Public Fields

string *MainScene*

string *StartingZone*

SpawnPoint *PlayerSpawn*

<<<

GiveCreature

PiRhoSoft.MonsterRpgEngine.GiveCreature : InstructionGraphNode

Description

Constructors

GiveCreature()

# Public Fields

InstructionGraphNode *Next*

CreatureReference *Creature*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

GiveItem

PiRhoSoft.MonsterRpgEngine.GiveItem : InstructionGraphNode

Description

Constructors

GiveItem()

# Public Fields

InstructionGraphNode *Next*
ItemVariableSource *Item*
IntegerVariableSource *Amount*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

# Protected Methods

IEnumerator **Run_(**InstructionGraph *graph*, InstructionStore *variables*, **int** *iteration***)** *(virtual)*

<<<

HasItem

PiRhoSoft.MonsterRpgEngine.HasItem : InstructionGraphNode

Description

Constructors

HasItem()

# Public Fields

InstructionGraphNode *OnTrue*
InstructionGraphNode *OnFalse*
ItemVariableSource *Item*
IntegerVariableSource *Amount*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

# Protected Methods

IEnumerator **Run_(**InstructionGraph *graph*, InstructionStore *variables*, **int** *iteration***)** *(virtual)*

<<<

IInteractable

PiRhoSoft.MonsterRpgEngine.IInteractable

Description

## Public Methods

bool IsInteracting() *(abstract)*

bool CanInteract(MovementDirection *direction***)** *(abstract)*

void Interact() *(abstract)*

<<<

InstructionTrigger

PiRhoSoft.MonsterRpgEngine.InstructionTrigger : IInteractable

Description

Constructors

InstructionTrigger()

# Public Fields

InteractionDirection *InteractionDirections*

InteractionDirection *EnteringDirections*

InteractionDirection *EnterDirections*

InteractionDirection *ExitingDirections*

InteractionDirection *ExitDirections*

InstructionCaller *InteractionInstructions*

InstructionCaller *EnteringInstructions*

InstructionCaller *EnterInstructions*

InstructionCaller *ExitingInstructions*

InstructionCaller *ExitInstructions*

# Public Methods

bool Equals(Object *obj***)** *(virtual)*

int GetHashCode() *(virtual)*

bool IsInteracting() *(virtual)*

bool CanInteract(MovementDirection *direction***)** *(virtual)*

void Interact() *(virtual)*

void Entering(MovementDirection *direction***)**

void Enter(MovementDirection *direction***)**

void Exiting(MovementDirection *direction***)**

void Exit(MovementDirection *direction***)**

<<<

Interaction

PiRhoSoft.MonsterRpgEngine.Interaction : MonoBehaviour, IVariableStore, IInteractable

Description

Constructors

Interaction()

# Public Fields

InteractionDirection *Directions*

InstructionCaller *Instructions*

# Public Methods

bool IsInteracting() *(virtual)*

bool CanInteract(MovementDirection *direction***)** *(virtual)*

void Interact() *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

# Protected Methods

void Awake() *(virtual)*

<<<

InteractionDirection

PiRhoSoft.MonsterRpgEngine.InteractionDirection

Description

Values

InteractionDirection *Any*

InteractionDirection *Left*

InteractionDirection *Right*

InteractionDirection *Down*

InteractionDirection *Up*

InteractionDirection *This*

<<<

Inventory

PiRhoSoft.MonsterRpgEngine.Inventory : IVariableStore

Description

Constructors

Inventory()

# Public Fields

int *Money*

InventoryItemList *Items*

# Public Methods

void Setup(Object *owner***)**

void Add(Item *item*, **int** *amount***)**

void Remove(InventoryItem *item*, **int** *amount***)**

bool Contains(Item *item*, **int** *amount***)**

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<**string**> **GetVariableNames()** *(virtual)*

<<<

InventoryItem

PiRhoSoft.MonsterRpgEngine.InventoryItem : IVariableStore

Description

Constructors

InventoryItem()

# Public Fields

Item *Item*

int *Count*

# Public Properties

Inventory *Inventory (read only)*

# Public Methods

IEnumerator UseInWorld(InstructionStore *variables***)**

IEnumerator UseInBattle(InstructionStore *variables***)**

void Toss(int *amount***)**

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

InventoryItemList

PiRhoSoft.MonsterRpgEngine.InventoryItemList : IndexedVariableStore<InventoryItem>

Description

Constructors

InventoryItemList()

<<<

Item

PiRhoSoft.MonsterRpgEngine.Item : Resource, IVariableStore

Description

Constructors

Item()

# Public Fields

string *Name*

string *Description*

ItemType *Type*

int *PurchaseCost*

int *SellCost*

Expression *UseInWorldCondition*

InstructionCaller *UseInWorldInstruction*

Expression *UseInBattleCondition*

InstructionCaller *UseInBattleInstruction*

# Public Methods

bool IsUsableInWorld(IVariableStore *variables***)** *(virtual)*

IEnumerator UseInWorld(InstructionStore *variables***)** *(virtual)*

bool IsUsableInBattle(IVariableStore *variables***)** *(virtual)*

IEnumerator UseInBattle(InstructionStore *variables***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

ItemSaveData

PiRhoSoft.MonsterRpgEngine.ItemSaveData

Description

Constructors

ItemSaveData()

# Public Fields

string *ItemPath*

int *Count*

<<<

ItemType

PiRhoSoft.MonsterRpgEngine.ItemType

Description

Values

ItemType *Durable*

ItemType *Consumable*

ItemType *Key*

<<<

ItemUseLocation

PiRhoSoft.MonsterRpgEngine.ItemUseLocation

Description

Values

ItemUseLocation *World*

ItemUseLocation *Battle*

<<<

ItemVariableSource

PiRhoSoft.MonsterRpgEngine.ItemVariableSource : ObjectVariableSource<Item>

Description

Constructors

ItemVariableSource()

<<<

ITrainer

PiRhoSoft.MonsterRpgEngine.ITrainer : IVariableStore

Description

Public Properties

Roster *Roster (read only) (abstract)*

BattleAi *Ai (read only) (abstract)*

bool *IsPlayer (read only) (abstract)*

# Public Methods

void ResetTraits(string *availability***)** *(abstract)*

void ResetTraits(IList<string> *traits***)** *(abstract)*

<<<

LayerSorting

PiRhoSoft.MonsterRpgEngine.LayerSorting : MonoBehaviour

Description

Static Fields

int *LayerCount*

# Static Methods

int GetSortingOrder(CollisionLayer *layer***)**

# Constructors

LayerSorting()

# Public Fields

CollisionLayer *Layer*

bool *ForceToBottom*

<<<

LineOfSightTrigger

PiRhoSoft.MonsterRpgEngine.LineOfSightTrigger : MonoBehaviour

Description

## Constructors

LineOfSightTrigger()

# Public Fields

int *Distance*

bool *SeesThroughWalls*

<<<

LoadInformation

PiRhoSoft.MonsterRpgEngine.LoadInformation

Description

Constructors

LoadInformation()

# Public Fields

Action *OnComplete*

Action<LoadState, LoadState> *OnProgress*

Action<string> *OnError*

# Public Properties

LoadState *State (read only)*

float *Progress (read only)*

string *Message (read only)*

# Public Methods

void UpdateProgress(LoadState *state*, **float** *progress***)**

void SetError(string *message***)**

void SetComplete()

<<<

LoadState

PiRhoSoft.MonsterRpgEngine.LoadState

Description

Values

LoadState *ReadingData*

LoadState *LoadingWorld*

LoadState *LoadingZones*

LoadState *LoadingUi*

LoadState *Error*

LoadState *Complete*

<<<

MapLayerList

PiRhoSoft.MonsterRpgEngine.MapLayerList : SerializedList<string>

Description

Constructors

MapLayerList()

<<<

MapProperties

PiRhoSoft.MonsterRpgEngine.MapProperties : MonoBehaviour

Description

Constructors

MapProperties()

# Public Fields

bool *ClampLeftBounds*

bool *ClampRightBounds*

bool *ClampBottomBounds*

bool *ClampTopBounds*

float *LeftBounds*

float *RightBounds*

float *BottomBounds*

float *TopBounds*

# Public Properties

bool *ClampBounds (read only)*

TileDictionary *Tiles (read only)*

# Public Methods

TileInfo AddOrGetTile(Vector2Int *position***)**

TileInfo GetTile(Vector2Int *position***)**

TileInfo AddTile(Vector2Int *position***)**

void AddConnections(List<int> *connections***)**

void AddSpawnPoints(Dictionary<string, string> *spawnPoints***)**

void RefreshTiles()

RectInt GetBounds()

float CalculateLeft()

float CalculateRight()

float CalculateTop()

float CalculateBottom()

<<<

MonsterRpg

PiRhoSoft.MonsterRpgEngine.MonsterRpg

Description

Static Fields

string *DocumentationUrl*

<<<

MountPoint

PiRhoSoft.MonsterRpgEngine.MountPoint : ValueType

Description

Public Fields

float *X*

float *Y*

float *Rotation*

<<<

MountPointDictionary

PiRhoSoft.MonsterRpgEngine.MountPointDictionary : SerializedDictionary<string, string>

Description

Constructors

MountPointDictionary()

<<<

Move

PiRhoSoft.MonsterRpgEngine.Move : ScriptableObject, IVariableStore

Description

Static Methods

Move Create(Creature *creature*, **MoveSaveData** *data***)**

MoveSaveData Save(Move *move***)**

# Constructors

Move()

# Public Fields

Ability *Ability*

Creature *Creature*

string *Name*

VariableList *Traits*

# Public Methods

void Setup(Creature *creature***)**

Move Clone(Creature *creature***)**

void ResetTraits(string *availability***)** *(virtual)*

void ResetTraits(IList<string> *traits***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

# Protected Methods

void AddPropertiesToMap<MoveType>(PropertyMap<MoveType> *map***)**

PropertyMap GetPropertyMap() *(virtual)*

void Load(MoveSaveData *data***)** *(virtual)*

void Save(MoveSaveData *data***)** *(virtual)*

<<<

MoveContext

PiRhoSoft.MonsterRpgEngine.MoveContext : LocalVariableStore<Move>

Description

## Constructors

MoveContext(BattleContext *battle*, **CreatureContext** *creature*, **Move** *move***)**

# Public Properties

Move *Move (read only)*

CreatureContext *Creature (read only)*

# Public Methods

VariableValue GetVariable(string *name*) *(virtual)*

SetVariableResult SetVariable(string *name*, VariableValue *value*) *(virtual)*

<<<

MoveContextList

PiRhoSoft.MonsterRpgEngine.MoveContextList : IndexedVariableStore<MoveContext>

Description

Constructors

MoveContextList()

<<<

MoveList

PiRhoSoft.MonsterRpgEngine.MoveList : IndexedVariableStore<Move>

Description

Constructors

MoveList()

# Public Methods

void Setup(Creature *creature*)

void Reset(string *availability*)

void Reset(IList<string> *traits*)

<<<

MovementDirection

PiRhoSoft.MonsterRpgEngine.MovementDirection

Description

Values

MovementDirection *None*

MovementDirection *Left*

MovementDirection *Right*

MovementDirection *Down*

MovementDirection *Up*

<<<

Mover

PiRhoSoft.MonsterRpgEngine.Mover : MonoBehaviour

Description

Static Fields

Vector2 *PositionOffset*

# Constructors

Mover()

# Public Fields

CollisionLayer *MovementLayer*

int *DirectionDelayFrames*

float *MoveSpeed*

Renderer *Shadow*

UnityAction<Vector2Int, Vector2Int> *OnTileChanged*

UnityAction<Vector2Int> *OnTileEntering*

UnityAction<Vector2Int> *OnTileExiting*

UnityAction<Vector2Int> *OnWarp*

UnityAction<MovementDirection, MovementDirection> *OnDirectionChanged*

# Public Properties

Vector2Int *TargetGridPosition (read only)*

Vector2Int *PreviousGridPosition (read only)*

Vector2Int *CurrentGridPosition (read only)*

MovementDirection *MovementDirection (read only)*

bool *Moving (read only)*

bool *DidWarp (read only)*

bool *CanInteract (read only)*

float *Speed (read only)*

Vector2Int *DirectionVector (read only)*

# Public Methods

void SkipNextUpdate()

void FaceDirection(MovementDirection *direction***)**

void UpdateMove(float *horizontal*, **float** *vertical***)**

void OccupyCurrentTiles()

void UnoccupyCurrentTiles()

void WarpToPosition(Vector2Int *position*, MovementDirection *direction*, CollisionLayer *layer*)

void Move(MovementDirection *direction*)

<<<

MoverAnimator

PiRhoSoft.MonsterRpgEngine.MoverAnimator : MonoBehaviour

Description

Constructors

MoverAnimator()

<<<

MoveSaveData

PiRhoSoft.MonsterRpgEngine.MoveSaveData

Description

Constructors

MoveSaveData()

# Public Fields

string *AbilityPath*

string *Name*

VariableList *Traits*

<<<

NeighborType

PiRhoSoft.MonsterRpgEngine.NeighborType

Description

Values

NeighborType *Any*

NeighborType *This*

NeighborType *NotThis*

NeighborType *Count*

<<<

NodeData

PiRhoSoft.MonsterRpgEngine.NodeData : ValueType

Description

## Public Fields

Vector2Int *Position*

MovementDirection *Direction*

float *Delay*

<<<

Npc

PiRhoSoft.MonsterRpgEngine.Npc : MonoBehaviour, IVariableStore

Description

Constructors

Npc()

# Public Fields

string *Name*

string *Guid*

# Public Properties

Controller *Controller (read only)*

Trainer *Trainer (read only)*

MappedVariableStore *Variables (read only)*

# Public Methods

void Load(NpcSaveData *saveData***)** *(virtual)*

void Save(NpcSaveData *saveData***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<**string**> **GetVariableNames()** *(virtual)*

# Protected Methods

void AddPropertiesToMap<**NpcType**>**(PropertyMap**<**NpcType**> *map***)**

void SetupVariables() *(virtual)*

<<<

NpcSaveData

PiRhoSoft.MonsterRpgEngine.NpcSaveData

Description

Constructors

NpcSaveData()

# Public Fields

string *Id*

Vector2Int *Position*

MovementDirection *Direction*

string *ControllerData*

VariableList *NpcTraits*

VariableList *TrainerTraits*

<<<

OccupyTiles

PiRhoSoft.MonsterRpgEngine.OccupyTiles : InstructionGraphNode

Description

Constructors

OccupyTiles()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

OrderTrainers

PiRhoSoft.MonsterRpgEngine.OrderTrainers : InstructionGraphNode

Description

Constructors

OrderTrainers()

# Public Fields

InstructionGraphNode *Next*
SortConditionList *SortConditions*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

Part

PiRhoSoft.MonsterRpgEngine.Part

Description

Constructors

Part()

# Public Fields

GameObject *GameObject*
int *OrderOffset*
SpriteRenderer *Renderer*
Rect *Bounds*

<<<

Path

PiRhoSoft.MonsterRpgEngine.Path

Description

Constructors

Path()

# Public Fields

PathType *Type*

int *RepeatCount*

bool *UseAbsolutePositioning*

bool *UsePathfinding*

bool *FindAlternateRoutes*

NodeData[] *Nodes*

<<<

PathController

PiRhoSoft.MonsterRpgEngine.PathController : Controller

Description

Constructors

PathController()

# Public Fields

bool *BeginOnAwake*

Path *Path*

# Public Methods

void StartPath()

<<<

Pathfinding

PiRhoSoft.MonsterRpgEngine.Pathfinding : MonoBehaviour

Description

Constructors

Pathfinding()

# Public Methods

void RegenerateNodes()

List<Vector2Int> **GetPath(CollisionLayer** *layer*, **Vector2Int** *fromPosition*, **Vector2Int** *toPosition*, **bool** *respectOccupied***)**

<<<

PathNode

PiRhoSoft.MonsterRpgEngine.PathNode : InstructionGraphNode

Description

Public Fields

InstructionGraphNode *Next*

bool *WaitForCompletion*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Constructors

PathNode()

# Protected Methods

Path GetPath(Mover *mover***)** *(abstract)*

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

PathState

PiRhoSoft.MonsterRpgEngine.PathState

Description

Constructors

PathState()

# Public Properties

bool *Running (read only)*

# Public Methods

void Start(Path *path*, **Mover** *mover*, **Controller** *controller***)**

void Stop()

void Load(string *saveData***)**

string Save()

<<<

PathType

PiRhoSoft.MonsterRpgEngine.PathType

Description

Values

PathType *Loop*

PathType *BackAndForth*

<<<

PlayBattleClip

PiRhoSoft.MonsterRpgEngine.PlayBattleClip : CreatureDisplayNode

Description

Constructors

PlayBattleClip()

# Public Fields

BattleAnimationClip *Clip*

# Public Properties

bool *IsExecutionImmediate (read only)*

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

PlayBattleEffect

PiRhoSoft.MonsterRpgEngine.PlayBattleEffect : CreatureDisplayNode

Description

Constructors

PlayBattleEffect()

# Public Fields

BattleAnimationEffect *Effect*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

PlayBattleSound

PiRhoSoft.MonsterRpgEngine.PlayBattleSound : CreatureDisplayNode

Description

Constructors

PlayBattleSound()

# Public Fields

BattleAnimationSound *Sound*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

Player

PiRhoSoft.MonsterRpgEngine.Player : SingletonBehaviour<Player>, IVariableStore, IVariableListener

Description

Constructors

Player()

# Public Fields

string *Name*

# Public Properties

ZoneData *Zone (read only)*

Mover *Mover (read only)*

Trainer *Trainer (read only)*

PlayerController *Controller (read only)*

IInteractable *Interaction (read only)*

MappedVariableStore *Variables (read only)*

bool *CanInteract (read only)*

bool *IsInteracting (read only)*

# Public Methods

void Interact()

void ForceInteract(Interaction *interaction***)**

void Load(PlayerSaveData *saveData***)** *(virtual)*

void Save(PlayerSaveData *saveData***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

void VariableChanged(int *index*, **VariableValue** *value***)** *(virtual)*

# Protected Methods

void Awake() *(virtual)*

void Start() *(virtual)*

void FixedUpdate() *(virtual)*

void AddPropertiesToMap<PlayerType>(PropertyMap<PlayerType> *map***)**

void SetupVariables(VariableList *savedVariables***)** *(virtual)*

void UpdateInteraction()

void OnSpawn(Vector2Int *postion***)** *(virtual)*

void OnTileEntering(Vector2Int *position***)** *(virtual)*

void OnTileExiting(Vector2Int *position***)** *(virtual)*

void OnTileChanged(Vector2Int *previous*, **Vector2Int** *current***)** *(virtual)*

<<<

PlayerController

PiRhoSoft.MonsterRpgEngine.PlayerController : Controller

Description

Constructors

PlayerController()

# Public Fields

string *HorizontalAxis*

string *VerticalAxis*

string *InteractButton*

KeyCode *InteractKey*

# Protected Fields

float *_horizontal*

float *_vertical*

bool *_interact*

# Protected Properties

bool *CanMove (read only) (virtual)*

# Protected Methods

void UpdateInput() *(virtual)*

void ClearInput() *(virtual)*

void ProcessInput() *(virtual)*

<<<

PlayerSaveData

PiRhoSoft.MonsterRpgEngine.PlayerSaveData

Description

Constructors

PlayerSaveData()

# Public Fields

string *Name*

VariableList *PlayerTraits*

VariableList *TrainerTraits*

List<**CreatureSaveData**> *Creatures*

List<**ItemSaveData**> *Items*

<<<

ProcessCreature

PiRhoSoft.MonsterRpgEngine.ProcessCreature : InstructionGraphNode, ILoopNode

Description

Constructors

ProcessCreature()

# Public Fields

InstructionGraphNode *ProcessSkill*

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

ValueTuple<InstructionGraphNode, InstructionGraphNode> GetBreakNode()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

PurchaseItem

PiRhoSoft.MonsterRpgEngine.PurchaseItem : InstructionGraphNode

Description

Constructors

PurchaseItem()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

RandomTile

PiRhoSoft.MonsterRpgEngine.RandomTile : TileBase

Description

Constructors

RandomTile()

# Public Fields

float *NoiseScale*

List<TileTransformInfo> *Tiles*

# Public Methods

void GetTileData(Vector3Int *position*, **ITilemap** *tilemap*, **TileData** *tileData (ref)***)** *(virtual)*

int GetIndex(Vector3Int *position*, **int** *length***)**

<<<

ReleaseCreature

PiRhoSoft.MonsterRpgEngine.ReleaseCreature : InstructionGraphNode

Description

Constructors

ReleaseCreature()

# Public Fields

InstructionGraphNode *ReleasedNode*

InstructionGraphNode *CantReleaseNode*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetCreature

PiRhoSoft.MonsterRpgEngine.ResetCreature : ResetTraits

Description

Constructors

ResetCreature()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetCreatureAvailability

PiRhoSoft.MonsterRpgEngine.ResetCreatureAvailability : ResetTraitsAvailability

Description

Constructors

ResetCreatureAvailability()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetMove

PiRhoSoft.MonsterRpgEngine.ResetMove : ResetTraits

Description

Constructors

ResetMove()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetMoveAvailability

PiRhoSoft.MonsterRpgEngine.ResetMoveAvailability : ResetTraitsAvailability

Description

Constructors

ResetMoveAvailability()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetTrainer

PiRhoSoft.MonsterRpgEngine.ResetTrainer : ResetTraits

Description

Constructors

ResetTrainer()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetTrainerAvailability

PiRhoSoft.MonsterRpgEngine.ResetTrainerAvailability : ResetTraitsAvailability

Description

Constructors

ResetTrainerAvailability()

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ResetTraitList

PiRhoSoft.MonsterRpgEngine.ResetTraitList : SerializedList<string>

Description

Constructors

ResetTraitList()

<<<

ResetTraits

PiRhoSoft.MonsterRpgEngine.ResetTraits : InstructionGraphNode

Description

Public Fields

InstructionGraphNode *Next*

ResetTraitList *Traits*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Constructors

ResetTraits()

<<<

ResetTraitsAvailability

PiRhoSoft.MonsterRpgEngine.ResetTraitsAvailability : InstructionGraphNode

Description

Public Fields

InstructionGraphNode *Next*

string *Availability*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Constructors

ResetTraitsAvailability()

<<<

Roster

PiRhoSoft.MonsterRpgEngine.Roster : SerializedList<CreatureReference>, IVariableStore, IIndexedVariableStore

Description

Constructors

Roster()

# Public Properties

List<**Creature**> *Creatures (read only)*

# Public Methods

void Setup()

void CreateCreatures(ITrainer *trainer***)**

void DestroyCreatures()

void ResetTraits(string *availability***)**

void ResetTraits(IList<string> *traits***)**

void AddCreature(Creature *creature***)**

void RemoveCreature(Creature *creature***)**

void TakeCreature(Creature *creature***)**

IVariableStore GetItem(int *index***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

Rule

PiRhoSoft.MonsterRpgEngine.Rule

Description

Constructors

Rule()

# Public Fields

TileTransformInfo *Tile*

bool *UseReference*

TileBase *Reference*

<<<

Rule

PiRhoSoft.MonsterRpgEngine.Rule

Description

Constructors

Rule()

# Public Fields

NeighborType[] *Neighbors*

TileTransformInfo *Tile*

bool *UseReference*

TileBase *Reference*

<<<

RuleOverrideTile

PiRhoSoft.MonsterRpgEngine.RuleOverrideTile : TileBase

Description

Constructors

RuleOverrideTile()

# Public Fields

RuleTile *OverrideTile*

Rule *DefaultRule*

List<**Rule**> *Rules*

# Public Methods

void GetTileData(Vector3Int *position*, **ITilemap** *tilemap*, **TileData** *tileData (ref)***)** *(virtual)*

bool GetTileAnimationData(Vector3Int *position*, **ITilemap** *tilemap*, **TileAnimationData** *tileAnimationData (ref)***)** *(virtual)*

void RefreshTile(Vector3Int *location*, **ITilemap** *tileMap***)** *(virtual)*

<<<

RuleTile

PiRhoSoft.MonsterRpgEngine.RuleTile : TileBase

Description

Constructors

RuleTile()

# Public Fields

Rule *DefaultRule*

List<TileBase> *NeighborTypes*

List<**Rule**> *Rules*

# Public Methods

void GetTileData(Vector3Int *position*, ITilemap *tilemap*, TileData *tileData (ref)***)** *(virtual)*

bool GetTileAnimationData(Vector3Int *position*, ITilemap *tilemap*, TileAnimationData *tileAnimationData (ref)***)** *(virtual)*

void RefreshTile(Vector3Int *location*, ITilemap *tileMap***)** *(virtual)*

int GetRuleIndex(Vector3Int *position*, ITilemap *tilemap*, TileBase *referenceTile***)**

<<<

SaveData

PiRhoSoft.MonsterRpgEngine.SaveData

Description

Constructors

SaveData()

# Public Fields

GameSaveData *Game*

WorldSaveData *World*

PlayerSaveData *Player*

<<<

SaveGame

PiRhoSoft.MonsterRpgEngine.SaveGame : InstructionGraphNode

Description

Constructors

SaveGame()

# Public Fields

InstructionGraphNode *Next*

GameObject *SavingIndicator*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

SaveInformation

PiRhoSoft.MonsterRpgEngine.SaveInformation

Description

Constructors

SaveInformation()

# Public Fields

Action *OnComplete*

Action<SaveState, SaveState> *OnProgress*

Action<string> *OnError*

# Public Properties

SaveState *State (read only)*

float *Progress (read only)*

string *Message (read only)*

# Public Methods

void UpdateProgress(SaveState *state*, **float** *progress***)**

void SetError(string *message***)**

void SetComplete()

<<<

SaveState

PiRhoSoft.MonsterRpgEngine.SaveState

Description

Values

SaveState *SavingWorld*

SaveState *WritingData*

SaveState *Error*

SaveState *Complete*

<<<

SetCreature

PiRhoSoft.MonsterRpgEngine.SetCreature : CreatureDisplayNode

Description

Constructors

SetCreature()

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

Shop

PiRhoSoft.MonsterRpgEngine.Shop : MonoBehaviour, IVariableStore

Description

Constructors

Shop()

# Public Fields

ShopItemList *Items*

# Public Methods

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

ShopItem

PiRhoSoft.MonsterRpgEngine.ShopItem : IVariableStore

Description

Constructors

ShopItem()

# Public Fields

Item *Item*

int *Count*

# Public Properties

int *PurchaseCost (read only)*

int *SellCost (read only)*

# Public Methods

bool CanPurchase(Inventory *inventory***)**

void Purchase(Inventory *inventory***)**

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

ShopItemList

PiRhoSoft.MonsterRpgEngine.ShopItemList : IndexedVariableStore<ShopItem>

Description

Constructors

ShopItemList()

<<<

Skill

PiRhoSoft.MonsterRpgEngine.Skill

Description

Constructors

Skill()

# Public Fields

string *Name*

int *LearnLimit*

Expression *Condition*

InstructionCaller *Instruction*

<<<

SkillList

PiRhoSoft.MonsterRpgEngine.SkillList : SerializedList<Skill>

Description

Constructors

SkillList()

# Public Methods

Skill Find(string *name*)

<<<

SkillsDictionary

PiRhoSoft.MonsterRpgEngine.SkillsDictionary : SerializedDictionary<string, string>

Description

Constructors

SkillsDictionary()

<<<

SortConditionList

PiRhoSoft.MonsterRpgEngine.SortConditionList : SerializedList<VariableReference>

Description

Constructors

SortConditionList()

<<<

SpawnPoint

PiRhoSoft.MonsterRpgEngine.SpawnPoint : ValueType

Description

Static Fields

SpawnPoint *Default*

# Public Fields

Vector2Int *Position*

string *Name*

Transition *Transition*

MovementDirection *Direction*

CollisionLayer *Layer*

bool *Move*

# Public Properties

bool *IsNamed (read only)*

<<<

Species

PiRhoSoft.MonsterRpgEngine.Species : Resource, IReloadable, IVariableStore

Description

Constructors

Species()

# Public Fields

Ecosystem *Ecosystem*

string *Name*

Sprite *Icon*

AnimatorOverrideController *Animations*

VariableList *Traits*

SkillList *Skills*

MountPointDictionary *MountPoints*

# Public Methods

void OnEnable() *(virtual)*

void OnDisable() *(virtual)*

Creature **CreateCreature(**ITrainer *trainer***)** *(virtual)*

List<**int**> **GetTriggeredTraits(int** *index***)**

List<**int**> **GetTriggeredSkills(int** *index***)**

VariableValue **GetVariable(string** *name***)** *(virtual)*

SetVariableResult **SetVariable(string** *name*, VariableValue *value***)** *(virtual)*

IEnumerable<**string**> **GetVariableNames()** *(virtual)*

# Protected Methods

void AddPropertiesToMap<SpeciesType>(PropertyMap<**SpeciesType**> *map***)**

PropertyMap **GetPropertyMap()** *(virtual)*

<<<

SpinnerController

PiRhoSoft.MonsterRpgEngine.SpinnerController : Controller

Description

Constructors

SpinnerController()

# Public Fields

SpinType *Type*

float *SpinDelay*

<<<

SpinType

PiRhoSoft.MonsterRpgEngine.SpinType

Description

Values

SpinType *Clockwise*

SpinType *CounterClockwise*

SpinType *Random*

<<<

StartPath

PiRhoSoft.MonsterRpgEngine.StartPath : PathNode

Description

Constructors

StartPath()

# Protected Methods

Path **GetPath(**Mover *mover***)** *(virtual)*

<<<

StaticCollider

PiRhoSoft.MonsterRpgEngine.StaticCollider : MonoBehaviour

Description

Constructors

StaticCollider()

# Public Fields

CollisionLayer *CollisionLayer*

# Public Methods

void OccupyCurrentTiles()

void UnoccupyCurrentTiles()

<<<

TeachAbility

PiRhoSoft.MonsterRpgEngine.TeachAbility : InstructionGraphNode

Description

Constructors

TeachAbility()

# Public Fields

InstructionGraphNode *Next*

AbilityVariableSource *Ability*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Public Methods

void GetInputs(List<VariableDefinition> *inputs***)** *(virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

TeachSkill

PiRhoSoft.MonsterRpgEngine.TeachSkill : InstructionGraphNode

Description

Constructors

TeachSkill()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

ThawController

PiRhoSoft.MonsterRpgEngine.ThawController : InstructionGraphNode

Description

Constructors

ThawController()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

TileDictionary

PiRhoSoft.MonsterRpgEngine.TileDictionary : SerializedDictionary<Vector2Int, Vector2Int>

Description

Constructors

TileDictionary()

<<<

TileInfo

PiRhoSoft.MonsterRpgEngine.TileInfo

Description

Constructors

TileInfo()

# Public Fields

Vector2Int *Position*

CollisionLayer *CollisionLayer*

CollisionLayer *LayerChange*

bool *HasSpawnPoint*

SpawnPoint *SpawnPoint*

bool *HasZoneTrigger*

ZoneTrigger *Zone*

bool *HasEncounter*

Encounter *Encounter*

bool *HasInstructions*

InstructionTrigger *Instructions*

bool *HasStairs*

int *Slope*

bool *HasOffset*

Vector2 *Offset*

bool *HasEdge*

MovementDirection *EdgeDirection*

# Public Methods

bool HasCollision(CollisionLayer *layer***)**

bool IsEdge(MovementDirection *direction***)**

bool IsJumpable(MovementDirection *direction***)**

bool IsEmpty()

bool IsSameCollisionLayer(TileInfo *other***)**

bool IsSameCollisionLayerIncrement(TileInfo *other***)**

bool IsSameZoneAs(TileInfo *other***)**

bool IsSameEncounterAs(TileInfo *other***)**

bool IsSameInstructionAs(TileInfo *other***)**

bool IsSameStairsAs(TileInfo *other***)**

bool IsSameOffsetAs(TileInfo *other***)**

bool IsSameEdgeAs(TileInfo *other***)**

<<<

TileTransformInfo

PiRhoSoft.MonsterRpgEngine.TileTransformInfo : ValueType

Description

Static Fields

int[] *Rotations*

# Public Fields

Sprite *Sprite*

int *Rotation*

bool *FlipHorizontal*

bool *FlipVertical*

# Public Methods

Matrix4x4 GetTransform()

<<<

TossItem

PiRhoSoft.MonsterRpgEngine.TossItem : InstructionGraphNode

Description

Constructors

TossItem()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

Trainer

PiRhoSoft.MonsterRpgEngine.Trainer : MonoBehaviour, IReloadable, IVariableStore, ITrainer

Description

Constructors

Trainer()

# Public Fields

Ecosystem *Ecosystem*

VariableList *Traits*

Inventory *Inventory*

Roster *_roster*

# Public Properties

string *Name (read only)*

Roster *Roster (read only) (virtual)*

BattleAi *Ai (read only) (virtual)*

bool *IsPlayer (read only) (virtual)*

# Public Methods

void OnEnable() *(virtual)*

void OnDisable() *(virtual)*

void ResetTraits(string *availability***)** *(virtual)*

void ResetTraits(IList<string> *traits***)** *(virtual)*

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

# Protected Methods

void AddPropertiesToMap<TrainerType>(PropertyMap<TrainerType> *map***)**

PropertyMap GetPropertyMap() *(virtual)*

<<<

TrainerContext

PiRhoSoft.MonsterRpgEngine.TrainerContext : LocalVariableStore<ITrainer>

Description

Constructors

TrainerContext(BattleContext *battle*, ITrainer *trainer*)

# Public Properties

ITrainer *Trainer (read only)*

CreatureContextList *Creatures (read only)*

# Public Methods

VariableValue **GetVariable(string** *name***)** *(virtual)*

SetVariableResult **SetVariable(string** *name*, VariableValue *value***)** *(virtual)*

<<<

TrainerContextList

PiRhoSoft.MonsterRpgEngine.TrainerContextList : IndexedVariableStore<TrainerContext>

Description

Constructors

TrainerContextList()

<<<

TrainerReferenceList

PiRhoSoft.MonsterRpgEngine.TrainerReferenceList : SerializedList<VariableReference>

Description

Constructors

TrainerReferenceList()

<<<

TypewriterControl

PiRhoSoft.MonsterRpgEngine.TypewriterControl : MessageControl

Description

Constructors

TypewriterControl()

# Public Fields

float *CharactersPerSecond*

# Protected Methods

void Setup() *(virtual)*

IEnumerator Run(string *text*, **MessageInteractionType** *interaction*, **bool** *isLast***)** *(virtual)*

<<<

UnoccupyTiles

PiRhoSoft.MonsterRpgEngine.UnoccupyTiles : InstructionGraphNode

Description

Constructors

UnoccupyTiles()

# Public Fields

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

UpdateCreature

PiRhoSoft.MonsterRpgEngine.UpdateCreature : CreatureDisplayNode

Description

Constructors

UpdateCreature()

# Public Fields

string *Group*

bool *WaitForCompletion*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

UseItem

PiRhoSoft.MonsterRpgEngine.UseItem : InstructionGraphNode

Description

Constructors

UseItem()

# Public Fields

ItemUseLocation *Location*

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

UseMove

PiRhoSoft.MonsterRpgEngine.UseMove : InstructionGraphNode

Description

Constructors

UseMove()

# Public Fields

AbilityUseLocation *Location*

InstructionGraphNode *Next*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator **Run_(InstructionGraph** *graph*, **InstructionStore** *variables*, **int** *iteration*) *(virtual)*

<<<

VariableState

PiRhoSoft.MonsterRpgEngine.VariableState

Description

Values

VariableState *Awake*

VariableState *Enabled*

VariableState *Changed*

<<<

WarpMover

PiRhoSoft.MonsterRpgEngine.WarpMover : InstructionGraphNode

Description

Constructors

WarpMover()

# Public Fields

InstructionGraphNode *Next*

Vector2Int *Position*

MovementDirection *Direction*

CollisionLayer *Layer*

# Public Properties

Color *NodeColor (read only) (virtual)*

# Protected Methods

IEnumerator Run_(InstructionGraph *graph*, **InstructionStore** *variables*, **int** *iteration***)** *(virtual)*

<<<

World

PiRhoSoft.MonsterRpgEngine.World : ScriptableObject

Description

Constructors

World()

# Public Fields

SceneReference *MainScene*

List<SceneReference> *UiScenes*

Transition *DefaultZoneTransition*

Transition *DefaultSpawnTransition*

AudioClip *BackgroundMusic*

VariableSchema *WorldSchema*

VariableSchema *PlayerSchema*

VariableSchema *NpcSchema*

MapLayerList *MapLayers*

List<Zone> *Zones*

# Public Methods

Zone GetZoneByName(string *name***)**

Zone GetZoneBySceneIndex(int *index***)**

<<<

WorldListener

PiRhoSoft.MonsterRpgEngine.WorldListener : MonoBehaviour, IVariableStore

Description

Constructors

WorldListener()

# Public Fields

WorldListenerSource *Source*

string *Variable*

InstructionCaller *Instructions*

# Public Properties

VariableState *State (read only)*

# Public Methods

void OnVariableChanged(WorldListenerSource *source*, **string** *variable***)**

VariableValue GetVariable(string *name***)** *(virtual)*

SetVariableResult SetVariable(string *name*, **VariableValue** *value***)** *(virtual)*

IEnumerable<string> GetVariableNames() *(virtual)*

<<<

WorldListenerGraph

PiRhoSoft.MonsterRpgEngine.WorldListenerGraph : InstructionGraph

Description

Constructors

WorldListenerGraph()

# Public Fields

InstructionGraphNode *OnAwake*

InstructionGraphNode *OnEnabled*

InstructionGraphNode *OnVariableChanged*

# Protected Methods

IEnumerator Run(InstructionStore *variables***)** *(virtual)*

<<<

WorldListenerSource

PiRhoSoft.MonsterRpgEngine.WorldListenerSource

Description

Values

WorldListenerSource *Zone*

WorldListenerSource *World*

WorldListenerSource *Player*

<<<

WorldLoader

PiRhoSoft.MonsterRpgEngine.WorldLoader : GlobalBehaviour<WorldLoader>

Description

Static Methods

LoadInformation **New(string** *startingZone*, **string** *playerSpawn***)**

LoadInformation **Load(string** *filename***)**

LoadInformation **Load(**GameSaveData *game*, **string** *filename***)**

SaveInformation **Save(**WorldManager *world***)**

# Constructors

WorldLoader()

<<<

WorldManager

PiRhoSoft.MonsterRpgEngine.WorldManager : SingletonBehaviour<WorldManager>, IVariableStore, IVariableListener

Description

## Constructors

WorldManager()

# Public Fields

World *World*

# Public Properties

Interface *Interface (read only)*

ZoneData**[]** *Zones (read only)*

List<ZoneData> *LoadedZones (read only)*

string *SaveFilename (read only)*

MappedVariableStore *Variables (read only)*

InstructionContext *Context (read only)*

bool *IsFrozen (read only)*

bool *IsTransitioning (read only)*

# Public Methods

ZoneData **GetZone(**Zone *zone***)**

ZoneData **GetZone(**Object *o***)**

void Load(string *filename*, WorldSaveData *saveData***)** *(virtual)*

string Save(WorldSaveData *saveData***)** *(virtual)*

void Freeze()

void Thaw()

TileInfo **FindTile(**Vector2Int *position***)**

bool IsOccupied(Vector2Int *position*, CollisionLayer *layer***)**

void SetOccupied(Vector2Int *position*, CollisionLayer *layer***)**

void SetUnoccupied(Vector2Int *position*, CollisionLayer *layer***)**

IInteractable **GetInteraction(**Vector2Int *position***)**

void AddInteraction(Vector2Int *position*, Interaction *interaction***)**

void RemoveInteraction(Vector2Int *position*, Interaction *interaction***)**

IEnumerator **LoadUi()**

void ChangeZone(Zone *zone***)**

void TransitionZone(Zone *zone*, SpawnPoint *spawnPoint*, Transition *transition***)**

VariableValue **GetVariable(string** *name***)** *(virtual)*

SetVariableResult **SetVariable(string** *name*, VariableValue *value***)** *(virtual)*

IEnumerable<**string**> **GetVariableNames()** *(virtual)*

void VariableChanged(int *index*, VariableValue *value***)** *(virtual)*

# Protected Methods

void Awake() *(virtual)*

void OnDestroy() *(virtual)*

void AddPropertiesToMap<**WorldManagerType**>**(**PropertyMap<**WorldManagerType**> *map***)**

void SetupVariables() *(virtual)*

void TeardownVariables() *(virtual)*

<<<

WorldSaveData

PiRhoSoft.MonsterRpgEngine.WorldSaveData

Description

Constructors

WorldSaveData()

# Public Fields

VariableList *PersistentVariables*

List<ZoneSaveData> *Zones*

<<<

Zone

PiRhoSoft.MonsterRpgEngine.Zone : ScriptableObject

Description

Static Fields

string *ZoneLoadedAvailability*

string *ZoneActiveAvailability*

# Constructors

Zone()

# Public Fields

World *World*

string *Name*

SceneReference *Scene*

string *MapLayer*

AudioClip *BackgroundMusic*

VariableSchema *Schema*

InstructionCaller *EnterInstructions*

InstructionCaller *ExitInstructions*

<<<

ZoneData

PiRhoSoft.MonsterRpgEngine.ZoneData : ScriptableObject, IVariableStore, IVariableListener

Description

## Constructors

ZoneData()

# Public Fields

Zone *Zone*

int *SceneIndex*

# Public Properties

ZoneState *State (read only)*

bool *IsActive (read only)*

bool *IsEnabled (read only)*

WorldManager *World (read only)*

MapProperties *Properties (read only)*

Pathfinding *Pathfinding (read only)*

List<**int**> *Connections (read only)*

Dictionary<**string, string**> *SpawnPoints (read only)*

List<WorldListener> *Listeners (read only)*

List<Npc> *Npcs (read only)*

MappedVariableStore *Variables (read only)*

# Public Methods

SpawnPoint **GetSpawnPoint(string** *name***)**

void Load(ZoneSaveData *saveData***)** *(virtual)*

void Save(ZoneSaveData *saveData***)** *(virtual)*

VariableValue **GetVariable(string** *name***)** *(virtual)*

SetVariableResult **SetVariable(string** *name*, VariableValue *value***)** *(virtual)*

IEnumerable<**string**> **GetVariableNames()** *(virtual)*

void VariableChanged(int *index*, VariableValue *value***)** *(virtual)*

void VariableChanged(WorldListenerSource *source*, **string** *variable***)**

# Protected Methods

void AddPropertiesToMap<**ZoneDataType**>**(**PropertyMap<**ZoneDataType**> *map***)**

void SetupVariables() *(virtual)*

<<<

ZoneLoadStatus

PiRhoSoft.MonsterRpgEngine.ZoneLoadStatus

Description

Constructors

ZoneLoadStatus()

# Public Fields

bool *IsDone*

<<<

ZoneSaveData

PiRhoSoft.MonsterRpgEngine.ZoneSaveData

Description

Constructors

ZoneSaveData()

# Public Fields

string *Name*

VariableList *PersistentVariables*

List<**NpcSaveData**> *Npcs*

<<<

ZoneState

PiRhoSoft.MonsterRpgEngine.ZoneState

Description

Values

ZoneState *Unloaded*

ZoneState *Loading*

ZoneState *Loaded*

ZoneState *Unloading*

<<<

ZoneTrigger

PiRhoSoft.MonsterRpgEngine.ZoneTrigger

Description

Constructors

ZoneTrigger()

# Public Fields

Zone *TargetZone*

string *TargetSpawn*

bool *HasTransition*

Transition *Transition*

# Public Methods

void Enter()

void Exit()