SimpleMan.Utilities Download

Standard utilities for any project on Unity engine.

Authors: Igor-Valerii Chebotar, Alexey Naumenko

Bool extensions

Function Name	Description
Mirror	Returns the reflected boolean value.
MirrorRef	Modifies the boolean value by reflecting it.

Int extensions

FunctionName	Description
Mirror	Returns the negative value of the input integer. Ex. 1 -> -1
MirrorRef	Modifies the input integer by setting it to its negative value.
Clamp	Clamps the input integer between specified minimum and maximum float values and returns the clamped integer.
ClampRef	Modifies the input integer by clamping it between specified minimum and maximum float values.
Clamp (Vector2)	Clamps the input integer between a range defined by a Vector2 (as float values) and returns the clamped integer.
ClampRef (Vector2)	Modifies the input integer by clamping it between a range defined by a Vector2 (as float values).
Clamp (FloatRange)	Clamps the input integer between a range defined by a FloatRange (as float values) and returns the clamped integer.
ClampRef (FloatRange)	Modifies the input integer by clamping it between a range defined by a FloatRange (as float values).
ClampPositive	Clamps the input integer to be 0 or greater.
ClampPositiveRef	Modifies the input integer by clamping it to be 0 or greater.

FunctionName	Description
ClampNegative	Clamps the input integer to be less than or equal to 0.
ClampNegativeRef	Modifies the input integer by clamping it to be less than or equal to 0.
SetMin	Sets the minimum value for the input integer, clamping it to be greater than or equal to the specified minimum.
SetMinRef	Modifies the input integer by setting its minimum value and clamping it to be greater than or equal to the specified minimum.
SetMax	Sets the maximum value for the input integer, clamping it to be less than or equal to the specified maximum.
SetMaxRef	Modifies the input integer by setting its maximum value and clamping it to be less than or equal to the specified maximum.
InRange	Checks if the input integer is within the specified range (inclusive).
InRange (Vector2)	Checks if the input integer is within the range specified by a Vector2 (inclusive).
InRange (FloatRange)	Checks if the input integer is within the range specified by a FloatRange (inclusive).
OutOfRange	Checks if the input integer is outside the specified range (exclusive).
OutOfRange (Vector2)	Checks if the input integer is outside the range specified by a Vector2 (exclusive).
OutOfRange (FloatRange)	Checks if the input integer is outside the range specified by a FloatRange (exclusive).
Abs	Returns the absolute value of the input integer.
AbsRef	Modifies the input integer by setting it to its absolute value.

Float extensions

FunctionName	Description
Mirror	Returns the negative value of the input float. Ex. 1 -> -1
MirrorRef	Modifies the input float by setting it to its negative value.

FunctionName	Description
Clamp	Clamps the input float between specified minimum and maximum values, returning the clamped value.
ClampRef	Modifies the input float by clamping it between specified minimum and maximum values.
Clamp (Vector2)	Clamps the input float between a range defined by a Vector2, returning the clamped value.
ClampRef (Vector2)	Modifies the input float by clamping it between a range defined by a Vector2.
Clamp (FloatRange)	Clamps the input float between a range defined by a FloatRange, returning the clamped value.
ClampRef (FloatRange)	Modifies the input float by clamping it between a range defined by a FloatRange.
ClampPositive	Clamps the input float to be 0 or greater.
ClampPositiveRef	Modifies the input float by clamping it to be 0 or greater.
ClampNegative	Clamps the input float to be less than or equal to 0.
ClampNegativeRef	Modifies the input float by clamping it to be less than or equal to 0.
SetMin	Sets the minimum value for the input float, clamping it to be greater than or equal to the specified minimum.
SetMinRef	Modifies the input float by setting its minimum value and clamping it to be greater than or equal to the specified minimum.
SetMax	Sets the maximum value for the input float, clamping it to be less than or equal to the specified maximum.
SetMaxRef	Modifies the input float by setting its maximum value and clamping it to be less than or equal to the specified maximum.
Clamp01	Clamps the input float between 0 and 1.
Clamp01Ref	Modifies the input float by clamping it between 0 and 1.
ClampAxis	Clamps the input float between -1 and 1.
ClampAxisRef	Modifies the input float by clamping it between -1 and 1.

FunctionName	Description
NormalizeAngle	Normalizes an input angle in degrees to the range of -180 to 180 degrees while preserving its direction.
NormalizeAngleRef	Modifies the input angle by normalizing it to the range of -180 to 180 degrees while preserving its direction.
ClampAngle	Clamps an input angle between specified minimum and maximum angles, considering their mid-angle.
ClampAngleRef	Modifies the input angle by clamping it between specified minimum and maximum angles, considering their mid-angle.
Abs	Returns the absolute value of the input float.
AbsRef	Modifies the input float by setting it to its absolute value.
Round	Rounds the input float to the nearest integer.
RoundRef	Modifies the input float by rounding it to the nearest integer.
RoundToInt	Rounds the input float to the nearest integer and returns it as an integer.
Floor	Returns the largest integer smaller than or equal to the input float.
FloorRef	Modifies the input float by rounding it down to the largest integer smaller than or equal to it.
FloorToInt	Rounds the input float down to the largest integer smaller than or equal to it and returns it as an integer.
Ceil	Returns the smallest integer greater than or equal to the input float.
CeilRef	Modifies the input float by rounding it up to the smallest integer greater than or equal to it.
CeilToInt	Rounds the input float up to the smallest integer greater than or equal to it and returns it as an integer.
InRange	Checks if the input float is within the specified range (inclusive).
InRange (Vector2)	Checks if the input float is within the range specified by a Vector2 (inclusive).
InRange (FloatRange)	Checks if the input float is within the range specified by a FloatRange (inclusive).

FunctionName	Description
OutOfRange	Checks if the input float is outside the specified range (exclusive).
OutOfRange (Vector2)	Checks if the input float is outside the range specified by a Vector2 (exclusive).
OutOfRange (FloatRange)	Checks if the input float is outside the range specified by a FloatRange (exclusive).

String extensions

FunctionName	Description
FirstCharToUpper	Capitalizes the first character of the input string.
ToSplitPascalCase	Converts a PascalCase string to space-separated words. Example: SadButTrue -> Sad But True
WithoutSpaces	Removes spaces from a string and capitalizes letters following spaces. Example: Sad But True -> SadButTrue
WithUnderscorePrefix	Adds a prefix with an underscore to the input string. Example: ObjectName -> Prefix_ObjectName
WithoutUnderscorePrefix	Removes a prefix with an underscore from the input string. Example: Prefix_ObjectName -> ObjectName
WithSquarePrefix	Adds a square bracket enclosed prefix to the input string. Example: ObjectName -> [Prefix]ObjectName
WithoutSquarePrefix	Removes a square bracket enclosed prefix from the input string. Example: [Prefix]ObjectName -> ObjectName
ExctractFileNameFromPath	Extracts the file name from a file path.
RemoveClone	Removes the "(Clone)" suffix from a string if present.
Bold	Wraps the input string in HTML tags to display it as bold in Unity console.
Italic	Wraps the input string in HTML tags to display it as italic in Unity console.
SetColor	Wraps the input string in HTML tags to color it in Unity console using a specified color code.

FunctionName	Description
SetColorRed	Wraps the input string in HTML tags to color it in red in Unity console.
SetColorGreen	Wraps the input string in HTML tags to color it in green in Unity console.
SetColorBlue	Wraps the input string in HTML tags to color it in blue in Unity console.
SetColorYellow	Wraps the input string in HTML tags to color it in yellow in Unity console.
SetColorGray	Wraps the input string in HTML tags to color it in gray in Unity console.
SetColorBlack	Wraps the input string in HTML tags to color it in black in Unity console.
SetColorWhite	Wraps the input string in HTML tags to color it in white in Unity console.

Collection extensions

FunctionName	Description
IsEmpty	Returns true if the count of elements in the collection is zero.
IsNotEmpty	Returns true if the count of elements in the collection is greater than zero.
IsEmpty	Returns true if the count of elements in the array is zero.
IsNotEmpty	Returns true if the count of elements in the array is greater than zero.
IsEmpty<TKey, TValue>	Returns true if the count of elements in the collection is zero.
Random	Returns a random element from the collection.
ForEach	Applies the specified action to each element in the collection.
ResetAllElements	Sets each element in the array to its default value.
Except	Returns a collection without the target element.
Validate	Returns a collection without null elements.

FunctionName	Description
AssertNoNullElements	Throws an exception if the collection contains null elements.
GetElementIndexByKey<TKey, TValue>	Returns the index of a key in the Dictionary.
AddUnique	Adds an item to the list if it is not already present.
AddUnique	Adds items to the list if they are not already present.
AddUnique	Adds an item to the queue if it is not already present.
AddUnique	Adds an item to the stack if it is not already present.
AddUnique<TKey, TValue>	Adds a key-value pair to the dictionary if the key is not already present.
IfContainsKey<TKey, TValue>	Executes an action if the dictionary contains a specified key.
AddAndWrap	Adds a value to the array and wraps elements when the array is full.
AddAndWrap	Adds a value to the array and wraps elements when the array is full, using a specified null element.
AddAndWrap	Adds a value to the list and wraps elements when the list is full.

Vector extensions

FunctionName	Description
XY2XZ(Vector2Int)	Converts a 2D vector (x, y) to a 3D vector (x, 0, z).
XY2XZ(Vector2)	Converts a 2D vector (x, y) to a 3D vector (x, 0, z).
XZ2XY(Vector3)	Converts a 3D vector (x, y, z) to a 2D vector (x, z).
XZ2XY(Vector3Int)	Converts a 3D vector (x, y, z) to a 2D vector (x, z).
ToFloatVector(Vector2Int)	Converts a 2D integer vector to a 2D float vector.
ToFloatVector(Vector3Int)	Converts a 3D integer vector to a 3D float vector.
ToIntVector(Vector2)	Converts a 2D float vector to a 2D integer vector.
ToIntVector(Vector3)	Converts a 3D float vector to a 3D integer vector.

FunctionName	Description
Normalized(Vector2Int)	Returns a 2D integer vector that is normalized to have components clamped between -1 and 1.
Normalized(Vector3Int)	Returns a 3D integer vector that is normalized to have components clamped between -1 and 1.
NormalizeRef(Vector2Int)	Normalizes a 2D integer vector in-place to have components clamped between -1 and 1.
NormalizeRef(Vector3Int)	Normalizes a 3D integer vector in-place to have components clamped between -1 and 1.

Quaternion extensions

FunctionName	Description
ExtractVector	Transforms a direction vector using the provided quaternion rotation.
ExtractForwardVector	Transforms the forward vector (Vector3.forward) using the quaternion rotation.
ExtractRightVector	Transforms the right vector (Vector3.right) using the quaternion rotation.

Transform extensions

Function name	Description
IsDirectChildOf	Retruns true if specified object is direct child of this object
IsDirectParentOf	Retruns true if specified object is direct parent of this object
GetDirectChildren	Get only direct children for this transform
GetDirectChildrenOfType	Returns array of direct children that have certain component
DestroyChildren	Destroy all children of current transform
DestroyChildrenImmediate	Destroy all children of current transform (for editor mode)
SetPositionAndRotation	Set position and rotation using *PositionAndRotation* structure
SetLocalPositionAndRotation	Set local position and rotation using *PositionAndRotation* structure

Function name	Description
GetPositionAndRotation	Returns *PositionAndRotation* structure
GetLocalPositionAndRotation	Returns *PositionAndRotation* structure

C# Examples

```
//Destroy all children
transform.DestroyChildren();
```

```
//Get direct children array
Transform[] children = transform.GetDirectChildren();
```

```
//Get only direct children with 'Health' component
Health[] children = transform.GetDirectChildrenOfType<Health>();
```

Component extensions

Function name	Description
TryGetComponentInChildren	Return true if at least one child of this object have certain component.
TryGetComponentInParent	Return true if at least one parent of this object have certain component.
StopCoroutineIfExist	You don't need to check if your coroutine is null using this extension.

C# Examples

```
//Throws exception if parent object don't have 'Animator' component
if(TryGetComponentInParent<Animator>(out Animator animator) == false)
    throw new NullReferenceException("Animator was not found in chilren");
```

Object extensions

Function name	Description
Exist	Returns true if object is not null (Supports UnityEngine.Object null checking).
NotExist	Returns true if object is null (Supports UnityEngine.Object null checking).

C# Examples

```
//Before
if(target != null)
{
    ...
}

//After
//Supports also UnityEngine.Object and their special null checking
if(target.Exist())
{
    ...
}
```

Mathematics

Function name	Description
GetClosest	Return closest Transform or other component to target point
GetPointsOnCircle	Return array with points positions
GetPointOnCircle	Return point position on circle by angle

C# Examples

```
//Get items around
IInteractable[] items = GetAvailableItems();

//Get closest interactable item to player
Vector3 playerPosition = transform.position;
IInteractable closestItem = Mathematics.GetClosest(playerPosition, items);
```

```
//Get angle between horizontal axes and mouse input
float inputAngle = Vector2.Angle(inputAxes, Vector2.right)
```

```
//Set top down crosshair position
_crosshair.transform.position = Mathematics.GetPointOnCircle(
        transform.position,
        _crosshairDistance,
        inputAngle).
        XY2XZ();
}
```

Project and inspector window blocker

You can block the inspector and project windows using CTRL+SPACE and CTRL+SHIFT+SPACE hot keys

Execute once system

Gives ability to execute method only one time per frame, no matter how many calls was received.

Function name	Description
ExecuteOncePerFrame	Execute target method. Ignore other execution calls for this method in current frame.

C# Examples

```
//Will be called once
ExecuteOnceSystem.ExecuteOncePerFrame(DoAction)
ExecuteOnceSystem.ExecuteOncePerFrame(DoAction)
ExecuteOnceSystem.ExecuteOncePerFrame(DoAction)
```

```
//Works also with parameters
ExecuteOnceSystem.ExecuteOncePerFrame(( ) => DoAction("Hello"))
```