Namespace PAC

Classes

Config

Config.Colours

Config.Tools

Finder

Functions

ImageConverter

KeyCodeFunctions

Preferences

Preferences.Preference<T>

Tex2DSprite

Structs

OutlineSideFill

Enums

FlipDirection

RotationAngle

Class Config

Namespace: PAC

```
public static class Config
```

Inheritance

object ← Config

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Class Config.Colours

Namespace: PAC

```
public static class Config.Colours
```

Inheritance

object ← Config.Colours

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

brushMask

```
public static readonly Color brushMask
```

Field Value

Color

transparent

```
public static readonly Color transparent
```

Field Value

Color

Class Config.Tools

Namespace: PAC

```
public static class Config.Tools
```

Inheritance

object ← Config.Tools

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

maxBrushSize

```
public const int maxBrushSize = 25
```

Field Value

int

minBrushSize

```
public const int minBrushSize = 1
```

Field Value

int

Class Finder

Namespace: PAC

```
public class Finder
```

Inheritance

object ← Finder

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Properties

animationManager

```
public static AnimationManager animationManager { get; }
```

Property Value

AnimationManager

clipboard

```
public static Clipboard clipboard { get; }
```

Property Value

Clipboard

colourPicker

```
public static GlobalColourPicker colourPicker { get; }
```

Property Value

GlobalColourPicker

dialogBoxManager

```
public static DialogBoxManager dialogBoxManager { get; }
```

Property Value

DialogBoxManager

drawingArea

```
public static DrawingArea drawingArea { get; }
```

Property Value

DrawingArea

fileManager

```
public static FileManager fileManager { get; }
```

Property Value

FileManager

fileTabsManager

```
public static FileTabsManager fileTabsManager { get; }
```

Property Value

FileTabsManager

gridManager

```
public static GridManager gridManager { get; }
```

Property Value

GridManager

imageEditManager

```
public static ImageEditManager imageEditManager { get; }
```

Property Value

ImageEditManager

inputSystem

```
public static InputSystem inputSystem { get; }
```

Property Value

InputSystem

keyboard

```
public static Keyboard keyboard { get; }
```

Property Value

Keyboard

layerManager

```
public static LayerManager layerManager { get; }
```

Property Value

LayerManager

mouse

```
public static Mouse mouse { get; }
```

Property Value

Mouse

themeManager

```
public static ThemeManager themeManager { get; }
```

Property Value

ThemeManager

tileOutlineManager

```
public static TileOutlineManager tileOutlineManager { get; }
```

Property Value

TileOutlineManager

tilesetManager

```
public static TilesetManager tilesetManager { get; }
```

Property Value

TilesetManager

toolbar

```
public static Toolbar toolbar { get; }
```

Property Value

Toolbar

uiManager

```
public static UIManager uiManager { get; }
```

Property Value

UIManager

undoRedoManager

```
public static UndoRedoManager undoRedoManager { get; }
```

Property Value

UndoRedoManager

Enum FlipDirection

Namespace: PAC

```
public enum FlipDirection
```

Fields

`None = 0`

`X = 1`

`Y = 2`

Class Functions

Namespace: PAC

```
public static class Functions
```

Inheritance

object ← Functions

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

ArrayToString<T>(T[])

```
public static string ArrayToString<T>(T[] array)
```

Parameters

`array` T[]

Returns

string

Type Parameters

`T`

CeilToMultiple(float, int)

```
public static int CeilToMultiple(float toRound, int multipleOf)
```

Parameters

`toRound` float

`multipleOf` int

Returns

int

CeilToMultiple(float, float)

```
public static float CeilToMultiple(float toRound, float multipleOf)
```

Parameters

`toRound` float

`multipleOf` float

Returns

float

CompareArrays<T>(T[], T[])

```
public static bool CompareArrays<T>(T[] array1, T[] array2)
```

Parameters

`array1` T[]

`array2` T[]

Returns

bool

Type Parameters

T

ConcatArrays<T>(T[], T[])

```
public static T[] ConcatArrays<T>(T[] array1, T[] array2)
```

Parameters

array1 T[]

array2 T[]

Returns

T[]

Type Parameters

T

CopyArray<T>(T[])

```
public static T[] CopyArray<T>(T[] array)
```

Parameters

array T[]

Returns

T[]

Type Parameters

T

FirstNChars(string, int)

```
public static string FirstNChars(string str, int numOfChars)
```

Parameters

`str` string

`numOfChars` int

Returns

string

FloorToMultiple(float, int)

```
public static int FloorToMultiple(float toRound, int multipleOf)
```

Parameters

`toRound` float

`multipleOf` int

Returns

int

FloorToMultiple(float, float)

```
public static float FloorToMultiple(float toRound, float multipleOf)
```

Parameters

`toRound` float

`multipleOf` float

Returns

float

Mod(int, int)

Returns a mod b, giving a non-negative result.

```
public static int Mod(int a, int b)
```

Parameters

`a` int

`b` int

Returns

int

Range(int, int)

```
public static int[] Range(int start, int end)
```

Parameters

`start` int

`end` int

Returns

int[]

RoundDecimalPlaces(float, int)

```
public static float RoundDecimalPlaces(float f, int decimalPlaces)
```

Parameters

`f` float

`decimalPlaces` int

Returns

float

RoundToMultiple(float, int)

```
public static int RoundToMultiple(float toRound, int multipleOf)
```

Parameters

`toRound` float

`multipleOf` int

Returns

int

RoundToMultiple(float, float)

```
public static float RoundToMultiple(float toRound, float multipleOf)
```

Parameters

`toRound` float

`multipleOf` float

Returns

float

SymmetricCeil(float)

```
public static float SymmetricCeil(float f)
```

Parameters

f float

Returns

float

SymmetricCeilToInt(float)

```
public static int SymmetricCeilToInt(float f)
```

Parameters

f float

Returns

int

SymmetricFloor(float)

```
public static float SymmetricFloor(float f)
```

Parameters

f float

Returns

float

SymmetricFloorToInt(float)

```
public static int SymmetricFloorToInt(float f)
```

Parameters

f float

Returns

int

ToArray<T>(HashSet<T>)

```
public static T[] ToArray<T>(this HashSet<T> hashSet)
```

Parameters

hashSet HashSet<T>

Returns

T[]

Type Parameters

T

TruncateDecimalPlaces(float, int)

```
public static float TruncateDecimalPlaces(float f, int decimalPlaces)
```

Parameters

`f` float

`decimalPlaces` int

Returns

float

Vector2ToVector3(Vector3)

```
public static Vector3 Vector2ToVector3(Vector3 vector2)
```

Parameters

`vector2` Vector3

Returns

Vector3

Vector3ToVector2(Vector3)

```
public static Vector2 Vector3ToVector2(Vector3 vector3)
```

Parameters

`vector3` Vector3

Returns

Vector2

Class ImageConverter

Namespace: PAC

```
public static class ImageConverter
```

Inheritance

object ← ImageConverter

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Class KeyCodeFunctions

Namespace: PAC

```
public static class KeyCodeFunctions
```

Inheritance

object ← KeyCodeFunctions

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

IsAlpha(KeyCode)

```
public static bool IsAlpha(KeyCode keyCode)
```

Parameters

`keyCode` KeyCode

Returns

bool

IsAlphanumeric(KeyCode)

```
public static bool IsAlphanumeric(KeyCode keyCode)
```

Parameters

`keyCode` KeyCode

Returns

bool

IsDigit(KeyCode)

```
public static bool IsDigit(KeyCode keyCode)
```

Parameters

`keyCode` KeyCode

Returns

bool

StrToKeyCode(string)

```
public static KeyCode StrToKeyCode(string str)
```

Parameters

`str` string

Returns

KeyCode

Struct OutlineSideFill

Namespace: PAC

```
public struct OutlineSideFill
```

Inherited Members
ValueType.Equals(object) , ValueType.GetHashCode() , ValueType.ToString() , object.Equals(object, object) , object.GetType() , object.ReferenceEquals(object, object)

Constructors

OutlineSideFill(bool, bool, bool, bool, bool, bool, bool, bool)

```
public OutlineSideFill(bool topLeft, bool topMiddle, bool topRight, bool middleLeft,
bool middleRight, bool bottomLeft, bool bottomMiddle, bool bottomRight)
```

Parameters

`topLeft` bool

`topMiddle` bool

`topRight` bool

`middleLeft` bool

`middleRight` bool

`bottomLeft` bool

`bottomMiddle` bool

`bottomRight` bool

Fields

bottomLeft

```
public bool bottomLeft
```

Field Value

bool

bottomMiddle

```
public bool bottomMiddle
```

Field Value

bool

bottomRight

```
public bool bottomRight
```

Field Value

bool

middleLeft

```
public bool middleLeft
```

Field Value

bool

middleRight

```
public bool middleRight
```

Field Value

bool

topLeft

```
public bool topLeft
```

Field Value

bool

topMiddle

```
public bool topMiddle
```

Field Value

bool

topRight

```
public bool topRight
```

Field Value

bool

Class Preferences

Namespace: PAC

```
public static class Preferences
```

Inheritance

object ← Preferences

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

startupBrushSize

```
public static readonly Preferences.Preference<int> startupBrushSize
```

Field Value

Preferences.Preference<int>

Class Preferences.Preference<T>

Namespace: PAC

```
public class Preferences.Preference<T>
```

Type Parameters

T

Inheritance

object ← Preferences.Preference<T>

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Preference(string, Func<T>, Action<T>)

```
public Preference(string displayName, Func<T> getter, Action<T> setter)
```

Parameters

displayName string

getter Func<T>

setter Action<T>

Properties

displayName

```
public string displayName { get; }
```

Property Value

string

Methods

Get()

```
public T Get()
```

Returns

T

Set(T)

```
public void Set(T value)
```

Parameters

value T

Enum RotationAngle

Namespace: PAC

```
public enum RotationAngle
```

Fields

`Minus90 = -90`

`_0 = 0`

`_180 = 180`

`_90 = 90`

Class Tex2DSprite

Namespace: PAC

```
public static class Tex2DSprite
```

Inheritance

object ← Tex2DSprite

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

ApplyMask(Texture2D, IntVector2[])

```
public static Texture2D ApplyMask(Texture2D texture, IntVector2[] mask)
```

Parameters

`texture` Texture2D

`mask` IntVector2[]

Returns

Texture2D

ApplyMask(Texture2D, Texture2D)

```
public static Texture2D ApplyMask(Texture2D texture, Texture2D mask)
```

Parameters

`texture` Texture2D

`mask` Texture2D

Returns

Texture2D

BlankTexture(int, int)

```
public static Texture2D BlankTexture(int width, int height)
```

Parameters

`width` int

`height` int

Returns

Texture2D

Blend(Color, Texture2D, BlendMode)

Overlays topColour onto each pixel of bottomTex using the given blend mode.

```
public static Texture2D Blend(Color topColour, Texture2D bottomTex,
BlendMode blendMode)
```

Parameters

`topColour` Color

`bottomTex` Texture2D

`blendMode` BlendMode

Returns

Blend(Texture2D, Texture2D, BlendMode)

Overlays topTex onto bottomTex using the given blend mode, placing the bottom-left corner on the bottom-left corner.

```
public static Texture2D Blend(Texture2D topTex, Texture2D bottomTex,
BlendMode blendMode)
```

Parameters

`topTex` Texture2D

`bottomTex` Texture2D

`blendMode` BlendMode

Returns

Texture2D

Blend(Texture2D, Texture2D, IntVector2, BlendMode)

Overlays topTex onto bottomTex using the given blend mode, placing the bottom-left corner at the coordinates topTexOffset (which don't have to be within the image).

```
public static Texture2D Blend(Texture2D topTex, Texture2D bottomTex, IntVector2
topTexOffset, BlendMode blendMode)
```

Parameters

`topTex` Texture2D

`bottomTex` Texture2D

`topTexOffset` IntVector2

`blendMode` BlendMode

Returns

Texture2D

ChangeRect(Texture2D, IntRect)

Changes the dimensions of the texture to the new rect.

```
public static Texture2D ChangeRect(Texture2D texture, IntRect newRect)
```

Parameters

`texture` Texture2D

`newRect` IntRect

The coords of the new rect relative to the coords of the old rect.

Returns

Texture2D

CheckerboardBackground(int, int)

```
public static Texture2D CheckerboardBackground(int width, int height)
```

Parameters

`width` int

`height` int

Returns

Texture2D

ContainsPixel(Texture2D, IntVector2)

```
public static bool ContainsPixel(this Texture2D texture, IntVector2 pixel)
```

Parameters

`texture` Texture2D

`pixel` IntVector2

Returns

bool

ContainsPixel(Texture2D, int, int)

```
public static bool ContainsPixel(this Texture2D texture, int x, int y)
```

Parameters

`texture` Texture2D

`x` int

`y` int

Returns

bool

Copy(Texture2D)

```
public static Texture2D Copy(Texture2D texture)
```

Parameters

`texture` Texture2D

Returns

Extend(Texture2D, int, int, int, int)

Adds the given number of transparent pixels to each side of the texture. Negative amounts will crop the image.

```
public static Texture2D Extend(Texture2D texture, int left, int right, int up,
int down)
```

Parameters

`texture` Texture2D

`left` int

`right` int

`up` int

`down` int

Returns

Texture2D

Fill(Texture2D, IntVector2, Color, int)

```
public static Texture2D Fill(Texture2D texture, IntVector2 startPoint, Color colour,
int maxNumOfIterations = 1000000)
```

Parameters

`texture` Texture2D

`startPoint` IntVector2

`colour` Color

`maxNumOfIterations` int

Returns

Texture2D

Flip(Texture2D, FlipDirection)

```
public static Texture2D Flip(Texture2D texture, FlipDirection direction)
```

Parameters

`texture` Texture2D

`direction` FlipDirection

Returns

Texture2D

FlipX(Texture2D)

```
public static Texture2D FlipX(Texture2D texture)
```

Parameters

`texture` Texture2D

Returns

Texture2D

FlipY(Texture2D)

```
public static Texture2D FlipY(Texture2D texture)
```

Parameters

`texture` Texture2D

Returns

Texture2D

GetFillMask(Texture2D, IntVector2, int)

```
public static Texture2D GetFillMask(Texture2D texture, IntVector2 startPoint, int
maxNumOfIterations = 100000)
```

Parameters

`texture` Texture2D

`startPoint` IntVector2

`maxNumOfIterations` int

Returns

Texture2D

GetPixel(Texture2D, IntVector2)

```
public static Color GetPixel(this Texture2D texture, IntVector2 coords)
```

Parameters

`texture` Texture2D

`coords` IntVector2

Returns

Color

GetPixelsToFill(Texture2D, IntVector2, int)

```
public static IntVector2[] GetPixelsToFill(Texture2D texture, IntVector2 startPoint,
int maxNumOfIterations = 100000)
```

Parameters

`texture` Texture2D

`startPoint` IntVector2

`maxNumOfIterations` int

Returns

IntVector2[]

HSLHueSaturationGrid(int, int)

```
public static Texture2D HSLHueSaturationGrid(int width, int height)
```

Parameters

`width` int

`height` int

Returns

Texture2D

LoadFromFile(string)

```
public static Texture2D LoadFromFile(string filePath)
```

Parameters

`filePath` string

Returns

Texture2D

Multiply(Texture2D, Color)

```
public static Texture2D Multiply(Texture2D texture, Color colour)
```

Parameters

`texture` Texture2D

`colour` Color

Returns

Texture2D

Offset(Texture2D, IntVector2)

```
public static Texture2D Offset(Texture2D texture, IntVector2 offset)
```

Parameters

`texture` Texture2D

`offset` IntVector2

Returns

Texture2D

Outline(Texture2D, Color, bool, OutlineSideFill)

Makes an outline around the non-transparent pixels of the given texture.

```
public static Texture2D Outline(Texture2D texture, Color outlineColour, bool
outlineOutside, OutlineSideFill outlineSideFill)
```

Parameters

`texture` Texture2D

`outlineColour` Color

`outlineOutside` bool

When true: the outline is created next to existing pixels (widens the sprite). When false: replaces the outer pixels.

`outlineSideFill` OutlineSideFill

Returns

Texture2D

Overlay(Texture2D, Texture2D)

Overlays topTex onto bottomTex, placing the bottom-left corner on the bottom-left corner. Uses Normal blend mode.

```
public static Texture2D Overlay(Texture2D topTex, Texture2D bottomTex)
```

Parameters

`topTex` Texture2D

`bottomTex` Texture2D

Returns

Texture2D

Overlay(Texture2D, Texture2D, IntVector2)

Overlays topTex onto bottomTex, placing the bottom-left corner at the coordinates topTexOffset (which don't have to be within the image). Uses Normal blend mode.

```
public static Texture2D Overlay(Texture2D topTex, Texture2D bottomTex,
IntVector2 topTexOffset)
```

Parameters

`topTex` Texture2D

`bottomTex` Texture2D

`topTexOffset` IntVector2

Returns

Texture2D

ReplaceColour(Texture2D, Color, Color)

```
public static Texture2D ReplaceColour(Texture2D texture, Color toReplace,
Color replaceWith)
```

Parameters

`texture` Texture2D

`toReplace` Color

`replaceWith` Color

Returns

Texture2D

Rotate(Texture2D, RotationAngle)

Rotation is clockwise.

```
public static Texture2D Rotate(Texture2D texture, RotationAngle angle)
```

Parameters

`texture` Texture2D

`angle` RotationAngle

Returns

Texture2D

Rotate180(Texture2D)

```
public static Texture2D Rotate180(Texture2D texture)
```

Parameters

`texture` Texture2D

Returns

Texture2D

Rotate90(Texture2D)

Rotation is clockwise.

```
public static Texture2D Rotate90(Texture2D texture)
```

Parameters

`texture` Texture2D

Returns

Texture2D

RotateMinus90(Texture2D)

Rotation is clockwise.

```
public static Texture2D RotateMinus90(Texture2D texture)
```

Parameters

`texture` Texture2D

Returns

Texture2D

Scale(Texture2D, int, int)

```
public static Texture2D Scale(Texture2D texture, int newWidth, int newHeight)
```

Parameters

`texture` Texture2D

`newWidth` int

`newHeight` int

Returns

Texture2D

Scale(Texture2D, float)

```
public static Texture2D Scale(Texture2D texture, float scaleFactor)
```

Parameters

`texture` Texture2D

`scaleFactor` float

Returns

Texture2D

Scale(Texture2D, float, float)

```
public static Texture2D Scale(Texture2D texture, float xScaleFactor,
float yScaleFactor)
```

Parameters

`texture` Texture2D

`xScaleFactor` float

`yScaleFactor` float

Returns

Texture2D

SetPixel(Texture2D, IntVector2, Color)

```
public static void SetPixel(this Texture2D texture, IntVector2 coords, Color colour)
```

Parameters

`texture` Texture2D

`coords` IntVector2

`colour` Color

SolidTexture(int, int, Color)

```
public static Texture2D SolidTexture(int width, int height, Color colour)
```

Parameters

`width` int

`height` int

`colour` Color

Returns

Texture2D

Subtract(Texture2D, Texture2D)

```
public static Texture2D Subtract(Texture2D topTex, Texture2D bottomTex)
```

Parameters

`topTex` Texture2D

`bottomTex` Texture2D

Returns

Texture2D

Tex2DToSprite(Texture2D)

```
public static Sprite Tex2DToSprite(Texture2D tex)
```

Parameters

`tex` Texture2D

Returns

Sprite

Namespace PAC.Animation

Classes

AnimationKeyFrame

A class representing a single keyframe for a layer.

AnimationManager

Handles the animation timeline and playback of animations. The animation system is still a work in progress.

FrameNotch

A class for the frame number markers on the animation timeline.

KeyFrameIcon

Stores the data about a keyframe icon on the animation timeline.

LayerAnimation

Class AnimationKeyFrame

Namespace: PAC.Animation

A class representing a single keyframe for a layer.

```
public class AnimationKeyFrame : IJSONable
```

Inheritance

object ← AnimationKeyFrame

Implements

IJSONable

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

AnimationKeyFrame(AnimationKeyFrame)

Creates a deep copy of the AnimationKeyFrame.

```
public AnimationKeyFrame(AnimationKeyFrame animationKeyFrame)
```

Parameters

`animationKeyFrame` AnimationKeyFrame

AnimationKeyFrame(int, Texture2D)

```
public AnimationKeyFrame(int frame, Texture2D texture)
```

Parameters

`frame` int

`texture` Texture2D

Fields

frame

The number of the frame this keyframe is on.

```
public int frame
```

Field Value

int

texture

The texture displayed at this keyframe.

```
public Texture2D texture
```

Field Value

Texture2D

Methods

DeepCopy()

Creates a deep copy of the AnimationKeyFrame.

```
public AnimationKeyFrame DeepCopy()
```

Returns

AnimationKeyFrame

FromJSON(JSON)

```
public static AnimationKeyFrame FromJSON(JSON json)
```

Parameters

`json` JSON

Returns

AnimationKeyFrame

ToJSON()

```
public JSON ToJSON()
```

Returns

JSON

Class AnimationManager

Namespace: PAC.Animation

Handles the animation timeline and playback of animations. The animation system is still a work in progress.

```
public class AnimationManager : MonoBehaviour
```

Inheritance

object ← AnimationManager

Fields

framerate

```
public int framerate
```

Field Value

int

onionSkinColour

```
public Color onionSkinColour
```

Field Value

Color

showOnionSkin

```
public bool showOnionSkin
```

Field Value

bool

Properties

currentFrameIndex

The number of the frame currently being displayed.

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

Property Value

int

Methods

AddKeyFrame()

Extends the length of the animation by adding one frame to the end.

```
public void AddKeyFrame()
```

DebugLogKeyFrames()

For debugging purposes. Prints the current frame number and the frame numbers of each keyframe.

```
public void DebugLogKeyFrames()
```

DeleteSelectedKeyFrame()

Deletes the selected key frame.

```
public void DeleteSelectedKeyFrame()
```

Pause()

Pauses the animation playback.

```
public void Pause()
```

Play()

Starts/resumes the animation playback.

```
public void Play()
```

RemoveKeyFrame()

Reduces the length of the animation by removing the final frame.

```
public void RemoveKeyFrame()
```

Stop()

Stops the animation playback and resets the current frame number back to the start.

```
public void Stop()
```

SubscribeToOnCurrentFrameIndexChange(UnityAction)

Event is invoked when the current frame number changes.

```
public void SubscribeToOnCurrentFrameIndexChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnKeyFrameAdded(UnityAction)

Event is invoked when a keyframe is added.

```
public void SubscribeToOnKeyFrameAdded(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnKeyFrameDeleted(UnityAction)

Event is invoked when a keyframe is deleted.

```
public void SubscribeToOnKeyFrameDeleted(UnityAction call)
```

Parameters

`call` UnityAction

Class FrameNotch

Namespace: PAC.Animation

A class for the frame number markers on the animation timeline.

```
public class FrameNotch : MonoBehaviour
```

Inheritance

object ← FrameNotch

Properties

frameNum

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

Property Value

int

Methods

SetFrameNumber(int)

```
public void SetFrameNumber(int num)
```

Parameters

num int

Class KeyFrameIcon

Namespace: PAC.Animation

Stores the data about a keyframe icon on the animation timeline.

```
public class KeyFrameIcon : MonoBehaviour
```

Inheritance

object ← KeyFrameIcon

Fields

frameIndex

The frame number this keyframe is on.

```
public int frameIndex
```

Field Value

int

layerIndex

The layer index this keyframe is on.

```
public int layerIndex
```

Field Value

int

Class LayerAnimation

Namespace: PAC.Animation

```
[Serializable]
[Obsolete("LayerAnimation has now been moved directly into the Layer class.")]
public class LayerAnimation
```

Inheritance

object ← LayerAnimation

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

LayerAnimation(LayerAnimation)

Creates a deep copy of the given layer animation, including deep copies of the key frames.

```
public LayerAnimation(LayerAnimation layerAnimation)
```

Parameters

`layerAnimation` LayerAnimation

LayerAnimation(int, int)

Create an animation with a blank starting key frame.

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

Parameters

`width` int

The width of the frames.

`height` int

The height of the frames.

LayerAnimation(Texture2D)

Create an animation with the given texture as the starting key frame.

```
public LayerAnimation(Texture2D texture)
```

Parameters

`texture` Texture2D

The texture for the starting key frame.

Properties

this[int]

Returns the most recent key frame before or at frame frame - i.e. the frame the animation will play at frame frame. Same as GetKeyFrame(i).

```
public AnimationKeyFrame this[int i] { get; }
```

Parameters

`i` int

Property Value

AnimationKeyFrame

height

```
public int height { get; }
```

Property Value

int

keyFrameCount

The number of key frames in the animation.

```
public int keyFrameCount { get; }
```

Property Value

int

keyFrameIndices

The frame indices of the key frames in the animation.

```
public int[] keyFrameIndices { get; }
```

Property Value

int[]

keyFrames

The key frames of the animation, in order of frame number.

```
public List<AnimationKeyFrame> keyFrames { get; }
```

Property Value

List<AnimationKeyFrame>

numOfFrames

```
public int numOfFrames { get; }
```

Property Value

int

width

```
public int width { get; }
```

Property Value

int

Methods

AddKeyFrame(AnimationKeyFrame)

Adds the given key frame. Returns true if it replaces an existing key frame, and false otherwise.

```
public bool AddKeyFrame(AnimationKeyFrame keyFrame)
```

Parameters

`keyFrame` AnimationKeyFrame

Returns

bool

AddKeyFrame(int)

Adds a key frame at frame frame. The texture will be that of the most recent key frame. Returns true if it replaces an existing key frame, and false otherwise.

```
public bool AddKeyFrame(int frame)
```

Parameters

`frame` int

Returns

bool

AddKeyFrame(int, Texture2D)

Adds a key frame with the given texture at frame frame. Returns true if it replaces an existing key frame, and false otherwise.

```
public bool AddKeyFrame(int frame, Texture2D texture)
```

Parameters

`frame` int

`texture` Texture2D

Returns

bool

Clear()

Deletes all key frames.

```
public void Clear()
```

DeleteKeyFrame(AnimationKeyFrame)

Deletes the given key frame, if it's in the animation, in which case it returns true. Otherwise it returns false.

```
public bool DeleteKeyFrame(AnimationKeyFrame keyFrame)
```

Parameters

`keyFrame` AnimationKeyFrame

Returns

bool

DeleteKeyFrame(int)

Deletes the key frame at the given frame index, if there is one, in which case it returns that key frame. Otherwise it returns null.

```
public AnimationKeyFrame DeleteKeyFrame(int keyframe)
```

Parameters

`keyframe` int

Returns

AnimationKeyFrame

DeleteMostRecentKeyFrame(int)

Deletes the most recent key frame before or at the given frame index and returns that key frame.

```
public AnimationKeyFrame DeleteMostRecentKeyFrame(int frame)
```

Parameters

`frame` int

Returns

AnimationKeyFrame

FromJSON(JSON)

```
public static LayerAnimation FromJSON(JSON json)
```

Parameters

`json` JSON

Returns

LayerAnimation

GetKeyFrame(int)

Returns the most recent key frame before or at frame frame - i.e. the frame the animation will play at frame frame.

```
public AnimationKeyFrame GetKeyFrame(int frame)
```

Parameters

`frame` int

Returns

AnimationKeyFrame

HasKeyFrameAt(int)

Returns whether or not there is a key frame at the given frame index.

```
public bool HasKeyFrameAt(int frame)
```

Parameters

`frame` int

Returns

bool

ToJSON()

```
public JSON ToJSON()
```

Returns

JSON

Namespace PAC.Clipboard

Classes

Clipboard

A class to handle copy/paste functionality. Not yet properly implemented.

Class Clipboard

Namespace: PAC.Clipboard

A class to handle copy/paste functionality. Not yet properly implemented.

```
public class Clipboard : MonoBehaviour
```

Inheritance

object ← Clipboard

Methods

Copy()

```
public void Copy()
```

Cut()

```
public void Cut()
```

Paste()

```
public void Paste()
```

Namespace PAC.Colour

Classes

BlendMode

A class for blend modes.

ColorExtensions

Structs

HSL

Represents a colour in HSL form.

HSV

Represents a colour in HSV form.

Class BlendMode

Namespace: PAC.Colour

A class for blend modes.

```
public class BlendMode
```

Inheritance

object ← BlendMode

Inherited Members

object.Equals(object, object) , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Fields

Add

Add blend mode.

```
public static readonly BlendMode Add
```

Field Value

BlendMode

Multiply

Multiply blend mode.

```
public static readonly BlendMode Multiply
```

Field Value

BlendMode

Normal

Normal blend mode.

```
public static readonly BlendMode Normal
```

Field Value

BlendMode

Overlay

Overlay blend mode.

```
public static readonly BlendMode Overlay
```

Field Value

BlendMode

Replace

Replace blend mode.

```
public static readonly BlendMode Replace
```

Field Value

BlendMode

Screen

Screen blend mode.

```
public static readonly BlendMode Screen
```

Field Value

BlendMode

Subtract

Subtract blend mode.

```
public static readonly BlendMode Subtract
```

Field Value

BlendMode

blendModes

All implemented blend modes.

```
public static readonly BlendMode[] blendModes
```

Field Value

BlendMode[]

Methods

Blend(Color, Color)

Blend the two colours using the blend mode's blend function.

```
public Color Blend(Color topColour, Color bottomColour)
```

Parameters

`topColour` Color

`bottomColour` Color

Returns

Color

Equals(object)

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

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

Parameters

`obj` object

The object to compare with the current object.

Returns

bool

true if the specified object is equal to the current object; otherwise, false.

GetHashCode()

Serves as the default hash function.

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

Returns

int

A hash code for the current object.

MultiplyColours(Color, Color)

Multiplies the colours component-wise.

```
public static Color MultiplyColours(Color colour1, Color colour2)
```

Parameters

`colour1` Color

`colour2` Color

Returns

Color

StringToBlendMode(string)

Returns the blend mode with that name (case-insensitive).

```
public static BlendMode StringToBlendMode(string blendModeName)
```

Parameters

`blendModeName` string

Returns

BlendMode

ToString()

Returns a string that represents the current object.

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

Returns

string

A string that represents the current object.

Operators

operator ==(BlendMode, BlendMode)

```
public static bool operator ==(BlendMode a, BlendMode b)
```

Parameters

a BlendMode

b BlendMode

Returns

bool

operator !=(BlendMode, BlendMode)

```
public static bool operator !=(BlendMode a, BlendMode b)
```

Parameters

a BlendMode

b BlendMode

Returns

bool

Class ColorExtensions

Namespace: PAC.Colour

```
public static class ColorExtensions
```

Inheritance

object ← ColorExtensions

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Invert(Color, bool)

Inverts the colour, i.e. does 1 - value for each component.

```
public static Color Invert(this Color colour, bool invertAlpha = false)
```

Parameters

`colour` Color

`invertAlpha` bool

Whether to invert the alpha value as well.

Returns

Color

ToHSL(Color)

```
public static HSL ToHSL(this Color color)
```

Parameters

`color` Color

Returns

HSL

ToHSV(Color)

```
public static HSV ToHSV(this Color color)
```

Parameters

`color` Color

Returns

HSV

Struct HSL

Namespace: PAC.Colour

Represents a colour in HSL form.

```
public struct HSL
```

Inherited Members
ValueType.Equals(object) , ValueType.GetHashCode() , object.Equals(object, object) , object.GetType() , object.ReferenceEquals(object, object)

Constructors

HSL(Color)

```
public HSL(Color rgb)
```

Parameters

`rgb` Color

HSL(float, float, float)

```
public HSL(float hue, float saturation, float lightness)
```

Parameters

`hue` float

`saturation` float

`lightness` float

HSL(float, float, float, float)

```
public HSL(float hue, float saturation, float lightness, float alpha)
```

Parameters

`hue` float

`saturation` float

`lightness` float

`alpha` float

Properties

a

Alpha.

```
public float a { readonly get; set; }
```

Property Value

float

color

```
public Color color { get; }
```

Property Value

Color

h

Hue.

```
public float h { readonly get; set; }
```

Property Value

float

hsv

```
public HSV hsv { get; }
```

Property Value

HSV

l

Lightness.

```
public float l { readonly get; set; }
```

Property Value

float

s

Saturation.

```
public float s { readonly get; set; }
```

Property Value

float

Methods

ToColor()

```
public Color ToColor()
```

Returns

Color

ToHSV()

```
public HSV ToHSV()
```

Returns

HSV

ToString()

Returns the fully qualified type name of this instance.

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

Returns

string

The fully qualified type name.

ToString(int)

```
public string ToString(int decimalPlaces)
```

Parameters

`decimalPlaces` int

Returns

string

Operators

explicit operator Color(HSL)

```
public static explicit operator Color(HSL hsl)
```

Parameters

`hsl` HSL

Returns

Color

explicit operator HSV(HSL)

```
public static explicit operator HSV(HSL hsl)
```

Parameters

`hsl` HSL

Returns

HSV

Struct HSV

Namespace: PAC.Colour

Represents a colour in HSV form.

```
public struct HSV
```

Inherited Members
ValueType.Equals(object) , ValueType.GetHashCode() , object.Equals(object, object) , object.GetType() , object.ReferenceEquals(object, object)

Constructors

HSV(Color)

```
public HSV(Color rgb)
```

Parameters

`rgb` Color

HSV(float, float, float)

```
public HSV(float hue, float saturation, float value)
```

Parameters

`hue` float

`saturation` float

`value` float

HSV(float, float, float, float)

```
public HSV(float hue, float saturation, float value, float alpha)
```

Parameters

`hue` float

`saturation` float

`value` float

`alpha` float

Properties

a

Alpha.

```
public float a { readonly get; set; }
```

Property Value

float

color

```
public Color color { get; }
```

Property Value

Color

h

Hue.

```
public float h { readonly get; set; }
```

Property Value

float

hsl

```
public HSL hsl { get; }
```

Property Value

HSL

s

Saturation.

```
public float s { readonly get; set; }
```

Property Value

float

v

Value.

```
public float v { readonly get; set; }
```

Property Value

float

Methods

ToColor()

```
public Color ToColor()
```

Returns

Color

ToHSL()

```
public HSL ToHSL()
```

Returns

HSL

ToString()

Returns the fully qualified type name of this instance.

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

Returns

string

The fully qualified type name.

ToString(int)

```
public string ToString(int decimalPlaces)
```

Parameters

`decimalPlaces` int

Returns

string

Operators

explicit operator Color(HSV)

```
public static explicit operator Color(HSV hsv)
```

Parameters

`hsv` HSV

Returns

Color

explicit operator HSL(HSV)

```
public static explicit operator HSL(HSV hsv)
```

Parameters

`hsv` HSV

Returns

HSL

Namespace PAC.ColourPicker

Classes

ColourPreview

A class to represent colour previews - the boxes on colour pickers that show what colours you have selected.

GlobalColourPicker

A class for the main colour picker in the program - the one that appears in the main view.

HSLColourPicker

A class for the HSL colour picker.

HSLHueSaturationBox

A class for the hue/saturation box of the HSL colour picker.

HSLLightnessSlider

A class for the lightness slider of the HSL colour picker.

Class ColourPreview

Namespace: PAC.ColourPicker

A class to represent colour previews - the boxes on colour pickers that show what colours you have selected.

```
public class ColourPreview : MonoBehaviour
```

Inheritance

object ← ColourPreview

Properties

colour

The colour displayed in the colour preview.

```
public Color colour { get; }
```

Property Value

Color

outlineThickness

```
public float outlineThickness { get; }
```

Property Value

float

Methods

SetColour(Color)

```
public void SetColour(Color colour)
```

Parameters

`colour` Color

SubscribeToOnDeselect(UnityAction)

Event invoked when colour preview is deselected.

```
public void SubscribeToOnDeselect(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnSelect(UnityAction)

Event invoked when colour preview is selected.

```
public void SubscribeToOnSelect(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnToggle(UnityAction)

Event invoked when colour preview is selected or deselected.

```
public void SubscribeToOnToggle(UnityAction call)
```

Parameters

`call` UnityAction

Class GlobalColourPicker

Namespace: PAC.ColourPicker

A class for the main colour picker in the program - the one that appears in the main view.

```
public class GlobalColourPicker : MonoBehaviour
```

Inheritance

object ← GlobalColourPicker

Properties

colour

The currenty-selected colour.

```
public Color colour { get; }
```

Property Value

Color

numOfColourPreviews

```
public int numOfColourPreviews { get; }
```

Property Value

int

primaryColour

The current primary colour.

```
public Color primaryColour { get; }
```

Property Value

Color

secondaryColour

The current secondary colour.

```
public Color secondaryColour { get; }
```

Property Value

Color

Methods

GetColour(int)

Gets the chosen colour at the given index: 0 - primary; 1 - secondary.

```
public Color GetColour(int colourPreviewIndex)
```

Parameters

`colourPreviewIndex` int

Returns

Color

SetColour(Color)

Sets the current colour.

```
public void SetColour(Color colour)
```

Parameters

`colour` Color

SubscribeToOnColourChange(UnityAction)

Event is invoked when the selected colour changes.

```
public void SubscribeToOnColourChange(UnityAction call)
```

Parameters

`call` UnityAction

Class HSLColourPicker

Namespace: PAC.ColourPicker

A class for the HSL colour picker.

```
public class HSLColourPicker : MonoBehaviour
```

Inheritance

object ← HSLColourPicker

Properties

alpha

```
public float alpha { get; }
```

Property Value

float

color

```
public Color color { get; }
```

Property Value

Color

hsl

```
public HSL hsl { get; }
```

Property Value

HSL

hue

```
public float hue { get; }
```

Property Value

float

lightness

```
public float lightness { get; }
```

Property Value

float

mouseSensitivity

```
public float mouseSensitivity { get; }
```

Property Value

float

saturation

```
public float saturation { get; }
```

Property Value

float

slowSensitivityScalar

The sensitivity of the mouse when holding the modifier keyboard shortcut to reduce the sensitivity.

```
public float slowSensitivityScalar { get; }
```

Property Value

float

Methods

SetColour(Color)

```
public void SetColour(Color colour)
```

Parameters

`colour` Color

SubscribeToOnColourChange(UnityAction)

Event invoked when the selected colour changes.

```
public void SubscribeToOnColourChange(UnityAction call)
```

Parameters

`call` UnityAction

UpdateColour()

```
public void UpdateColour()
```

Class HSLHueSaturationBox

Namespace: PAC.ColourPicker

A class for the hue/saturation box of the HSL colour picker.

```
public class HSLHueSaturationBox : MonoBehaviour
```

Inheritance

object ← HSLHueSaturationBox

Properties

hue

```
public float hue { get; }
```

Property Value

float

saturation

```
public float saturation { get; }
```

Property Value

float

Methods

SetHue(float)

```
public void SetHue(float hue)
```

Parameters

`hue` float

SetSaturation(float)

```
public void SetSaturation(float saturation)
```

Parameters

`saturation` float

Class HSLLightnessSlider

Namespace: PAC.ColourPicker

A class for the lightness slider of the HSL colour picker.

```
public class HSLLightnessSlider : MonoBehaviour
```

Inheritance

object ← HSLLightnessSlider

Properties

lightness

```
public float lightness { get; }
```

Property Value

float

Methods

SetDisplayHueSaturation(float, float)

Sets the hue/saturation to use for displaying the lightness gradient.

```
public void SetDisplayHueSaturation(float hue, float saturation)
```

Parameters

`hue` float

`saturation` float

SetLightness(float)

```
public void SetLightness(float lightness)
```

Parameters

`lightness` float

Namespace PAC.DataStructures

Classes

CustomStack<T>

A custom implementation of a stack to allow removal of items at a specific index.

Structs

IntRect

A struct to represent a rectangular region of integer coordinates.

IntVector2

A struct to represent a 2-dimensional vector with integer coordinates.

Class CustomStack<T>

Namespace: PAC.DataStructures

A custom implementation of a stack to allow removal of items at a specific index.

```
public class CustomStack<T>
```

Type Parameters

T

Inheritance

object ← CustomStack<T>

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Properties

Count

```
public int Count { get; }
```

Property Value

int

Methods

Clear()

Removes all items from the stack.

```
public void Clear()
```

Peek()

Returns the item on top of the stack.

```
public T Peek()
```

Returns

T

Pop()

Removes and returns the item on top of the stack.

```
public T Pop()
```

Returns

T

Push(T)

Adds the item to the top of the stack.

```
public void Push(T item)
```

Parameters

`item` T

Remove(T)

Removes the first occurrence (starting from the top) of the item in the stack.

```
public bool Remove(T item)
```

Parameters

`item` T

Returns

bool

true if the item is successfully removed.

RemoveAll(T)

Removes all occurrences of the item in the stack.

```
public void RemoveAll(T item)
```

Parameters

`item` T

RemoveAt(int)

Removes the item at the given index and returns it.

```
public T RemoveAt(int index)
```

Parameters

`index` int

Returns

T

ToArray()

```
public T[] ToArray()
```

Returns

T[]

ToList()

```
public List<T> ToList()
```

Returns

List <T>

Struct IntRect

Namespace: PAC.DataStructures

A struct to represent a rectangular region of integer coordinates.

```
public struct IntRect : IEnumerable
```

Implements
IEnumerable

Inherited Members
object.Equals(object, object) , object.GetType() , object.ReferenceEquals(object, object)

Constructors

IntRect(IntVector2, IntVector2)

```
public IntRect(IntVector2 corner, IntVector2 oppositeCorner)
```

Parameters

`corner` IntVector2

`oppositeCorner` IntVector2

Properties

area

```
public int area { get; }
```

Property Value

int

bottomLeft

```
public IntVector2 bottomLeft { get; set; }
```

Property Value

IntVector2

bottomRight

```
public IntVector2 bottomRight { get; set; }
```

Property Value

IntVector2

centre

```
public Vector2 centre { get; }
```

Property Value

Vector2

height

```
public int height { get; }
```

Property Value

int

isSquare

True if the rect is a square.

```
public bool isSquare { get; }
```

Property Value

bool

points

Gets the points in the rect, starting with the bottom row, read left to right, then the next row, etc.

WARNING: this is very expensive for large rects.

```
public IntVector2[] points { get; }
```

Property Value

IntVector2[]

topLeft

```
public IntVector2 topLeft { get; set; }
```

Property Value

IntVector2

topRight

```
public IntVector2 topRight { get; set; }
```

Property Value

IntVector2

width

```
public int width { get; }
```

Property Value

int

Methods

Area(IntRect)

```
public static int Area(IntRect rect)
```

Parameters

`rect` IntRect

Returns

int

Clamp(IntRect)

Shifts the given rect so it is (weakly) contained within the rect.

```
public IntRect Clamp(IntRect intRect)
```

Parameters

`intRect` IntRect

Returns

IntRect

Clamp(IntVector2)

Clamps the vector component-wise so its coordinates are within the rect.

```
public IntVector2 Clamp(IntVector2 intVector)
```

Parameters

`intVector` IntVector2

Returns

IntVector2

Contains(IntRect)

Returns true if the given rect is (weakly) contained in this rect.

```
public bool Contains(IntRect intRect)
```

Parameters

`intRect` IntRect

Returns

bool

Contains(IntVector2)

Returns true if the point is in the rect.

```
public bool Contains(IntVector2 point)
```

Parameters

`point` IntVector2

Returns

bool

Contains(int, int)

Returns true if the point is in the rect.

```
public bool Contains(int x, int y)
```

Parameters

`x` int

`y` int

Returns

bool

Contains(float, float)

```
public bool Contains(float x, float y)
```

Parameters

`x` float

`y` float

Returns

bool

Contains(Vector2)

```
public bool Contains(Vector2 point)
```

Parameters

`point` Vector2

Returns

bool

Equals(object)

Indicates whether this instance and a specified object are equal.

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

Parameters

`obj` object

The object to compare with the current instance.

Returns

bool

true if `obj` and this instance are the same type and represent the same value; otherwise, false.

FilterPointsInside(IntVector2[])

Removes all IntVector2s outside the rect.

```
public IntVector2[] FilterPointsInside(IntVector2[] intVectors)
```

Parameters

`intVectors` IntVector2[]

Returns

IntVector2[]

FilterPointsOutside(IntVector2[])

Removes all IntVector2s inside the rect.

```
public IntVector2[] FilterPointsOutside(IntVector2[] intVectors)
```

Parameters

`intVectors` IntVector2[]

Returns

IntVector2[]

GetEnumerator()

Enumerates the points in the rect, starting with the bottom row, read left to right, then the next row, etc.

```
public IEnumerator GetEnumerator()
```

Returns

IEnumerator

GetHashCode()

Returns the hash code for this instance.

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

Returns

int

A 32-bit signed integer that is the hash code for this instance.

GetPoints()

Gets the points in the rect, starting with the bottom row, read left to right, then the next row, etc.

WARNING: this is very expensive for large rects.

```
public IntVector2[] GetPoints()
```

Returns

IntVector2[]

IsContainedIn(IntRect)

Returns true if this rect is (weakly) contained in the given rect.

```
public bool IsContainedIn(IntRect intRect)
```

Parameters

`intRect` IntRect

Returns

bool

IsSquare(IntRect)

Returns true if the rect is a square.

```
public static bool IsSquare(IntRect intRect)
```

Parameters

`intRect` IntRect

Returns

bool

Overlap(IntRect, IntRect)

Returns true if the two rects overlap at all.

```
public static bool Overlap(IntRect rect1, IntRect rect2)
```

Parameters

`rect1` IntRect

`rect2` IntRect

Returns

bool

Overlaps(IntRect)

Returns true if this rect overlaps the given rect at all.

```
public bool Overlaps(IntRect intRect)
```

Parameters

`intRect` IntRect

Returns

bool

ToRect()

Cast to Unity Rect.

```
public Rect ToRect()
```

Returns

Rect

ToString()

Returns the fully qualified type name of this instance.

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

Returns

string

The fully qualified type name.

Operators

operator +(IntRect, IntVector2)

Shifts the whole rect by the given vector.

```
public static IntRect operator +(IntRect intRect, IntVector2 intVector)
```

Parameters

`intRect` IntRect

`intVector` IntVector2

Returns

IntRect

operator +(IntVector2, IntRect)

Shifts the whole rect by the given vector.

```
public static IntRect operator +(IntVector2 intVector, IntRect intRect)
```

Parameters

`intVector` IntVector2

`intRect` IntRect

Returns

IntRect

operator ==(IntRect, IntRect)

```
public static bool operator ==(IntRect rect1, IntRect rect2)
```

Parameters

`rect1` IntRect

`rect2` IntRect

Returns

bool

explicit operator Rect(IntRect)

Cast to Unity Rect.

```
public static explicit operator Rect(IntRect intRect)
```

Parameters

`intRect` IntRect

Returns

Rect

operator !=(IntRect, IntRect)

```
public static bool operator !=(IntRect a, IntRect b)
```

Parameters

`a` IntRect

`b` IntRect

Returns

bool

operator -(IntRect, IntVector2)

Shifts the whole rect by the given vector.

```
public static IntRect operator -(IntRect intRect, IntVector2 intVector)
```

Parameters

`intRect` IntRect

`intVector` IntVector2

Returns

IntRect

Struct IntVector2

Namespace: PAC.DataStructures

A struct to represent a 2-dimensional vector with integer coordinates.

```
public struct IntVector2
```

Inherited Members
object.Equals(object, object) , object.GetType() , object.ReferenceEquals(object, object)

Constructors

IntVector2(int, int)

```
public IntVector2(int x, int y)
```

Parameters

x int

y int

IntVector2(float, float)

Rounds the coords towards zero.

```
public IntVector2(float x, float y)
```

Parameters

x float

y float

IntVector2(Vector2)

Rounds the coords towards zero.

```
public IntVector2(Vector2 vector2)
```

Parameters

`vector2` Vector2

Fields

down

The vector (0, -1).

```
public static IntVector2 down
```

Field Value

IntVector2

left

The vector (-1, 0).

```
public static IntVector2 left
```

Field Value

IntVector2

one

The vector (1, 1).

```
public static IntVector2 one
```

Field Value

IntVector2

right

The vector (1, 0).

```
public static IntVector2 right
```

Field Value

IntVector2

up

The vector (0, 1).

```
public static IntVector2 up
```

Field Value

IntVector2

x

```
public int x
```

Field Value

int

y

```
public int y
```

Field Value

int

zero

The vector (0, 0).

```
public static IntVector2 zero
```

Field Value

IntVector2

Properties

magnitude

```
public float magnitude { get; }
```

Property Value

float

squaredMagnitude

```
public float squaredMagnitude { get; }
```

Property Value

float

Methods

CeilToIntVector2(Vector2)

Ceils component-wise.

```
public static IntVector2 CeilToIntVector2(Vector2 vector2)
```

Parameters

`vector2` Vector2

Returns

IntVector2

Distance(IntVector2, IntVector2)

Computes the Euclidean distance between the vectors.

```
public static float Distance(IntVector2 a, IntVector2 b)
```

Parameters

`a` IntVector2

`b` IntVector2

Returns

float

Dot(IntVector2, IntVector2)

Computes the dot product.

```
public static int Dot(IntVector2 a, IntVector2 b)
```

Parameters

`a` IntVector2

`b` IntVector2

Returns

int

Equals(object)

Indicates whether this instance and a specified object are equal.

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

Parameters

`obj` object

The object to compare with the current instance.

Returns

bool

true if `obj` and this instance are the same type and represent the same value; otherwise, false.

FloorToIntVector2(Vector2)

Floors component-wise.

```
public static IntVector2 FloorToIntVector2(Vector2 vector2)
```

Parameters

`vector2` Vector2

Returns

IntVector2

GetHashCode()

Returns the hash code for this instance.

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

Returns

int

A 32-bit signed integer that is the hash code for this instance.

Magnitude(IntVector2)

```
public static float Magnitude(IntVector2 a)
```

Parameters

a IntVector2

Returns

float

Max(IntVector2, IntVector2)

Takes the maximum of a and b component-wise.

```
public static IntVector2 Max(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

IntVector2

Max(params IntVector2[])

Takes the maximum of the vectors component-wise.

```
public static IntVector2 Max(params IntVector2[] intVectors)
```

Parameters

intVectors IntVector2[]

Returns

IntVector2

Min(IntVector2, IntVector2)

Takes the minimum of a and b component-wise.

```
public static IntVector2 Min(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

IntVector2

Min(params IntVector2[])

Takes the minimum of the vectors component-wise.

```
public static IntVector2 Min(params IntVector2[] intVectors)
```

Parameters

`intVectors` IntVector2[]

Returns

IntVector2

RoundToIntVector2(Vector2)

Rounds component-wise.

```
public static IntVector2 RoundToIntVector2(Vector2 vector2)
```

Parameters

`vector2` Vector2

Returns

IntVector2

SqrDistance(IntVector2, IntVector2)

```
public static float SqrDistance(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

float

SqrMagnitude(IntVector2)

```
public static float SqrMagnitude(IntVector2 a)
```

Parameters

a IntVector2

Returns

float

ToString()

Returns the fully qualified type name of this instance.

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

Returns

string

The fully qualified type name.

ToVector2()

Cast to Unity Vector2.

```
public Vector2 ToVector2()
```

Returns

Vector2

ToVector3()

Cast to Unity Vector3, with a 0 in the z-coord.

```
public Vector3 ToVector3()
```

Returns

Vector3

Operators

operator +(IntVector2, IntVector2)

Adds component-wise.

```
public static IntVector2 operator +(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

IntVector2

operator +(IntVector2, IntVector2[])

Adds the vector to each element of the array.

```
public static IntVector2[] operator +(IntVector2 intVector,
IntVector2[] intVectorArray)
```

Parameters

`intVector` IntVector2

`intVectorArray` IntVector2[]

Returns

IntVector2[]

operator +(IntVector2[], IntVector2)

Adds the vector to each element of the array.

```
public static IntVector2[] operator +(IntVector2[] intVectorArray,
IntVector2 intVector)
```

Parameters

`intVectorArray` IntVector2[]

`intVector` IntVector2

Returns

IntVector2[]

operator /(IntVector2, IntVector2)

Divides (integer division) component-wise.

```
public static IntVector2 operator /(IntVector2 a, IntVector2 b)
```

Parameters

`a` IntVector2

`b` IntVector2

Returns

IntVector2

operator /(IntVector2, IntVector2[])

Divides the vector by each element of the array.

```
public static IntVector2[] operator /(IntVector2 intVector,
IntVector2[] intVectorArray)
```

Parameters

`intVector` IntVector2

`intVectorArray` IntVector2[]

Returns

IntVector2[]

operator /(IntVector2, int)

Divides (integer division) component-wise.

```
public static IntVector2 operator /(IntVector2 vector, int scalar)
```

Parameters

`vector` IntVector2

`scalar` int

Returns

IntVector2

operator /(IntVector2[], IntVector2)

Divides each element of the array by the vector.

```
public static IntVector2[] operator /(IntVector2[] intVectorArray,
IntVector2 intVector)
```

Parameters

`intVectorArray` IntVector2[]

`intVector` IntVector2

Returns

IntVector2[]

operator ==(IntVector2, IntVector2)

```
public static bool operator ==(IntVector2 a, IntVector2 b)
```

Parameters

`a` IntVector2

`b` IntVector2

Returns

bool

operator >(IntVector2, IntVector2)

Compares component-wise.

```
public static bool operator >(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

bool

operator >=(IntVector2, IntVector2)

Compares component-wise.

```
public static bool operator >=(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

bool

implicit operator Vector2(IntVector2)

Cast to Unity Vector2.

```
public static implicit operator Vector2(IntVector2 intVector)
```

Parameters

intVector IntVector2

Returns

Vector2

implicit operator Vector3(IntVector2)

Cast to Unity Vector3, with a 0 in the z-coord.

```
public static implicit operator Vector3(IntVector2 intVector)
```

Parameters

`intVector` IntVector2

Returns

Vector3

operator !=(IntVector2, IntVector2)

```
public static bool operator !=(IntVector2 a, IntVector2 b)
```

Parameters

`a` IntVector2

`b` IntVector2

Returns

bool

operator <(IntVector2, IntVector2)

Compares component-wise.

```
public static bool operator <(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

bool

operator <=(IntVector2, IntVector2)

Compares component-wise.

```
public static bool operator <=(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

bool

operator *(IntVector2, IntVector2)

Multiplies component-wise.

```
public static IntVector2 operator *(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

`b` IntVector2

Returns

IntVector2

operator *(IntVector2, IntVector2[])

Multiplies each element of the array by the vector.

```
public static IntVector2[] operator *(IntVector2 intVector,
IntVector2[] intVectorArray)
```

Parameters

`intVector` IntVector2

`intVectorArray` IntVector2[]

Returns

IntVector2[]

operator *(IntVector2, int)

Multiplies component-wise.

```
public static IntVector2 operator *(IntVector2 vector, int scalar)
```

Parameters

`vector` IntVector2

`scalar` int

Returns

IntVector2

operator *(IntVector2[], IntVector2)

Multiplies each element of the array by the vector.

```
public static IntVector2[] operator *(IntVector2[] intVectorArray,
IntVector2 intVector)
```

Parameters

`intVectorArray` IntVector2[]

`intVector` IntVector2

Returns

IntVector2[]

operator *(int, IntVector2)

Multiplies component-wise.

```
public static IntVector2 operator *(int scalar, IntVector2 vector)
```

Parameters

`scalar` int

`vector` IntVector2

Returns

IntVector2

operator -(IntVector2, IntVector2)

Subtracts component-wise.

```
public static IntVector2 operator -(IntVector2 a, IntVector2 b)
```

Parameters

a IntVector2

b IntVector2

Returns

IntVector2

operator -(IntVector2, IntVector2[])

Subtracts each element of the array from the vector.

```
public static IntVector2[] operator -(IntVector2 intVector,
IntVector2[] intVectorArray)
```

Parameters

intVector IntVector2

intVectorArray IntVector2[]

Returns

IntVector2[]

operator -(IntVector2[], IntVector2)

Subtracts the vector from each element of the array.

```
public static IntVector2[] operator -(IntVector2[] intVectorArray,
IntVector2 intVector)
```

Parameters

intVectorArray IntVector2[]

intVector IntVector2

Returns

IntVector2[]

operator -(IntVector2)

Negates component-wise.

```
public static IntVector2 operator -(IntVector2 a)
```

Parameters

a IntVector2

Returns

IntVector2

Namespace PAC.Drawing

Classes

DrawingArea

GridManager

Handles drawing a grid over the image when the grid is enabled.

Shapes

A class to define how different shapes are drawn.

Tool

Defines the tools available to use and their properties.

Toolbar

Handles selecting tools, brush size, etc.

Tools

Defines how different tools act. I may rework this to be more like the BlendMode class where each Tool instance defines how it acts. Then to use a tool you would just do tool.Use(pixel) or similar.

Enums

BrushShape

GradientMode

SelectionMode

Shape

Enum BrushShape

Namespace: PAC.Drawing

```
public enum BrushShape
```

Fields

`Circle = 0`

`Custom = -1`

`Diamond = 2`

`Square = 1`

Class DrawingArea

Namespace: PAC.Drawing

```
public class DrawingArea : MonoBehaviour
```

Inheritance

object ← DrawingArea

Properties

file

```
public File file { get; }
```

Property Value

File

hasSelection

```
public bool hasSelection { get; }
```

Property Value

bool

pixelsPerUnit

```
public float pixelsPerUnit { get; }
```

Property Value

float

scrollSpeed

```
public float scrollSpeed { get; }
```

Property Value

float

selectionMask

```
public Texture2D selectionMask { get; }
```

Property Value

Texture2D

selectionRect

```
public IntRect selectionRect { get; }
```

Property Value

IntRect

zoomScrollSpeed

```
public float zoomScrollSpeed { get; }
```

Property Value

float

Methods

DeleteSelection()

```
public void DeleteSelection()
```

PixelsToWorldPos(IntVector2)

Turns the pixel coordinate in the drawing into a world coordinate (the resulting coord is the centre of the pixel in the world).

```
public Vector2 PixelsToWorldPos(IntVector2 pixel)
```

Parameters

`pixel` IntVector2

Returns

Vector2

PixelsToWorldPos(int, int)

Turns the pixel coordinate in the drawing into a world coordinate (the resulting coord is the centre of the pixel in the world).

```
public Vector2 PixelsToWorldPos(int x, int y)
```

Parameters

`x` int

`y` int

Returns

Vector2

PixelsToWorldPos(float, float)

Turns the pixel coordinate in the drawing into a world coordinate.

```
public Vector2 PixelsToWorldPos(float x, float y)
```

Parameters

`x` float

`y` float

Returns

Vector2

PixelsToWorldPos(Vector2)

Turns the pixel coordinate in the drawing into a world coordinate.

```
public Vector2 PixelsToWorldPos(Vector2 pixel)
```

Parameters

`pixel` Vector2

Returns

Vector2

PreviewGradient(IntVector2, IntVector2, Color, Color, GradientMode)

```
public void PreviewGradient(IntVector2 start, IntVector2 end, Color startColour,
Color endColour, GradientMode gradientMode)
```

Parameters

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`gradientMode` GradientMode

UpdateDrawing()

Makes any changes to the file visible on the drawing area.

```
public void UpdateDrawing()
```

WorldPosToPixel(Vector2)

Turns the world coord into a pixel coord in the drawing.

```
public IntVector2 WorldPosToPixel(Vector2 worldPos)
```

Parameters

`worldPos` Vector2

Returns

IntVector2

WorldPosToPixels(float, float)

Turns the world coord into a pixel coord in the drawing.

```
public IntVector2 WorldPosToPixels(float x, float y)
```

Parameters

x float

y float

Returns

IntVector2

Enum GradientMode

Namespace: PAC.Drawing

```
public enum GradientMode
```

Fields

`Linear = 0`

`Radial = 1`

Class GridManager

Namespace: PAC.Drawing

Handles drawing a grid over the image when the grid is enabled.

```
public class GridManager : MonoBehaviour
```

Inheritance

object ← GridManager

Fields

lineThickness

```
public float lineThickness
```

Field Value

float

Properties

height

```
public int height { get; }
```

Property Value

int

on

```
public bool on { get; }
```

Property Value

bool

width

```
public int width { get; }
```

Property Value

int

xOffset

```
public int xOffset { get; }
```

Property Value

int

yOffset

```
public int yOffset { get; }
```

Property Value

int

Methods

SetGrid(int, int, int, int)

```
public void SetGrid(int width, int height, int xOffset, int yOffset)
```

Parameters

`width` int

`height` int

`xOffset` int

`yOffset` int

SetOnOff(bool)

```
public void SetOnOff(bool on)
```

Parameters

`on` bool

SetOnOffNoDisplayUpdate(bool)

```
public void SetOnOffNoDisplayUpdate(bool on)
```

Parameters

`on` bool

Enum SelectionMode

Namespace: PAC.Drawing

```
public enum SelectionMode
```

Fields

`Draw = 0`

`Ellipse = 11`

`MagicWand = 1`

`Rectangle = 10`

Enum Shape

Namespace: PAC.Drawing

```
public enum Shape
```

Fields

`Ellipse = 1`

`Rectangle = 0`

`Triangle = 2`

Class Shapes

Namespace: PAC.Drawing

A class to define how different shapes are drawn.

```
public static class Shapes
```

Inheritance

object ← Shapes

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

Circle(int, int, IntVector2, IntVector2, Color, bool, bool)

```
public static Texture2D Circle(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool filled, bool stayWithinImageBounds)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

`stayWithinImageBounds` bool

Returns

Texture2D

Diamond(int, int, IntVector2, IntVector2, Color, bool)

```
public static Texture2D Diamond(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool filled)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

Returns

Texture2D

Ellipse(int, int, IntVector2, IntVector2, Color, bool)

```
public static Texture2D Ellipse(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool filled)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

Returns

Texture2D

Gradient(int, int, IntVector2, IntVector2, Color, Color, GradientMode)

```
public static Texture2D Gradient(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color startColour, Color endColour, GradientMode gradientMode)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`gradientMode` GradientMode

Returns

Texture2D

GradientLinear(int, int, IntVector2, IntVector2, Color, Color)

```
public static Texture2D GradientLinear(int texWidth, int texHeight, IntVector2
start, IntVector2 end, Color startColour, Color endColour)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

Returns

Texture2D

GradientRadial(int, int, IntVector2, IntVector2, Color, Color)

```
public static Texture2D GradientRadial(int texWidth, int texHeight, IntVector2
start, IntVector2 end, Color startColour, Color endColour)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

Returns

Texture2D

IsoBox(int, int, IntVector2, IntVector2, IntVector2, Color, bool)

```
public static Texture2D IsoBox(int texWidth, int texHeight, IntVector2 baseStart,
IntVector2 baseEnd, IntVector2 heightEnd, Color colour, bool filled)
```

Parameters

`texWidth` int

`texHeight` int

`baseStart` IntVector2

`baseEnd` IntVector2

`heightEnd` IntVector2

`colour` Color

`filled` bool

Returns

Texture2D

IsoRectangle(int, int, IntVector2, IntVector2, Color, bool)

```
public static Texture2D IsoRectangle(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool filled)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

Returns

Texture2D

Line(int, int, IntVector2, IntVector2, Color)

```
public static Texture2D Line(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

Returns

Texture2D

LineCoords(IntVector2, IntVector2)

Gets the coords of a pixel-perfect line between two points, ordered from start to end.

```
public static IntVector2[] LineCoords(IntVector2 start, IntVector2 end)
```

Parameters

`start` IntVector2

`end` IntVector2

Returns

IntVector2[]

Rectangle(int, int, IntVector2, IntVector2, Color, bool)

```
public static Texture2D Rectangle(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool filled)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

Returns

Texture2D

RightTriangle(int, int, IntVector2, IntVector2, Color, bool, bool)

```
public static Texture2D RightTriangle(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool rightAngleOnBottom, bool filled)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`rightAngleOnBottom` bool

`filled` bool

Returns

Texture2D

SnapEndCoordToSquare(IntVector2, IntVector2)

Snaps the end coord so that the rect it forms with the start coord is a square.

```
public static IntVector2 SnapEndCoordToSquare(IntVector2 start, IntVector2 end)
```

Parameters

`start` IntVector2

`end` IntVector2

Returns

IntVector2

SnapEndCoordToSquare(int, int, IntVector2, IntVector2, bool)

Snaps the end coord so that the rect it forms with the start coord is a square.

```
public static IntVector2 SnapEndCoordToSquare(int texWidth, int texHeight,
IntVector2 start, IntVector2 end, bool stayWithinImageBounds)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`stayWithinImageBounds` bool

Returns

IntVector2

Square(int, int, IntVector2, IntVector2, Color, bool, bool)

```
public static Texture2D Square(int texWidth, int texHeight, IntVector2 start,
IntVector2 end, Color colour, bool filled, bool stayWithinImageBounds)
```

Parameters

`texWidth` int

`texHeight` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

`stayWithinImageBounds` bool

Returns

Class Tool

Namespace: PAC.Drawing

Defines the tools available to use and their properties.

```
public class Tool
```

Inheritance

object ← Tool

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

Brush

```
public static Tool Brush
```

Field Value

Tool

EyeDropper

```
public static Tool EyeDropper
```

Field Value

Tool

Fill

```
public static Tool Fill
```

Field Value

Tool

GlobalEyeDropper

```
public static Tool GlobalEyeDropper
```

Field Value

Tool

Gradient

```
public static Tool Gradient
```

Field Value

Tool

IsoBox

```
public static Tool IsoBox
```

Field Value

Tool

Line

```
public static Tool Line
```

Field Value

Tool

Move

```
public static Tool Move
```

Field Value

Tool

None

```
public static Tool None
```

Field Value

Tool

Pencil

```
public static Tool Pencil
```

Field Value

Tool

Rubber

```
public static Tool Rubber
```

Field Value

Tool

Selection

```
public static Tool Selection
```

Field Value

Tool

Shape

```
public static Tool Shape
```

Field Value

Tool

canBeCancelled

```
public bool canBeCancelled
```

Field Value

bool

tools

All implemented tools.

```
public static readonly Tool[] tools
```

Field Value

Tool[]

Properties

finishMode

What action causes a use of the tool to be ended.

```
public MouseTargetDeselectMode finishMode { get; }
```

Property Value

MouseTargetDeselectMode

name

```
public string name { get; }
```

Property Value

string

showBrushBorder

Whether the outline of the brush shape should be shown.

```
public bool showBrushBorder { get; }
```

Property Value

bool

useMovementInterpolation

When the mouse position jumps between frames: true - the tool should be applied to each coord the mouse moved through; false - just applied to the ending coord.

```
public bool useMovementInterpolation { get; }
```

Property Value

bool

Methods

StringToTool(string)

Gets the tool with that name.

```
public static Tool StringToTool(string toolName)
```

Parameters

`toolName` string

Returns

Tool

Class Toolbar

Namespace: PAC.Drawing

Handles selecting tools, brush size, etc.

```
public class Toolbar : MonoBehaviour
```

Inheritance

object ← Toolbar

Properties

brushPixels

The pixels, given relative to the position of the mouse, that will be affected by the current brush.

```
public IntVector2[] brushPixels { get; }
```

Property Value

IntVector2[]

brushPixelsHeight

```
public int brushPixelsHeight { get; }
```

Property Value

int

brushPixelsIsEmpty

```
public bool brushPixelsIsEmpty { get; }
```

Property Value

bool

brushPixelsIsSingleCentralPixel

```
public bool brushPixelsIsSingleCentralPixel { get; }
```

Property Value

bool

brushPixelsWidth

```
public int brushPixelsWidth { get; }
```

Property Value

int

brushShape

```
public BrushShape brushShape { get; set; }
```

Property Value

BrushShape

brushSize

```
public int brushSize { get; }
```

Property Value

int

brushTexture

```
public Texture2D brushTexture { get; }
```

Property Value

Texture2D

gradientMode

```
public GradientMode gradientMode { get; }
```

Property Value

GradientMode

lineSmoothingTime

The amount of time you have to draw a new pixel in for an old one to be potentially smoothed.

```
public float lineSmoothingTime { get; }
```

Property Value

float

previousTool

```
public Tool previousTool { get; }
```

Property Value

Tool

selectedTool

```
public Tool selectedTool { get; }
```

Property Value

Tool

selectionMode

```
public SelectionMode selectionMode { get; }
```

Property Value

SelectionMode

shapeToolShape

```
public Shape shapeToolShape { get; }
```

Property Value

Shape

Methods

DeselectGlobalEyeDropper()

```
public void DeselectGlobalEyeDropper()
```

LoadCustomBrush(Texture2D)

Turns the given texture into a brush shape by taking any pixels with non-zero alpha.

```
public void LoadCustomBrush(Texture2D brushShape)
```

Parameters

`brushShape` Texture2D

SelectGlobalEyeDropper()

```
public void SelectGlobalEyeDropper()
```

SetBrushSize(int)

```
public bool SetBrushSize(int brushSize)
```

Parameters

`brushSize` int

Returns

bool

SubscribeToOnBrushPixelsChanged(UnityAction)

Event invoked when brush pixels change.

```
public void SubscribeToOnBrushPixelsChanged(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnBrushSizeChanged(UnityAction)

Event invoked when brush size changes.

```
public void SubscribeToOnBrushSizeChanged(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnToolChanged(UnityAction)

Event invoked when selected tool changes.

```
public void SubscribeToOnToolChanged(UnityAction call)
```

Parameters

`call` UnityAction

Class Tools

Namespace: PAC.Drawing

Defines how different tools act. I may rework this to be more like the BlendMode class where each Tool instance defines how it acts. Then to use a tool you would just do tool.Use(pixel) or similar.

```
public static class Tools
```

Inheritance

object ← Tools

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

PencilLineSmoothing(File, int, int, IntVector2[], IntVector2[], Color)

Smooth the meeting point of the two lines (given as coords) so it is pixel-perfect - i.e. no hard 90-degree corner.

```
public static bool PencilLineSmoothing(File file, int layer, int frame, IntVector2[]
line, IntVector2[] previousLine, Color colourLineMeetingPoint)
```

Parameters

`file` File

`layer` int

`frame` int

`line` IntVector2[]

`previousLine` IntVector2[]

`colourLineMeetingPoint` Color

Returns

bool

UseBrush(File, int, int, IntVector2, IntVector2[], Color)

```
public static void UseBrush(File file, int layer, int frame, IntVector2 pixel,
IntVector2[] brushBorderMaskPixels, Color colour)
```

Parameters

`file` File

`layer` int

`frame` int

`pixel` IntVector2

`brushBorderMaskPixels` IntVector2[]

`colour` Color

UseBrush(File, int, int, int, int, IntVector2[], Color)

```
public static void UseBrush(File file, int layer, int frame, int x, int y,
IntVector2[] brushBorderMaskPixels, Color colour)
```

Parameters

`file` File

`layer` int

`frame` int

`x` int

`y` int

`brushBorderMaskPixels` IntVector2[]

`colour` Color

UseCircle(File, int, int, IntVector2, IntVector2, Color, bool, bool)

```
public static void UseCircle(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color colour, bool filled, bool stayWithinImageBounds)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

`stayWithinImageBounds` bool

UseEllipse(File, int, int, IntVector2, IntVector2, Color, bool)

```
public static void UseEllipse(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color colour, bool filled)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

UseEyeDropper(File, int, int, IntVector2)

```
public static Color UseEyeDropper(File file, int layer, int frame, IntVector2 pixel)
```

Parameters

`file` File

`layer` int

`frame` int

`pixel` IntVector2

Returns

Color

UseEyeDropper(File, int, int, int, int)

```
public static Color UseEyeDropper(File file, int layer, int frame, int x, int y)
```

Parameters

`file` File

`layer` int

`frame` int

`x` int

`y` int

Returns

Color

UseFill(File, int, int, IntVector2, Color, int)

```
public static void UseFill(File file, int layer, int frame, IntVector2 pixel, Color
colour, int maxNumOfIterations = 1000000)
```

Parameters

`file` File

`layer` int

`frame` int

`pixel` IntVector2

`colour` Color

`maxNumOfIterations` int

UseFill(File, int, int, int, int, Color, int)

```
public static void UseFill(File file, int layer, int frame, int x, int y, Color
colour, int maxNumOfIterations = 1000000)
```

Parameters

`file` File

`layer` int

`frame` int

`x` int

`y` int

`colour` Color

`maxNumOfIterations` int

UseGradient(File, int, int, IntVector2, IntVector2, Color, Color, GradientMode)

```
public static void UseGradient(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color startColour, Color endColour, GradientMode gradientMode)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`gradientMode` GradientMode

UseGradient(File, int, int, IntVector2, IntVector2, Color, Color, GradientMode, IntVector2[])

```
public static void UseGradient(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color startColour, Color endColour, GradientMode gradientMode,
```

```
IntVector2[] mask)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`gradientMode` GradientMode

`mask` IntVector2[]

UseGradient(File, int, int, IntVector2, IntVector2, Color, Color, GradientMode, Texture2D)

```
public static void UseGradient(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color startColour, Color endColour, GradientMode gradientMode,
Texture2D mask)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`gradientMode` GradientMode

`mask` Texture2D

UseGradientLinear(File, int, int, IntVector2, IntVector2, Color, Color)

```
public static void UseGradientLinear(File file, int layer, int frame, IntVector2 start, IntVector2 end, Color startColour, Color endColour)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

UseGradientLinear(File, int, int, IntVector2, IntVector2, Color, Color, IntVector2[])

```
public static void UseGradientLinear(File file, int layer, int frame, IntVector2 start, IntVector2 end, Color startColour, Color endColour, IntVector2[] mask)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`mask` IntVector2[]

UseGradientLinear(File, int, int, IntVector2, IntVector2, Color, Color, Texture2D)

```
public static void UseGradientLinear(File file, int layer, int frame, IntVector2
start, IntVector2 end, Color startColour, Color endColour, Texture2D mask)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`mask` Texture2D

UseGradientRadial(File, int, int, IntVector2, IntVector2, Color, Color)

```
public static void UseGradientRadial(File file, int layer, int frame, IntVector2
```

```
start, IntVector2 end, Color startColour, Color endColour)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

UseGradientRadial(File, int, int, IntVector2, IntVector2, Color, Color, IntVector2[])

```
public static void UseGradientRadial(File file, int layer, int frame, IntVector2
start, IntVector2 end, Color startColour, Color endColour, IntVector2[] mask)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`mask` IntVector2[]

UseGradientRadial(File, int, int, IntVector2, IntVector2, Color, Color, Texture2D)

```
public static void UseGradientRadial(File file, int layer, int frame, IntVector2 start, IntVector2 end, Color startColour, Color endColour, Texture2D mask)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`startColour` Color

`endColour` Color

`mask` Texture2D

UseIsoBox(File, int, int, IntVector2, IntVector2, IntVector2, Color, bool)

```
public static void UseIsoBox(File file, int layer, int frame, IntVector2 baseStart, IntVector2 baseEnd, IntVector2 heightEnd, Color colour, bool filled)
```

Parameters

`file` File

`layer` int

`frame` int

`baseStart` IntVector2

`baseEnd` IntVector2

`heightEnd` IntVector2

`colour` Color

`filled` bool

UseIsoRectangle(File, int, int, IntVector2, IntVector2, Color, bool)

```
public static void UseIsoRectangle(File file, int layer, int frame, IntVector2 start, IntVector2 end, Color colour, bool filled)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

UseLine(File, int, int, IntVector2, IntVector2, Color)

```
public static void UseLine(File file, int layer, int frame, IntVector2 start, IntVector2 end, Color colour)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

UsePencil(File, int, int, IntVector2, Color)

```
public static void UsePencil(File file, int layer, int frame, IntVector2 pixel,
Color colour)
```

Parameters

`file` File

`layer` int

`frame` int

`pixel` IntVector2

`colour` Color

UsePencil(File, int, int, int, int, Color)

```
public static void UsePencil(File file, int layer, int frame, int x, int y,
Color colour)
```

Parameters

`file` File

`layer` int

`frame` int

`x` int

`y` int

`colour` Color

UseRectangle(File, int, int, IntVector2, IntVector2, Color, bool)

```
public static void UseRectangle(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color colour, bool filled)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

UseRightTriangle(File, int, int, IntVector2, IntVector2, Color, bool, bool)

```
public static void UseRightTriangle(File file, int layer, int frame, IntVector2
start, IntVector2 end, Color colour, bool rightAngleOnBottom, bool filled)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

`rightAngleOnBottom` bool

`filled` bool

UseRubber(File, int, int, IntVector2)

```
public static void UseRubber(File file, int layer, int frame, IntVector2 pixel)
```

Parameters

`file` File

`layer` int

`frame` int

`pixel` IntVector2

UseRubber(File, int, int, IntVector2, IntVector2[])

```
public static void UseRubber(File file, int layer, int frame, IntVector2 pixel,
IntVector2[] brushBorderMaskPixels)
```

Parameters

`file` File

`layer` int

`frame` int

`pixel` IntVector2

`brushBorderMaskPixels` IntVector2[]

UseRubber(File, int, int, int, int)

```
public static void UseRubber(File file, int layer, int frame, int x, int y)
```

Parameters

`file` File

`layer` int

`frame` int

`x` int

`y` int

UseRubber(File, int, int, int, int, IntVector2[])

```
public static void UseRubber(File file, int layer, int frame, int x, int y,
IntVector2[] brushBorderMaskPixels)
```

Parameters

`file` File

`layer` int

`frame` int

`x` int

`y` int

`brushBorderMaskPixels` IntVector2[]

UseSquare(File, int, int, IntVector2, IntVector2, Color, bool, bool)

```
public static void UseSquare(File file, int layer, int frame, IntVector2 start,
IntVector2 end, Color colour, bool filled, bool stayWithinImageBounds)
```

Parameters

`file` File

`layer` int

`frame` int

`start` IntVector2

`end` IntVector2

`colour` Color

`filled` bool

`stayWithinImageBounds` bool

Namespace PAC.EffectPanels

Classes

BlurPanel

A panel that blurs the view behind it.

EffectPanel

A class to represent a type of panel that applies an effect to the view behind it - e.g. a blur panel or a pixellate panel.

PixellatePanel

A panel that pixellates the view behind it.

Class BlurPanel

Namespace: PAC.EffectPanels

A panel that blurs the view behind it.

```
public class BlurPanel : MonoBehaviour
```

Inheritance

object ← BlurPanel

Properties

blurEnabled

```
public bool blurEnabled { get; }
```

Property Value

bool

Methods

EnableDisable(bool)

```
public void EnableDisable(bool enabled)
```

Parameters

`enabled` bool

Class EffectPanel

Namespace: PAC.EffectPanels

A class to represent a type of panel that applies an effect to the view behind it - e.g. a blur panel or a pixellate panel.

```
public class EffectPanel : MonoBehaviour
```

Inheritance

object ← EffectPanel

Methods

EnableDisable(bool)

```
public void EnableDisable(bool enabled)
```

Parameters

`enabled` bool

Class PixellatePanel

Namespace: PAC.EffectPanels

A panel that pixellates the view behind it.

```
public class PixellatePanel : MonoBehaviour
```

Inheritance

object ← PixellatePanel

Properties

pixellateOn

```
public bool pixellateOn { get; }
```

Property Value

bool

Namespace PAC.Files

Classes

File

A class to represent a single Pixel Art Creator file.

FileManager

FileTab

FileTabsManager

ImageEditManager

Class File

Namespace: PAC.Files

A class to represent a single Pixel Art Creator file.

```
public class File : IJSONable
```

Inheritance

object ← File

Implements

IJSONable

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

File(File)

Creates a deep copy of the File.

```
public File(File file)
```

Parameters

`file` File

File(string, int, int)

Creates a blank file.

```
public File(string name, int width, int height)
```

Parameters

`name` string

`width` int

`height` int

Properties

height

```
public int height { get; }
```

Property Value

int

keyFrameIndices

The frame indices at which some layer has a key frame.

```
public int[] keyFrameIndices { get; }
```

Property Value

int[]

layers

```
public List<Layer> layers { get; }
```

Property Value

List<Layer>

liveRender

```
public Texture2D liveRender { get; }
```

Property Value

Texture2D

mostRecentSavePath

```
public string mostRecentSavePath { get; }
```

Property Value

string

name

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

Property Value

string

numOfFrames

The number of frames the animation lasts for.

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

Property Value

int

rect

```
public IntRect rect { get; }
```

Property Value

IntRect

savedSinceLastEdit

```
public bool savedSinceLastEdit { get; }
```

Property Value

bool

tiles

All the tile objects currently in the file.

```
public Tile[] tiles { get; }
```

Property Value

Tile[]

width

```
public int width { get; }
```

Property Value

int

Methods

AddLayer(Layer)

Adds the given layer on top of all existing layers.

```
public void AddLayer(Layer layer)
```

Parameters

`layer` Layer

AddLayer(Layer, int)

Adds the given layer at the given index. Throws an error if the layer is already in the file.

```
public void AddLayer(Layer layer, int index)
```

Parameters

`layer` Layer

`index` int

AddLayers(params Layer[])

Adds the layers on top of all existing layers, retaining the order they have in the array - i.e. first layer in the array is the one on top, etc.

```
public void AddLayers(params Layer[] layers)
```

Parameters

`layers` Layer[]

AddNormalLayer()

Adds a blank normal layer on top of all existing layers.

```
public void AddNormalLayer()
```

AddNormalLayer(int)

Adds a blank normal layer at the given index.

```
public void AddNormalLayer(int index)
```

Parameters

`index` int

AddNormalLayer(Texture2D, int)

Adds a normal layer at the given index with the given texture.

```
public void AddNormalLayer(Texture2D texture, int index = 0)
```

Parameters

`texture` Texture2D

`index` int

AddTile(File, IntVector2, int)

Adds the given file as a tile.

```
public void AddTile(File file, IntVector2 bottomLeft, int lowestLayer)
```

Parameters

`file` File

`bottomLeft` IntVector2

`lowestLayer` int

AddTile(Tile)

Adds the tile to the file.

```
public void AddTile(Tile tile)
```

Parameters

`tile` Tile

AddTileLayer()

Adds a blank tile layer on top of all existing layers.

```
public void AddTileLayer()
```

AddTileLayer(int)

Adds a blank tile layer at the given index.

```
public void AddTileLayer(int index)
```

Parameters

`index` int

ContainsTile(Tile)

```
public bool ContainsTile(Tile tile)
```

Parameters

`tile` Tile

Returns

bool

DeepCopy()

Creates a deep copy of the File.

```
public File DeepCopy()
```

Returns

File

ExportAnimation(string)

Exports each frame of the animation and puts them in a folder. The file path specifies the folder name and location, and the file extension to export each frame as.

```
public bool ExportAnimation(string filePath)
```

Parameters

`filePath` string

Returns

bool

ExportFrame(int, string)

Exports the frame to the file path, according to the file extension.

```
public bool ExportFrame(int frame, string filePath)
```

Parameters

`frame` int

`filePath` string

Returns

bool

Extend(int, int, int, int)

Extends the dimensions of the file in each direction by the given amounts.

```
public void Extend(int left, int right, int up, int down)
```

Parameters

`left` int

`right` int

`up` int

`down` int

Flip(FlipDirection)

Flips the file.

```
public void Flip(FlipDirection direction)
```

Parameters

`direction` FlipDirection

FromJSON(JSON)

Gets a file from its JSON representation.

```
public static File FromJSON(JSON json)
```

Parameters

`json` JSON

Returns

File

ImportImage(string)

Adds the image at the file path as a new layer.

```
public void ImportImage(string filePath)
```

Parameters

`filePath` string

ImportLayers(params Layer[])

Adds deep copies of the layers on top of all existing layers, retaining the order they have in the array - i.e. first layer in the array is the one on top, etc.

```
public void ImportLayers(params Layer[] layers)
```

Parameters

`layers` Layer[]

IndexOfLayer(Layer)

Gets the index of the layer within the file.

```
public int IndexOfLayer(Layer layer)
```

Parameters

`layer` Layer

Returns

int

The index of the layer, or -1 if the layer is not in the file.

IsBlank()

Returns true if and only if all pixels on all layers are completely transparent.

```
public bool IsBlank()
```

Returns

bool

MoveLayer(Layer, int)

Moves the layer to indexToMoveTo. Throws an error if the layer is not in the file.

```
public void MoveLayer(Layer layerToMove, int indexToMoveTo)
```

Parameters

`layerToMove` Layer

`indexToMoveTo` int

MoveLayer(int, int)

Moves the layer at indexToMove to indexToMoveTo.

```
public void MoveLayer(int layerToMove, int indexToMoveTo)
```

Parameters

`layerToMove` int

`indexToMoveTo` int

OpenFile(string)

Opens the file according to its file extension.

```
public static File OpenFile(string filePath)
```

Parameters

`filePath` string

Returns

File

OpenImage(string)

Opens the file at the file path if it is an image file type, e.g. PNG or JPEG.

```
public static File OpenImage(string filePath)
```

Parameters

`filePath` string

Returns

File

OpenJPEG(string)

Opens the JPEG/JPG file. Throws an error if the file is not a JPEG/JPG file.

```
public static File OpenJPEG(string filePath)
```

Parameters

`filePath` string

Returns

File

OpenPAC(string)

Opens the PAC file. Throws an error if the file is not a PAC file.

```
public static File OpenPAC(string filePath)
```

Parameters

`filePath` string

Returns

File

OpenPNG(string)

Opens the PNG file. Throws an error if the file is not a PNG file.

```
public static File OpenPNG(string filePath)
```

Parameters

`filePath` string

Returns

File

RemoveAllLayers()

Removes all layers, then adds a blank normal layer.

```
public void RemoveAllLayers()
```

RemoveLayer(Layer)

Removes the layer from the file. Throws an error if the layer is not in the file.

```
public void RemoveLayer(Layer layer)
```

Parameters

`layer` Layer

RemoveLayer(int)

Removes the layer at the given index.

```
public void RemoveLayer(int layer)
```

Parameters

`layer` int

RemoveLayers(Layer[])

Removes the given layers from the file. Throws an error if a layer is not in the file.

```
public void RemoveLayers(Layer[] layers)
```

Parameters

`layers` Layer[]

RemoveLayers(int[])

Removes the layers at the given indices.

```
public void RemoveLayers(int[] layers)
```

Parameters

`layers` int[]

RemoveTile(Tile)

Removes the tile. Throws an error if the tile is not in the file.

```
public void RemoveTile(Tile tile)
```

Parameters

`tile` Tile

Render(int)

Renders all layers down to a Texture2D. Does not apply the texture.

```
public Texture2D Render(int frame)
```

Parameters

`frame` int

Returns

Texture2D

RenderLayers(int, int, int, bool)

Renders all the layers with index between lowestLayer and highestLayer (inclusive iff inclusive == true) down to a Texture2D. highestLayer and lowestLayer are automatically clamped to be in the valid range of layers. Does not apply the texture.

```
public Texture2D RenderLayers(int highestLayer, int lowestLayer, int frame, bool
inclusive = true)
```

Parameters

`highestLayer` int

The higher layer (so lower index).

`lowestLayer` int

The lower layer (so higher index).

`frame` int

`inclusive` bool

Returns

Texture2D

RenderLayers(int[], int)

Renders the layers at the given layer indices down to a Texture2D. Does not apply the texture.

```
public Texture2D RenderLayers(int[] layerIndices, int frame)
```

Parameters

`layerIndices` int[]

The layer indices in the order you want them to be rendered, from highest layer (so lowest index) to lowest.

`frame` int

Returns

Texture2D

RenderLayersAbove(int, int, bool)

Renders all layers that appear above the given layer (includes the layer iff inclusive == true) down to a Texture2D. Returns a blank texture if there are none. Does not apply the texture.

```
public Texture2D RenderLayersAbove(int layer, int frame, bool inclusive = false)
```

Parameters

`layer` int

`frame` int

`inclusive` bool

Returns

Texture2D

RenderLayersBelow(int, int, bool)

Renders all layers that appear strictly beneath the given layer (includes the layer iff inclusive == true) down to a Texture2D. Returns a blank texture if there are none. Does not apply the texture.

```
public Texture2D RenderLayersBelow(int layer, int frame, bool inclusive = false)
```

Parameters

`layer` int

`frame` int

`inclusive` bool

Returns

Texture2D

RenderPixel(IntVector2, int)

Renders the colour of the given pixel.

```
public Color RenderPixel(IntVector2 pixel, int frame)
```

Parameters

`pixel` IntVector2

`frame` int

Returns

Color

RenderPixel(IntVector2, int, int, int, bool)

```
public Color RenderPixel(IntVector2 pixel, int highestLayer, int lowestLayer, int
frame, bool inclusive = true)
```

Parameters

`pixel` IntVector2

`highestLayer` int

`lowestLayer` int

`frame` int

`inclusive` bool

Returns

Color

RenderPixel(IntVector2, int[], int)

Renders the colour of the pixel on the layers at the given layer indices. Throws an error if there are no layer indices.

```
public Color RenderPixel(IntVector2 pixel, int[] layerIndices, int frame)
```

Parameters

`pixel` IntVector2

`layerIndices` int[]

`frame` int

Returns

Color

RenderPixel(int, int, int)

Renders the colour of the given pixel.

```
public Color RenderPixel(int x, int y, int frame)
```

Parameters

`x` int

`y` int

`frame` int

Returns

Color

RenderPixel(int, int, int, int, int, bool)

```
public Color RenderPixel(int x, int y, int highestLayer, int lowestLayer, int frame,
bool inclusive = true)
```

Parameters

`x` int

`y` int

`highestLayer` int

`lowestLayer` int

`frame` int

`inclusive` bool

Returns

Color

RenderPixel(int, int, int[], int)

Renders the colour of pixel (x, y) on the layers at the given layer indices. Throws an error if there are no layer indices.

```
public Color RenderPixel(int x, int y, int[] layerIndices, int frame)
```

Parameters

`x` int

`y` int

`layerIndices` int[]

`frame` int

Returns

Color

ReplaceLayer(Layer, Layer)

Replaces the layer with the given layer. Throws an error if the layer to replace is not in the file, or if the layer to replace with is already in the file.

```
public void ReplaceLayer(Layer layerToReplace, Layer layerToReplaceWith)
```

Parameters

`layerToReplace` Layer

`layerToReplaceWith` Layer

ReplaceLayer(int, Layer)

Replaces the layer at the given index with the given layer. Throws an error if the layer to replace with is already in the file.

```
public void ReplaceLayer(int layerToReplace, Layer layerToReplaceWith)
```

Parameters

`layerToReplace` int

`layerToReplaceWith` Layer

Rotate(RotationAngle)

Rotates the file. Rotation is clockwise.

```
public void Rotate(RotationAngle angle)
```

Parameters

`angle` RotationAngle

SaveAsPAC(string)

Saves the file as a PAC file at the given file path. Throws an error if the file is not a PAC file.

```
public void SaveAsPAC(string filePath)
```

Parameters

`filePath` string

SavePAC()

Saves the file as a PAC file at the location it was most recently saved at. Throws an error if the file hasn't been saved before.

```
public void SavePAC()
```

Scale(int, int)

Resizes the file to the new width and height.

```
public void Scale(int newWidth, int newHeight)
```

Parameters

`newWidth` int

`newHeight` int

Scale(float)

Resizes the dimensions of the file by the scale factor.

```
public void Scale(float scaleFactor)
```

Parameters

`scaleFactor` float

Scale(float, float)

Resizes the dimensions of the file by the scale factors.

```
public void Scale(float xScaleFactor, float yScaleFactor)
```

Parameters

`xScaleFactor` float

`yScaleFactor` float

SetLiveRenderFrame(int)

Sets the frame that the live render will be for.

```
public void SetLiveRenderFrame(int frame)
```

Parameters

`frame` int

SubscribeToOnPixelsChanged(UnityAction<IntVector2[], Layer[], int[]>)

Event invoked when some pixels have been changed on a layer. Passes the pixels, layers and frames that were affected.

```
public void SubscribeToOnPixelsChanged(UnityAction<IntVector2[], Layer[],
int[]> call)
```

Parameters

`call` UnityAction<IntVector2[], Layer[], int[]>

SubscribeToOnSavedSinceEditChanged(UnityAction)

Event invoked when the a change to file is made or when the file is saved.

```
public void SubscribeToOnSavedSinceEditChanged(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnTileAdded(UnityAction)

Event invoked when a tile is added to the file.

```
public void SubscribeToOnTileAdded(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnTileRemoved(UnityAction)

Event invoked when a tile is removed from the file.

```
public void SubscribeToOnTileRemoved(UnityAction call)
```

Parameters

`call` UnityAction

ToJSON()

Gets a JSON representation of the file.

```
public JSON ToJSON()
```

Returns

JSON

Class FileManager

Namespace: PAC.Files

```
public class FileManager : MonoBehaviour
```

Inheritance

object ← FileManager

Fields

defaultFileName

```
public const string defaultFileName = "Untitled Image"
```

Field Value

string

previousFile

```
public File previousFile
```

Field Value

File

Properties

currentFile

```
public File currentFile { get; }
```

Property Value

File

currentFileIndex

```
public int currentFileIndex { get; }
```

Property Value

int

files

```
public List<File> files { get; }
```

Property Value

List<File>

Methods

AddFile(File)

```
public bool AddFile(File file)
```

Parameters

`file` File

Returns

bool

CloseFile(File)

Closes the file, with no checks for unsaved changes.

```
public bool CloseFile(File file)
```

Parameters

`file` File

Returns

bool

CloseFile(int)

Closes the file, with no checks for unsaved changes.

```
public File CloseFile(int fileIndex)
```

Parameters

`fileIndex` int

Returns

File

ExportAnimation(int, string)

```
public bool ExportAnimation(int fileIndex, string filePath)
```

Parameters

`fileIndex` int

`filePath` string

Returns

bool

ExportCurrentAnimation(string)

```
public bool ExportCurrentAnimation(string filePath)
```

Parameters

`filePath` string

Returns

bool

ExportCurrentAnimationDialog()

```
public void ExportCurrentAnimationDialog()
```

ExportCurrentFrame(int, string)

```
public bool ExportCurrentFrame(int frameIndex, string filePath)
```

Parameters

`frameIndex` int

`filePath` string

Returns

bool

ExportCurrentFrameDialog()

```
public void ExportCurrentFrameDialog()
```

ExportFrame(int, int, string)

```
public bool ExportFrame(int frameIndex, int fileIndex, string filePath)
```

Parameters

`frameIndex` int

`fileIndex` int

`filePath` string

Returns

bool

ImportDialog()

```
public void ImportDialog()
```

NewFile(int, int)

```
public void NewFile(int width, int height)
```

Parameters

`width` int

`height` int

OpenFile(File)

```
public bool OpenFile(File file)
```

Parameters

`file` File

Returns

bool

OpenFile(string)

```
public bool OpenFile(string filePath)
```

Parameters

`filePath` string

Returns

bool

OpenFileDialog()

```
public void OpenFileDialog()
```

ReloadFile()

```
public void ReloadFile()
```

SaveAsCurrentFile(string)

```
public void SaveAsCurrentFile(string filePath)
```

Parameters

`filePath` string

SaveAsCurrentFileDialog()

```
public void SaveAsCurrentFileDialog()
```

SaveAsFile(File, string)

```
public void SaveAsFile(File file, string filePath)
```

Parameters

`file` File

`filePath` string

SaveAsFile(int, string)

```
public void SaveAsFile(int fileIndex, string filePath)
```

Parameters

`fileIndex` int

`filePath` string

SaveAsFileDialog(File)

```
public void SaveAsFileDialog(File file)
```

Parameters

`file` File

SaveCurrentFileDialog()

```
public void SaveCurrentFileDialog()
```

SaveFileDialog(File)

```
public void SaveFileDialog(File file)
```

Parameters

`file` File

SubscribeToFileSwitched(UnityAction)

```
public void SubscribeToFileSwitched(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToFilesChanged(UnityAction)

```
public void SubscribeToFilesChanged(UnityAction call)
```

Parameters

`call` UnityAction

SwitchToFile(int)

```
public void SwitchToFile(int fileIndex)
```

Parameters

`fileIndex` int

TryCloseFile(File)

Will close the file if there are no unsaved changes. Otherwise it will open a dialog box asking if you want to save.

```
public bool TryCloseFile(File file)
```

Parameters

`file` File

Returns

bool

TryCloseFile(int)

Will close the file if there are no unsaved changes. Otherwise it will open a dialog box asking if you want to save.

```
public bool TryCloseFile(int fileIndex)
```

Parameters

`fileIndex` int

Returns

bool

Class FileTab

Namespace: PAC.Files

```
public class FileTab : MonoBehaviour
```

Inheritance

object ← FileTab

Properties

closed

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

Property Value

bool

file

```
public File file { get; }
```

Property Value

File

selected

```
public bool selected { get; }
```

Property Value

bool

tileToggle

```
public UIToggleButton tileToggle { get; }
```

Property Value

UIToggleButton

Methods

SetFile(File)

```
public void SetFile(File file)
```

Parameters

`file` File

SubscribeToClose(UnityAction)

```
public void SubscribeToClose(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToNameChange(UnityAction)

```
public void SubscribeToNameChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToSelect(UnityAction)

```
public void SubscribeToSelect(UnityAction call)
```

Parameters

`call` UnityAction

Class FileTabsManager

Namespace: PAC.Files

```
public class FileTabsManager : MonoBehaviour
```

Inheritance

object ← FileTabsManager

Properties

selectedFile

```
public File selectedFile { get; }
```

Property Value

File

selectedFileIndex

```
public int selectedFileIndex { get; }
```

Property Value

int

Methods

CloseFile()

```
public void CloseFile()
```

OnFilesChanged()

```
public void OnFilesChanged()
```

SelectFile()

```
public void SelectFile()
```

SubscribeToOnFilesChanged(UnityAction)

```
public void SubscribeToOnFilesChanged(UnityAction call)
```

Parameters

`call` UnityAction

Class ImageEditManager

Namespace: PAC.Files

```
public class ImageEditManager : MonoBehaviour
```

Inheritance

object ← ImageEditManager

Methods

ExtendFile(int, int, int, int)

```
public void ExtendFile(int left, int right, int up, int down)
```

Parameters

`left` int

`right` int

`up` int

`down` int

ExtendFileDown(int)

```
public void ExtendFileDown(int amount)
```

Parameters

`amount` int

ExtendFileLeft(int)

```
public void ExtendFileLeft(int amount)
```

Parameters

`amount` int

ExtendFileRight(int)

```
public void ExtendFileRight(int amount)
```

Parameters

`amount` int

ExtendFileUp(int)

```
public void ExtendFileUp(int amount)
```

Parameters

`amount` int

FlipFile(FlipDirection)

```
public void FlipFile(FlipDirection direction)
```

Parameters

`direction` FlipDirection

FlipFileX()

```
public void FlipFileX()
```

FlipFileY()

```
public void FlipFileY()
```

FlipSelectedLayers(FlipDirection)

```
public void FlipSelectedLayers(FlipDirection direction)
```

Parameters

`direction` FlipDirection

FlipSelectedLayersX()

```
public void FlipSelectedLayersX()
```

FlipSelectedLayersY()

```
public void FlipSelectedLayersY()
```

RotateFile(RotationAngle)

```
public void RotateFile(RotationAngle angle)
```

Parameters

`angle` RotationAngle

RotateSelectedLayers(RotationAngle)

```
public void RotateSelectedLayers(RotationAngle angle)
```

Parameters

`angle` RotationAngle

RotateSelectedLayers180()

```
public void RotateSelectedLayers180()
```

RotateSelectedLayers90()

```
public void RotateSelectedLayers90()
```

RotateSelectedLayersMinus90()

```
public void RotateSelectedLayersMinus90()
```

ScaleFile(int, int)

```
public void ScaleFile(int newWidth, int newHeight)
```

Parameters

`newWidth` int

`newHeight` int

ScaleFile(float)

```
public void ScaleFile(float scaleFactor)
```

Parameters

`scaleFactor` float

ScaleFile(float, float)

```
public void ScaleFile(float xScaleFactor, float yScaleFactor)
```

Parameters

`xScaleFactor` float

`yScaleFactor` float

SubscribeToEdit(UnityAction)

```
public void SubscribeToEdit(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToImageSizeChanged(UnityAction)

```
public void SubscribeToImageSizeChanged(UnityAction call)
```

Parameters

`call` UnityAction

Namespace PAC.Input

Classes

InputSystem

InputTarget

Keyboard

KeyboardTarget

Mouse

MouseTarget

Tooltip

Enums

CursorState

MouseButton

MouseTargetDeselectMode

MouseTargetState

Enum CursorState

Namespace: PAC.Input

```
public enum CursorState
```

Fields

```
CrossArrows = 7

CurrentTool = -1

EyeDropper = 5

Grab = 12

Hover = 2

Invisible = 4

LeftRightArrow = 9

LinearGradient = 30

MagicWand = 11

MagnifyingGlass = 10

Normal = 1

Press = 3

RadialGradient = 31

SelectionEllipse = 101

SelectionRectangle = 100

Text = 6

Unspecified = 0

UpDownArrow = 8
```

Class InputSystem

Namespace: PAC.Input

```
public class InputSystem : MonoBehaviour
```

Inheritance

object ← InputSystem

Fields

timeBetweenKeySpam

```
public float timeBetweenKeySpam
```

Field Value

float

timeUntilKeySpam

```
public float timeUntilKeySpam
```

Field Value

float

Properties

globalInputTarget

```
public InputTarget globalInputTarget { get; }
```

Property Value

InputTarget

globalKeyboardTarget

```
public KeyboardTarget globalKeyboardTarget { get; }
```

Property Value

KeyboardTarget

globalMouseTarget

```
public MouseTarget globalMouseTarget { get; }
```

Property Value

MouseTarget

hasInputTarget

```
public bool hasInputTarget { get; }
```

Property Value

bool

inputTarget

```
public InputTarget inputTarget { get; }
```

Property Value

InputTarget

keyboard

```
public Keyboard keyboard { get; }
```

Property Value

Keyboard

mouse

```
public Mouse mouse { get; }
```

Property Value

Mouse

Methods

LockForAFrame()

```
public void LockForAFrame()
```

SubscribeToGlobalKeyboard(UnityAction)

```
public void SubscribeToGlobalKeyboard(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToGlobalLeftClick(UnityAction)

```
public void SubscribeToGlobalLeftClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToGlobalMouseScroll(UnityAction)

```
public void SubscribeToGlobalMouseScroll(UnityAction call)
```

Parameters

`call` UnityAction

Target(InputTarget)

```
public bool Target(InputTarget newTarget)
```

Parameters

`newTarget` InputTarget

Returns

bool

Untarget()

```
public InputTarget Untarget()
```

Returns

InputTarget

Class InputTarget

Namespace: PAC.Input

```
public class InputTarget : MonoBehaviour
```

Inheritance

object ← InputTarget

Fields

allowHoldingKeySpam

```
public bool allowHoldingKeySpam
```

Field Value

bool

allowLeftClick

```
public bool allowLeftClick
```

Field Value

bool

allowMiddleClick

```
public bool allowMiddleClick
```

Field Value

bool

allowRightClick

```
public bool allowRightClick
```

Field Value

bool

allowScroll

```
public bool allowScroll
```

Field Value

bool

cursorHover

```
public CursorState cursorHover
```

Field Value

CursorState

cursorPress

```
public CursorState cursorPress
```

Field Value

CursorState

cursorSelected

```
public CursorState cursorSelected
```

Field Value

CursorState

disableMouseWhenSelected

```
public bool disableMouseWhenSelected
```

Field Value

bool

isHoverTrigger

```
public bool isHoverTrigger
```

Field Value

bool

keyboardInputEnabled

```
public bool keyboardInputEnabled
```

Field Value

bool

mouseDeselectMode

```
public MouseTargetDeselectMode mouseDeselectMode
```

Field Value

MouseTargetDeselectMode

mouseInputEnabled

```
public bool mouseInputEnabled
```

Field Value

bool

receiveAlreadyHeldKeys

```
public bool receiveAlreadyHeldKeys
```

Field Value

bool

uiElement

```
public UIElement uiElement
```

Field Value

UIElement

viewport

```
public UIViewport viewport
```

Field Value

UIViewport

Properties

collider

```
public Collider2D collider { get; }
```

Property Value

Collider2D

cursorScroll

```
public CursorState cursorScroll { get; set; }
```

Property Value

CursorState

keyboardTarget

```
public KeyboardTarget keyboardTarget { get; }
```

Property Value

KeyboardTarget

mouseTarget

```
public MouseTarget mouseTarget { get; }
```

Property Value

MouseTarget

targetName

```
public string targetName { get; }
```

Property Value

string

targetTag

```
public int targetTag { get; }
```

Property Value

int

Methods

GetUntargeted()

```
public void GetUntargeted()
```

SubscribeToTarget(UnityAction)

```
public void SubscribeToTarget(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToUntarget(UnityAction)

```
public void SubscribeToUntarget(UnityAction call)
```

Parameters

`call` UnityAction

Target()

```
public void Target()
```

Untarget()

```
public void Untarget()
```

Class Keyboard

Namespace: PAC.Input

```
public class Keyboard : MonoBehaviour
```

Inheritance

object ← Keyboard

Fields

canInteract

```
public bool canInteract
```

Field Value

bool

Class KeyboardTarget

Namespace: PAC.Input

```
public class KeyboardTarget
```

Inheritance

object ← KeyboardTarget

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

KeyboardTarget()

```
public KeyboardTarget()
```

Fields

allowHoldingKeySpam

```
public bool allowHoldingKeySpam
```

Field Value

bool

receiveAlreadyHeldKeys

```
public bool receiveAlreadyHeldKeys
```

Field Value

bool

Properties

inputThisFrame

```
public bool inputThisFrame { get; }
```

Property Value

bool

keysHeld

The keys that are currently held down, in order of when they were pressed (most recent first).

```
public CustomKeyCode[] keysHeld { get; }
```

Property Value

CustomKeyCode[]

keysPressed

```
public CustomKeyCode[] keysPressed { get; }
```

Property Value

CustomKeyCode[]

Methods

IsHeld(CustomKeyCode)

Returns true if all the given key (and potentially some other keys) is held (and is a key detectable by KeyboardTarget).

```
public bool IsHeld(CustomKeyCode key)
```

Parameters

`key` CustomKeyCode

Returns

bool

IsHeld(params CustomKeyCode[])

Returns true if all the given keys (and potentially some other keys) are held (and are keys detectable by KeyboardTarget).

```
public bool IsHeld(params CustomKeyCode[] keys)
```

Parameters

`keys` CustomKeyCode[]

Returns

bool

IsHeld(KeyboardShortcut)

Returns true if all the given key (and potentially some other keys) is held (and is a key detectable by KeyboardTarget).

```
public bool IsHeld(KeyboardShortcut shortcut)
```

Parameters

`shortcut` KeyboardShortcut

Returns

bool

IsHeldExactly(params CustomKeyCode[])

Returns true if all and only the given keys are held (and are keys detectable by KeyboardTarget).

```
public bool IsHeldExactly(params CustomKeyCode[] keys)
```

Parameters

`keys` CustomKeyCode[]

Returns

bool

IsHeldExactly(KeyboardShortcut)

Returns true if all and only the given keys are held (and are keys detectable by KeyboardTarget).

```
public bool IsHeldExactly(KeyboardShortcut shortcut)
```

Parameters

`shortcut` KeyboardShortcut

Returns

bool

IsPressed(CustomKeyCode)

Returns true if all the given key (and potentially some other keys) has been pressed this frame (and is a key detectable by KeyboardTarget).

```
public bool IsPressed(CustomKeyCode key)
```

Parameters

`key` CustomKeyCode

Returns

bool

IsPressed(params CustomKeyCode[])

Returns true if all the given keys (and potentially some other keys) have been pressed this frame (and are keys detectable by KeyboardTarget).

```
public bool IsPressed(params CustomKeyCode[] keys)
```

Parameters

`keys` CustomKeyCode[]

Returns

bool

IsPressed(KeyboardShortcut)

Returns true if all the given keys (and potentially some other keys) have been pressed this frame (and are keys detectable by KeyboardTarget).

```
public bool IsPressed(KeyboardShortcut shortcut)
```

Parameters

`shortcut` KeyboardShortcut

Returns

bool

KeyDown(CustomKeyCode)

Simulates the key being pressed, if it is a key detectable by KeyboardTarget.

```
public void KeyDown(CustomKeyCode key)
```

Parameters

`key` CustomKeyCode

KeyDownNoSpamReset(CustomKeyCode)

Simulates the key being pressed, if it is a key detectable by KeyboardTarget, without restting the timer until key spamming occurs.

```
public void KeyDownNoSpamReset(CustomKeyCode key)
```

Parameters

`key` CustomKeyCode

KeyUp(CustomKeyCode)

Simulates the key being unpressed.

```
public void KeyUp(CustomKeyCode key)
```

Parameters

`key` CustomKeyCode

KeysDown(params CustomKeyCode[])

Simulates the keys being pressed, if they are keys detectable by KeyboardTarget.

```
public void KeysDown(params CustomKeyCode[] keys)
```

Parameters

`keys` CustomKeyCode[]

KeysDownNoSpamReset(params CustomKeyCode[])

Simulates the keys being pressed, if they are keys detectable by KeyboardTarget, without restting the timer until key spamming occurs.

```
public void KeysDownNoSpamReset(params CustomKeyCode[] keys)
```

Parameters

`keys` CustomKeyCode[]

KeysUp(params CustomKeyCode[])

Simulates the keys being unpressed.

```
public void KeysUp(params CustomKeyCode[] keys)
```

Parameters

`keys` CustomKeyCode[]

ManualUpdate()

Only to be called in InputTarget's Update() method.

```
public void ManualUpdate()
```

OneIsHeld(IEnumerable<KeyboardShortcut>)

Returns true if all the given keys (and potentially some other keys) are held (and are keys detectable by KeyboardTarget) for one of the given keyboard shortcuts.

```
public bool OneIsHeld(IEnumerable<KeyboardShortcut> shortcuts)
```

Parameters

`shortcuts` IEnumerable<KeyboardShortcut>

Returns

bool

OneIsHeldExactly(IEnumerable<KeyboardShortcut>)

Returns true if all and only the given keys are held (and are keys detectable by KeyboardTarget) for one of the given keyboard shortcuts.

```
public bool OneIsHeldExactly(IEnumerable<KeyboardShortcut> shortcuts)
```

Parameters

`shortcuts` IEnumerable<KeyboardShortcut>

Returns

bool

OneIsPressed(IEnumerable<KeyboardShortcut>)

Returns true if all the given keys (and potentially some other keys) have been pressed this frame (and are keys detectable by KeyboardTarget) for one of the given keyboard shortcuts.

```
public bool OneIsPressed(IEnumerable<KeyboardShortcut> shortcuts)
```

Parameters

`shortcuts` IEnumerable<KeyboardShortcut>

Returns

bool

SubscribeToOnInput(UnityAction)

```
public void SubscribeToOnInput(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnKeyDown(UnityAction<CustomKeyCode>)

```
public void SubscribeToOnKeyDown(UnityAction<CustomKeyCode> call)
```

Parameters

`call` UnityAction<CustomKeyCode>

SubscribeToOnKeyUp(UnityAction<CustomKeyCode>)

```
public void SubscribeToOnKeyUp(UnityAction<CustomKeyCode> call)
```

Parameters

`call` UnityAction<CustomKeyCode>

SubscribeToUntarget(UnityAction)

```
public void SubscribeToUntarget(UnityAction call)
```

Parameters

`call` UnityAction

Untarget()

```
public void Untarget()
```

Class Mouse

Namespace: PAC.Input

```
public class Mouse : MonoBehaviour
```

Inheritance

object ← Mouse

Fields

canInteract

```
public bool canInteract
```

Field Value

bool

lockAllUIInteractions

```
public bool lockAllUIInteractions
```

Field Value

bool

Properties

click

```
public bool click { get; }
```

Property Value

bool

cursorState

```
public CursorState cursorState { get; }
```

Property Value

CursorState

hasHoverTrigger

```
public bool hasHoverTrigger { get; }
```

Property Value

bool

held

```
public bool held { get; }
```

Property Value

bool

hoverTarget

```
public InputTarget hoverTarget { get; }
```

Property Value

InputTarget

leftClick

```
public bool leftClick { get; }
```

Property Value

bool

leftHold

```
public bool leftHold { get; }
```

Property Value

bool

leftUnclick

```
public bool leftUnclick { get; }
```

Property Value

bool

middleClick

```
public bool middleClick { get; }
```

Property Value

bool

middleHold

```
public bool middleHold { get; }
```

Property Value

bool

middleUnclick

```
public bool middleUnclick { get; }
```

Property Value

bool

nothingClick

```
public bool nothingClick { get; }
```

Property Value

bool

nothingHeld

```
public bool nothingHeld { get; }
```

Property Value

bool

numButtonsHeld

```
public int numButtonsHeld { get; }
```

Property Value

int

rightClick

```
public bool rightClick { get; }
```

Property Value

bool

rightHold

```
public bool rightHold { get; }
```

Property Value

bool

rightUnclick

```
public bool rightUnclick { get; }
```

Property Value

bool

scrollCursorDisplayTime

```
public float scrollCursorDisplayTime { get; }
```

Property Value

float

scrollDelta

```
public float scrollDelta { get; }
```

Property Value

float

unclick

```
public bool unclick { get; }
```

Property Value

bool

worldPos

```
public Vector2 worldPos { get; }
```

Property Value

Vector2

Methods

IsClicked(MouseButton)

```
public bool IsClicked(MouseButton mouseButton)
```

Parameters

`mouseButton` MouseButton

Returns

bool

IsHeld(MouseButton)

```
public bool IsHeld(MouseButton mouseButton)
```

Parameters

`mouseButton` MouseButton

Returns

bool

IsUnclicked(MouseButton)

```
public bool IsUnclicked(MouseButton mouseButton)
```

Parameters

`mouseButton` MouseButton

Returns

bool

SetCursorSprite(CursorState)

```
public void SetCursorSprite(CursorState cursorState)
```

Parameters

`cursorState` CursorState

SetCursorState(CursorState)

```
public void SetCursorState(CursorState cursorState)
```

Parameters

`cursorState` CursorState

SubscribeToClick(UnityAction)

```
public void SubscribeToClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToLeftClick(UnityAction)

```
public void SubscribeToLeftClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToLeftUnclick(UnityAction)

```
public void SubscribeToLeftUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToMiddleClick(UnityAction)

```
public void SubscribeToMiddleClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToMiddleUnclick(UnityAction)

```
public void SubscribeToMiddleUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToRightClick(UnityAction)

```
public void SubscribeToRightClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToRightUnclick(UnityAction)

```
public void SubscribeToRightUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToScroll(UnityAction)

```
public void SubscribeToScroll(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToUnclick(UnityAction)

```
public void SubscribeToUnclick(UnityAction call)
```

Parameters

`call` UnityAction

Enum MouseButton

Namespace: PAC.Input

```
public enum MouseButton
```

Fields

`Left = 0`

`Middle = 2`

`Right = 1`

Class MouseTarget

Namespace: PAC.Input

```
public class MouseTarget
```

Inheritance

object ← MouseTarget

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Fields

allowLeftClick

```
public bool allowLeftClick
```

Field Value

bool

allowMiddleClick

```
public bool allowMiddleClick
```

Field Value

bool

allowRightClick

```
public bool allowRightClick
```

Field Value

bool

allowScroll

```
public bool allowScroll
```

Field Value

bool

buttonTargetedWith

```
public MouseButton buttonTargetedWith
```

Field Value

MouseButton

cursorHover

```
public CursorState cursorHover
```

Field Value

CursorState

cursorPress

```
public CursorState cursorPress
```

Field Value

CursorState

cursorScroll

```
public CursorState cursorScroll
```

Field Value

CursorState

cursorSelected

```
public CursorState cursorSelected
```

Field Value

CursorState

deselectMode

```
public MouseTargetDeselectMode deselectMode
```

Field Value

MouseTargetDeselectMode

isHoverTrigger

```
public bool isHoverTrigger
```

Field Value

bool

Properties

selected

```
public bool selected { get; }
```

Property Value

bool

state

```
public MouseTargetState state { get; }
```

Property Value

MouseTargetState

timeHovered

```
public float timeHovered { get; }
```

Property Value

float

Methods

Deselect()

```
public void Deselect()
```

Hover()

```
public void Hover()
```

Idle()

```
public void Idle()
```

Press()

```
public void Press()
```

Scroll()

```
public void Scroll()
```

Select()

```
public void Select()
```

SubscribeToClick(UnityAction)

```
public void SubscribeToClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToDeselect(UnityAction)

```
public void SubscribeToDeselect(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToHover(UnityAction)

```
public void SubscribeToHover(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToIdle(UnityAction)

```
public void SubscribeToIdle(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToLeftClick(UnityAction)

```
public void SubscribeToLeftClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToLeftUnclick(UnityAction)

```
public void SubscribeToLeftUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToMiddleClick(UnityAction)

```
public void SubscribeToMiddleClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToMiddleUnclick(UnityAction)

```
public void SubscribeToMiddleUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToPress(UnityAction)

```
public void SubscribeToPress(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToRightClick(UnityAction)

```
public void SubscribeToRightClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToRightUnclick(UnityAction)

```
public void SubscribeToRightUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToScroll(UnityAction)

```
public void SubscribeToScroll(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToSelect(UnityAction)

```
public void SubscribeToSelect(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToStateChange(UnityAction)

```
public void SubscribeToStateChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToUnclick(UnityAction)

```
public void SubscribeToUnclick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToUntarget(UnityAction)

```
public void SubscribeToUntarget(UnityAction call)
```

Parameters

`call` UnityAction

Untarget()

```
public void Untarget()
```

Enum MouseTargetDeselectMode

Namespace: PAC.Input

```
public enum MouseTargetDeselectMode
```

Fields

`ClickAgain = 1`

`Manual = 100`

`Unclick = 0`

Enum MouseTargetState

Namespace: PAC.Input

```
public enum MouseTargetState
```

Fields

`Hover = 1`

`Idle = 0`

`Pressed = 2`

Class Tooltip

Namespace: PAC.Input

```
public class Tooltip : MonoBehaviour
```

Inheritance

object ← Tooltip

Fields

hoverTimeUntilTooltip

```
public float hoverTimeUntilTooltip
```

Field Value

float

padding

```
public Vector2 padding
```

Field Value

Vector2

Properties

text

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

Property Value

string

tooltipVisible

```
public bool tooltipVisible { get; }
```

Property Value

bool

Methods

HideTooltip()

```
public void HideTooltip()
```

ShowTooltip(Vector2)

```
public void ShowTooltip(Vector2 worldCoords)
```

Parameters

`worldCoords` Vector2

Namespace PAC.JSON

Classes

JSON

A class to represent data in a JSON format.

JSON.JSONProperty

A class to store an individual piece of JSON data.

Interfaces

IJSONable

An interface for objects that can be converted to JSON.

Interface IJSONable

Namespace: PAC.JSON

An interface for objects that can be converted to JSON.

```
public interface IJSONable
```

Methods

ToJSON()

```
JSON ToJSON()
```

Returns

JSON

Class JSON

Namespace: PAC.JSON

A class to represent data in a JSON format.

```
public class JSON
```

Inheritance

object ← JSON

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

JSON()

Create an empty JSON object.

```
public JSON()
```

JSON(string)

Parse the JSON-format string into a JSON object.

```
public JSON(string jsonToParse)
```

Parameters

`jsonToParse` string

Properties

this[string]

Accesses the string form of the data for the given key - the same as Get(key) and Add(key, value). To access the JSONProperty for the key, use GetJSONProperty(key).

```
public string this[string key] { get; set; }
```

Parameters

`key` string

Property Value

string

Keys

The keys appearing in the outermost scope of the JSON data.

```
public string[] Keys { get; }
```

Property Value

string[]

Methods

Add(string, IJSONable)

Adds the string form of the JSON of the given object to this JSON object.

```
public void Add(string key, IJSONable jsonableObject)
```

Parameters

`key` string

`jsonableObject` IJSONable

Add(string, JSON)

Adds the string form of the given JSON to this JSON object.

```
public void Add(string key, JSON json)
```

Parameters

`key` string

`json` JSON

Add(string, JSONProperty)

Adds the given data at the given key, overriding any existing data.

```
public void Add(string key, JSON.JSONProperty data)
```

Parameters

`key` string

`data` JSON.JSONProperty

Add(string, bool)

Adds the given bool to the JSON.

```
public void Add(string key, bool boolean)
```

Parameters

`key` string

`boolean` bool

Add(string, IEnumerable<IJSONable>)

Adds the string forms of the JSON of the given objects to this JSON object in the format of a JSON array.

```
public void Add(string key, IEnumerable<IJSONable> jsonableObjects)
```

Parameters

`key` string

`jsonableObjects` IEnumerable<IJSONable>

Add(string, IEnumerable<string>, bool)

Adds the given strings to the JSON in the format of a JSON array, adding quotation marks around each string.

```
public void Add(string key, IEnumerable<string> strings, bool separateLines)
```

Parameters

`key` string

`strings` IEnumerable<string>

`separateLines` bool

Whether to start a new line for each element of the array.

Add(string, IEnumerable<string>, bool, bool)

Adds the given data (given in string form) to the JSON in the format of a JSON array.

```
public void Add(string key, IEnumerable<string> strings, bool separateLines,
bool addQuotationMarks)
```

Parameters

`key` string

`strings` IEnumerable<string>

`separateLines` bool

Whether to start a new line for each element of the array.

`addQuotationMarks` bool

Whether to add quotation marks around each element of the array.

Add(string, IEnumerable, bool)

Adds the string form of the given objects to the JSON in the format of a JSON array, adding no quotation marks.

```
public void Add(string key, IEnumerable objects, bool separateLines)
```

Parameters

`key` string

`objects` IEnumerable

`separateLines` bool

Whether to start a new line for each element of the array.

Add(string, IEnumerable, bool, bool)

Adds the string form of the given objects to the JSON in the format of a JSON array.

```
public void Add(string key, IEnumerable objects, bool separateLines,
bool addQuotationMarks)
```

Parameters

`key` string

`objects` IEnumerable

`separateLines` bool

Whether to start a new line for each element of the array.

`addQuotationMarks` bool

Whether to add quotation marks around each element of the array.

Add(string, object)

Adds the string form of the given object to the JSON, adding no quotation marks.

```
public void Add(string key, object obj)
```

Parameters

`key` string

`obj` object

Add(string, object, bool)

Adds the string form of the given object to the JSON.

```
public void Add(string key, object obj, bool addQuotationMarks)
```

Parameters

`key` string

`obj` object

`addQuotationMarks` bool

Add(string, string)

Adds the given string to the JSON, adding quotation marks around the string.

```
public void Add(string key, string str)
```

Parameters

`key` string

`str` string

Add(string, string, bool)

Adds the given data (given in string form) to the JSON.

```
public void Add(string key, string str, bool addQuotationMarks)
```

Parameters

`key` string

`str` string

`addQuotationMarks` bool

AddParse(string)

Parses the JSON string and adds all data to this JSON object. Does not delete any existing data in this object, except when keys collide, in which case new data will override old data.

```
public void AddParse(string jsonToParse)
```

Parameters

`jsonToParse` string

Append(JSON)

Adds the data of the given JSON object to the end of this JSON object. Throws error if they have a key in common.

```
public void Append(JSON json)
```

Parameters

`json` JSON

ContainsKey(string)

Returns true if this JSON object has the given key.

```
public bool ContainsKey(string key)
```

Parameters

`key` string

Returns

bool

Get(string)

Returns the string form of the data at the given key.

```
public string Get(string key)
```

Parameters

`key` string

Returns

string

GetJSONProperty(string)

Returns the JSONProperty at the given key.

```
public JSON.JSONProperty GetJSONProperty(string key)
```

Parameters

`key` string

Returns

JSON.JSONProperty

Parse(string)

Parses the JSON string into a new JSON object.

```
public static JSON Parse(string jsonToParse)
```

Parameters

`jsonToParse` string

Returns

JSON

SplitArray(string)

Splits a JSON string representing an array into a string[] of the data.

```
public static string[] SplitArray(string jsonString)
```

Parameters

`jsonString` string

Include the start/end square brackets in the string.

Returns

string[]

StripQuotationMarks(string)

Removes the " " pair enclosing a string, if there is one; otherwise throws an error.

```
public static string StripQuotationMarks(string str)
```

Parameters

`str` string

Returns

string

ToJSONString(IEnumerable<IJSONable>)

Returns the JSON of the given objects in a JSON-format array in string form.

```
public static string ToJSONString(IEnumerable<IJSONable> jsonableObjects)
```

Parameters

`jsonableObjects` IEnumerable<IJSONable>

Returns

string

ToJSONString(IEnumerable<string>, bool)

Returns the given strings in a JSON-format array in string form, adding quotation marks around each element.

```
public static string ToJSONString(IEnumerable<string> strings, bool separateLines)
```

Parameters

`strings` IEnumerable<string>

`separateLines` bool

Returns

string

ToJSONString(IEnumerable<string>, bool, bool)

Returns the given strings in a JSON-format array in string form.

```
public static string ToJSONString(IEnumerable<string> strings, bool separateLines,
bool addQuotationMarks)
```

Parameters

`strings` IEnumerable<string>

`separateLines` bool

`addQuotationMarks` bool

Whether to add quotation marks around each element of the array.

Returns

string

ToJSONString(IEnumerable, bool)

Returns the JSON of the given objects in a JSON-format array in string form, adding no quotation marks.

```
public static string ToJSONString(IEnumerable objects, bool separateLines)
```

Parameters

`objects` IEnumerable

`separateLines` bool

Returns

string

ToJSONString(IEnumerable, bool, bool)

Returns the JSON of the given objects in a JSON-format array in string form.

```
public static string ToJSONString(IEnumerable objects, bool separateLines,
bool addQuotationMarks)
```

Parameters

`objects` IEnumerable

`separateLines` bool

`addQuotationMarks` bool

Whether to add quotation marks around each element of the array.

Returns

string

ToString()

Returns the data as a string in JSON format.

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

Returns

string

Class JSON.JSONProperty

Namespace: PAC.JSON

A class to store an individual piece of JSON data.

```
public class JSON.JSONProperty
```

Inheritance

object ← JSON.JSONProperty

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

JSONProperty(string, bool)

```
public JSONProperty(string data, bool addQuotationMarks)
```

Parameters

`data` string

`addQuotationMarks` bool

Whether quotation marks should be added around the data - e.g. if the data represents a string.

Fields

addQuotationMarks

Whether quotation marks should be added around the data - e.g. if the data represents a string.

```
public bool addQuotationMarks
```

Field Value

bool

data

The data.

```
public string data
```

Field Value

string

Namespace PAC.KeyboardShortcuts

Classes

CustomKeyCode

A class to supersede Unity's KeyCode enum to allow keycodes to refer to multiple keys - e.g. Shift instead of separated into LeftShift and RightShift.

KeyboardShortcut

A class to represent a single keyboard shortcut.

KeyboardShortcutOptionsManager

Handles the window where you can view / change keyboard shortcuts.

KeyboardShortcuts

A class for storing, loading and accessing keyboard shortcuts for defined actions.

Class CustomKeyCode

Namespace: PAC.KeyboardShortcuts

A class to supersede Unity's KeyCode enum to allow keycodes to refer to multiple keys - e.g. Shift instead of separated into LeftShift and RightShift.

```
public class CustomKeyCode : IEnumerable
```

Inheritance

object ← CustomKeyCode

Implements

IEnumerable

Inherited Members

object.Equals(object, object) , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Fields

Alt

```
public static readonly CustomKeyCode Alt
```

Field Value

CustomKeyCode

Ctrl

```
public static readonly CustomKeyCode Ctrl
```

Field Value

CustomKeyCode

GreaterThan

```
public static readonly CustomKeyCode GreaterThan
```

Field Value

CustomKeyCode

LessThan

```
public static readonly CustomKeyCode LessThan
```

Field Value

CustomKeyCode

Minus

```
public static readonly CustomKeyCode Minus
```

Field Value

CustomKeyCode

Plus

```
public static readonly CustomKeyCode Plus
```

Field Value

CustomKeyCode

Shift

```
public static readonly CustomKeyCode Shift
```

Field Value

CustomKeyCode

_0

The keycode for 0.

```
public static readonly CustomKeyCode _0
```

Field Value

CustomKeyCode

_1

The keycode for 1.

```
public static readonly CustomKeyCode _1
```

Field Value

CustomKeyCode

_2

The keycode for 2.

```
public static readonly CustomKeyCode _2
```

Field Value

CustomKeyCode

_3

The keycode for 3.

```
public static readonly CustomKeyCode _3
```

Field Value

CustomKeyCode

_4

The keycode for 4.

```
public static readonly CustomKeyCode _4
```

Field Value

CustomKeyCode

_5

The keycode for 5.

```
public static readonly CustomKeyCode _5
```

Field Value

CustomKeyCode

_6

The keycode for 6.

```
public static readonly CustomKeyCode _6
```

Field Value

CustomKeyCode

_7

The keycode for 7.

```
public static readonly CustomKeyCode _7
```

Field Value

CustomKeyCode

_8

The keycode for 8.

```
public static readonly CustomKeyCode _8
```

Field Value

CustomKeyCode

_9

The keycode for 9.

```
public static readonly CustomKeyCode _9
```

Field Value

CustomKeyCode

allKeyCodes

All Unity KeyCodes and all combined keycodes.

```
public static readonly CustomKeyCode[] allKeyCodes
```

Field Value

CustomKeyCode[]

alphabet

The keycodes A-Z.

```
public static readonly CustomKeyCode[] alphabet
```

Field Value

CustomKeyCode[]

combinedKeyCodes

All the defined keycodes that combine multiple Unity KeyCodes.

```
public static readonly CustomKeyCode[] combinedKeyCodes
```

Field Value

CustomKeyCode[]

digits

The keycodes 0-9.

```
public static readonly CustomKeyCode[] digits
```

Field Value

CustomKeyCode[]

Properties

displayName

What will be returned from ToString().

```
public string displayName { get; }
```

Property Value

string

keyCodes

The Unity KeyCodes comprising this keycode.

```
public List<KeyCode> keyCodes { get; }
```

Property Value

List<KeyCode>

Methods

Contains(KeyCode)

Returns true if the Unity KeyCode forms part of this CustomKeyCode.

```
public bool Contains(KeyCode keyCode)
```

Parameters

`keyCode` KeyCode

Returns

bool

Equals(object)

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

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

Parameters

`obj` object

The object to compare with the current object.

Returns

bool

true if the specified object is equal to the current object; otherwise, false.

FromString(string)

Converts a string into the keycode with that display name. Returns null if there isn't one.

```
public static CustomKeyCode FromString(string displayName)
```

Parameters

`displayName` string

Returns

CustomKeyCode

GetEnumerator()

Returns an enumerator that iterates through a collection.

```
public IEnumerator GetEnumerator()
```

Returns

IEnumerator

An IEnumerator object that can be used to iterate through the collection.

GetHashCode()

Serves as the default hash function.

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

Returns

int

A hash code for the current object.

GetKey()

Returns true while this key is being held down.

```
public bool GetKey()
```

Returns

bool

GetKeyDown()

Returns true the frame this key is pressed.

```
public bool GetKeyDown()
```

Returns

bool

GetKeyUp()

Returns true the frame this key is released.

```
public bool GetKeyUp()
```

Returns

bool

ToString()

Returns a string that represents the current object.

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

Returns

string

A string that represents the current object.

Operators

operator ==(CustomKeyCode, CustomKeyCode)

```
public static bool operator ==(CustomKeyCode keyCode1, CustomKeyCode keyCode2)
```

Parameters

`keyCode1` CustomKeyCode

`keyCode2` CustomKeyCode

Returns

bool

implicit operator CustomKeyCode(KeyCode)

```
public static implicit operator CustomKeyCode(KeyCode keyCode)
```

Parameters

`keyCode` KeyCode

Returns

CustomKeyCode

operator !=(CustomKeyCode, CustomKeyCode)

```
public static bool operator !=(CustomKeyCode keyCode1, CustomKeyCode keyCode2)
```

Parameters

`keyCode1` CustomKeyCode

`keyCode2` CustomKeyCode

Returns

bool

Class KeyboardShortcut

Namespace: PAC.KeyboardShortcuts

A class to represent a single keyboard shortcut.

```
public class KeyboardShortcut : IJSONable
```

Inheritance

object ← KeyboardShortcut

Implements

IJSONable

Inherited Members

object.Equals(object, object) , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object)

Constructors

KeyboardShortcut(params CustomKeyCode[])

Creates a shortcut from the given keycodes. Sorts the keycodes into the order they would be read.

```
public KeyboardShortcut(params CustomKeyCode[] keyCodes)
```

Parameters

`keyCodes` CustomKeyCode[]

Properties

None

The empty keyboard shortcut - i.e. no keycodes.

```
public static KeyboardShortcut None { get; }
```

Property Value

KeyboardShortcut

keyCodes

The keycodes in this shortcut, kept in the order they would be read.

```
public List<CustomKeyCode> keyCodes { get; }
```

Property Value

List<CustomKeyCode>

Methods

Add(CustomKeyCode)

Adds the keycode to the shortcut. Returns false if it was already in the shortcut; otherwise returns true.

```
public bool Add(CustomKeyCode keyCode)
```

Parameters

`keyCode` CustomKeyCode

Returns

bool

Contains(CustomKeyCode)

Returns true if the keycode is in the shortcut.

```
public bool Contains(CustomKeyCode keyCode)
```

Parameters

`keyCode` CustomKeyCode

Returns

bool

Equals(object)

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

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

Parameters

`obj` object

The object to compare with the current object.

Returns

bool

true if the specified object is equal to the current object; otherwise, false.

GetHashCode()

Serves as the default hash function.

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

Returns

int

A hash code for the current object.

GetKeys()

Returns true while all keys are being held down.

```
public bool GetKeys()
```

Returns

bool

GetKeysDown()

Returns true the frame the final key is pressed.

```
public bool GetKeysDown()
```

Returns

bool

GetKeysUp()

Returns true the frame (any key of) this shortcut is released.

```
public bool GetKeysUp()
```

Returns

bool

Remove(CustomKeyCode)

Removes the keycode from the shortcut. Returns false if the keycode already wasn't in the shortcut; otherwise returns true.

```
public bool Remove(CustomKeyCode keyCode)
```

Parameters

`keyCode` CustomKeyCode

Returns

bool

ToArray()

Returns an array of the keycodes in this shortcut, in the order they would be read.

```
public CustomKeyCode[] ToArray()
```

Returns

CustomKeyCode[]

ToJSON()

```
public JSON ToJSON()
```

Returns

JSON

ToString()

Returns a string that represents the current object.

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

Returns

string

A string that represents the current object.

Operators

operator ==(KeyboardShortcut, KeyboardShortcut)

Checks if the shortcuts have the same set of keycodes.

```
public static bool operator ==(KeyboardShortcut shortcut1,
KeyboardShortcut shortcut2)
```

Parameters

`shortcut1` KeyboardShortcut

`shortcut2` KeyboardShortcut

Returns

bool

operator !=(KeyboardShortcut, KeyboardShortcut)

Checks if the shortcuts do not have the same set of keycodes.

```
public static bool operator !=(KeyboardShortcut shortcut1,
KeyboardShortcut shortcut2)
```

Parameters

`shortcut1` KeyboardShortcut

`shortcut2` KeyboardShortcut

Returns

bool

Class KeyboardShortcutOptionsManager

Namespace: PAC.KeyboardShortcuts

Handles the window where you can view / change keyboard shortcuts.

```
public class KeyboardShortcutOptionsManager : MonoBehaviour
```

Inheritance

object ← KeyboardShortcutOptionsManager

Class KeyboardShortcuts

Namespace: PAC.KeyboardShortcuts

A class for storing, loading and accessing keyboard shortcuts for defined actions.

```
public class KeyboardShortcuts : MonoBehaviour
```

Inheritance

object ← KeyboardShortcuts

Properties

shortcuts

A key is an action name. A value is a list of the keyboard shortcuts for that action.

```
public static Dictionary<string, List<KeyboardShortcut>> shortcuts { get; }
```

Property Value

Dictionary<string, List<KeyboardShortcut>>

Methods

AddShortcut(string, params CustomKeyCode[])

Combines the given keycodes into a single shortcut and adds it for the given action.

```
public static void AddShortcut(string actionName, params CustomKeyCode[] keyCodes)
```

Parameters

`actionName` string

`keyCodes` CustomKeyCode[]

AddShortcut(string, KeyboardShortcut)

Adds the given shortcut for the given action.

```
public static void AddShortcut(string actionName, KeyboardShortcut shortcut)
```

Parameters

`actionName` string

`shortcut` KeyboardShortcut

ClearShortcutsFor(string)

Removes all shortcuts for the action.

```
public static void ClearShortcutsFor(string actionName)
```

Parameters

`actionName` string

ContainsKey(string)

Returns true if the action is defined for having keyboard shortcuts.

```
public static bool ContainsKey(string actionName)
```

Parameters

`actionName` string

Returns

bool

ContainsShortcutFor(string)

Returns true if the action has a shortcut.

```
public static bool ContainsShortcutFor(string actionName)
```

Parameters

`actionName` string

Returns

bool

GetShortcutsFor(string)

Returns all shortcuts for the action.

```
public static List<KeyboardShortcut> GetShortcutsFor(string actionName)
```

Parameters

`actionName` string

Returns

List<KeyboardShortcut>

LoadShortcuts()

Loads saved shortcuts from the disk.

```
public static void LoadShortcuts()
```

SaveShortcuts()

Saves the current assignment of each shortcut to the disk.

```
public static void SaveShortcuts()
```

SetShortcut(string, KeyboardShortcut)

Removes any existing shortcuts for the action then adds the given shortcut.

```
public static void SetShortcut(string actionName, KeyboardShortcut shortcut)
```

Parameters

`actionName` string

`shortcut` KeyboardShortcut

ToJSON()

Returns the list of keyboard shortcuts as a JSON object.

```
public static JSON ToJSON()
```

Returns

JSON

Namespace PAC.Layers

Classes

Layer

An abstract class to define what each type of layer must have.

LayerManager

LayerTile

NormalLayer

A class to represent a normal layer - one that can be drawn on as a regular image.

TileLayer

A class to represent a tile layer - one for placing and editing tileset tiles.

Enums

AnimFrameRefMode

LayerType

Enum AnimFrameRefMode

Namespace: PAC.Layers

```
public enum AnimFrameRefMode
```

Fields

`MostRecentKeyFrame = 1`

`NewKeyFrame = 0`

Class Layer

Namespace: PAC.Layers

An abstract class to define what each type of layer must have.

```
public abstract class Layer : IJSONable
```

Inheritance

object ← Layer

Implements

IJSONable

Derived

NormalLayer, TileLayer

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Layer(Layer)

```
public Layer(Layer layer)
```

Parameters

`layer` Layer

Layer(string, Texture2D)

```
public Layer(string name, Texture2D texture)
```

Parameters

`name` string

`texture` Texture2D

Fields

_blendMode

```
protected BlendMode _blendMode
```

Field Value

BlendMode

_visible

```
protected bool _visible
```

Field Value

bool

onPixelsChanged

Called when the SetPixel() or SetPixels() has been called. Passes the array of pixels changed and the frames it was called on.

```
protected UnityEvent<IntVector2[], int[]> onPixelsChanged
```

Field Value

UnityEvent<IntVector2[], int[]>

Properties

this[int]

Returns the texture of the most recent key frame before or at the frame index. Same as GetTexture(frame).

```
public AnimationKeyFrame this[int frame] { get; }
```

Parameters

`frame` int

Property Value

AnimationKeyFrame

blendMode

```
public BlendMode blendMode { get; set; }
```

Property Value

BlendMode

height

```
public int height { get; protected set; }
```

Property Value

int

keyFrameIndices

The frame indices of the key frames in the animation, in order.

```
public int[] keyFrameIndices { get; }
```

Property Value

int[]

keyFrameTextures

The textures of the key frames in the animation, in order.

```
public Texture2D[] keyFrameTextures { get; }
```

Property Value

Texture2D[]

keyFrames

The key frames of the animation, in order.

```
public List<AnimationKeyFrame> keyFrames { get; protected set; }
```

Property Value

List<AnimationKeyFrame>

layerType

```
public abstract LayerType layerType { get; }
```

Property Value

LayerType

locked

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

Property Value

bool

name

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

Property Value

string

opacity

```
public float opacity { get; set; }
```

Property Value

float

rect

```
public IntRect rect { get; }
```

Property Value

IntRect

visible

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

Property Value

bool

width

```
public int width { get; protected set; }
```

Property Value

int

Methods

AddKeyFrame(AnimationKeyFrame)

Adds the given key frame. Returns true if it replaces an existing key frame, and false otherwise.

```
protected bool AddKeyFrame(AnimationKeyFrame keyFrame)
```

Parameters

`keyFrame` AnimationKeyFrame

Returns

bool

AddKeyFrame(int)

Adds a key frame at frame frame. The texture will be that of the most recent key frame. Returns true if it replaces an existing key frame, and false otherwise.

```
protected bool AddKeyFrame(int frame)
```

Parameters

`frame` int

Returns

bool

AddKeyFrame(int, Texture2D)

Adds a key frame with the given texture at frame frame. Returns true if it replaces an existing key frame, and false otherwise.

```
protected bool AddKeyFrame(int frame, Texture2D texture)
```

Parameters

`frame` int

`texture` Texture2D

Returns

bool

ClearFrames()

Deletes all key frames.

```
public abstract void ClearFrames()
```

DeepCopy()

Creates a deep copy of the Layer.

```
public abstract Layer DeepCopy()
```

Returns

Layer

DeleteKeyFrame(AnimationKeyFrame)

Deletes the given key frame, if it's in the animation, in which case it returns true. Otherwise it returns false.

```
public bool DeleteKeyFrame(AnimationKeyFrame keyFrame)
```

Parameters

`keyFrame` AnimationKeyFrame

Returns

bool

DeleteKeyFrame(int)

```
public AnimationKeyFrame DeleteKeyFrame(int keyFrame)
```

Parameters

`keyFrame` int

Returns

AnimationKeyFrame

DeleteKeyFrameNoEvent(int)

Deletes the key frame at the given frame index, if there is one, in which case it returns that key frame. Otherwise it returns null.

```
protected abstract AnimationKeyFrame DeleteKeyFrameNoEvent(int keyframe)
```

Parameters

`keyframe` int

Returns

AnimationKeyFrame

DeleteMostRecentKeyFrame(int)

Deletes the most recent key frame before or at the given frame index and returns that key frame.

```
public AnimationKeyFrame DeleteMostRecentKeyFrame(int frame)
```

Parameters

`frame` int

Returns

AnimationKeyFrame

Extend(int, int, int, int)

Extends the dimensions of the layer in each direction by the given amounts.

```
public void Extend(int left, int right, int up, int down)
```

Parameters

`left` int

`right` int

`up` int

`down` int

ExtendNoEvent(int, int, int, int)

Extends the dimensions of the layer in each direction by the given amounts, but does not invoke the onPixelsChanged event.

```
protected abstract void ExtendNoEvent(int left, int right, int up, int down)
```

Parameters

`left` int

`right` int

`up` int

`down` int

Flip(FlipDirection)

Flips the layer.

```
public void Flip(FlipDirection direction)
```

Parameters

`direction` FlipDirection

FlipNoEvent(FlipDirection)

Flips the layer, but does not invoke the onPixelsChanged event.

```
protected abstract void FlipNoEvent(FlipDirection direction)
```

Parameters

`direction` FlipDirection

FromJSON(JSON)

```
public static Layer FromJSON(JSON json)
```

Parameters

`json` JSON

Returns

Layer

GetKeyFrame(int)

Returns the most recent key frame before or at the frame index - i.e. the key frame the animation will play at the frame index.

```
public AnimationKeyFrame GetKeyFrame(int frame)
```

Parameters

`frame` int

Returns

AnimationKeyFrame

GetPixel(IntVector2, int, bool)

Gets the colour of the pixel.

```
public abstract Color GetPixel(IntVector2 pixel, int frame, bool useLayerOpacity
= true)
```

Parameters

`pixel` IntVector2

`frame` int

`useLayerOpacity` bool

Returns

Color

GetPixel(int, int, int, bool)

Gets the colour of the pixel (x, y).

```
public Color GetPixel(int x, int y, int frame, bool useLayerOpacity = true)
```

Parameters

`x` int

`y` int

`frame` int

`useLayerOpacity` bool

Returns

Color

HasKeyFrameAt(int)

Returns whether or not there is a key frame at the given frame index.

```
public bool HasKeyFrameAt(int frame)
```

Parameters

`frame` int

Returns

bool

IsBlank()

Returns true if and only if all pixels are completely transparent.

```
public bool IsBlank()
```

Returns

bool

LoadJSON(JSON)

```
protected abstract void LoadJSON(JSON json)
```

Parameters

`json` JSON

Rotate(RotationAngle)

Rotates the layer. Rotation is clockwise.

```
public void Rotate(RotationAngle angle)
```

Parameters

`angle` RotationAngle

RotateNoEvent(RotationAngle)

Rotates the layer, but does not invoke the onPixelsChanged event. Rotation is clockwise.

```
protected abstract void RotateNoEvent(RotationAngle angle)
```

Parameters

`angle` RotationAngle

Scale(int, int)

Resizes the dimensions of the file by the scale factor.

```
public void Scale(int newWidth, int newHeight)
```

Parameters

`newWidth` int

`newHeight` int

Scale(float)

Resizes the dimensions of the file by the scale factor.

```
public void Scale(float scaleFactor)
```

Parameters

`scaleFactor` float

Scale(float, float)

Resizes the dimensions of the file by the scale factors.

```
public void Scale(float xScaleFactor, float yScaleFactor)
```

Parameters

`xScaleFactor` float

`yScaleFactor` float

ScaleNoEvent(int, int)

Resizes the dimensions to the new width and height, but does not invoke the onPixelsChanged event.

```
protected abstract void ScaleNoEvent(int newWidth, int newHeight)
```

Parameters

`newWidth` int

`newHeight` int

ScaleNoEvent(float, float)

Resizes the dimensions of the file by the scale factors, but does not invoke the onPixelsChanged event.

```
protected abstract void ScaleNoEvent(float xScaleFactor, float yScaleFactor)
```

Parameters

`xScaleFactor` float

`yScaleFactor` float

SetPixel(IntVector2, int, Color, AnimFrameRefMode)

Sets the colour of the pixel. Throws an error if the pixel is outside the layer.

```
public IntVector2[] SetPixel(IntVector2 pixel, int frame, Color colour,
AnimFrameRefMode frameRefMode)
```

Parameters

`pixel` IntVector2

`frame` int

`colour` Color

`frameRefMode` AnimFrameRefMode

Returns

IntVector2[]

SetPixel(int, int, int, Color, AnimFrameRefMode)

Sets the colour of the pixel (x, y). Throws an error if the pixel is outside the layer.

```
public IntVector2[] SetPixel(int x, int y, int frame, Color colour,
AnimFrameRefMode frameRefMode)
```

Parameters

`x` int

`y` int

`frame` int

`colour` Color

`frameRefMode` AnimFrameRefMode

Returns

IntVector2[]

SetPixels(IntVector2[], int, Color, AnimFrameRefMode)

Sets the colour of the pixels. Throws an error if a pixel is outside the layer.

```
public IntVector2[] SetPixels(IntVector2[] pixels, int frame, Color colour,
AnimFrameRefMode frameRefMode)
```

Parameters

`pixels` IntVector2[]

`frame` int

`colour` Color

`frameRefMode` AnimFrameRefMode

Returns

IntVector2[]

SetPixelsNoEvent(IntVector2[], int, Color, AnimFrameRefMode)

Sets the colour of the pixels. You do not need to check the pixels are in the layer as this check is done in Layer.SetPixels(), which is the only way this method is called.

```
protected abstract IntVector2[] SetPixelsNoEvent(IntVector2[] pixels, int frame,
Color colour, AnimFrameRefMode frameRefMode)
```

Parameters

`pixels` IntVector2[]

`frame` int

`colour` Color

`frameRefMode` AnimFrameRefMode

Returns

IntVector2[]

SubscribeToOnBlendModeChanged(UnityAction)

```
public void SubscribeToOnBlendModeChanged(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnKeyFrameAdded(UnityAction<int>)

```
public void SubscribeToOnKeyFrameAdded(UnityAction<int> call)
```

Parameters

`call` UnityAction<int>

SubscribeToOnKeyFrameRemoved(UnityAction<int>)

```
public void SubscribeToOnKeyFrameRemoved(UnityAction<int> call)
```

Parameters

`call` UnityAction<int>

SubscribeToOnPixelsChanged(UnityAction<IntVector2[], int[]>)

```
public void SubscribeToOnPixelsChanged(UnityAction<IntVector2[], int[]> call)
```

Parameters

`call` UnityAction<IntVector2[], int[]>

SubscribeToOnVisibilityChanged(UnityAction)

```
public void SubscribeToOnVisibilityChanged(UnityAction call)
```

Parameters

`call` UnityAction

ToJSON()

```
public abstract JSON ToJSON()
```

Returns

JSON

Class LayerManager

Namespace: PAC.Layers

```
public class LayerManager : MonoBehaviour
```

Inheritance

object ← LayerManager

Fields

layerTypeSprite

```
public Sprite layerTypeSprite
```

Field Value

Sprite

tileLayerTypeSprite

```
public Sprite tileLayerTypeSprite
```

Field Value

Sprite

Properties

selectedLayer

The last layer that was selected.

```
public Layer selectedLayer { get; }
```

Property Value

Layer

selectedLayerIndex

The index of the last layer that was selected.

```
public int selectedLayerIndex { get; }
```

Property Value

int

selectedLayerIndices

The indices of all selected layers, in increasing order.

```
public int[] selectedLayerIndices { get; }
```

Property Value

int[]

selectedLayers

The selected layers, in order (highest layer first etc).

```
public Layer[] selectedLayers { get; }
```

Property Value

Layer[]

Methods

AddLayer()

```
public void AddLayer()
```

AddLayer(Texture2D)

```
public void AddLayer(Texture2D texture)
```

Parameters

`texture` Texture2D

AddTileLayer()

```
public void AddTileLayer()
```

DuplicateSelectedLayers()

```
public void DuplicateSelectedLayers()
```

FlattenSelectedLayers()

```
public void FlattenSelectedLayers()
```

HideAllBut(Layer)

```
public void HideAllBut(Layer layer)
```

Parameters

`layer` Layer

MoveSelectedLayersDown()

```
public void MoveSelectedLayersDown()
```

MoveSelectedLayersUp()

```
public void MoveSelectedLayersUp()
```

OnFileSwitched()

```
public void OnFileSwitched()
```

OnLayersChanged()

```
public void OnLayersChanged()
```

RemoveSelectedLayers()

```
public void RemoveSelectedLayers()
```

SetLayerBlendMode(int, BlendMode)

```
public void SetLayerBlendMode(int layerIndex, BlendMode blendMode)
```

Parameters

`layerIndex` int

`blendMode` BlendMode

SetLayerName(int, string)

```
public void SetLayerName(int layerIndex, string layerName)
```

Parameters

`layerIndex` int

`layerName` string

SetLayerOpacity(int, float)

```
public void SetLayerOpacity(int layerIndex, float opacity)
```

Parameters

`layerIndex` int

`opacity` float

SubscribeToLayerChange(UnityAction)

```
public void SubscribeToLayerChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToVisibilityChange(UnityAction)

```
public void SubscribeToVisibilityChange(UnityAction call)
```

Parameters

`call` UnityAction

WorldYCoordOfLayerTile(int)

```
public float WorldYCoordOfLayerTile(int layerTileIndex)
```

Parameters

`layerTileIndex` int

Returns

float

Class LayerTile

Namespace: PAC.Layers

```
public class LayerTile : MonoBehaviour
```

Inheritance

object ← LayerTile

Properties

layer

```
public Layer layer { get; }
```

Property Value

Layer

selected

```
public bool selected { get; }
```

Property Value

bool

tileToggle

```
public UIToggleButton tileToggle { get; }
```

Property Value

UIToggleButton

Methods

SetLayer(Layer)

```
public void SetLayer(Layer layer)
```

Parameters

`layer` Layer

SubscribeToLockChange(UnityAction)

```
public void SubscribeToLockChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToNameChange(UnityAction)

```
public void SubscribeToNameChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToRightClick(UnityAction)

```
public void SubscribeToRightClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToVisibilityChange(UnityAction)

```
public void SubscribeToVisibilityChange(UnityAction call)
```

Parameters

`call` UnityAction

Enum LayerType

Namespace: PAC.Layers

```
public enum LayerType
```

Fields

`Normal = 0`

`Tile = 1`

Class NormalLayer

Namespace: PAC.Layers

A class to represent a normal layer - one that can be drawn on as a regular image.

```
public class NormalLayer : Layer, IJSONable
```

Inheritance

object ← Layer ← NormalLayer

Implements

IJSONable

Inherited Members

Layer.name , Layer.width , Layer.height , Layer.rect , Layer._visible , Layer.visible , Layer.locked , Layer.opacity , Layer._blendMode , Layer.blendMode , Layer.keyFrames , Layer.keyFrameIndices , Layer.keyFrameTextures , Layer.onPixelsChanged , Layer.SetPixel(int, int, int, Color, AnimFrameRefMode) , Layer.SetPixel(IntVector2, int, Color, AnimFrameRefMode) , Layer.SetPixels(IntVector2[], int, Color, AnimFrameRefMode) , Layer.GetPixel(int, int, int, bool) , Layer.IsBlank() , Layer.Flip(FlipDirection) , Layer.Rotate(RotationAngle) , Layer.Extend(int, int, int, int) , Layer.Scale(float) , Layer.Scale(float, float) , Layer.Scale(int, int) , Layer.FromJSON(JSON) , Layer.this[int] , Layer.GetKeyFrame(int) , Layer.HasKeyFrameAt(int) , Layer.AddKeyFrame(int) , Layer.AddKeyFrame(int, Texture2D) , Layer.AddKeyFrame(AnimationKeyFrame) , Layer.DeleteMostRecentKeyFrame(int) , Layer.DeleteKeyFrame(int) , Layer.DeleteKeyFrame(AnimationKeyFrame) , Layer.SubscribeToOnPixelsChanged(UnityAction<IntVector2[], int[]>) , Layer.SubscribeToOnVisibilityChanged(UnityAction) , Layer.SubscribeToOnBlendModeChanged(UnityAction) , Layer.SubscribeToOnKeyFrameAdded(UnityAction<int>) , Layer.SubscribeToOnKeyFrameRemoved(UnityAction<int>) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

NormalLayer(NormalLayer)

Creates a deep copy of the NormalLayer.

```
public NormalLayer(NormalLayer layer)
```

Parameters

`layer` NormalLayer

NormalLayer(int, int)

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

Parameters

`width` int

`height` int

NormalLayer(string, int, int)

```
public NormalLayer(string name, int width, int height)
```

Parameters

`name` string

`width` int

`height` int

NormalLayer(string, Texture2D)

```
public NormalLayer(string name, Texture2D texture)
```

Parameters

`name` string

`texture` Texture2D

NormalLayer(Texture2D)

```
public NormalLayer(Texture2D texture)
```

Parameters

`texture` Texture2D

Properties

layerType

```
public override LayerType layerType { get; }
```

Property Value

LayerType

Methods

ClearFrames()

Deletes all key frames.

```
public override void ClearFrames()
```

DeepCopy()

Creates a deep copy of the Layer.

```
public override Layer DeepCopy()
```

Returns

Layer

DeleteKeyFrameNoEvent(int)

Deletes the key frame at the given frame index, if there is one, in which case it returns that key frame. Otherwise it returns null.

```
protected override AnimationKeyFrame DeleteKeyFrameNoEvent(int keyframe)
```

Parameters

`keyframe` int

Returns

AnimationKeyFrame

ExtendNoEvent(int, int, int, int)

Extends the dimensions of the layer in each direction by the given amounts, but does not invoke the onPixelsChanged event.

```
protected override void ExtendNoEvent(int left, int right, int up, int down)
```

Parameters

`left` int

`right` int

`up` int

`down` int

Flip(int, FlipDirection, AnimFrameRefMode)

Flips the given frame of the layer.

```
public void Flip(int frame, FlipDirection direction, AnimFrameRefMode frameRefMode)
```

Parameters

`frame` int

`direction` FlipDirection

`frameRefMode` AnimFrameRefMode

FlipNoEvent(FlipDirection)

Flips the layer, but does not invoke the onPixelsChanged event.

```
protected override void FlipNoEvent(FlipDirection direction)
```

Parameters

`direction` FlipDirection

GetPixel(IntVector2, int, bool)

Gets the colour of the pixel.

```
public override Color GetPixel(IntVector2 pixel, int frame, bool useLayerOpacity
= true)
```

Parameters

`pixel` IntVector2

`frame` int

`useLayerOpacity` bool

Returns

Color

LoadJSON(JSON)

```
protected override void LoadJSON(JSON json)
```

Parameters

`json` JSON

Offset(IntVector2)

Offsets the texture of every frame. (Moves the texture so the bottom-left corner is at the coordinates 'offset'.

```
public void Offset(IntVector2 offset)
```

Parameters

`offset` IntVector2

Offset(int, IntVector2, AnimFrameRefMode)

Offsets the texture at the given frame. (Moves the texture so the bottom-left corner is at the coordinates 'offset'.

```
public void Offset(int frame, IntVector2 offset, AnimFrameRefMode frameRefMode)
```

Parameters

`frame` int

`offset` IntVector2

`frameRefMode` AnimFrameRefMode

OverlayTexture(int, Texture2D, IntVector2, AnimFrameRefMode)

Overlays the texture onto the given frame, placing the bottom-left corner at the coordinates 'offset' (which don't have to be within the image). Uses Normal blend mode.

```
public void OverlayTexture(int frame, Texture2D overlayTex, IntVector2 offset,
AnimFrameRefMode frameRefMode)
```

Parameters

`frame` int

`overlayTex` Texture2D

`offset` IntVector2

`frameRefMode` AnimFrameRefMode

OverlayTexture(int, Texture2D, AnimFrameRefMode)

Overlays the texture onto the given frame. Uses Normal blend mode.

```
public void OverlayTexture(int frame, Texture2D overlayTex,
AnimFrameRefMode frameRefMode)
```

Parameters

`frame` int

`overlayTex` Texture2D

`frameRefMode` AnimFrameRefMode

OverlayTexture(Texture2D)

Overlays the texture onto every frame. Uses Normal blend mode.

```
public void OverlayTexture(Texture2D overlayTex)
```

Parameters

`overlayTex` Texture2D

OverlayTexture(Texture2D, IntVector2)

Overlays the texture onto every frame, placing the bottom-left corner at the coordinates 'offset' (which don't have to be within the image). Uses Normal blend mode.

```
public void OverlayTexture(Texture2D overlayTex, IntVector2 offset)
```

Parameters

`overlayTex` Texture2D

`offset` IntVector2

Rotate(int, RotationAngle, AnimFrameRefMode)

Rotates the given frame of the layer. Rotation is clockwise.

```
public void Rotate(int frame, RotationAngle angle, AnimFrameRefMode frameRefMode)
```

Parameters

`frame` int

`angle` RotationAngle

`frameRefMode` AnimFrameRefMode

RotateNoEvent(RotationAngle)

Rotates the layer, but does not invoke the onPixelsChanged event. Rotation is clockwise.

```
protected override void RotateNoEvent(RotationAngle angle)
```

Parameters

`angle` RotationAngle

ScaleNoEvent(int, int)

Resizes the dimensions to the new width and height, but does not invoke the onPixelsChanged event.

```
protected override void ScaleNoEvent(int newWidth, int newHeight)
```

Parameters

`newWidth` int

`newHeight` int

ScaleNoEvent(float, float)

Resizes the dimensions of the file by the scale factors, but does not invoke the onPixelsChanged event.

```
protected override void ScaleNoEvent(float xScaleFactor, float yScaleFactor)
```

Parameters

`xScaleFactor` float

`yScaleFactor` float

SetPixelsNoEvent(IntVector2[], int, Color, AnimFrameRefMode)

Sets the colour of the pixels. You do not need to check the pixels are in the layer as this check is done in Layer.SetPixels(), which is the only way this method is called.

```
protected override IntVector2[] SetPixelsNoEvent(IntVector2[] pixels, int frame,
Color colour, AnimFrameRefMode frameRefMode)
```

Parameters

`pixels` IntVector2[]

`frame` int

`colour` Color

`frameRefMode` AnimFrameRefMode

Returns

IntVector2[]

SetTexture(int, Texture2D, AnimFrameRefMode)

Sets the texture at the given frame.

```
public void SetTexture(int frame, Texture2D texture, AnimFrameRefMode frameRefMode)
```

Parameters

`frame` int

`texture` Texture2D

`frameRefMode` AnimFrameRefMode

ToJSON()

```
public override JSON ToJSON()
```

Returns

JSON

Class TileLayer

Namespace: PAC.Layers

A class to represent a tile layer - one for placing and editing tileset tiles.

```
public class TileLayer : Layer, IJSONable
```

Inheritance

object ← Layer ← TileLayer

Implements

IJSONable

Inherited Members

Layer.name , Layer.width , Layer.height , Layer.rect , Layer._visible , Layer.visible , Layer.locked , Layer.opacity , Layer._blendMode , Layer.blendMode , Layer.keyFrames , Layer.keyFrameIndices , Layer.keyFrameTextures , Layer.onPixelsChanged , Layer.SetPixel(int, int, int, Color, AnimFrameRefMode) , Layer.SetPixel(IntVector2, int, Color, AnimFrameRefMode) , Layer.SetPixels(IntVector2[], int, Color, AnimFrameRefMode) , Layer.GetPixel(int, int, int, bool) , Layer.IsBlank() , Layer.Flip(FlipDirection) , Layer.Rotate(RotationAngle) , Layer.Extend(int, int, int, int) , Layer.Scale(float) , Layer.Scale(float, float) , Layer.Scale(int, int) , Layer.FromJSON(JSON) , Layer.this[int] , Layer.GetKeyFrame(int) , Layer.HasKeyFrameAt(int) , Layer.AddKeyFrame(int) , Layer.AddKeyFrame(int, Texture2D) , Layer.AddKeyFrame(AnimationKeyFrame) , Layer.DeleteMostRecentKeyFrame(int) , Layer.DeleteKeyFrame(int) , Layer.DeleteKeyFrame(AnimationKeyFrame) , Layer.SubscribeToOnPixelsChanged(UnityAction<IntVector2[], int[]>) , Layer.SubscribeToOnVisibilityChanged(UnityAction) , Layer.SubscribeToOnBlendModeChanged(UnityAction) , Layer.SubscribeToOnKeyFrameAdded(UnityAction<int>) , Layer.SubscribeToOnKeyFrameRemoved(UnityAction<int>) , object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

TileLayer(TileLayer)

Creates a deep copy of the TileLayer.

```
public TileLayer(TileLayer layer)
```

Parameters

`layer` TileLayer

TileLayer(string, int, int)

```
public TileLayer(string name, int width, int height)
```

Parameters

`name` string

`width` int

`height` int

Properties

layerType

```
public override LayerType layerType { get; }
```

Property Value

LayerType

tiles

The tiles on this layer.

```
public List<Tile> tiles { get; }
```

Property Value

List<Tile>

Methods

AddTile(Tile)

Adds the tile to the layer.

```
public void AddTile(Tile tile)
```

Parameters

`tile` Tile

ClearFrames()

Deletes all key frames.

```
public override void ClearFrames()
```

ClearTiles()

Removes all tiles.

```
public void ClearTiles()
```

ContainsTile(Tile)

Returns true if the tile appears on this layer.

```
public bool ContainsTile(Tile tile)
```

Parameters

`tile` Tile

Returns

bool

DeepCopy()

Creates a deep copy of the Layer.

```
public override Layer DeepCopy()
```

Returns

Layer

DeleteKeyFrameNoEvent(int)

Deletes the key frame at the given frame index, if there is one, in which case it returns that key frame. Otherwise it returns null.

```
protected override AnimationKeyFrame DeleteKeyFrameNoEvent(int keyframe)
```

Parameters

`keyframe` int

Returns

AnimationKeyFrame

ExtendNoEvent(int, int, int, int)

Extends the dimensions of the layer in each direction by the given amounts, but does not invoke the onPixelsChanged event.

```
protected override void ExtendNoEvent(int left, int right, int up, int down)
```

Parameters

`left` int

`right` int

`up` int

`down` int

FlipNoEvent(FlipDirection)

Flips the layer, but does not invoke the onPixelsChanged event.

```
protected override void FlipNoEvent(FlipDirection direction)
```

Parameters

`direction` FlipDirection

GetLinkedPixels(IntVector2)

Get the pixels that are linked to the given pixel due to multiple tiles having the same file - i.e. they point to the same pixel within the tiles' file.

```
public IntVector2[] GetLinkedPixels(IntVector2 pixel)
```

Parameters

`pixel` IntVector2

Returns

IntVector2[]

GetLinkedPixels(IntVector2[])

Get the pixels that are linked to the given pixels due to multiple tiles having the same file - i.e. they point to the same pixels within the tiles' file.

```
public IntVector2[] GetLinkedPixels(IntVector2[] pixels)
```

Parameters

`pixels` IntVector2[]

Returns

IntVector2[]

GetPixel(IntVector2, int, bool)

Gets the colour of the pixel.

```
public override Color GetPixel(IntVector2 pixel, int frame, bool useLayerOpacity
= true)
```

Parameters

`pixel` IntVector2

`frame` int

`useLayerOpacity` bool

Returns

Color

LoadJSON(JSON)

```
protected override void LoadJSON(JSON json)
```

Parameters

`json` JSON

PixelToTile(IntVector2)

Gets the tile that the pixel lands in, or null if there isn't one.

```
public Tile PixelToTile(IntVector2 pixel)
```

Parameters

`pixel` IntVector2

Returns

Tile

PixelToTile(int, int)

Gets the tile that the pixel (x, y) lands in, or null if there isn't one.

```
public Tile PixelToTile(int x, int y)
```

Parameters

`x` int

`y` int

Returns

Tile

RemoveTile(Tile)

Removes the tile from the layer. Throws an error if the tile is not in the layer.

```
public void RemoveTile(Tile tile)
```

Parameters

`tile` Tile

RerenderKeyFrame(int)

Rerenders the keyframe.

```
public void RerenderKeyFrame(int frame)
```

Parameters

`frame` int

RerenderKeyFrame(int, IntRect)

Rerenders the section of the keyframe within the given rect.

```
public void RerenderKeyFrame(int frame, IntRect rect)
```

Parameters

`frame` int

`rect` IntRect

RerenderKeyFrame(int, IntVector2[])

Rerenders the given pixels of the keyframe.

```
public void RerenderKeyFrame(int frame, IntVector2[] pixels)
```

Parameters

`frame` int

`pixels` IntVector2[]

RerenderKeyFrames()

Rerenders all keyframes.

```
public void RerenderKeyFrames()
```

RerenderKeyFrames(IntRect)

Rerenders the section of every keyframe within the given rect.

```
public void RerenderKeyFrames(IntRect rect)
```

Parameters

`rect` IntRect

RerenderKeyFrames(IntVector2[])

Rerenders the given pixels of every keyframe.

```
public void RerenderKeyFrames(IntVector2[] pixels)
```

Parameters

`pixels` IntVector2[]

RerenderKeyFrames(int[])

Rerenders the given keyframes.

```
public void RerenderKeyFrames(int[] keyFrames)
```

Parameters

`keyFrames` int[]

RerenderKeyFrames(int[], IntRect)

Rerenders the section of the given keyframes within the given rect.

```
public void RerenderKeyFrames(int[] keyFrames, IntRect rect)
```

Parameters

`keyFrames` int[]

`rect` IntRect

RerenderKeyFrames(int[], IntVector2[])

Rerenders the given pixels of the given keyframes.

```
public void RerenderKeyFrames(int[] keyFrames, IntVector2[] pixels)
```

Parameters

`keyFrames` int[]

`pixels` IntVector2[]

RotateNoEvent(RotationAngle)

Rotates the layer, but does not invoke the onPixelsChanged event. Rotation is clockwise.

```
protected override void RotateNoEvent(RotationAngle angle)
```

Parameters

`angle` RotationAngle

ScaleNoEvent(int, int)

Resizes the dimensions to the new width and height, but does not invoke the onPixelsChanged event.

```
protected override void ScaleNoEvent(int newWidth, int newHeight)
```

Parameters

`newWidth` int

`newHeight` int

ScaleNoEvent(float, float)

Resizes the dimensions of the file by the scale factors, but does not invoke the onPixelsChanged event.

```
protected override void ScaleNoEvent(float xScaleFactor, float yScaleFactor)
```

Parameters

`xScaleFactor` float

`yScaleFactor` float

SetPixelsNoEvent(IntVector2[], int, Color, AnimFrameRefMode)

Sets the colour of the pixels. You do not need to check the pixels are in the layer as this check is done in Layer.SetPixels(), which is the only way this method is called.

```
protected override IntVector2[] SetPixelsNoEvent(IntVector2[] pixels, int frame,
Color colour, AnimFrameRefMode frameRefMode)
```

Parameters

`pixels` IntVector2[]

`frame` int

`colour` Color

`frameRefMode` AnimFrameRefMode

Returns

IntVector2[]

ToJSON()

```
public override JSON ToJSON()
```

Returns

JSON

Namespace PAC.Screen

Classes

MaximiseWindow

ScreenInfo

Class MaximiseWindow

Namespace: PAC.Screen

```
public class MaximiseWindow : MonoBehaviour
```

Inheritance

object ← MaximiseWindow

Methods

ShowWindowAsync(int, int)

```
public static extern bool ShowWindowAsync(int hWnd, int nCmdShow)
```

Parameters

`hWnd` int

`nCmdShow` int

Returns

bool

Class ScreenInfo

Namespace: PAC.Screen

```
public static class ScreenInfo
```

Inheritance

object ← ScreenInfo

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Properties

screenWorldHeight

Returns the height of the screen in world coords.

```
public static float screenWorldHeight { get; }
```

Property Value

float

screenWorldWidth

Returns the width of the screen in world coords.

```
public static float screenWorldWidth { get; }
```

Property Value

float

Methods

GetScreenPixelColour(IntVector2)

Returns the colour of the screen pixel at the given coords. Best results when called after end of frame, using 'yield return new WaitForEndOfFrame()'.

```
public static Color GetScreenPixelColour(IntVector2 coords)
```

Parameters

`coords` IntVector2

Returns

Color

GetScreenPixelColour(int, int)

Returns the colour of the screen pixel at coords (x, y). Best results when called after end of frame, using 'yield return new WaitForEndOfFrame()'.

```
public static Color GetScreenPixelColour(int x, int y)
```

Parameters

`x` int

`y` int

Returns

Color

Namespace PAC.Serialization

Classes

LayerSerializationSurrogate

Texture2DSerializationSurrogate

Class LayerSerializationSurrogate

Namespace: PAC.Serialization

```
[Obsolete]
public class LayerSerializationSurrogate : ISerializationSurrogate
```

Inheritance

object ← LayerSerializationSurrogate

Implements

ISerializationSurrogate

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

GetObjectData(object, SerializationInfo, StreamingContext)

Populates the provided SerializationInfo with the data needed to serialize the object.

```
public void GetObjectData(object obj, SerializationInfo info,
StreamingContext context)
```

Parameters

`obj` object

The object to serialize.

`info` SerializationInfo

The SerializationInfo to populate with data.

`context` StreamingContext

The destination (see StreamingContext) for this serialization.

Exceptions

SecurityException

The caller does not have the required permission.

SetObjectData(object, SerializationInfo, StreamingContext, ISurrogateSelector)

Populates the object using the information in the SerializationInfo.

```
public object SetObjectData(object obj, SerializationInfo info, StreamingContext
context, ISurrogateSelector selector)
```

Parameters

`obj` object

The object to populate.

`info` SerializationInfo

The information to populate the object.

`context` StreamingContext

The source from which the object is deserialized.

`selector` ISurrogateSelector

The surrogate selector where the search for a compatible surrogate begins.

Returns

object

The populated deserialized object.

Exceptions

SecurityException

The caller does not have the required permission.

Class Texture2DSerializationSurrogate

Namespace: PAC.Serialization

```
[Obsolete]
public class Texture2DSerializationSurrogate : ISerializationSurrogate
```

Inheritance

object ← Texture2DSerializationSurrogate

Implements

ISerializationSurrogate

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

GetObjectData(object, SerializationInfo, StreamingContext)

Populates the provided SerializationInfo with the data needed to serialize the object.

```
public void GetObjectData(object obj, SerializationInfo info,
StreamingContext context)
```

Parameters

`obj` object

The object to serialize.

`info` SerializationInfo

The SerializationInfo to populate with data.

`context` StreamingContext

The destination (see StreamingContext) for this serialization.

Exceptions

SecurityException

The caller does not have the required permission.

SetObjectData(object, SerializationInfo, StreamingContext, ISurrogateSelector)

Populates the object using the information in the SerializationInfo.

```
public object SetObjectData(object obj, SerializationInfo info, StreamingContext
context, ISurrogateSelector selector)
```

Parameters

`obj` object

The object to populate.

`info` SerializationInfo

The information to populate the object.

`context` StreamingContext

The source from which the object is deserialized.

`selector` ISurrogateSelector

The surrogate selector where the search for a compatible surrogate begins.

Returns

object

The populated deserialized object.

Exceptions

SecurityException

The caller does not have the required permission.

Namespace PAC.Shaders

Classes

HueSaturationBoxShader

LightnessSliderShader

Outline

RainbowOutline

Class HueSaturationBoxShader

Namespace: PAC.Shaders

```
public class HueSaturationBoxShader : MonoBehaviour
```

Inheritance

object ← HueSaturationBoxShader

Fields

hueSaturationBoxMaterial

```
public Material hueSaturationBoxMaterial
```

Field Value

Material

Class LightnessSliderShader

Namespace: PAC.Shaders

```
public class LightnessSliderShader : MonoBehaviour
```

Inheritance

object ← LightnessSliderShader

Fields

lightnessSliderMaterial

```
public Material lightnessSliderMaterial
```

Field Value

Material

Class Outline

Namespace: PAC.Shaders

```
public class Outline : MonoBehaviour
```

Inheritance

object ← Outline

Fields

outlineMaterial

```
public Material outlineMaterial
```

Field Value

Material

Properties

colour

```
public Color colour { get; set; }
```

Property Value

Color

keepExistingTexture

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

Property Value

bool

outlineEnabled

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

Property Value

bool

thickness

```
public float thickness { get; set; }
```

Property Value

float

Class RainbowOutline

Namespace: PAC.Shaders

```
public class RainbowOutline : MonoBehaviour
```

Inheritance

object ← RainbowOutline

Fields

rainbowOutlineMaterial

```
public Material rainbowOutlineMaterial
```

Field Value

Material

Properties

keepExistingTexture

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

Property Value

bool

outlineEnabled

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

Property Value

bool

thickness

```
public float thickness { get; set; }
```

Property Value

float

Namespace PAC.Themes

Classes

Theme

ThemeManager

UITheme

Enums

ThemeObjectType

Class Theme

Namespace: PAC.Themes

```
public class Theme : ScriptableObject
```

Inheritance

object ← Theme

Fields

backgroundColour

```
public Color backgroundColour
```

Field Value

Color

buttonColour

```
public Color buttonColour
```

Field Value

Color

buttonHoverColour

```
public Color buttonHoverColour
```

Field Value

Color

buttonPressedColour

```
public Color buttonPressedColour
```

Field Value

Color

layerTileHoverTint

```
public Color layerTileHoverTint
```

Field Value

Color

layerTileOffColour

```
public Color layerTileOffColour
```

Field Value

Color

layerTileOnColour

```
public Color layerTileOnColour
```

Field Value

Color

layerTilePressedColour

```
public Color layerTilePressedColour
```

Field Value

Color

panelColour

```
public Color panelColour
```

Field Value

Color

scaleColour

```
public Color scaleColour
```

Field Value

Color

scaleHoverColour

```
public Color scaleHoverColour
```

Field Value

Color

scalePressedColour

```
public Color scalePressedColour
```

Field Value

Color

scrollbarBackgroundColour

```
public Color scrollbarBackgroundColour
```

Field Value

Color

scrollbarBackgroundHoverColour

```
public Color scrollbarBackgroundHoverColour
```

Field Value

Color

scrollbarBackgroundPressedColour

```
public Color scrollbarBackgroundPressedColour
```

Field Value

Color

scrollbarHandleColour

```
public Color scrollbarHandleColour
```

Field Value

Color

scrollbarHandleHoverColour

```
public Color scrollbarHandleHoverColour
```

Field Value

Color

scrollbarHandlePressedColour

```
public Color scrollbarHandlePressedColour
```

Field Value

Color

shadowColour

```
public Color shadowColour
```

Field Value

Color

subPanelColour

```
public Color subPanelColour
```

Field Value

Color

textboxColour

```
public Color textboxColour
```

Field Value

Color

textboxHoverColour

```
public Color textboxHoverColour
```

Field Value

Color

textboxPressedColour

```
public Color textboxPressedColour
```

Field Value

Color

textboxSelectedColour

```
public Color textboxSelectedColour
```

Field Value

Color

themeName

```
public string themeName
```

Field Value

string

toggleButtonHoverTint

```
public Color toggleButtonHoverTint
```

Field Value

Color

toggleButtonOffColour

```
public Color toggleButtonOffColour
```

Field Value

Color

toggleButtonOnColour

```
public Color toggleButtonOnColour
```

Field Value

Color

toggleButtonPressedColour

```
public Color toggleButtonPressedColour
```

Field Value

Color

Methods

SubscribeToOnChanged(UnityAction)

```
public void SubscribeToOnChanged(UnityAction call)
```

Parameters

`call` UnityAction

Class ThemeManager

Namespace: PAC.Themes

```
public class ThemeManager : MonoBehaviour
```

Inheritance

object ← ThemeManager

Properties

currentTheme

```
public Theme currentTheme { get; }
```

Property Value

Theme

themes

```
public List<Theme> themes { get; }
```

Property Value

List<Theme>

Methods

SetTheme(Theme)

```
public void SetTheme(Theme theme)
```

Parameters

`theme` Theme

SetTheme(string)

```
public void SetTheme(string themeName)
```

Parameters

`themeName` string

SubscribeToThemeChanged(UnityAction)

```
public void SubscribeToThemeChanged(UnityAction call)
```

Parameters

`call` UnityAction

Enum ThemeObjectType

Namespace: PAC.Themes

```
public enum ThemeObjectType
```

Fields

`Background = 1`

`None = 0`

`Panel = 2`

`RadioButton = 5`

`Shadow = 4`

`SubPanel = 3`

Class UITheme

Namespace: PAC.Themes

```
public class UITheme : MonoBehaviour
```

Inheritance

object ← UITheme

Namespace PAC.Tilesets

Classes

Tile

TileOutlineManager

Tileset

TilesetManager

Class Tile

Namespace: PAC.Tilesets

```
public class Tile
```

Inheritance

object ← Tile

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Tile(File, IntVector2, TileLayer[])

```
public Tile(File file, IntVector2 bottomLeft, TileLayer[] linkedTileLayers)
```

Parameters

`file` File

`bottomLeft` IntVector2

`linkedTileLayers` TileLayer[]

The tile layers that the tile is on that are associated with each of the layers in the tile's file. Given in the same order as the layers in the tile's file.

Properties

bottomLeft

```
public IntVector2 bottomLeft { get; set; }
```

Property Value

IntVector2

bottomRight

```
public IntVector2 bottomRight { get; set; }
```

Property Value

IntVector2

centre

```
public Vector2 centre { get; }
```

Property Value

Vector2

file

```
public File file { get; }
```

Property Value

File

height

```
public int height { get; }
```

Property Value

int

rect

```
public IntRect rect { get; }
```

Property Value

IntRect

tileLayersAppearsOn

```
public TileLayer[] tileLayersAppearsOn { get; }
```

Property Value

TileLayer[]

topLeft

```
public IntVector2 topLeft { get; set; }
```

Property Value

IntVector2

topRight

```
public IntVector2 topRight { get; set; }
```

Property Value

IntVector2

width

```
public int width { get; }
```

Property Value

int

Methods

LayerInTileToTileLayer(Layer)

Takes in the layer (that must be in the tile's file) and returns the tile layer (that the tile is on) that the layer is associated with.

```
public TileLayer LayerInTileToTileLayer(Layer layer)
```

Parameters

`layer` Layer

Returns

TileLayer

SubscribeToOnMoved(UnityAction<IntRect>)

```
public void SubscribeToOnMoved(UnityAction<IntRect> call)
```

Parameters

`call` UnityAction<IntRect>

TileLayerToLayerInTile(TileLayer)

Takes in the tile layer (that this tile must be on) and returns the layer it is associated with in the tile's file.

```
public Layer TileLayerToLayerInTile(TileLayer tileLayer)
```

Parameters

`tileLayer` TileLayer

Returns

Layer

Class TileOutlineManager

Namespace: PAC.Tilesets

```
public class TileOutlineManager : MonoBehaviour
```

Inheritance

object ← TileOutlineManager

Methods

HideShowTileOutline(Tile, bool)

```
public void HideShowTileOutline(Tile tile, bool show)
```

Parameters

`tile` Tile

`show` bool

HideTileOutline(Tile)

```
public void HideTileOutline(Tile tile)
```

Parameters

`tile` Tile

RefreshTileOutlines()

```
public void RefreshTileOutlines()
```

ShowTileOutline(Tile)

```
public void ShowTileOutline(Tile tile)
```

Parameters

`tile` Tile

Class Tileset

Namespace: PAC.Tilesets

```
public class Tileset
```

Inheritance

object ← Tileset

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

Tileset(string)

```
public Tileset(string name)
```

Parameters

`name` string

Tileset(string, List<File>)

```
public Tileset(string name, List<File> tiles)
```

Parameters

`name` string

`tiles` List<File>

Properties

Count

```
public int Count { get; }
```

Property Value

int

tilesetName

```
public string tilesetName { get; }
```

Property Value

string

Methods

AddTile(File)

```
public void AddTile(File tile)
```

Parameters

`tile` File

RemoveTile(File)

```
public bool RemoveTile(File tile)
```

Parameters

`tile` File

Returns

bool

Class TilesetManager

Namespace: PAC.Tilesets

```
public class TilesetManager : MonoBehaviour
```

Inheritance

object ← TilesetManager

Methods

SubscribeToOnTileIconSelected(UnityAction<File>)

```
public void SubscribeToOnTileIconSelected(UnityAction<File> call)
```

Parameters

`call` UnityAction<File>

Namespace PAC.UI

Classes

DialogBoxManager

UIButton

UICollapser

UIColourField

UIColourFieldGroup

UIColourPicker

UIDropdown

UIDropdownChoice

UIElement

UIGridPacker

UIKeyboardShortcut

UIManager

UIModalWindow

UINumberField

UIScale

UIScrollbar

UITabManager

UITextbox

UITileIcon

UIToggleButton

UIToggleGroup

UIToolButton

UITooltip

UIViewport

Enums

CollapsedState

DropdownCloseMode

DropdownDirection

GridAlignment

ScrollDirection

UIAnchorPoint

UIElementDeselectMode

UIElementSelectMode

UITextboxAnchorPoint

ViewportSide

Enum CollapsedState

Namespace: PAC.UI

```
public enum CollapsedState
```

Fields

`Collapsed = 1`

`Uncollapsed = 0`

Class DialogBoxManager

Namespace: PAC.UI

```
public class DialogBoxManager : MonoBehaviour
```

Inheritance

object ← DialogBoxManager

Methods

CloseBrushSettingsWindow()

```
public void CloseBrushSettingsWindow()
```

CloseDialogBox(GameObject)

```
public void CloseDialogBox(GameObject dialogBox)
```

Parameters

`dialogBox` GameObject

CloseDialogBox(GameObject, Action)

```
public void CloseDialogBox(GameObject dialogBox, Action onComplete)
```

Parameters

`dialogBox` GameObject

`onComplete` Action

CloseExtendCropWindow()

```
public void CloseExtendCropWindow()
```

CloseGridWindow()

```
public void CloseGridWindow()
```

CloseImportPACWindow()

```
public void CloseImportPACWindow()
```

CloseKeyboardShortcutsWindow()

```
public void CloseKeyboardShortcutsWindow()
```

CloseLayerPropertiesWindow()

```
public void CloseLayerPropertiesWindow()
```

CloseModalWindow(UIModalWindow)

```
public void CloseModalWindow(UIModalWindow modalWindow)
```

Parameters

`modalWindow` UIModalWindow

CloseNewFileWindow()

```
public void CloseNewFileWindow()
```

CloseOutlineWindow()

```
public void CloseOutlineWindow()
```

CloseReplaceColourWindow()

```
public void CloseReplaceColourWindow()
```

CloseScaleWindow()

```
public void CloseScaleWindow()
```

ConfirmExtendCropWindow()

```
public void ConfirmExtendCropWindow()
```

ConfirmGridWindow()

```
public void ConfirmGridWindow()
```

ConfirmImportPACWindow()

```
public void ConfirmImportPACWindow()
```

ConfirmNewFileWindow()

```
public void ConfirmNewFileWindow()
```

ConfirmOutlineWindow()

```
public void ConfirmOutlineWindow()
```

ConfirmReplaceColourWindow()

```
public void ConfirmReplaceColourWindow()
```

ConfirmScaleWindow()

```
public void ConfirmScaleWindow()
```

OpenBrushSettingsWindow()

```
public void OpenBrushSettingsWindow()
```

OpenDialogBox(GameObject)

```
public void OpenDialogBox(GameObject dialogBox)
```

Parameters

`dialogBox` GameObject

OpenExtendCropWindow()

```
public void OpenExtendCropWindow()
```

OpenGridWindow()

```
public void OpenGridWindow()
```

OpenImportPACWindow(File)

```
public void OpenImportPACWindow(File file)
```

Parameters

`file` File

OpenKeyboardShortcutsWindow()

```
public void OpenKeyboardShortcutsWindow()
```

OpenLayerPropertiesWindow(int)

```
public void OpenLayerPropertiesWindow(int layerIndex)
```

Parameters

`layerIndex` int

OpenModalWindow()

```
public UIModalWindow OpenModalWindow()
```

Returns

UIModalWindow

OpenModalWindow(string, string)

```
public UIModalWindow OpenModalWindow(string title, string message)
```

Parameters

`title` string

`message` string

Returns

UIModalWindow

OpenModalWindow(string, string, string[], UnityAction[])

```
public UIModalWindow OpenModalWindow(string title, string message, string[]
buttonTexts, UnityAction[] buttonOnClicks)
```

Parameters

`title` string

`message` string

`buttonTexts` string[]

`buttonOnClicks` UnityAction[]

Returns

UIModalWindow

OpenNewFileWindow()

```
public void OpenNewFileWindow()
```

OpenOutlineWindow()

```
public void OpenOutlineWindow()
```

OpenReplaceColourWindow()

```
public void OpenReplaceColourWindow()
```

OpenScaleWindow()

```
public void OpenScaleWindow()
```

OpenUnsavedChangesWindow(int)

```
public void OpenUnsavedChangesWindow(int fileIndex)
```

Parameters

`fileIndex` int

SetNewFileHeight(int)

```
public void SetNewFileHeight(int height)
```

Parameters

`height` int

SetNewFileWidth(int)

```
public void SetNewFileWidth(int width)
```

Parameters

`width` int

UnsavedChangesNo()

```
public void UnsavedChangesNo()
```

UnsavedChangesYes()

```
public void UnsavedChangesYes()
```

UpdateBrushSettingsShape()

```
public void UpdateBrushSettingsShape()
```

UpdateImportPACPreview()

```
public void UpdateImportPACPreview()
```

Enum DropdownCloseMode

Namespace: PAC.UI

```
public enum DropdownCloseMode
```

Fields

`ClickOff = 0`

`MouseOff = 1`

Enum DropdownDirection

Namespace: PAC.UI

```
public enum DropdownDirection
```

Fields

`down = -1`

`up = 1`

Enum GridAlignment

Namespace: PAC.UI

```
public enum GridAlignment
```

Fields

`Centre = 0`

`Left = -1`

`Right = 1`

Enum ScrollDirection

Namespace: PAC.UI

```
public enum ScrollDirection
```

Fields

`Horizontal = 1`

`Vertical = 0`

Enum UIAnchorPoint

Namespace: PAC.UI

```
public enum UIAnchorPoint
```

Fields

bottomCentre = 7

bottomLeft = 6

bottomRight = 8

centre = 4

leftCentre = 3

rightCentre = 5

topCentre = 1

topLeft = 0

topRight = 2

Class UIButton

Namespace: PAC.UI

```
public class UIButton : MonoBehaviour
```

Inheritance

object ← UIButton

Fields

backgroundColour

```
public Color backgroundColour
```

Field Value

Color

backgroundHoverColour

```
public Color backgroundHoverColour
```

Field Value

Color

backgroundPressedColour

```
public Color backgroundPressedColour
```

Field Value

Color

Properties

height

```
public float height { get; set; }
```

Property Value

float

hoverImage

```
public Sprite hoverImage { get; }
```

Property Value

Sprite

image

```
public Sprite image { get; }
```

Property Value

Sprite

isPressed

```
public bool isPressed { get; }
```

Property Value

bool

pressedImage

```
public Sprite pressedImage { get; }
```

Property Value

Sprite

width

```
public float width { get; set; }
```

Property Value

float

Methods

Press()

```
public void Press()
```

RightClick()

```
public void RightClick()
```

SetImages(Sprite, Sprite, Sprite)

```
public void SetImages(Sprite image, Sprite hoverImage, Sprite pressedImage)
```

Parameters

`image` Sprite

`hoverImage` Sprite

`pressedImage` Sprite

SetSortingLayer(string)

```
public void SetSortingLayer(string sortingLayer)
```

Parameters

`sortingLayer` string

SetSortingLayer(string, int)

```
public void SetSortingLayer(string sortingLayer, int sortingOrder)
```

Parameters

`sortingLayer` string

`sortingOrder` int

SetSortingOrder(int)

```
public void SetSortingOrder(int sortingOrder)
```

Parameters

`sortingOrder` int

SetText(string)

```
public void SetText(string text)
```

Parameters

`text` string

SubscribeToClick(UnityAction)

```
public void SubscribeToClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToHover(UnityAction)

```
public void SubscribeToHover(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToIdle(UnityAction)

```
public void SubscribeToIdle(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToLeftClick(UnityAction)

```
public void SubscribeToLeftClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToRightClick(UnityAction)

```
public void SubscribeToRightClick(UnityAction call)
```

Parameters

`call` UnityAction

UpdateDisplay()

```
public void UpdateDisplay()
```

Class UICollapser

Namespace: PAC.UI

```
public class UICollapser : MonoBehaviour
```

Inheritance

object ← UICollapser

Methods

Collapse()

```
public void Collapse()
```

SetCollapsed(CollapsedState)

```
public void SetCollapsed(CollapsedState collapsedState)
```

Parameters

`collapsedState` CollapsedState

Uncollapse()

```
public void Uncollapse()
```

Class UIColourField

Namespace: PAC.UI

```
public class UIColourField : MonoBehaviour
```

Inheritance

object ← UIColourField

Properties

colour

```
public Color colour { get; }
```

Property Value

Color

colourPickerOpen

```
public bool colourPickerOpen { get; }
```

Property Value

bool

outlineThickness

```
public float outlineThickness { get; }
```

Property Value

float

Methods

CloseColourPicker()

```
public void CloseColourPicker()
```

OpenColourPicker()

```
public void OpenColourPicker()
```

SetColour(Color)

```
public void SetColour(Color colour)
```

Parameters

`colour` Color

SubscribeToColourChange(UnityAction)

```
public void SubscribeToColourChange(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToColourPickerClose(UnityAction)

```
public void SubscribeToColourPickerClose(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToColourPickerOpen(UnityAction)

```
public void SubscribeToColourPickerOpen(UnityAction call)
```

Parameters

`call` UnityAction

Class UIColourFieldGroup

Namespace: PAC.UI

```
public class UIColourFieldGroup : MonoBehaviour
```

Inheritance

object ← UIColourFieldGroup

Properties

Count

```
public int Count { get; }
```

Property Value

int

colourFields

```
public List<UIColourField> colourFields { get; }
```

Property Value

List<UIColourField>

currentOpenColourField

```
public UIColourField currentOpenColourField { get; }
```

Property Value

UIColourField

Methods

Add(UIColourField)

```
public bool Add(UIColourField colourField)
```

Parameters

`colourField` UIColourField

Returns

bool

Clear()

```
public void Clear()
```

Contains(UIColourField)

```
public bool Contains(UIColourField colourField)
```

Parameters

`colourField` UIColourField

Returns

bool

DestroyColourFields()

```
public void DestroyColourFields()
```

Opened(UIColourField)

```
public void Opened(UIColourField openedColourField)
```

Parameters

`openedColourField` UIColourField

Remove(UIColourField)

```
public bool Remove(UIColourField colourField)
```

Parameters

`colourField` UIColourField

Returns

bool

SubscribeToColourFieldOpen(UnityAction)

```
public void SubscribeToColourFieldOpen(UnityAction call)
```

Parameters

`call` UnityAction

Class UIColourPicker

Namespace: PAC.UI

```
public class UIColourPicker : MonoBehaviour
```

Inheritance

object ← UIColourPicker

Fields

colourPreview

```
public ColourPreview colourPreview
```

Field Value

ColourPreview

Properties

colour

```
public Color colour { get; }
```

Property Value

Color

Methods

Close()

```
public void Close()
```

SelectDeselectEyeDropper()

```
public void SelectDeselectEyeDropper()
```

SetColour(Color)

```
public void SetColour(Color colour)
```

Parameters

`colour` Color

SubscribeToClose(UnityAction)

```
public void SubscribeToClose(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToColourChange(UnityAction)

```
public void SubscribeToColourChange(UnityAction call)
```

Parameters

`call` UnityAction

UpdateColour(Color)

```
public void UpdateColour(Color colour)
```

Parameters

`colour` Color

Class UIDropdown

Namespace: PAC.UI

```
public class UIDropdown : MonoBehaviour
```

Inheritance

object ← UIDropdown

Fields

openingButton

```
public GameObject openingButton
```

Field Value

GameObject

parentDropdown

```
public UIDropdown parentDropdown
```

Field Value

UIDropdown

Properties

isRootDropdown

```
public bool isRootDropdown { get; }
```

Property Value

bool

open

```
public bool open { get; }
```

Property Value

bool

rootDropdown

```
public UIDropdown rootDropdown { get; }
```

Property Value

UIDropdown

Methods

Close()

```
public void Close()
```

CloseRoot()

Closes the highest-level dropdown containing this one.

```
public void CloseRoot()
```

FullyOpen()

Opens this dropdown and all child dropdowns, and all their child dropdowns, etc.

```
public void FullyOpen()
```

Initialise()

```
public void Initialise()
```

MouseOff()

```
public bool MouseOff()
```

Returns

bool

Open()

```
public void Open()
```

SetOpen(bool)

```
public void SetOpen(bool open)
```

Parameters

`open` bool

SetOpenEditor(bool)

```
public void SetOpenEditor(bool open)
```

Parameters

`open` bool

ToggleOpen()

```
public void ToggleOpen()
```

Class UIDropdownChoice

Namespace: PAC.UI

```
public class UIDropdownChoice : MonoBehaviour
```

Inheritance

object ← UIDropdownChoice

Fields

direction

```
public DropdownDirection direction
```

Field Value

DropdownDirection

options

```
public List<string> options
```

Field Value

List<string>

selectedOptionIndex

```
public int selectedOptionIndex
```

Field Value

int

Properties

selectedOption

```
public string selectedOption { get; }
```

Property Value

string

Methods

Select(string)

```
public bool Select(string option)
```

Parameters

`option` string

Returns

bool

SetUpDropdown()

```
public void SetUpDropdown()
```

SubscribeToOptionChanged(UnityAction)

```
public void SubscribeToOptionChanged(UnityAction call)
```

Parameters

`call` UnityAction

Class UIElement

Namespace: PAC.UI

```
public class UIElement : MonoBehaviour
```

Inheritance

object ← UIElement

Fields

viewport

```
public UIViewport viewport
```

Field Value

UIViewport

Properties

elementName

```
public string elementName { get; }
```

Property Value

string

Enum UIElementDeselectMode

Namespace: PAC.UI

```
public enum UIElementDeselectMode
```

Fields

`Manual = 0`

`OnInputTargetUntargeted = 1`

Enum UIElementSelectMode

Namespace: PAC.UI

```
public enum UIElementSelectMode
```

Fields

`Manual = 0`

`OnInputTargetTargeted = 1`

Class UIGridPacker

Namespace: PAC.UI

```
public class UIGridPacker : MonoBehaviour
```

Inheritance

object ← UIGridPacker

Methods

Repack()

```
public void Repack()
```

Class UIKeyboardShortcut

Namespace: PAC.UI

```
public class UIKeyboardShortcut : MonoBehaviour
```

Inheritance

object ← UIKeyboardShortcut

Fields

actionName

```
public string actionName
```

Field Value

string

Properties

shortcut

```
public KeyboardShortcut shortcut { get; set; }
```

Property Value

KeyboardShortcut

Methods

SubscribeToOnShortcutSet(UnityAction<KeyboardShortcut>)

```
public void SubscribeToOnShortcutSet(UnityAction<KeyboardShortcut> call)
```

Parameters

`call` UnityAction<KeyboardShortcut>

Class UIManager

Namespace: PAC.UI

```
public class UIManager : MonoBehaviour
```

Inheritance

object ← UIManager

Properties

selectedUIElement

```
public UIElement selectedUIElement { get; }
```

Property Value

UIElement

Methods

CanTargetInputTarget(InputTarget)

```
public bool CanTargetInputTarget(InputTarget inputTarget)
```

Parameters

`inputTarget` InputTarget

Returns

bool

TryTarget(UIElement)

```
public bool TryTarget(UIElement uiElement)
```

Parameters

`uiElement` UIElement

Returns

bool

TryUntarget(UIElement)

```
public bool TryUntarget(UIElement uiElement)
```

Parameters

`uiElement` UIElement

Returns

bool

Class UIModalWindow

Namespace: PAC.UI

```
public class UIModalWindow : MonoBehaviour
```

Inheritance

object ← UIModalWindow

Methods

AddButton(string, UnityAction)

```
public UIModalWindow AddButton(string text, UnityAction onClick)
```

Parameters

`text` string

`onClick` UnityAction

Returns

UIModalWindow

AddCloseButton(string)

```
public UIModalWindow AddCloseButton(string text)
```

Parameters

`text` string

Returns

UIModalWindow

Close()

```
public void Close()
```

SetMessage(string)

```
public UIModalWindow SetMessage(string message)
```

Parameters

`message` string

Returns

UIModalWindow

SetTitle(string)

```
public UIModalWindow SetTitle(string title)
```

Parameters

`title` string

Returns

UIModalWindow

Class UINumberField

Namespace: PAC.UI

```
public class UINumberField : MonoBehaviour
```

Inheritance

object ← UINumberField

Properties

max

```
public float max { get; set; }
```

Property Value

float

min

```
public float min { get; set; }
```

Property Value

float

value

```
public float value { get; set; }
```

Property Value

float

Methods

AddNumOfIncrements(int)

```
public void AddNumOfIncrements(int numOfIncrements)
```

Parameters

`numOfIncrements` int

Decrement()

```
public void Decrement()
```

Increment()

```
public void Increment()
```

SubscribeToValueChanged(UnityAction)

```
public void SubscribeToValueChanged(UnityAction call)
```

Parameters

`call` UnityAction

Class UIScale

Namespace: PAC.UI

```
public class UIScale : MonoBehaviour
```

Inheritance

object ← UIScale

Fields

backgroundColour

```
public Color backgroundColour
```

Field Value

Color

backgroundHoverColour

```
public Color backgroundHoverColour
```

Field Value

Color

backgroundPressedColour

```
public Color backgroundPressedColour
```

Field Value

height

```
public float height
```

Field Value

float

width

```
public float width
```

Field Value

float

Properties

decimalPlaces

```
public int decimalPlaces { get; }
```

Property Value

int

value

```
public float value { get; }
```

Property Value

float

Methods

SetValue(float)

```
public bool SetValue(float value)
```

Parameters

`value` float

Returns

bool

SetValueNoNotify(float)

```
public bool SetValueNoNotify(float value)
```

Parameters

`value` float

Returns

bool

SubscribeToValueChange(UnityAction)

```
public void SubscribeToValueChange(UnityAction call)
```

Parameters

`call` UnityAction

UpdateDisplay()

```
public void UpdateDisplay()
```

Class UIScrollbar

Namespace: PAC.UI

```
public class UIScrollbar : MonoBehaviour
```

Inheritance

object ← UIScrollbar

Fields

backgroundColour

```
public Color backgroundColour
```

Field Value

Color

backgroundHoverColour

```
public Color backgroundHoverColour
```

Field Value

Color

backgroundPressedColour

```
public Color backgroundPressedColour
```

Field Value

Color

handleColour

```
public Color handleColour
```

Field Value

Color

handleHoverColour

```
public Color handleHoverColour
```

Field Value

Color

handlePressedColour

```
public Color handlePressedColour
```

Field Value

Color

height

```
public float height
```

Field Value

float

width

```
public float width
```

Field Value

float

Properties

scrollAmount

```
public float scrollAmount { get; set; }
```

Property Value

float

Methods

GetScrollAmount()

```
public float GetScrollAmount()
```

Returns

float

SetScrollAmount(float)

```
public void SetScrollAmount(float scrollAmount)
```

Parameters

`scrollAmount` float

UpdateDisplay()

```
public void UpdateDisplay()
```

Class UITabManager

Namespace: PAC.UI

```
public class UITabManager : MonoBehaviour
```

Inheritance

object ← UITabManager

Methods

AddTab(GameObject)

```
public void AddTab(GameObject tab)
```

Parameters

`tab` GameObject

SelectTab(int)

```
public void SelectTab(int tabIndex)
```

Parameters

`tabIndex` int

SelectTabEditor(int)

```
public void SelectTabEditor(int tabIndex)
```

Parameters

`tabIndex` int

Class UITextbox

Namespace: PAC.UI

```
public class UITextbox : MonoBehaviour
```

Inheritance

object ← UITextbox

Fields

allowLetters

```
public bool allowLetters
```

Field Value

bool

allowNumbers

```
public bool allowNumbers
```

Field Value

bool

allowPunctuation

```
public bool allowPunctuation
```

Field Value

bool

allowSpaces

```
public bool allowSpaces
```

Field Value

bool

anchorPoint

```
public UITextboxAnchorPoint anchorPoint
```

Field Value

UITextboxAnchorPoint

backgroundColour

```
public Color backgroundColour
```

Field Value

Color

backgroundHoverColour

```
public Color backgroundHoverColour
```

Field Value

Color

backgroundPressedColour

```
public Color backgroundPressedColour
```

Field Value

Color

backgroundSelectedColour

```
public Color backgroundSelectedColour
```

Field Value

Color

height

```
public float height
```

Field Value

float

width

```
public float width
```

Field Value

float

Properties

prefix

```
public string prefix { get; }
```

Property Value

string

suffix

```
public string suffix { get; }
```

Property Value

string

text

```
public string text { get; }
```

Property Value

string

Methods

SetPrefix(string)

```
public void SetPrefix(string prefix)
```

Parameters

`prefix` string

SetSuffix(string)

```
public void SetSuffix(string suffix)
```

Parameters

`suffix` string

SetText(string)

```
public void SetText(string text)
```

Parameters

`text` string

SubscribeToFinishEvent(UnityAction)

```
public void SubscribeToFinishEvent(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToInputEvent(UnityAction)

```
public void SubscribeToInputEvent(UnityAction call)
```

Parameters

`call` UnityAction

UpdateAnchor()

```
public void UpdateAnchor()
```

UpdateDisplay()

```
public void UpdateDisplay()
```

Enum UITextboxAnchorPoint

Namespace: PAC.UI

```
public enum UITextboxAnchorPoint
```

Fields

`Centre = 4`

`Left = 3`

`Right = 5`

Class UITileIcon

Namespace: PAC.UI

```
public class UITileIcon : MonoBehaviour
```

Inheritance

object ← UITileIcon

Properties

file

```
public File file { get; }
```

Property Value

File

height

```
public float height { get; }
```

Property Value

float

width

```
public float width { get; }
```

Property Value

float

Methods

SetFile(File)

```
public void SetFile(File file)
```

Parameters

`file` File

SubscribeToOnClick(UnityAction)

```
public void SubscribeToOnClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnLeftClick(UnityAction)

```
public void SubscribeToOnLeftClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToOnRightClick(UnityAction)

```
public void SubscribeToOnRightClick(UnityAction call)
```

Parameters

`call` UnityAction

Class UIToggleButton

Namespace: PAC.UI

```
public class UIToggleButton : MonoBehaviour
```

Inheritance

object ← UIToggleButton

Fields

height

```
public float height
```

Field Value

float

hoverBackgroundTint

```
public Color hoverBackgroundTint
```

Field Value

Color

width

```
public float width
```

Field Value

float

Properties

hoverImage

```
public Sprite hoverImage { get; }
```

Property Value

Sprite

inToggleGroup

```
public bool inToggleGroup { get; }
```

Property Value

bool

offBackgroundColour

```
public Color offBackgroundColour { get; set; }
```

Property Value

Color

offImage

```
public Sprite offImage { get; }
```

Property Value

Sprite

on

```
public bool on { get; }
```

Property Value

bool

onBackgroundColour

```
public Color onBackgroundColour { get; set; }
```

Property Value

Color

onImage

```
public Sprite onImage { get; }
```

Property Value

Sprite

pressedBackgroundColour

```
public Color pressedBackgroundColour { get; set; }
```

Property Value

Color

pressedImage

```
public Sprite pressedImage { get; }
```

Property Value

Sprite

toggleGroup

```
public UIToggleGroup toggleGroup { get; }
```

Property Value

UIToggleGroup

toggleName

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

Property Value

string

Methods

JoinToggleGroup(UIToggleGroup)

```
public void JoinToggleGroup(UIToggleGroup toggleGroup)
```

Parameters

`toggleGroup` UIToggleGroup

LeaveToggleGroup()

```
public void LeaveToggleGroup()
```

Press()

```
public void Press()
```

RightClick()

```
public void RightClick()
```

SetImages(Sprite, Sprite, Sprite, Sprite)

```
public void SetImages(Sprite offImage, Sprite onImage, Sprite hoverImage,
Sprite pressedImage)
```

Parameters

`offImage` Sprite

`onImage` Sprite

`hoverImage` Sprite

`pressedImage` Sprite

SetOnOff(bool)

```
public void SetOnOff(bool on)
```

Parameters

on bool

SetText(string)

```
public void SetText(string text)
```

Parameters

text string

SetTextAlignment(TextAnchor)

```
public void SetTextAlignment(TextAnchor textAlignment)
```

Parameters

textAlignment TextAnchor

SubscribeToHover(UnityAction)

```
public void SubscribeToHover(UnityAction call)
```

Parameters

call UnityAction

SubscribeToLeftClick(UnityAction)

```
public void SubscribeToLeftClick(UnityAction call)
```

Parameters

call UnityAction

SubscribeToRightClick(UnityAction)

```
public void SubscribeToRightClick(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToTurnOff(UnityAction)

```
public void SubscribeToTurnOff(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToTurnOn(UnityAction)

```
public void SubscribeToTurnOn(UnityAction call)
```

Parameters

`call` UnityAction

UpdateDisplay()

```
public void UpdateDisplay()
```

Class UIToggleGroup

Namespace: PAC.UI

```
public class UIToggleGroup : MonoBehaviour
```

Inheritance

object ← UIToggleGroup

Properties

Count

```
public int Count { get; }
```

Property Value

int

canSelectMultiple

```
public bool canSelectMultiple { get; }
```

Property Value

bool

canSelectNone

```
public bool canSelectNone { get; }
```

Property Value

bool

currentToggle

```
public UIToggleButton currentToggle { get; }
```

Property Value

UIToggleButton

currentToggleIndex

```
public int currentToggleIndex { get; }
```

Property Value

int

hasSelectedToggle

```
public bool hasSelectedToggle { get; }
```

Property Value

bool

selectedIndices

```
public int[] selectedIndices { get; }
```

Property Value

int[]

selectedToggles

```
public UIToggleButton[] selectedToggles { get; }
```

Property Value

UIToggleButton[]

swapClickAndCtrlClick

```
public bool swapClickAndCtrlClick { get; }
```

Property Value

bool

toggles

```
public List<UIToggleButton> toggles { get; }
```

Property Value

List<UIToggleButton>

Methods

Add(UIToggleButton)

```
public bool Add(UIToggleButton toggle)
```

Parameters

`toggle` UIToggleButton

Returns

bool

Clear()

```
public void Clear()
```

Contains(UIToggleButton)

```
public bool Contains(UIToggleButton toggle)
```

Parameters

`toggle` UIToggleButton

Returns

bool

CtrlPress(UIToggleButton)

```
public bool CtrlPress(UIToggleButton toggle)
```

Parameters

`toggle` UIToggleButton

Returns

bool

CtrlPress(int)

```
public bool CtrlPress(int index)
```

Parameters

`index` int

Returns

bool

DestroyToggles()

```
public void DestroyToggles()
```

Press(UIToggleButton)

```
public bool Press(UIToggleButton toggle)
```

Parameters

`toggle` UIToggleButton

Returns

bool

Press(int)

```
public bool Press(int index)
```

Parameters

`index` int

Returns

bool

PressForceEvent(UIToggleButton)

```
public bool PressForceEvent(UIToggleButton toggle)
```

Parameters

`toggle` UIToggleButton

Returns

bool

PressForceEvent(int)

```
public bool PressForceEvent(int index)
```

Parameters

`index` int

Returns

bool

PressOrCtrlPress(UIToggleButton, bool)

```
public bool PressOrCtrlPress(UIToggleButton toggle, bool ctrlClick)
```

Parameters

`toggle` UIToggleButton

`ctrlClick` bool

Returns

bool

Remove(UIToggleButton)

```
public bool Remove(UIToggleButton toggle)
```

Parameters

`toggle` UIToggleButton

Returns

bool

SubscribeToSelectedToggleChange(UnityAction)

```
public void SubscribeToSelectedToggleChange(UnityAction call)
```

Parameters

`call` UnityAction

Class UIToolButton

Namespace: PAC.UI

```
public class UIToolButton : MonoBehaviour
```

Inheritance

object ← UIToolButton

Properties

buttons

```
public UIButton[] buttons { get; }
```

Property Value

UIButton[]

currentButton

```
public UIButton currentButton { get; }
```

Property Value

UIButton

Class UITooltip

Namespace: PAC.UI

```
public class UITooltip : MonoBehaviour
```

Inheritance

object ← UITooltip

Fields

padding

```
public Vector2 padding
```

Field Value

Vector2

Properties

globalHeight

```
public float globalHeight { get; }
```

Property Value

float

globalWidth

```
public float globalWidth { get; }
```

Property Value

float

localHeight

```
public float localHeight { get; }
```

Property Value

float

localWidth

```
public float localWidth { get; }
```

Property Value

float

text

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

Property Value

string

Methods

Awake()

```
public void Awake()
```

GoesOffBottomOfScreen()

```
public bool GoesOffBottomOfScreen()
```

Returns

bool

GoesOffLeftOfScreen()

```
public bool GoesOffLeftOfScreen()
```

Returns

bool

GoesOffRightOfScreen()

```
public bool GoesOffRightOfScreen()
```

Returns

bool

GoesOffScreen()

```
public bool GoesOffScreen()
```

Returns

bool

GoesOffTopOfScreen()

```
public bool GoesOffTopOfScreen()
```

Returns

bool

SetText(string)

```
public void SetText(string text)
```

Parameters

`text` string

Class UIViewport

Namespace: PAC.UI

```
public class UIViewport : MonoBehaviour
```

Inheritance

object ← UIViewport

Fields

boundScrollToContents

```
public bool boundScrollToContents
```

Field Value

bool

maxScrollAmount

```
public float maxScrollAmount
```

Field Value

float

maxScrollEnabled

```
public bool maxScrollEnabled
```

Field Value

bool

minScrollAmount

```
public float minScrollAmount
```

Field Value

float

minScrollEnabled

```
public bool minScrollEnabled
```

Field Value

bool

scrollDirection

```
public ScrollDirection scrollDirection
```

Field Value

ScrollDirection

Properties

collider

```
public BoxCollider2D collider { get; }
```

Property Value

defaultScrollSide

```
public ViewportSide defaultScrollSide { get; }
```

Property Value

ViewportSide

rectTransform

```
public RectTransform rectTransform { get; }
```

Property Value

RectTransform

scrollAmount

```
public float scrollAmount { get; }
```

Property Value

float

scrollingArea

```
public Transform scrollingArea { get; }
```

Property Value

Transform

Methods

AddScrollAmount(float)

```
public void AddScrollAmount(float scrollAmount)
```

Parameters

`scrollAmount` float

GetScrollMinMaxX()

Get the min and max x value of the space that the objects in this viewport's scroll area occupy. (Scaled to the scale of the viewport object.)

```
public Vector2 GetScrollMinMaxX()
```

Returns

Vector2

(min x, max x)

GetScrollMinMaxY()

Get the min and max y value of the space that the objects in this viewport's scroll area occupy. (Scaled to the scale of the viewport object.)

```
public Vector2 GetScrollMinMaxY()
```

Returns

Vector2

(min y, max y)

RefreshViewport()

```
public void RefreshViewport()
```

SetScrollAmount(float)

```
public void SetScrollAmount(float scrollAmount)
```

Parameters

`scrollAmount` float

SubscribeToRefresh(UnityAction)

```
public void SubscribeToRefresh(UnityAction call)
```

Parameters

`call` UnityAction

Enum ViewportSide

Namespace: PAC.UI

```
public enum ViewportSide
```

Fields

`Centre = 0`

`Negative = -1`

`Positive = 1`

Namespace PAC.UndoRedo

Classes

UndoRedoManager

UndoRedoState

Enums

UndoRedoAction

Enum UndoRedoAction

Namespace: PAC.UndoRedo

```
public enum UndoRedoAction
```

Fields

`Draw = 1`

`None = -1`

`ReorderLayers = 2`

`Undefined = 0`

Class UndoRedoManager

Namespace: PAC.UndoRedo

```
public class UndoRedoManager : MonoBehaviour
```

Inheritance

object ← UndoRedoManager

Methods

AddUndoState(UndoRedoState, int)

```
public void AddUndoState(UndoRedoState undoState, int fileIndex)
```

Parameters

`undoState` UndoRedoState

`fileIndex` int

SubscribeToRedo(UnityAction)

```
public void SubscribeToRedo(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToUndo(UnityAction)

```
public void SubscribeToUndo(UnityAction call)
```

Parameters

`call` UnityAction

SubscribeToUndoOrRedo(UnityAction)

```
public void SubscribeToUndoOrRedo(UnityAction call)
```

Parameters

`call` UnityAction

Class UndoRedoState

Namespace: PAC.UndoRedo

```
public class UndoRedoState
```

Inheritance

object ← UndoRedoState

Inherited Members

object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Constructors

UndoRedoState(UndoRedoAction, Layer, int)

```
public UndoRedoState(UndoRedoAction action, Layer layer, int layerIndex)
```

Parameters

`action` UndoRedoAction

`layer` Layer

`layerIndex` int

UndoRedoState(UndoRedoAction, Layer[], int[])

```
public UndoRedoState(UndoRedoAction action, Layer[] affectedLayers,
int[] affectedLayersIndices)
```

Parameters

`action` UndoRedoAction

`affectedLayers` Layer[]

`affectedLayersIndices` int[]

Properties

action

```
public UndoRedoAction action { get; }
```

Property Value

UndoRedoAction

affectedLayers

```
public Layer[] affectedLayers { get; }
```

Property Value

Layer[]

affectedLayersIndices

```
public int[] affectedLayersIndices { get; }
```

Property Value

int[]