LAYER SCRIPT INTERFACE

View online: https://www.construct.net/en/make-games/manuals/construct-3/scripting/scripting-reference/layout-interfaces/ilayout/ilayer

The `ILayer` script interface represents a layer on a layout.

Examples

See the Input event position example for a demonstration of using the layer `cssPxToLayer()` method, which is useful when handling input events. The Custom Layer Drawing example also demonstrates using code to draw custom content to a layer with the `"afterdraw"` event.

Layer events

The following events can be listened for using the `addEventListener` method.

"beforedraw"

"afterdraw"

Fired when the layer is drawn, allowing custom drawing code to draw on this layer. As Construct uses a back-to-front renderer, content drawn in the `"beforedraw"` event will appear underneath other content on the layer, and content drawn in the `"afterdraw"` event will appear on top of other content on the layer. The event object has the following properties:

- `renderer`: the IRenderer script interface with which you can draw content.
- `layer`: the `ILayer` of the layer being drawn.

Only draw content inside this event. Construct does not expect you to draw anything outside of this event (which includes code after an `await`), and if you attempt to do so anyway, it may not appear correctly.

Layer APIs

runtime

A reference back to the IRuntime interface.

name

A read-only string of the layer name.

index

A read-only number with the zero-based index of the layer on its layout. The bottom layer has an index of 0, with the index increasing upwards in Z order.

addEventListener(eventName, callback)

removeEventListener(eventName, callback)

Add or remove a callback function for an event. See *Layer events* above for the available events.

layout

The ILayout interface representing the layout this layer belongs to.

parentLayer

A reference to the layer's parent `ILayer` if it is a sub-layer, else `null` if it is a top-level layer.

***parentLayers()**

Iterates all the layer's parent layers, moving up towards the top of the hierarchy.

***subLayers()**

Iterates the layer's own sub-layers in increasing Z order. This does not iterate any sub-layers at lower levels in the hierarchy.

***allSubLayers()**

Iterates the layer's sub-layers and further sub-layers beneath those recursively, in increasing Z order.

isInteractive

A boolean indicating if the layer is interactive, allowing its content to respond to mouse and touch input.

> *Note that this returns the layer's own interactive state. If it has a non-interactive parent layer, this property can be `true` but the layer will still not be interactive. Use `isSelfAndParentsInteractive` to check if the layer and all its parents are interactive.*

isSelfAndParentsInteractive

A read-only boolean indicating if both this layer and all its parent layers are set to be interactive. If this is true the layer content will respond to mouse and touch input.

isVisible

A boolean indicating if the layer is visible. When invisible, the layer skips drawing entirely.

> *Note that this returns the layer's own visibility state. If it has an invisible parent layer, this property can be `true` but the layer will still not be visible. Use `isSelfAndParentsVisible` to check if the layer and all its parents are visible.*

isSelfAndParentsVisible

A read-only boolean indicating if both this layer and all its parent layers are set to visible. If this is true the layer will be drawn.

isTransparent

A boolean indicating if the layer background is transparent. When transparent, the background color is ignored.

backgroundColor

Set or get the background color of a layer as an array with 3 elements specifying the red, green and blue components with values in the 0-1 range. Note this is ignored if the layer is transparent.

isHTMLElementsLayer

A boolean indicating if this layer acts as a HTML layer. For more information see HTML layers.

scrollX

scrollY

scrollTo(x, y)

getScrollPosition()

restoreScrollPosition()

Independently scroll a layer, regardless of where the layout is scrolled to. By default layers all follow the layout scroll position. Upon setting a layer's scroll position, the layer will stop following the layout scroll position, and remain scrolled at the position specified. The `restoreScrollPosition()` method reverts the layer to the default mode where it follows the layout scroll position. When not independently scrolling a layer, the `scrollX` and `scrollY` getters return the layout scroll position.

parallaxX

parallaxY

-

Set or get the horizontal and vertical parallax rates of a layer.

opacity

The opacity of the layer, as a floating point number in the range [0, 1], where 0 is fully transparent and 1 is fully opaque. Note that changing the opacity to a value other than 1 will force the layer to render via its own texture.

scale

Set or get the layer scale, taking in to account its scale rate property.

scaleRate

Set or get the *scale rate* property of a layer, which affects how quickly it scales (if at all).

angle

Set or get the layer angle in radians.

zElevation

Set or get the Z elevation of the entire layer. By default the camera is at Z = 100, and looking down to Z = 0. The default Z elevation is 0. Increasing it will move the layer upwards (towards the camera) and decreasing it will move it downwards (away from the camera).

renderingMode

Set or get the *Rendering mode* layer property as a string of either `"2d"` or `"3d"`. This allows dynamically changing a layer between rendering modes at runtime.

getViewport()

Return a DOMRect representing the bounds of the viewport on this layer in layout co-ordinates.

isForceOwnTexture

A boolean indicating the layer's *Force own texture* property. For more information see the property in the Layers manual entry.

blendMode

A string indicating the blend mode of the layer, controlling how it draws over the other layers behind it. The allowed strings are the same as accepted by the IRenderer method `setBlendMode()`.

effects

An array of IEffectInstance representing the effect parameters of the effects on this layer.

cssPxToLayer(clientX, clientY, z = 0)

layerToCssPx(layerX, layerY, z = 0)

Convert between positions in CSS pixels, such as the `clientX/Y` properties of an input event, and layer co-ordinates within the project. An optional Z value can be provided to do the conversion taking in to account Z elevation to a certain height on the layer. This is useful for purposes like identifying what position in a layer was clicked in an input event, or positioning a HTML element in layer co-ordinates. Both methods return a pair of co-ordinates in the form `[x, y]`.

drawSurfaceToLayer(dsX, dsY, z = 0)

layerToDrawSurface(layerX, layerY, z = 0)

Convert between positions in layer co-ordinates and the draw surface in units of device pixels. An optional Z value can be provided to do the conversion taking in to account Z elevation to a certain height on the layer. Both methods return a pair of co-ordinates in the form `[x, y]`. These methods are intended for the Addon SDK and are for identifying the draw surface co-ordinates that correspond to an object.

renderScale

A read-only number with the render scale factor for this layer. This is intended for the Addon SDK where plugins need this value to render taking in to account Construct's scale factor.