IANIMATIONFRAME SCRIPT INTERFACE

View online: https://www.construct.net/en/make-games/manuals/construct-3/scripting/scripting-reference/object-interfaces/ianimationframe

The `IAnimationFrame` interface represents a single animation frame within an IAnimation. It derives from the IImageInfo script interface.

> *Note that when accessing the origin, image points, or collision polygon points on `IAnimationFrame`, the positions are returned in normalized form, in the range 0-1, representing the position set in the editor. For example (0, 0) is the top-left point, (1, 1) is the bottom-right, and (0.5, 0.5) is the middle. This differs from the ISpriteInstance methods which return positions in layout co-ordinates based on the current location and orientation of the Sprite instance it is called on.*

IAnimationFrame APIs

duration

A read-only number with the relative duration of this animation frame, i.e. 1 for standard speed, 2 for twice as long, etc.

tag

A string of the tag assigned for this frame in the Animation Editor.

originX

originY

getOrigin()

Read-only numbers with the normalized position of the origin within this animation frame, ranging from 0-1. The method returns both values at the same time.

getImagePointCount()

Return the number of image points on the animation frame.

getImagePointX(nameOrIndex)

getImagePointY(nameOrIndex)

getImagePoint(nameOrIndex)

Return the location of an image point on the animation frame in normalized co-ordinates, i.e. ranging from 0-1. Image points are identified either by a case-insensitive string of their name, or their index. If the image point is not found, this returns the origin instead. The `getImagePoint` variant returns `[x, y]`.

getPolyPointCount()

Return the number of collision polygon points on the animation frame.

getPolyPointX(index)

getPolyPointY(index)

getPolyPoint(index)

Return the location of a collision polygon point on the animation frame in normalized co-ordinates relative to the origin, by its zero-based index. The `getPolyPoint` variant returns `[x, y]`.

> *Note that the returned positions are both normalized and relative to the origin. For example when the origin is at (0.5, 0.5), a collision poly point in the top-left corner has the co-ordinates (-0.5, -0.5).*