TWEEN BEHAVIOR SCRIPT INTERFACE

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

The `ITweenBehaviorInstance` interface derives from IBehaviorInstance to add APIs specific to the Tween behavior.

An actively running tween is represented by ITweenState, which also derives from ITimelineState. These interfaces can be used to control playback, including identifying when tweens end via the `finished` promise.

Examples

See the Scripting tweens example for a demonstration of using tweens from JavaScript code.

Basic tween usage

A code sample is shown below of starting a tween and waiting for it to finish.

```
async function doTween(runtime)
{
	// Get a Sprite instance with the Tween behavior
	const inst = runtime.objects.Sprite.getFirstInstance();

	// Create a tween that moves it to (300, 300) over 2 seconds
	const tween = inst.behaviors.Tween.startTween("position", [300, 300], 2, "in-out-sine");

	// Wait for the tween to finish
	await tween.finished;

	// Log to the console now the tween has finished
	console.log("Tween finished");
}
```

More examples

Some examples of valid calls to `startTween` are shown below (assuming `Tween` represents this behavior).

```
// Tween X position to 300 over 2 seconds linearly
Tween.startTween("x", 300, 2, "linear");
```

```
// Tween position to (300, 300) over 2 seconds with ease "in-out-sine"
Tween.startTween("position", [300, 300], 2, "in-out-sine");

// Looping ping-pong tween to size 200x200 every 0.5 seconds
Tween.startTween("size", [200, 200], 0.5, "out-sine", {
        loop: true,
        pingPong: true
});

// Tween color to blue over 1.5 seconds linearly
Tween.startTween("color", [0, 0, 1], 1.5, "linear");

// Value tween from 100 to 200 linearly over 3 seconds
const t = Tween.startTween("value", 200, 3, "linear", {
        startValue: 100
});
// (then read t.value over time)
```

Tween properties

When using the `startTween` method, the `prop` parameter must be one of the strings given in the table below. Each property also lists how many values are expected for the `endValue` parameter; if more than 1, they should be passed as an array.

Property	Number of values
`"x"`	1
`"y"`	1
`"position"`	2
`"width"`	1
`"height"`	1
`"x-scale"`	1
`"y-scale"`	1
`"size"`	2
`"scale"`	2
`"angle"`	1 *(in radians)*
`"opacity"`	1 *(in 0-1 range)*
`"color"`	3 *(RGB values in 0-1 range)*
`"z-elevation"`	1
`"value"`	1

Ease names

When using the `startTween` method, the `ease` parameter must be one of the strings given in the table below, or the name of a custom ease in the project.

`"linear"`		
`"in-sine"`	`"out-sine"`	`"in-out-sine"`
`"in-elastic"`	`"out-elastic"`	`"in-out-elastic"`
`"in-back"`	`"out-back"`	`"in-out-back"`

`"in-bounce"`	`"out-bounce"`	`"in-out-bounce"`
`"in-cubic"`	`"out-cubic"`	`"in-out-cubic"`
`"in-quadratic"`	`"out-quadratic"`	`"in-out-quadratic"`
`"in-quartic"`	`"out-quartic"`	`"in-out-quartic"`
`"in-quintic"`	`"out-quintic"`	`"in-out-quintic"`
`"in-circular"`	`"out-circular"`	`"in-out-circular"`
`"in-exponential"`	`"out-exponential"`	`"in-out-exponential"`

Tween behavior APIs

startTween(prop, endValue, time, ease, opts)

Start a tween running for a property to a given end value, over a `time` given in seconds, with an ease function specified by `ease`. Returns an ITweenState representing the running tween.

- `prop` must be a string of one of the property names given in the table in the section *Tween properties* above.
- `endValue` must be either a number, or an array of numbers, depending on `prop`. In the table of properties above, where the *Number of values* is 1, this must be a number; where it is greater than 1, it must be an array with that many values.
- `time` is the duration the tween will run for in seconds.
- `ease` is a string of the name of one of the built-in eases in the section *Ease names* above, or the name of a custom ease in the project.

The `opts` parameter is optional for providing further parameters via object properties. The following properties can be used:

- `tags`: a list of tags to assign to the tween, specified either as a space-separated string, or an array of strings
- `destroyOnComplete`: a boolean indicating whether to automatically destroy the instance once the tween completes (default false)
- `loop`: a boolean indicating whether to repeat the tween when it reaches the end (default false)
- `repeatCount`: the number of times to repeat the tween (default 1).
- `pingPong`: a boolean indicating whether to alternate the playback direction when repeating (default false)

- `startValue` : for value tweens only, specifies the start value (default 0).

See above for some code examples demonstrating some of the ways this method can be called.

***allTweens()**

Iterates all actively running tweens created by the behavior, represented with ITweenState.

***tweensByTags(tags)**

Iterates all actively running tweens matching the given set of tags, represented with ITweenState. The tags may be specified as either a space-separated string, or an array of strings.

---

isEnabled

A boolean indicating if the behavior is enabled. If disabled, the behavior no longer has any effect on the object.