CONFIGURING BEHAVIORS

View online: https://www.construct.net/en/make-games/manuals/addon-sdk/guide/configuring-behaviors

The main configuration for a behavior is set in **behavior.js**.

Behavior constants

The following constants are defined in the file-level scope:

```
const BEHAVIOR_ID = "MyCompany_MyAddon";
const BEHAVIOR_CATEGORY = "general";
```

The ID and version constants must match the values specified in addon.json.

BEHAVIOR_ID

This is a unique ID that identifies your behavior from all other addons. This must not be used by any other addon ever published for Construct 3. **It must never change** after you first publish your addon. (The name is the only visible identifier of the addon in the Construct 3 editor, so that can be changed any time, but the ID must always be the same.) To ensure it is unique, it is recommended to use a vendor-specific prefix, e.g. `MyCompany_MyAddon`.

BEHAVIOR_CATEGORY

The category for the behavior when displaying it in the *Add behavior* dialog. This must be one of `"attributes"`, `"general"`, `"movements"`, `"other"`.

Updating behavior identifiers

The main class declaration of the behavior looks like this:

```
const BEHAVIOR_CLASS = SDK.Behaviors.MyCompany_MyAddon = class MyCustomBehavior extends SDK.IBeh
```

Be sure to update the identifiers to describe your own behavior, in both the SDK namespace and the class name.

Updating in type.js and instance.js

Likewise in both type.js and instance.js, you must update the following:

- `BEHAVIOR_CLASS` to refer to your behavior's name
- The `class` name suffixed with `Type` or `Instance`. (For example the Bullet behavior uses `BulletBehavior`, `BulletType` and `BulletInstance` as the three names.)

Optional behavior scripts

With the addon SDK, you may omit the editor script files type.js and instance.js, as well as the runtime script files plugin.js and type.js. If these files are omitted, it uses the default base class with no modifications. To remove these files, be sure to do the following:

1. Delete any unused script files
2. Remove the files from the `"file-list"` array in addon.json
3. Remove any unused editor script files from the `"editor-scripts"` array in addon.json
4. In the editor behavior script, call `this._info.SetC3RuntimeScripts()` with an array of the runtime script files in use, as the default list includes c3runtime/behavior.js and c3runtime/type.js.

The behavior constructor

The main function of behavior.js is to define a class representing your behavior. In the class constructor, the configuration for the behavior is set via the `this._info` member, which is an IBehaviorInfo interface. The constructor also reads potentially translated strings from the language subsystem.

For more information about the possible behavior configurations, see the `IBehaviorInfo` reference.

Specifying behavior properties

The behavior properties appear in the Properties Bar when instances using the behavior are selected. To set which properties appear, pass an array of PluginProperty to `this._info.SetProperties`. An example is shown below. For more details see the `PluginProperty` reference. (Note that behaviors use the same property class as plugins, hence re-using the PluginProperty class for behaviors.)

```
        this._info.SetProperties([
                new SDK.PluginProperty("integer", "test-property", 0)
        ]);
```