

ADDON METADATA

View online: <https://www.construct.net/en/make-games/manuals/addon-sdk/guide/addon-metadata>

The metadata for your addon, specifying details like its ID and type, is defined by **addon.json**. An example is shown below.

```
{
  "is-c3-addon": true,
  "sdk-version": 2,
  "type": "plugin",
  "name": "My custom plugin",
  "id": "MyCompany_MyAddon",
  "version": "1.0.0.0",
  "author": "Scirra",
  "website": "https://www.construct.net",
  "documentation": "https://www.construct.net",
  "description": "Example custom Construct 3 plugin.",
  "editor-scripts": [
    "plugin.js",
    "type.js",
    "instance.js"
  ],
  "file-list": [
    "c3runtime/plugin.js",
    "c3runtime/type.js",
    "c3runtime/instance.js",
    "c3runtime/conditions.js",
    "c3runtime/actions.js",
    "c3runtime/expressions.js",
    "lang/en-US.json",
    "aces.json",
    "addon.json",
    "icon.svg",
    "instance.js",
    "plugin.js",
    "type.js"
  ]
}
```

Note some of the information is duplicated elsewhere in the addon's files. This is because the editor reads this file before it loads any other files when asking the user if they want to install the

addon. Note information specified here, such as the ID, must exactly match everywhere else it is used.

The addon SDK provides a JSON schema to help you write addon.json files, as it provides autocomplete and validation in compatible editors.

Each field and its possible values are described below.

is-c3-addon

Boolean set to `true`. This is used by Construct 3 to identify valid addons.

sdk-version

The addon SDK version the addon has been built with. Currently only SDK version 2 is supported. Effects and themes are not affected by the addon SDK version and can omit this field.

type

One of `"plugin"`, `"behavior"`, `"effect"` or `"theme"`, indicating the kind of addon this is.

name

The displayed name of the addon, in English.

id

The unique ID of the addon. This is not displayed and is only used internally. This must not be used by any other addon ever published for Construct 3, and 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`. It must match the ID set in plugin.js.

version

A string specifying the addon version in four parts (major, minor, patch, revision). Be sure to update this when releasing updates to your addon. It must match the version set in plugin.js/behavior.js.

author

A string identifying the author of the addon.

website

A string of a URL to the author's website. It is recommended to provide updates to the addon at this URL if any become available. The website should use HTTPS.

documentation

A string of a URL to the online documentation for the addon. It is important to provide documentation for your addon to be useful to users.

description

A string of a brief description of what the addon does, displayed when prompting the user to install the addon.

min-construct-version

The minimum Construct version required to load your addon, e.g. "r399". If not specified, the addon will be allowed to be installed with any version of Construct. If specified and the user attempts to install the addon with a version lower than the minimum, then Construct will prevent installation and show a message indicating that a newer version of Construct must be used. If the user installs the addon with a newer version of Construct and then rolls back to an older version of Construct lower than the minimum, then Construct will refuse to load the addon (a message will be logged to the console) and the editor will act as if the addon is not installed.

supports-worker-mode

A boolean indicating whether the addon supports Construct's worker mode, where the entire runtime is hosted in a [Web Worker](#) instead of the main thread. This defaults to `true`. Providing the addon only uses APIs available in a Web Worker, then it is compatible; where access to the DOM is necessary, then a DOM script can be used to still access those features in worker mode - see [Runtime scripts](#) for more details. Therefore it should be possible for every addon to support worker mode, and supporting it is strongly recommended as worker mode can bring performance benefits. This can be set to `false` to indicate that the addon does not yet support worker mode, which may be useful to expedite addon development or if the addon makes use of extremely complex DOM operations. This will cause worker mode "auto" to switch to DOM mode which may degrade the performance of the project. If the user attempts to switch worker mode to "Yes" in project using the addon, then Construct will show an error message highlighting the addon that does not support the mode, and prevent changing the setting.

editor-scripts

For plugins and behaviors only. An array of script files in the addon package to load in the editor. It is recommended to leave this at the default unless you have large editor dependency scripts, or if you want to minify your addon in to a single script. Note themes do not use editor scripts.

stylesheets

For themes only. An array of CSS files in the addon package to apply to the document. These are the CSS files that define the theme's appearance.

file-list

For developer mode addons only. A list of all files used by the addon. This is required for Developer Mode addons since there is no other mechanism for Construct to determine the list of files when serving files from a web server. Be sure to update this property if you add, rename or remove any files in your addon.

Additional properties for effects

When developing an effect addon, additional information about the effect is included in the addon.json file. For more information see [Configuring effects](#).