BROWSER

View online: https://www.construct.net/en/make-games/manuals/construct-3/plugin-reference/browser

The **Browser** object accesses features of the browser or platform running the project. It also provides features like switching to and from fullscreen mode, detecting if an update is available, and determining if the page is visible.

Security limitations sometimes prevent browser actions. For example to prevent abuse, the window *Close* action can only be used when the window was opened automatically by a web page and not when it was opened by the user.

Scripting

This object has no script interface, because when using JavaScript or TypeScript coding you can use the browser built-in APIs to access the same features.

Browser conditions

Cookies enabled

True if the user has cookies enabled in their browser.

Is online

True if the browser thinks it currently has an active connection to the Internet. Construct projects can work offline - see Offline games for more information.

> *The "is online" detection of the Browser plugin is more or less a guess. It is actually difficult to categorically determine if an Internet connection is available. For example the user may have intermittent signal, and online features sometimes work and sometimes fail unpredictably. In this case the "Is online" detection is probably wrong, as it can't actually tell whether a connection will work. Rather than using this condition to check if online features will work, usually it's better to just go ahead and use online features anyway as if the user is online, and handle any errors that occur as a result.*

On went online

On went offline

Triggered when the browser thinks the connection to the Internet has become available or unavailable during the running of the project. This is common on mobile devices which may be moving in and out of signal areas. The *Is online* condition also changes to reflect the connection status.

See the caveat about online detection above.

Is portrait/landscape

Determine if the current display is portrait (height is greater than width) or landscape (width is greater than height). This is performed by making a simple check on the window size of the browser, so also returns accordingly on a desktop browser depending on its dimensions.

On back button

Triggered when the user presses the device's 'Back' button. Note not all devices have this button (e.g. iOS devices only have a 'Home' button) and not all platforms support this trigger.

On hash change

Triggered when the part of the URL after the hash character (e.g. `example.com/index.html#hashpart`) changes. The *Hash* expression will return the new value.

On offline ready

Triggers the first time the project runs when it has finished downloading and is ready to use offline.

On update found

Triggers when an update is detected. The update will download in the background and trigger *On update ready* when it is ready to be used.

On update ready

Triggered when an updated version has finished downloading in the background. If the user is still on the project's menu or title screen, you may wish to prompt them to refresh the page (or just do it automatically) so the new version is loaded. See Offline games for more information.

Compare display mode

Test the current display mode of the project. This corresponds to the CSS `display-mode` media query. Normal display in a web browser usually counts as the *browser* display mode. However if a web app install has been performed and the project is being displayed in its web app form without the usual browser interface elements like the address bar, then usually that counts as the *standalone* display mode.

On install available

Is install available

Even in browsers that support web app installs, the option to install may not be immediately available. *On install available* triggers when a web app install first becomes available, and after that *Is install available* remains true, indicating that the *Request install* action may be used.

On install result

Triggered after the *Request install* action has been used to display an install prompt to the user. The possible install results are:

- Accepted: the user chose to install the web app.
- Dismissed: the user cancelled or otherwise declined to install the web app.
- Error: something went wrong with the install prompt and it was not able to complete successfully.
- Any: run the trigger regardless of the result. The *InstallResult* expression can be used to retrieve a string representing the result.

On app installed

Triggered after the *Request install* action has been used to display an install prompt to the user, and the user chose to accept installation, and then the web app installation completed successfully. Usually this also corresponds to the browser changing the display mode of the project to its app form (typically *standalone* display mode).

Has focus

On focus

On blur

Has focus is true if the window currently has focus, which means keyboard input will be received by the window. *On focus* is triggered when the window becomes focused, and *On blur* is triggered when the window loses focus.

> *In some cases focus refers to a frame instead of an actual window - for example inside an iframe, focus means that the user is interacting with the iframe rather than the parent frame.*

Is fullscreen

True if the browser is running in fullscreen mode after using the *Request fullscreen* action.

This condition does not detect if the user pressed F11 or used another browser-provided shortcut to enter fullscreen mode - it only checks if it's fullscreen due to using Request fullscreen.

On fullscreen error

Triggered when a request to enter fullscreen mode was refused by the browser. This may be because the request was made outside of a user input trigger (e.g. mouse click or touch), or when running inside an iframe which blocks fullscreen permission.

On resized

Triggered when the browser window displaying the project is resized. This includes when changing orientation on a mobile device.

Browser actions

Start group

End group

Start or end a group in the browser console. Groups appear indented, and the browser may give the option to expand/collapse the group easily. Groups can optionally be named. To create a group, use *Start group*, then a series of *Log* actions, then the *End group* action.

Log

Log a message, warning or error to the browser console. This can be useful for debugging, testing and diagnostics.

Maximize window

Minimize window

Restore window

Maximize or minimize the main app window, or restore it to its original size and position if it was previously maximized or minimized. These actions only work in desktop exports (Windows, macOS and Linux) as browsers do not allow controlling the window in this way, and mobile devices generally do not provide windowing capabilities.

Vibrate

Vibrate the device with a given pattern, if the device/platform supports vibration. The pattern is given as a comma-separated list of times in milliseconds, alternating between vibrate time and waiting. For example the string "200,100,200" specifies a 200ms vibration, 100ms pause, then another 200ms vibration. This allows a single action to specify a whole vibrate pattern.

Load stylesheet

Load a Cascading Style Sheet (CSS) from a URL and applies its styles to the document. This can be useful for styling form controls and other DOM elements. The URL can also be the name of a project file, such as "myfile.css", to load a CSS file included in the project.

CSS can only style DOM elements such as form controls. Note that most objects like Sprite and Text render directly in to the canvas and are not DOM elements, so cannot be styled with CSS.

Set CSS style

Set a CSS style on the style attribute of some HTML elements in the document, based on a CSS property name and a string for its value. Setting the value to an empty string will remove the property from the style attribute. The element to change the style for is set by a CSS selector, e.g. `".myclass"` will mean to update the CSS style of an element with the class *myclass*; if the *Type* is set to *all*, it will update the style of all elements matching the selector.

This is useful for setting document-wide CSS variables that can be used with other HTML features like form controls or HTML Element.

This action is asynchronous because in worker mode the document cannot be accessed directly. When the action finishes, the change has been made to the document.

Get CSS style

Retrieve a string with the computed style value for a CSS property on an element given by a CSS selector. When the action finishes, the value is returned by the *CSSStyleValue* expression. A tag is used to identify different CSS style values.

This is useful for retrieving the value of CSS variables from stylesheets in the project.

The action is asynchronous because in worker mode the document cannot be accessed directly. When the action finishes, the result is available via the CSSStyleValue expression, passing the same tag.

Go back

Go forward

Move through the browser navigation history as if clicking the Back and Forward buttons on the browser.

Go to URL

Navigate to a given URL. Note this uses the same window/tab as is showing the project, so this action will end the project. The *Target* can be used to select which frame to redirect, which is only useful if the project is displayed within a frame (e.g. an iframe embed), and the frame has permission to redirect the parent frame (i.e. it is not sandboxed). Possible targets are:

- **Self:** redirect only the frame that is currently showing the project.
- **Parent:** redirect the parent frame.
- **Top:** redirect the top level frame (only different to the parent if more than one frame is used)

Invoke download

Invoke a URL as a file download in the browser. Even if this points to a web page or document, it will be downloaded as a file in the browser interface. The URL can point to any address on the Internet, or it can be the name of any imported project file, or it can be a data URL (useful for downloading canvas snapshots). The filename parameter allows you to choose the filename the browser gives to the download, which can be different to the name of the resource being downloaded.

> *Downloading is a browser feature and depends on the browser UI. Note that mobile apps don't run in browsers (there is no address bar etc), so the download feature isn't available there. Consider using the Share plugin to share the file instead.*

Invoke download of string

As with *Invoke download*, but instead of providing a URL to download, a string of the actual data to download as a file is used. A data URI combining the MIME type and data is created, then passed to the browser to download. This is convenient for downloading strings in JSON format as files, e.g. object data from the *AsJSON* expression.

Open URL in new window

Navigate to a given URL in a new window (or tab if the browser settings override). This continues to run the project in the old window or tab.

Reload

Force the page to refresh. This effectively restarts the project.

Set hash

Set the part of the URL after the hash character (e.g. `example.com/index.html#hashpart`). Note this does not reload the page, so the hash part of the URL can be used as a kind of mode or identifier that can be changed while the project is running.

Set warn on close

Enable a warning upon closing the browser tab or window. By default this is disabled, but when enabled, the user will receive a prompt to confirm closing the browser tab or window. A generic message will be shown, usually with text along the lines of "Leave site? Changes you made may not be saved." This provides the option to cancel closing the tab/window, which may help prevent accidental closure or losing unsaved progress. Note that browsers show this message at their discretion and in some circumstances may decide not to show a prompt for privacy or security reasons.

Request install

Where available, prompt the user to install the current page as a web app. This is normally only available in a web browser which supports Progressive Web App (PWA) installs. This is only allowed when installation is available, indicated by the *On install available* or *Is install available* conditions. Further it normally can only be used in a user input trigger, such as a button click or tap, and may only be used if not already installed. If the install prompt is shown, then *On install result* triggers afterwards with the outcome; if the install prompt was successful and the web app successfully installed, then *On app installed* triggers, and the display mode changes. Note that the style of the install dialog depends on how much information your project provides - see the section on *Installable web apps* in Publishing to the web for more details.

Alert

Bring up a simple 'alert' message box.

Blur

Unfocus the browser window.

Cancel fullscreen

Return to windowed mode if the browser is currently in fullscreen mode.

Close

Close the current window, if the script has permission to do so.

Focus

Request to focus the window. Browsers may disregard this request to prevent abuse. In desktop exports, this will attempt to bring the main app window to the top, but similarly the system may not always allow this in order to prevent abuse.

Lock orientation

Unlock orientation

Lock the display of the project to a portrait or landscape mode only, if the current platform supports this. This only has effect on mobile devices. The project may have to already be displaying in fullscreen (using the *Request fullscreen* action) before the orientation can be locked. Unlocking the orientation restores whatever behavior was set before locking, such as automatically changing orientation depending on the way the device is being held.

Request fullscreen

Request that the browser enter fullscreen mode. Note the browser may refuse this request unless the action is in a user-initiated event, such as a mouse click, key press, touch event or button press. It may also be refused if fullscreen permission is not available, such as if running inside an iframe which does not grant fullscreen permission. If the fullscreen request is ignored, *On fullscreen error* will trigger.

> *In desktop exports (Windows, macOS and Linux), fullscreen requests are always allowed at any time.*

The fullscreen modes that can be entered correspond to the *Fullscreen mode* project property. *Navigation UI* where supported sets whether the browser should show browser elements such as back buttons or the address bar, or hide them (for a true fullscreen experience). Typically this setting only affects mobile browsers.

Set window size

Set window position

Set the size and position of the main window. This is only applicable on desktop-style systems - mobile devices typically use fullscreen apps and therefore windows cannot be repositioned or resized.

> *Note that when running in a web browser, the browser may sometimes refuse to change the window size or position, as a measure to protect the user from unwanted changes. This can also include being unable to alter the main window when the project is running in an iframe.*

Browser expressions

Language

Get the browser's current language setting, e.g. *en-US*.

Platform

Get the current platform the browser reports itself running on, e.g. *Win32* for Windows.

UserAgent

Return the full user agent string for the browser, e.g. *Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.101 Safari/537.36*.

CSSStyleValue(tag)

Get the CSS style value retrieved from a prior call to the *Get CSS style* action. Once the action has finished (using *Wait for previous actions to complete*), pass the same tag as the action was called with to retrieve the result.

Title

The current HTML document's title.

Domain

The current domain, e.g. *construct.net*.

Hash

The string after the hash at the end of the URL, including the hash. For example, if the current URL is `example.com/index.html#hashpart`, this returns *#hashpart*.

PathName

The path relative to the domain in the URL. For example the path name of *https://construct.net/myproject/index.html#teapot* is */myproject/index.html*.

Port

A string of the port specified in the URL, if any. Note while ports are numbers, this expression returns a string, since if no port is specified in the URL it will return an empty string.

Protocol

The current protocol, usually either *http:* or *https:*.

QueryParam

Return a query string parameter by name. For example, if the URL ends with *index.html?foo=bar&baz=ban*, `QueryParam("foo")` returns *bar* and `QueryParam("baz")` returns *ban*.

QueryString

Return the full URL query string including the question mark. For example, if the URL ends with *index.html?foo=bar&baz=ban*, this returns *?foo=bar&baz=ban*.

Referrer

Get the previous page that linked to this page, if any.

URL

Get the complete current URL in the browser address bar, including the protocol.

DisplayMode

A string indicating the current browser display mode - see the *Compare display mode* condition for more details. This expression returns one of the strings "browser", "standalone", "minimal-ui" or "fullscreen".

InstallResult

In the *On install result* trigger, a string representing the install result. This may be one of the strings "accepted", "declined", "unavailable", "failed", or an empty string ("") if no install yet attempted.