# AJAX

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

The **AJAX plugin** allows you to fetch the content of a URL, or post data to a website. You can also use it to load project files. Its name derives from "Asynchronous JavaScript and XML", a technique familiar to most web developers.

### Scripting

This object has no script interface, because when using JavaScript or TypeScript coding you can use the browser built-in Fetch API to make network requests.

## How to make a request

The basic usage of the AJAX object consists of:

1. Use the *Request* action to load a URL.
2. A moment later after the request completes, *On completed* triggers.
3. The *LastData* expression can be used to access the content of the response.

The *tokenat* system expression may be useful to split simple responses. Alternatively, you can read *LastData* in other formats by using other plugins, such as the XML object, loading Array data, and so on.

## Tags

A different tag can be provided for each request. This is a simple string you set to tell apart different requests. For example, on startup you may request both *foo.json* with tag *"foo"* and *bar.json* with tag *"bar"*. When the first request completes, *On "foo" completed* triggers; when the second request completes, *On "bar" completed* triggers. Requests can complete in a different order to the order they were made, so without tags it would be impossible to tell which request was completing.

## Making AJAX requests cross-domain or in preview

By default, browsers **block AJAX requests across domains**. This means, for example, a game on *construct.net* can request other pages on *construct.net*, but cannot request pages on *facebook.com*. This is an important security feature of web browsers (it is not specific to Construct or its AJAX object).

Also, when previewing in Construct the game runs on its own domain at *preview.construct.net*. Therefore AJAX requests to any other domain will typically fail during preview, unless the server explicitly allows cross-domain requests.

If you want AJAX requests to your server to work from any domain, or in preview, you can configure the server to send the following HTTP header:

```
Access-Control-Allow-Origin: *
```

This will enable AJAX requests from any domain, but you should still be aware of the possible security implications of this. You may need to ensure this is set for all HTTP methods used, including GET, POST, and also OPTIONS since cross-domain requests sometimes use that method for "preflighted" requests. For more information on cross-domain requests see HTTP access control (CORS) on MDN.

### Use HTTPS

Since *preview.construct.net* runs on a secure server (HTTPS), you cannot make AJAX requests in preview to insecure servers (HTTP). Browsers block this for security reasons. You may see warnings related to "mixed content", which refers to this problem.

Therefore for cross-domain AJAX requests to work in preview mode, you must also make sure your server is secure (using HTTPS). On the modern web this is best practice anyway, especially since many other features only work on secure servers.

## In NW.js

When exporting desktop applications with NW.js, the AJAX object can also load files from the application folder. Simply use the *Request URL* action and enter the name of a file in the same directory as the application, e.g. `"example.txt"`. Note if a project file exists with the same name, this will always load the project file instead.

## MIME types

AJAX requests for files on your own server requires that your server has the correct MIME types set up.

## Binary data

The AJAX object can receive resources as binary, and also post binary data, using the Binary Data object. This is also useful to fetch local resources like canvas snapshot URLs or video recording URLs, and load them in to a Binary Data object to do something else with them, like save it to storage or upload it to a server.

To request a resource that is received as a binary, use it as follows:

1. Use the *Set response binary* action to specify a Binary Data object to receive the next request's response.
2. Use the *Request* action to load a URL.
3. After the request completes and the response has finished downloading, *On completed* triggers.

4 Now the chosen Binary Data object has automatically been filled with the response data. (Note in this case the *LastData* expression is not used.)

## AJAX conditions

### On completed

Triggered when a request with the same tag has completed successfully. The *LastData* expression contains the response, unless the *Set response binary* action was used, in which case the selected Binary Data object now contains the response.

### On any completed

Triggered when any request has completed successfully. The *Tag* expression identifies the request, and *LastData* contains the response.

### On error

Triggered when a request with the same tag has failed. This can be for a number of reasons, such as the server being down or the request timing out. (The *LastData* expression is not set since there is no response.)

### On any error

Triggered when any request has failed. The *Tag* expression identifies the request.

### On progress

For long running downloads, *On progress* triggers periodically and updates the *Progress* expression with the state of the request. This is useful for making progress bars for things like large file downloads.

### On upload progress

For long running uploads, *On upload progress* triggers periodically and updates the *Progress* expression with the state of the request. This is useful for making progress bars for things like large POST data uploads.

## AJAX actions

### Override MIME type

In some cases you may wish to interpret the server's response with a different MIME type to the one the server indicates. For example a misconfigured server may return a text file with the wrong character set, and you want to force the response to be interpreted as UTF-8. In this case you could override the MIME type as `text/plain; charset=utf-8` to avoid garbling the text. This action only applies to the next AJAX request that is made, after which the

MIME type will be set back to the default setting of accepting what the server response indicates.

### Post to URL

### Post binary to URL

Send a request with data to a URL and retrieve the response. A tag is provided to match it up with the *On completed*, *On progress* and *On error* triggers. The *binary* variant can post the contents of a Binary Data object to the server; otherwise a string is used. Construct does not automatically URL encode the string - use the *URLEncode* system expression to ensure the data is in the correct format for posting. Note string data is in the same format as a query string, e.g. `"foo=1&bar=2"`. The method can also be specified: by default it is POST, but for some APIs you may need to change this to PUT, DELETE or another HTTP method.

### Request URL

Send a GET request to retrieve the contents of a URL. A tag is provided to match it up with the *On completed*, *On progress* and *On error* triggers.

### Request project file

Request the contents of a project file. A tag is provided to match it up with the *On completed*, *On progress* and *On error* triggers.

### Set request header

Set a HTTP header on the next AJAX request that is made. After the next AJAX request all the headers set with this action are cleared again, so it only takes effect once.

### Set timeout

Set the amount of time a request has to complete in seconds; if the timeout expires without the request completing successfully, it will instead fail and trigger *On error*. This action only affects subsequent requests, and does not affect any requests that have already started. If the timeout is set to -1 it restores the default browser timeout.

### Set with credentials

Set the *with credentials* setting for the next AJAX request that is made. After the next AJAX request the setting will revert to its default (off), so it only takes effect once. When enabled, sending a request with credentials will cause cross-site requests to be made using credentials such as cookies and authorization headers. Internally this sets the `withCredentials` property of XMLHttpRequest. More details can be found at the MDN withCredentials documentation.

### Set response binary

Use this action before a *Request* action to read the response in to a Binary Data object instead of returning it as a string in the *LastData* expression. This allows for non-text resources like images to be fetched and processed directly.

## AJAX expressions

### LastData

The contents of the last response. This is set in the *On completed* trigger. Note if *Set response binary* was used, the response is in the chosen Binary Data object instead, and this expression will return an empty string.

### LastStatusCode

The HTTP status code of the last response, e.g. 200 for OK or 404 for Not Found. This is set in the *On completed* trigger.

> *You can find a complete list of possible status codes and what they mean on the MDN page HTTP response status codes.*

### Progress

Return the progress of the AJAX request in an *On progress* event. The progress is represented as a number from 0 to 1, e.g. 0.5 for half completed.

### Tag

The tag of the AJAX request in a trigger. This is useful to identify requests in *On any completed* or *On any error*.