

JavaScript API

- [Overview](#)
- [Display](#)
 - [Report](#)
 - [Dashboard](#)
- [Example Usage](#)
 - [Embedding a Report](#)
 - [Embedding a Dashboard](#)
- [User Access](#)
- [Limitations](#)

Overview

[top](#)

Using the JavaScript API, Yellowfin Reports and Dashboards can be embedded into external webpages, which can be hosted on a separate web server to Yellowfin. The only requirement is that the end user's browser be able to access the Yellowfin server. This is achieved by placing a scriptlet into the HTML code. The following example shows what a JavaScript scriptlet looks like:

```
<script src="http://reporting.example.com/JsAPI?reportUUID=e5e5aaf3-c3b8-4f9b-8280-e21e4d848e63&width=500&height=500" type="text/javascript"></script>
```

The example scriptlet above will render a report within the enclosing page. If the user does not have a current session on the instance that the embedded report is coming from, the user will be prompted to log in unless the [Guest User login](#) is enabled and the displayed content is not secured. For more details, see the [User Access](#) section below.

If the embedded report is being used on a dynamic page or portal, a [SSO](#) can be used to automatically log the user in, using Yellowfin's LOGINUSER web service. The token is appended to the URL (how to use the token with JavaScript API link).

The widget that is embedded cannot be greatly styled or modified. A compromise needs to be made between ease of integration with the JSAPI with limited customizability or complex integration using the Report Service API with unrestricted customizability. Refer to the [Limitations](#) section below for more details.

Some third-party application implementations use a combination of Administration Service web services to return the available reports/dashboards for a user and render these for the user using the JavaScript API.

The HTML page can include multiple reports, however each separate report may only be displayed once on each page. **Note:** multiple dashboards on a single page are not currently supported.

Interactive elements such as Drill Through, Drill Down, User Prompt Filters, Series Selection, and Export to PDF are available.

Display

[top](#)

Report

The embedded report contains a Title Bar and the Report itself. The Title Bar contains the name of the report, as well as buttons to control any interactive elements used on the report. The interactive features include:

	Description
	Displays basic Report Information including; Name, Description, Last Modified Date, Category, Sub Category, and Current User
	Displays the JavaScript used to embed the report
	Allows the user to toggle to Chart view if Table and Chart display is used

	Allows the user to toggle to Table view if Table and Chart display is used
	Opens a panel of User Prompt Filters enabled on the report
	Opens a Series Selection panel if the Visible Series Selection option is enabled
	Opens a Unit Selection panel if the Visible Units Selection option is enabled
	Opens a Page Navigation panel if the report contains multiple pages
	Opens a Section Navigation panel if the report contains sections
	Opens the GIS Map Navigation panel if the map has Hover Navigation enabled
	Opens the Export panel, allowing the user to export the report to various formats, including: •  Export to CSV •  Export to DOC •  Export to PDF •  Export to Rich Text Format •  Export to Text •  Export to XLS

Dashboard

The embedded dashboard contains a Title Bar for the entire dashboard, and then each of the dashboard reports (displayed the same as embedded reports), laid out the same way as they are when viewed through the Yellowfin front end.

Note: dashboard filters and KPI dashboards are supported.

	Description
	Displays basic Dashboard Information including; Name, Description, Last Modified Date, and Current User
	Displays the Javascript used to embed the dashboard
	Opens a panel of User Prompt Filters enabled on the dashboard
	Opens the Export panel, allowing the user to export the dashboard to various formats, including: •  Export to CSV •  Export to DOC •  Export to PDF •  Export to Rich Text Format •  Export to XLS

Example Usage

[top](#)

Embedding a Report

Using the [Share](#) menu, copy the [Embed](#) JavaScript, which identifies the report to load, for use in your HTML page, wiki, or blog. It will take a form similar to this:

```
<script src="http://reporting.example.com/JsAPI?reportUUID=e5e5aaf3-c3b8-4f9b-8280-e21e4d848e63&width=500&height=500" type="text/javascript"></script>
```

See the [Basic Use](#) for more information.

Embedding a Dashboard

Using the [Distribute](#) menu, copy the [Embed](#) Javascript, which identifies the dashboard to load, for use in your HTML page, wiki, or blog. It will take a form similar to this:

```
<script type="text/javascript" src="http://reporting.example.com/JsAPI?dashUUID=3b0b6c9a-9dfb-41f0-b85a-eb17bb8aeeb9"></script>
```

See the [Basic Use](#) for more information.

User Access

top

Users are authenticated when accessing a report or dashboard through the Javascript API. If a user has not yet logged in, they may be able to view the content as a Guest user. (Click [here](#) to learn how to do this.) There are a number of factors used to determine whether a report or dashboard can be run as a Guest user, or whether unauthenticated users are required to log in.

Authentication details can be passed to the API when the dashboard or report is loaded, either by passing a username and password, or [authenticating via web services](#) and passing the login token to the API. This allows you to use a specific user to load content rather than requiring the user to enter their details.

See the [Basic Use](#) for more information.

When a report or dashboard is requested through the Javascript API, the following process is used to authenticate the user:

- if user authentication details are passed to the API when a dashboard or report is loaded, they will be used to log the user in
- if the user is already logged in to the Javascript API on the page, the existing login session is used
- if the user has previously logged in to the Javascript API, they are automatically logged in with the same account (see below)
- if a multicast licence is installed, and a Guest role is available, the user will be logged in as a Guest

When the above process is complete, there are several cases that will not result in the user immediately loading the requested content:

- if none of the conditions above are met, the user cannot be logged in, and they will be prompted to enter their Yellowfin username and password manually.
- if the process above results in the user logging in with the Guest account, but the requested report or dashboard is not available to guest users, they will be prompted to enter their Yellowfin username and password.
- if the process above results in the user being authenticated with a non-guest account, but the account does not have access to the requested content, an appropriate error message will be displayed.

When a user enters their Yellowfin username and password to log in to the Javascript API, a cookie is stored in their browser to remember the user they logged in with. The next time they visit a page that includes a report or dashboard embedded using the Javascript API, they will automatically be logged in again using the same user. This prevents users from needing to enter their details every time they request a report or dashboard. By default, the cookie is valid for seven days from the last time the user logs in. This period can be changed by setting the **External API cookie timeout** value in **Administration > Configuration > System > General Settings**. To disable the cookie altogether, set the value to **0**.

Note: The cookie value stored in the browser is encrypted. The cookie will not work across a restart of Yellowfin (i.e. when Yellowfin is restarted, all users will be required to log in again).

Limitations

top

The current limitations of the Javascript API include:

- Multiple reports may be added to a single HTML page, but each report may only be included once per page
- Only one dashboard is supported per HTML page
- The calendar selector for date filters is not available
- Drill Anywhere is not available
- Saved Filter Sets are not available
- Chart Animation is not available

[top](#)