WebSDK Qual Module

API quick start guide

INFO TRAX

Revised November 5, 2019

Contents

Overview of guide

The WebSDK Qual module API allows you to display a qualification module on your website. Before starting, you should have basic working knowledge of HTML and CSS.

Prerequisites

In order to implement the Qualification module API, you'll need the following:

- Node.js https://nodejs.org/en/download/ installed on your machine (either PC or Mac)
- A command-line http server installed. For more information, see https://www.npmjs.com/package/http-server
- Username and password for the InfoTrax private NPM package repository. (Get this from your InfoTrax Account Manager)
- Username and password for the Qual Module authorization. (Get this from your InfoTrax Account Manager)
- A folder in a directory to store the SDK bundle and HTML files needed for the qualification module. Example:

  `c:\temp\InfoTraxSDK`

Make sure that you have the node.js and the NMP package installed correctly by running **npm -v** at the command prompt. If the command returns a version number, then you have the packages installed correctly.

Setting up the Qualification module

You need to perform the following steps to install and set up the Qualification module.

To set up the Qualification module

1. Install the InfoTrax SDK
2. Create an index.html file
3. Create a callback.html file
4. Test the API

Installing InfoTrax's WebSDK

To install the InfoTrax WebSDK

1. In the directory that you created to store the SDK bundle, type **`npm login`** at the command prompt.
2. Type the username, password, and public email address for the NMP package (that you've gotten from your InfoTrax Account Manager) and press **Return**.
3. Type **`npm install @infotraxsdk/frontend_sdk`** and press **Return**. The SDK installs.

After the SDK is installed, a **`node_modules`** folder with the required packages is created in the working directory.

Creating the index.html file

For your website, you’ll need an index.html file that calls the Qualification module.

With your favorite text editor, create an index.html file. Copy and paste the following into your index.html file:

```
<!DOCTYPE html>

<html lang=”en”>

      <head>

         <link rel="stylesheet"
         href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"/>

         <link
         rel="stylesheet"="https://cdn.jsdelivr.net/npm/semantic-ui@2.4.2/dist/semantic.min.css"/>

         <script crossorigin
         src="https://unpkg.com/react@16/umd/react.development.js"></script>

         <script crossorigin
         src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>

         <script
         src=node_modules/@infotraxsdk/frontend_sdk/dist/bundle.js></script>

      </head>

      <body>

            <div style=”width: 100%; overflow: hidden;display: flex;”>

            <div>

            <Qualifications></Qualifications>

            </div>

            </div>

      </body>

</html>
```

For more information on the various tags used in the index.html file, see the appendix.

Creating a callback page

For the website to authorize your sign in, you need to create a callback file.

With your favorite text editor, create a callback.html file. Copy and paste the following into your callback.html file:

```
<!DOCTYPE html>

<html lang=”en”>

      <head>
```

```
<script src=”./node_modules/@infotraxsdk/infotrax_rest_api/build/bundle.js></script>
</head>
<body>
<div>
<h1>You are logged in successfully!</h1>
<script type=”text/javascript”>
InfotraxApi.getAuth().auth0.popup.callback();

</script>
</div>
</body>
</html>
```

For more information on the various tags used in the callback.html file, see the appendix.

After installing the WebSDK, creating the index.html file, and creating the callback.html file, you should have the node_modules package, callback.html file, index.html file, and the package-lock.json in your directory.

Testing the project

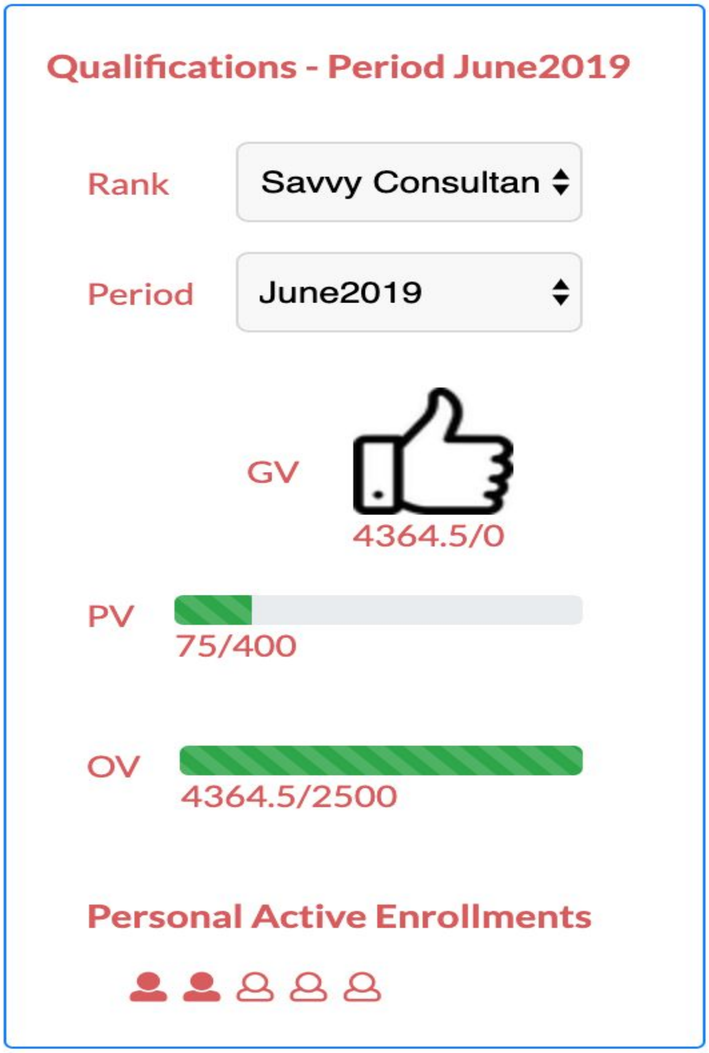
Once you have installed the SDK and created the HTML files, you’ll need to test the Qual module to make sure that it’s returning the right data and displaying correctly. To test the Qual module, you’ll use the http-server package to create a simple web server to host the index.html file on your local machine.

To test the project

1. At the command prompt, type `http-server .`

   Important: Make sure to include the “.” character in the http-server command. The “.” in the command allows the http-server to host all the files that are in the directory.

2. Open a browser and paste `localhost:8080/index.html` into the address bar.
3. Sign into the SDK with your Qual Module authorization credentials provided by your InfoTrax Account Manager.
4. After you sign in, the **Qualifications** module should appear.



Qualifications - Period June2019

Rank: Savvy Consultan

Period: June2019

GV 4364.5/0

PV 75/400

OV 4364.5/2500

Personal Active Enrollments

●●○○○

Implementing the Qual module

After testing the Qual module in the test server, you'll need to incorporate the Qual module into your own website. You must save the index.html and callback.html file that you've created into your website's file structure. This placement is dependent upon how you've organized your website. Make sure that the target web page for the Qual module calls to those html files that you created for the Qual module.

Appendix

About the index.html file tags

Here’s more information about the different tags in the index.html file.

Tag	Definition
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"/> **<link rel="stylesheet"="https://cdn.jsdelivr.net/npm/semantic-ui@2.4.2/dist/semantic.min.css"/>**	Links to the Bootstrap and semantic CSS stylesheets.
<script crossorigin src="https://unpkg.com/react@16/umd/react.development.js"></script> **<script crossorigin src="https://unpkg.com/react-dom@16/umd/react-dom.development.js"></script>**	Retrieves components developed with the React.js library.
<script src=node_modules/@infotraxsdk/frontend_sdk/dist/bundle.js></script>	Links to the web SDK bundle from InfoTrax’s webSDK.
<div style=”width: 100%; overflow: hidden;display: flex;”>	Provides style information for the UI element.
<div> **<Qualifications></Qualifications>** **</div>**	Imports the Qualification module.

About the callback.html tags

Here’s more information about the different tags in the callback.html file.

Tag	Definition
<script src=”./node_modules/@infotraxsdk/infotrax_rest_api/build/bundle.js></script >	Links to the API script.
<script type=”text/javascript”> InfotraxApi.getAuth().auth0.popup.callback(); </script>	Calls to the login authentication script
<h1>You are logged in successfully!</h1>	Message that appears when user successfully is authenticated.