Wave2 Integration Guide

Website and Mobile App WebView Integration

WEBSITE & MOBILE APP WEBVIEW INTEGRATION

OVERVIEW

The Wave2 Locator provides real-time aggregated multi-network branch and ATM search and mapping functionality in a mobile-responsive ADA-accessible user interface that is intended to be integrated seamlessly into the website, mobile web, and mobile app offerings of financial institutions. Integration of the Locator into either a website or a mobile app should be a fairly simple process.

In Section 1 of this Integration Guide we will focus on website integrations. While there are a variety of configurations available, for most website integrations you will need nothing more than the 'Standard Website Integration' which amounts to adding a one-line Javascript tag into the HTML source of your locations page. More complex integrations include the ability to add configuration parameters to the integration tag to provide some control over the Locator directly from the integration tag itself.

In Section 2 of this Guide, we will focus on mobile app integrations. The mobile app WebView integration technique presented here is the preferred approach, as it offers some important advantages over API integration.

1. WEBSITE INTEGRATION

Integration of the Wave2 Locator into your website generally requires one-line script to be added into the HTML source of your locations page at the appropriate place. This script will then build and display your locator in a mobile responsive embedded iframe. In this section, we will detail two ways to integrate the Locator - using the "standard" integration tag, as well as using the more configurable "advanced" integration tag.

1.1 MOBILE RESPONSIVE WEBSITES

When accessing your custom Locator URL in a browser, you should notice that the locator automatically fills the entire browser window, adjusting its GUI features according the available size of the browser. This is a demonstration of the mobile responsive nature of the Locator. It will function in a similar manner when it is loaded inside your mobile-responsive website, filling the "container" object to which it is added.

The container object is usually a DIV with various CSS rules applied by your website software. Your website will govern how that container behaves at different screen sizes and on different device types, and the Locator will respond accordingly to fill the container and make the best use of its available space.

In the ideal scenario, you will provide a mobile-responsive container object that has both its width and its height explicitly controlled by your website CSS rules. Unfortunately, the more common scenario is that your website CSS rules may not explicitly set the height of that container DIV. In that situation, the Locator will not have enough information about the available height to optimally set its own height. Therefore, you will need to use a variation of the standard site insertion script that explicitly sets the height for your Locator. Both scenarios are detailed in section 1.2 below.

1.2 STANDARD WEBSITE INTEGRATION

Once your Locator is ready for integration, you will receive your unique Wave2 Locator URL – something like https://demolocator.wave2.io/. Your unique URL will provide the basis for all the integration tags and instructions in this document, so substitute your own URL wherever you see the demo URL.

CONTAINER OBJECTS WITH AN EXPLICIT HEIGHT VALUE SET BY CSS

If your website CSS rules do explicitly set the mobile responsive width and height of your container object, you can use this most basic site integration tag. Simply edit the HTML source of the page and paste your integration tag directly inside the intended container.

```
<script id="W2L" src="https://demolocator.wave2.io/load.min.js"></script>
```

Note that the SRC URL root will be the same as your unique Locator URL. Note also that the Script has an ID value of “W2L” – it is important that you do not change the script ID value.

CONTAINER OBJECTS WITH NO EXPLICIT HEIGHT VALUE

If you try to use the integration tag given above and your resulting locator looks to be vertically short, or it is not filling the page the way you’d like it to, you can use this alternate tag to explicitly set the height as you see fit.

```
<DIV style="height:800px;">
  <script id="W2L" src="https://demolocator.wave2.io/load.min.js"></script>
</DIV>
```

Note that this is simple the first integration tag with a container DIV that has its own inline CSS that explicitly defines the height of the container to be 800 pixels high. You can accomplish this same thing with a variety of similar CSS techniques depending on your website structure. The important factor is explicitly setting the height of the container. The Wave2 Locator code will then completely fill its container, whatever size it happens to be.

That’s it! Basic website integration is usually as simple as a one-line copy and paste into your HTML source.

However, there are a variety of customizations that can be applied to the integration tags, which we will detail in the next section.

1.3 ADVANCED WEBSITE INTEGRATION

Sometimes you may have special requirements for your website integration. For example, you may need to have your default locator configuration (as set up in the Admin Portal) showing on your main locator page, but you may want to show a different configuration elsewhere in your website using different initial search filter settings and a different initial location. These types of changes can be accomplished by supplying a variety of configuration parameters injected directly into the advanced site integration tag, as detailed below.

BASE ADVANCED INTEGRATION TAG – “DATA LOADER”

The advanced integration tag looks very similar to the standard tag; however, it uses a different ID value (“W2DATA”) and a different Javascript filename. The advanced tag also uses your own unique locator URL in place of “demolocator.wave2.io”.

```
<script id="W2DATA" src="https://demolocator.wave2.io/load_data.min.js"></script>
```

This simplest version of the advanced tag will simply load your locator just like the standard integration does. Next, we can add whatever configuration options and parameters we need, right inside the integration tag itself.

For example, instead of writing out a container DIV to set the height of the locator, you could accomplish this using the advanced integration tag as follows:

```
<script id="W2DATA" data-height="750"
                    src="https://demolocator.wave2.io/load_data.min.js"></script>
```

In this case, the parameter *data-height="750"* tells the integration tag to build a container DIV with height of 750 pixels. Please note that the unit "px" is not supposed to be included in the parameter.

ADVANCED INTEGRATION CONFIGURATION PARAMETERS

Parameter	Purpose / Notes	Example
data-height	To explicitly set the height of the locator in the page. This parameter expects a number value only, with no units.	data-height="750"
data-filter	To override the initially selected search filters for your locator as they are configured in the Admin Portal. This parameter expects a comma-separated list of the filters that should be enabled, using the Search Filter codes specified in the Admin. (e.g. FCS for your own branches, FIATM for your own ATMs, ATMSF for surcharge-free ATMs, ESC for shared branches, etc.)	data-filter="FCS,FIATM"
data-autogeo	To turn on or off the automatic geolocate-on-load option. This parameter expects a value of 1 to enable automatic geolocation on initial load, and a value of 0 to disable that feature.	data-autogeo="1"
data-lat data-lon	To set the initial location of the center of the search and the center of the map view using Latitude and Longitude values. These parameters expect decimal number values.	data-lat="29.35235" data-lon="-77.5326"
data-search	To set a free-form initial search phrase as if it had been typed into the search box in the Locator. This can be a full address, a city, state, a Zipcode, a keyword, or some combination of those.	data-search="1600 Pennsylvania Ave, Washington D.C."
data-address	To set just the street address portion of the initial search location. Intended to be used in coordination with the other individual search inputs detailed below.	data-address="21 Jump Street"
data-city	To set just the city portion of the initial search location. Intended to be used in coordination with the other individual search inputs.	data-city="Chicago"

data-state	To set just the state portion of the initial search location. Intended to be used in coordination with the other individual search inputs.	data-state=”Illinois”
data-zip	To set just the Zipcode portion of the initial search location. Intended to be used in coordination with the other individual search inputs.	data-zip=”60609”
data-country	To set just the country portion of the initial search location. Intended to be used in coordination with the other individual search inputs.	data-country=”USA”

Example of a complete advanced integration tag using a variety of configuration parameters:

```
<script id="W2DATA" data-height="750" data-filter="ATMSF" data-search="Chicago, IL"
src="https://demolocator.wave2.io/load_data.min.js"></script>
```

2. MOBILE APP INTEGRATION

Although Wave2 offers extensive third-party app integration in the form of two different APIs (see separate technical specification documentation on the Locator API and the Legacy API) there is a much simpler mobile app integration approach available that offers the advantages of minimal software development, automatic pass-through of software upgrade, bug fixes, and new features, and a well-designed mobile-responsive and ADA accessible GUI that delivers a consistent user experience to the website locator. This approach is known as a WebView integration.

2.1 WEBVIEW MOBILE APP INTEGRATION

The WebView approach is quite simple – your mobile app software will include what is essentially a full-screen web browser, a WebView component, embedded directly inside the mobile app GUI itself. Into that WebView, it will load your unique locator URL, including a parameter to designate this as a mobile app integration for usage tracking purposes.

WEBVIEW INTEGRATION INSTRUCTIONS

Detailed instruction on the inclusion of a WebView component into your app software is out of scope of this document, however there is a wealth of information online on this topic. Platform-specific instructions can be found at the following links:
https://developer.android.com/guide/webapps/webview
https://developer.apple.com/documentation/uikit/uiwebview
https://developer.apple.com/documentation/webkit/wkwebview

Ideally, your WebView component will be setup to fill as much of the device screen as possible. The WebView layout should set the layout width and layout height to fill the parent for a full app screen experience. It is also

recommended to remove any title bar or address bar features when possible, to maximize the screen space available to the locator.

WebView Permissions Required
There are a small number of permissions that need to be enabled for your WebView integration to support full functionality of the Wave2 Locator. These are:

- Internet Access Enabled
- Javascript Enabled
- Geolocation Enabled

IOS CONSIDERATIONS

In iOS apps, it is recommended to use WKWebView which is a faster and more configurable WebView component versus UIWebView iOS implementations. This component should have all the required permissions enabled by default, so long as the deployed app has permissions to access location data, everything should work properly simply by loading your unique mobile URL into the WebView.

ANDROID CONSIDERATIONS

In Android apps, implementing a WebView object is just as simple, however you have to manually set all the proper permissions, which requires a few more steps in your app code.

To get access to internet and geolocation permissions, you will need to setup those permissions in your app manifest as follows:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

Then inside the app code, import both android.webkit.WebChromeClient and android.webkit.GeolocationPermissions into your app and set proper permissions as follows:

```
webview.setWebChromeClient(new WebChromeClient());
webview.getSettings().setJavaScriptEnabled(true);
webview.getSettings().setGeolocationEnabled(true);
webview.getSettings().setJavaScriptCanOpenWindowsAutomatically(true);
```

These permissions make sure that your Android WebView can access the internet, run the locator code, access device GPS location data, and open new windows for driving directions.

If these settings are all in place and the Android app is still not seeming to grant geolocation permissions, you may need to override the onGeolocationPermissionsShowPrompt method in the WebChromeClient and invoke the GeolocationPermissions.Callback to tell the WebView whether the user granted the permission to access location information or not when they were prompted, as detailed below.

Below is a sample Android implementation to make sure the user permission dialogue is handled correctly.

```
webView.setWebChromeClient(new WebChromeClient() {
  public void onGeolocationPermissionsShowPrompt(String origin,
                        android.webkit.GeolocationPermissions.Callback callback) {
    callback.invoke(origin, true, false);
 }
});
```

UNIQUE WEBVIEW URL

For every platform, your unique mobile app WebView URL will look like the following, substituting your own unique base URL for the demolocator URL. Please note the inclusion of the mobileapp=1 parameter. This is included to make sure your mobile WebView integration is tracked as a mobile app session.

```
https://demolocator.wave2.io/?mobileapp=1
```

2.2 SUPPORT

For support, please call 301.652.1245 or send an email to support@wave2.io or visit https://wave2locator.com/ and click on *Support* under the *Contact Us* menu