**Basic DemoWidget development and configuration :**

**For this widget to work, create a Web AppBuilder app with webmap, that has at least one Dynamic Map Service in it.**

Basic widget consist of few required files

1. CSS folder :-

* style.css file containing styles from widget UI
* Can have other css files for additional UI components, if any.

1. Images folder :-

* Icon.png image file for widget icon . Required
* Additional images if needed anywhere in widget

1. nls folder :- localization files

* String.js files that contains definition for configuration text for various components in the widget. Required
* Other localization files downloaded from esri website

1. Widget.html and Widget.js files

Main component of widget where widget UI and JavaScript logis are written

1. Settings folder with other settings and configuration files:- Certain widget requires some configuration based on the layers, map units, or attribute format etc. for it to work. Good example is Query widget in-build in WAB, when you add the widget to you app from Builder, it prompts you options on how user what the widget to work with layer and its attributes. Aslo, default measurement widget, when you add to app, show 2 dropdowns for selecting default units for you widget.

All the pre configuration of widget is defined in files in this folder.

For this basic version of widget development, we will be skipping this top. However, you can refer to the code in settings folder of default measurement widget and other widgets to get some idea on how it works.

**Development Steps :-**

1) Copy the DemoWidget folder from the zipped folder into **\client\stemapp\widgets**  of your WAB.

2) open manifest.json file and see the json structure

Name value pair contains the name for the widget. In this case it is **DemoWidget**

you can change the name to some other name as you like.

Other values in json file are other details about the widget and WAB version that it supports.

3) config.json file : This file contains the configuration if any required for this widget. In our example, we have json structure defined for a polygon symbol. For understanding purpose, we use this for defining symbols for selected polygon type feature in \_onDrawEnd method in Wdiget.js file

At this point Widget is ready to run in the App.

4) In many cases there is some pre-defined text information that needs to be displayed on the widget UI. In WAB standard, all this text is defined nls/strings.js file

Open nls/strings.js file in a text editor, and you will see below two key-value in roots function

\_widgetLabel: "DemoWidget", // this has to be same as WidgetName defined in manifest.js

developerName: "Your Name"

We can any number of key value pair variable listed in this file, To access values inside our widget code. For, example if we have to access any particular value in our widget code, we can simple call it by writing

nls.developerName

this will give us value associated with this key.

In our example, if you see Widget.html file in text editor, you will notice the line

Hello !! My Name is ${nls.developerName}. This is my first custom widget

So , when you launch you app, and open this widget, you will see text written on top of the widget as

**Hello !! My Name is “Your Name Value as defined” This is my first custom widget.**

**For understanding rest of the code for widget development, open Widget.js and Widget.html file in text editor. The code is fully ordered based on how widgets lifecycle runs, and follow the comments to understand the workflow. Follow ArcGIS JavaScript API reference, as the whole map workflow logic is from API only.**