JMAP Theme Manual

[**About The Current Theme**](#_nqaxz7aq9qzh) **2**

[**Installation Guide**](#_h1zn6aek0mnq) **2**

[**Startup Process**](#_72zmijibfhuo) **3**

[**Editing Process**](#_6kqin79sam0y) **3**

[**For custom theme testing:**](#_bcld9bl5jaup) **4**

[**For custom theme colours testing:**](#_uqgiqvrhua11) **7**

Understanding Dojo

**Understanding Config.js 8**

**Invoking a function in another widget 7**

**JMAP Theme Manual**

# About The Current Theme

The JMAP theme is made up of 3 different theme components

1. Launchpad Theme ( Base Theme )
2. Tab Theme
3. Jewelry Box Theme

Most widgets can be added through the Web App Builder Interface.

In case of custom widgets, follow the guides below to edit the codes.

# Installation Guide

You will need to have the following software to begin:

Visual Code

WAB ( Web App Builder )

If you do not have the developer edition for web app builder, follow the installation guide;

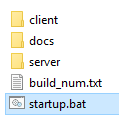
Installation Guide:

1. Go to <https://developers.arcgis.com/web-appbuilder/>
2. Download the SDK
3. Install the SDK
4. Get account credentials from the person you are working under
5. Wait for installation to be completed

# Startup Process

Startup Guide:

1. Open WebAppBuilderForArcGIS folder
2. Run the startup.bat file



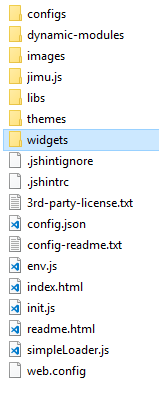
1. Create new/ edit existing by clicking on  or 
2. After entering the editing interface, click to see the web page in real time

# Editing Process

For dynamic custom widget testing:

How to edit codes for Widgets

1. Open WebAppBuilderForArcGIS folder
2. Go to 
3. Select the app number that you are working on which can be seen here 
4. All the widgets that you added in the interface will be inside this file



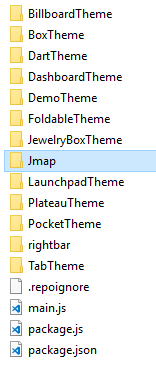
1. As all widget’s codes are appended using JavaScript, you can find all the functions in 
2. For editing of custom widgets go to [Editing of custom widgets](#_bbbvh3ib1u5q)
3. Any changes made to the codes will be implemented after you refresh the launched webpage
4. After that you need to copy the code block / file over to the client side to commit your changes

> **whatever you are editing**

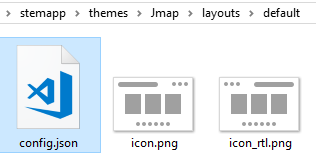
# For custom theme testing:

How to edit codes for UI

1. Go to 
2. Select



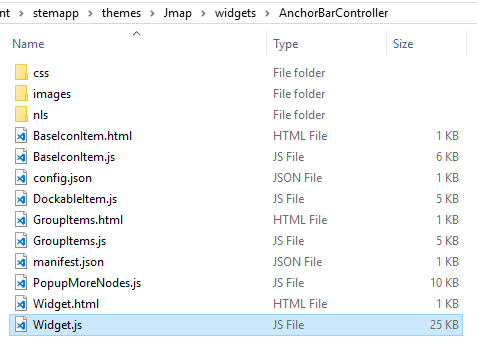
1. For layout changes

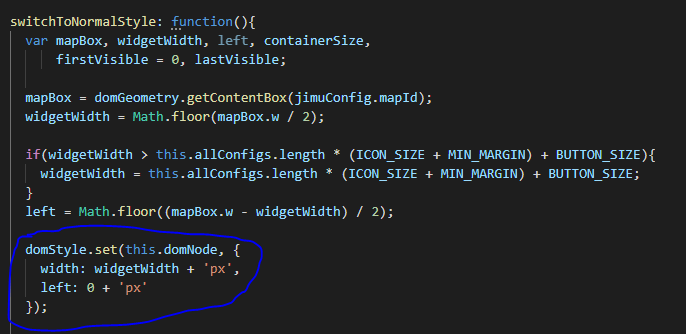


1. For right slider panel you can edit the position and height here

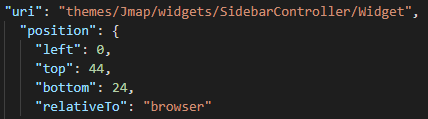


1. For theme widgets, you have to Ctrl + F codes
   1. Anchor bar controller positions are changed in the widget.js file





* 1. Side bar controller is as default but if needed;



1. In order to see commited changes, you will need to re-create the 2D web app

# For custom theme colours testing:

How to edit codes for UI colours

1. From the web app itself when you run it, right click inspect element
2. Select the element you want to edit
3. There should be a file path that is given to you when you hover it



1. Just follow the pathing and edit from there
2. For JTC colour scheme, etc, go to <https://ucc-uiux.firebaseapp.com/>

# **Understanding Dojo**

# Understanding Config.json

As customization by ArcGis AppBuilder is not supported by default, adding and removing a widget should be done via Config.json and not through the interface drag-and-drop.

In Config.json, be aware of the following glossaries:

1. URI: path of the widget
2. Position: Dimensions and paddings of a widget *(similar to CSS)*
3. Version: AppBuilder version. Leave the default value as it is.
4. ID: A unique identifier to distinguish between widget. **ID MUST BE DISTINCT.** If duplicated IDs are found. Appbuilder will automatically append underscore to the end of the ID to distinguish between them; this can lead to erratic and unexpected behavior for components using a specific ID format.
5. Name: Type of widget
6. Label: Custom title for widget displayed to user
7. ControlledWidgets: Smaller widgets that is contained or house by bigger widget such as AnchorBarController or SideBarController
8. ControlledGroups: Group widgets that is contained or house by bigger widget such as AnchorBarController or SideBarController
9. Index: position of a widget or widget group in an array
10. MapTo: a custom attribute to map a widget to another entity (be it a widget or layer). Value can be of any attribute associated to the entity you wish to map.

# Invoking a function in another widget

Invoking a function in another widget is done by WidgetManager. As a result, the invoked widget can receive parameters from WidgetManager. Each widget folder (*e.g. AnchorBarController, LayerList, About*) contains a script known as a widget.js. Only functions found in widget.js can be invoked by the WidgetManager.

The following diagram illustrates the workflow suppose we wish to invoke some functions in GroupItem.js in AnchorBarController through LayerListView.js in LayerList.

AnchorBarController

LayerList

Widget.Js

Widget.Js

ca

WidgetManager

DockableItem.js

GroupItems.js

BaseIconItems.js

LayerListView.js

PopupMenu.js

LayerFilter.js

WidgetManager.getInstance().getWidgetById(“../AnchorBarController”).anchorBarMethod(…)

Steps

1. Import WidgetManager by adding “jimu/Widgetmanger” to define[..] at the top, followed by adding “WidgetManager” to the function (…)
2. Next, assign a variable to the widget you wish to invoked.

var temp = WidgetManager.getInstance().getWidgetById(“widgettocall”)

1. Using the variable to subsequently call function in the widget.

temp.methodInWidget(param1, param2,…)

1. Widget.js can further invoke functions supplementary scripts in the same folder. (e.g. GroupItem.js)

*\* Also check out how to send and fetch information between widgets using DataManger in the API link below:*

https://developers.arcgis.com/web-appbuilder/guide/communication-between-widgets.htm