Coding Your AppsFramework Application

Published 2014-10-28 | (Compatible with SDK 3.5,4.5,5.0,5.1 and 2012,2013,2014 models)

> Coding Your AppsFramework Application

Contents

**** This class will not be supported in 2015.**

All functionalities of AppsFramework are more improved, integrating with CAPH. Therefore Appsframework is not supported since 2015 Smart TV. To use functionalities of Appsframework, refer to ***here****.*

The AppsFramework streamlines application development, thus allowing you to concentrate on the business logic and services. For more information about the AppsFramework and about creating an AppsFramework project type application, see AppsFramework.

To get started with the Samsung TV Apps Framework:

1. Choose the Apps Framework version.

   Choose the version of Apps Framework you wish to use. Change the Apps Framework core file path including the version as shown below.

   ```
   index.html
   ...
   <script type="text/javascript" src="$MANAGER_WIDGET/Common/af/2.0.0/loader.js"></script>
   ...
   ```
2. Modify the app.json file.
3. Modify the init.js file. Add the following code for the onStart function that is the entry point of all AppsFramework-based applications.

   ```
   function onStart() {
       alert('onStart()');

       // TODO : Add your Initilize code here

       sf.scene.show('Main');
       sf.scene.focus('Main');
   }

   function onDestroy () {
       alert('onDestroy()');

       // TODO : stop your XHR or Ajax operation and put codes to destroy your application here

   }
   ```

4. Define scenes and add the code in the scene class files.
5. Add the scene. You must add the 3 files each directory - htmls, scenes, stylesheets.

Writing the app.json File

The app.json file is descriptor for the AppsFramework. It is located at the root of each application.

A sample app.json fils shown below.

```
{
    "scenes" : ["Main", "Sub"],
    "files" : [],
    "theme" : "base",
    "languages" : ["en", "ko", "fr-US", "es-US", "pt-US"],
    "resolutions" : ["540p"],
    "modules" : ["ui.*"]
}
```

In the above example:

scenes

An Array<String>. If scene-based architecture is used, provide the scene list with this property.

files

An Array<String>. This is an optional property. If files are to be included with the application, add the file path to this array.

theme

A String. Define the theme of the Apps Framework UI components with this property. Currently, only the “base” theme is supported.

languages

An Array<String>. This is an optional property to set the supported languages in the application. The language resources have to be located in the lang directory. Add the file names in the form [languagecode].js to include while initializing the application. The first value is the default language. If the detected language does not match any of these values, the default language file (first one) is included. For more details, see Multi-Language Support.

resolutions

An Array<String>. This is an optional property to set the supported resolutions in the application. The options are '540p', '720p'. If this property exists, the scene stylesheet file (CSS) is loaded from /app/stylesheets/[RESOLUTION] instead of /app/stylesheets for the current resolution.

modules

An Array<String>. This is an optional property to use external modules. The table below lists the available values.

Value	Description
ime	IME for entering text
sso	Single Sign-On which is account managing module for Samsung Smart TV