Sepia Settings

This document describes how to implement app settings in an Ayla Mobile Foundry based app.

Define settings in config file.

The application settings json array should be defined in the user experience section of the config file. Each setting needs to include a name, type and default value. The framework currently supports Boolean, Integer, String, Decimal, MultipleChoice and LaunchScreen setting types.

The following is a sample settings json.

```
"applicationSettings": [{
    "name": "use_fingerprint_auth",
    "type": "Boolean",
    "defaultValue": false
  },
  {
    "name": "choose_config",
    "description": "choose_config_description",
    "type": "LaunchScreen",
    "defaultValue": "choose_config"
  },
  {
    "name": "user_data_grants",
    "description": "user_data_grants_description_settings",
    "type": "Boolean",
    "defaultValue": false
  },
  {
    "name": "temperatureUnits",
    "type": "MultipleChoice",
    "choices": ["F", "C"],
    "defaultValue": "C"
  }
]
```

Boolean

Default value is either true or false. Boolean settings are represented by widgets with On-Off values such as switches or toggle bars.

Integer, String and Decimal

These settings have default values of the corresponding type, and can be represented in a settings screen using EditText UI to accept values.

LaunchScreen

LaunchScreen type is used when the setting is too complicated to handle in the settings page of the app. Instead, a new screen will be launched, and the app can add the UI to handle this setting in the new screen. An example of the LaunchScreen type can be found in the “Choose config” setting in Ayla Mobile Foundry.

MultipleChoice

MutipleChoice is used when the settings value is to be chosen from a list of options. For example, a setting for temperature units as either “C” or “F”.

App settings are represented in the framework using SepiaAppSettings class. Screens using settings can instantiate an object of this class to create a new SepiaAppSettings object with the application settings defined in the config file. The screen UI can be adjusted dynamically based on the setting type as shown in AppSettingsScreen class. SepiaAppSettings class has getValue() and setValue() methods for reading and writing setting values.

Receiving settings change events

To receive settings change events, apps need to implement the SettingsChangeListener interface and register with SepiaActivity using the method SepiaActivity.addSettingsChangedListener(). Settings changes are received in the settingsChanged() callback. Any code to be executed on settings change events should be placed in this callback method.