Aura OEM Configuration

What is Aura?

Aura is Ayla's IoT engineer focused mobile application. Aura provides basic reference code to create a new user account, connect a new device to that account (wifi setup), register that device to the account, and configure properties in the cloud or in LAN Mode (WiFi peer networks). These are core Ayla platform features that can be used as a reference during early stage development.

Do I need OEM Config?

Aura is pre-configured to connect to the Ayla development service in the USA using the Aura app ID and app secret, and to manage properties for a few devices known to Ayla (currently the Ayla EVB (Evaluation Board or Dev Kit), the Ayla Generic Gateway (Raspberry Pi edition) and a couple of Smart Plugs).

IoT software engineers can easily change these values and rebuild the Aura application, so they may not need or want to use OEM Config, other than for convenience.

IoT HW/FW engineers also want to connect to different services or have built-in support for their own devices. However, many times they lack the time or skills to rebuild Aura. Additionally, early stage project typically focus on HW/FW, so mobile developers may not be available. To enable HW/FW engineers, Aura supports an external configuration file that can be installed via an email attachment or URL to allow Aura to be configured for any system. Customers can create an Aura configuration file containing the details required to connect to their own OEM account (app ID / secret) and manage their own devices.

How do I use OEM configuration files?

The Aura configuration file is recognized by its extension, ".auraconfig". The mobile applications register themselves as handlers for these files, and will import or update the configuration into Aura, and set it as the default when the file is opened on a mobile device with Aura installed.

Once the configuration has been installed, it will appear in the "Developer Options" section of the application in a drop-down list, along with the pre-defined configurations that ship with Aura. An installed configuration may be chosen by selecting the appropriate item from the drop-down list in the developer options screen.

If you are unsure of the values to use, check with the project lead responsible for

creating the Ayla platform template on the OEM Dashboard. The documentation in the iOS_AylaSDK and/or Android_AylaSDK github repos are also useful for better understanding these fields.

Decoding the .auraconfig file

The .auraconfig file is a JSON file containing the following fields:

name	Name of the configuration, required, visible to the user
appId	Application ID to use to sign in to the service Default: N/A - Created on the Dashboard by the project lead
appSecret	Application secret to use to sign in to the service Default: N/A - Created on the Dashboard by the project lead
serviceType	One of: Dynamic, Field, Development Default: Development
serviceLocation	One of: USA, China, Europe Default: USA
allowDSS	Allows or disallows the use of the Device Subscription Service, boolean value. This new feature must also be enabled on the cloud. Consult with the project team leader. Default: false
allowOfflineUse	Allows or disallows the application from signing in while the cloud service is unavailable. Use cautiously, as the app is now responsible for user authentication! Default: false
defaultNetworkTimeoutMs	Default timeout for network connections in milliseconds. Default: 5000.

managedDevices	Array of devices the application manages. This includes a mapping of DSN, model or OEM model to a set of property names that are to be managed for matching devices. Default: N/A

The array of managed devices is an array of objects with one or more of the following fields set:

dsn	DSN of a specific device to manage
oemModel	OEM model to match against a device
model	Model to match against a device
managedProperties[]	Array of property names that should be managed for matching devices. This is typically a small subset of all the properties from the template. Only include properties that are required by the mobile app.

Aura will check each device against the active configuration's managed device array and attempt to match them. If all defined fields from a configuration match a device, then that set of property names is returned to the Mobile SDK as the set of managed properties for that device.

A sample Aura configuration file is included below:

```
{
        "allowDSS": false,
        "allowOfflineUse": false,
        "appId": "aura_0dfc7900-id",
        "appSecret": "aura_0dfc7900-eUo-3Se7Rr5Z_QLeEiX1YkQDUNA",
        "defaultNetworkTimeoutMs": 5000,
        "name": "Brian's Cool Config",
        "serviceLocation": "USA",
        "serviceType": "Development",
        "managedDevices": [
            {
                // Ayla EVB
                "model" : "AY001MTC1",
                "managedProperties" : ["Blue_button", "Blue_LED",
"Green_LED"]
            },
            {
```

```
                // Ayla EVB
                "model" : "AY001MUS1",
                "managedProperties" : ["Red_button", "Blue_LED",
"Green_LED"]
            },
            {
                // Generic Gateway
                "model" : "AY001MRT1",
                "oemModel" : "generic",
                "managedProperties" : ["num_nodes", "log"]
            },
            {
                // Smart plug
                "oemModel" : "smartplug1",
                "managedProperties" : ["outlet1"]
            }
        ]
}
```