Arrowhead Adapter Quick Demo

Introduction

The Arrowhead Framework can be used in various industrial environments to provide a cloud of collaborative IoT devices. These collaborations are based on runtime service providing and consuming. These services can be anything from business logic to sensor measurements data.

In this demo we take a legacy system, which has no ability to register its service on its own. Therefore, we need an Adapter that registers the given system in the Arrowhead Cloud and serves its measurement data to the consumers that requested them. Here the legacy system will be represented by a DHT11 temperature sensor and the provided service will be the temperature data itself.

About this Demo

The process of integrating a new service – especially through a legacy system – into the Arrowhead ecosystem can be a complex task, primarily due to security reasons. This *Quick Demo* aims to simplify this process by providing pre-prepared scripts and a well-defined environment. With this package in your hand you can get a functional and interesting introduction to the Arrowhead Framework.

This demo is prepared to run on **Windows**. A Linux version is also available.

Core System

The Arrowhead Cloud hosts the frameworks' core components that the runtime collaborations between service providers and consumers. There is an available test cloud that can be used in this demo. (This system's URLs are preconfigured in the example codebase.)

You can also build your own Cloud. For this, you can get further information at https://github.com/arrowhead-f/core-java-spring.

How-To

This section will lead you through the steps of putting your adapter into operation, registering your device into the Arrowhead Cloud and testing it with a primitive consumer.

Assemble

Connect the DHT11 sensor VCC, DATA and GND pins to the appropriate pins on the Adapter respectively. Power up your device by plugging into your computer through USB. (Later on, the power source can also be a general mobile phone USB charger.)

Install dependencies

This demo requires a few programs as its environment. To install these dependencies, click on `install.bat`. This script does the following:

1. Checks whether you already have *Python* installed. If it is installed, you can see the version number.
2. Asks if you would like to install *Python*. If the version number (that is showed above) is less than 3.7 or *Python* is not installed at all, type **Y** and press *Enter*. The script will install *Python* for you. (You will need to navigate through its installer.) Otherwise type **N** and press *Enter*.
3. Local Java environment will be installed.
4. In addition, *MSYS2* will be installed. (You will need to navigate through its installer.) You must leave the installation path as it is: C:\msys64 (otherwise you have to include C:\msys64\usr\bin in your PATH environment variable).

Generate certificates

In order to gain access to the secure Arrowhead Cloud, you have to possess the appropriate certificates for your client programs/devices. In this demo it means that you need a certificate for your adapter and one for your test consumer. To generate these certificates, click on `generate_certificates.bat`. Here you must provide a unique name for your adapter and **an other** for your consumer. The names cannot contain spaces or any special characters.

Different parts of the demo need different formats of the certificates generated beforehand. To create these different formats as well, click on `transform_certificates.bat`. Here you must provide your system names that you used earlier. It might also ask for a passphrase a few times. Type **123456** in every case.

Initialize

In this step you will upload the adapter program to the initially empty device.

First connect the microcontroller to your computer through USB. You need to know on which serial port it is available. To find this, open *Device Manager* and search for the Ports entry. You can see your device port here (e.g. COM9).

Run `init_adapter.bat` to upload the program. (Enter the port that has been previously discovered.)

Configure

You have to configure the adapter to fit your environment. Navigate to the source/data folder. Here you can find 3 JSON config files that you will modify accordingly.

- *netConfig.json*: Provide your WiFi name and password to which your adapter device can connect.
- *providerConfig.json*: Here only the *systemName* parameter has to be modified. Put here your adapter name that you used earlier.
- *sslConfig.json*: Here only the *publicKey* parameter has to be modified. Open the *pubkey.pem* file that is in the same folder. Copy its content without the -----BEGIN PUBLIC KEY----- and -----END PUBLIC KEY----- parts as the *publicKey* value. Remove the newlines to fit in one row (as in the example)!

Navigate back to the root folder and run `update_adapter.bat` that will upload your configuration to the device.

Register

If you reached this step, you have successfully set up and configured your adapter. In order to connect your adapter to the cloud, you need to register it first appropriately in the Arrowhead ecosystem. Run `register_adapter.bat`, it will achieve this for you. You have to provide the system names that you used earlier as well as the public key that you already copied in the previous step. It is important to remove the newlines to fit in one row as you did earlier otherwise the program won't work.

At this point, the adapter device has to be reset for the registration come into effect. To achieve this, press the reset button on the device for about 3 seconds then release it. After 10-15 seconds the device should be back into operation.

Run test consumer

Now you have a functional adapter that serves the current temperature upon request through the Arrowhead ecosystem. To test whether it really works, you will use a test consumer program that will ask the Arrowhead Cloud for the current temperature (at this point it can only reach one provider that you registered in the previous step).

Navigate to the consumer folder and open the *application.properties* file. Here you have to modify the *client_system_name*, *key-store* and *key-alias* values. The *client_system_name* will be your consumer**(!)** system name as you provided earlier. In the key-store value you only need to modify the filename itself (it will be the same as your consumer**(!)** system name). The latter will be your consumer system name again.

Navigate back to the root folder and run `run_consumer.bat`. If everything goes well, the device response and the current temperature will be presented.