Connector Build addition to SSB

Author: Keith Smith

Problem:

In later versions of IdentityIQ, the lib-connectors folder was implemented in order to divide the connectors from the main code. The effect was to make existing connectors, built upon the openconnectors framework and using a single POJO, no longer work when deployed in the WEB-INF/classes folder.

The following Compass article provides the direction for fixing this:

https://community.sailpoint.com/t5/IdentityIQ-Forum/Custom-Connector-which-extends-OOTB-connector/td-p/226176

The simple solution:

Package all of the class files into a jar and save to the lib-connectors folder, then add the reference to the JSON file. While this can be done manually, an SSB automation is better.

Proposed Solution which has been integrated into the SSB 7.0.2 version:

In a similar fashion to the plugins solution, the following connector solution is proposed:

- Create a folder in the SSD Base folder called connectorsrc
- Create a folder for each connector, which is the name of the connector (this will be the name of the jar file and should be the name used as often as possible in the files). This is the home folder for that connector.
- In the home folder the following folders are created:
    - define/applications - to put the connector xhtml file in
    - install/Configuration - to put the connector registry merge file into
    - src - to put the source code into
- In the home folder a manifest.xml file is needed, looking similar to a plugin manifest file but the following differences:
    - No dtd line (line 2) - no need to validate
    - Structure of Connector/Attributes/Map
    - Single entry key="connectorName" and value = the full class name of the connector
    - Followed by jar file references, the last of which is the Connector.jar file (Connector being the name of the connector). The easiest way to add these is something like in the example:
      <entry key="jar1" value="connector-bundle-webservices.jar"/>
      <entry key="jar2" value="ExampleConnector.jar"/>
    - The will edit the connectorVsConnectorBundleMapping.json file adding it to the file.

- The following changes are needed in the files:
- build.xml - Add the following to the imports:

```
<import file="scripts/build.connectors.xml"/>
```

Add the following after the definition of pluginsrc:

```
<property name="connectorSrc" location="connectorsrc"/>
```

After the prepareCustomConfig antcall section add the following:

```
<!-- Build and deploy any connectors in the connectors source folder -->
<if>
  <and>
    <not>
        <equals arg1="${pre7.1}" arg2="true"/>
    </not>
    <available file="${connectorSrc}" type="dir"/>
  </and>
  <then>
    <antcall inheritall="true" target="buildConnectors"/>
  </then>
</if>
```

- Add the build.connectors.xml file (attached)
- Add the bsf.jar file to the lib folder
- Add the json-simple-1.1.1.jar file to the lib folder

The basic process works like this:

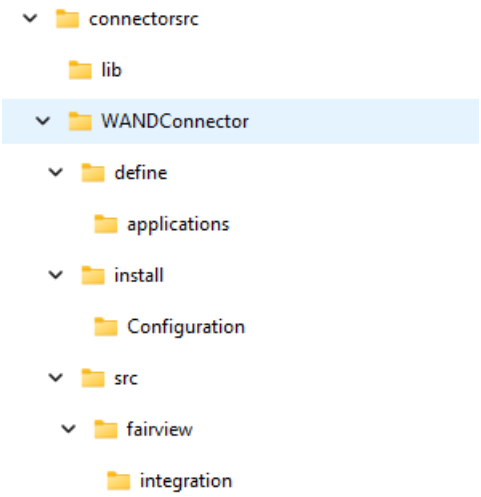
For each connector in the connectorsrc folder:

- Verify presence of the manifest.xml file
- Copy ConnectorRegistry the merge file to the (existing) extract/WEB-INF/config/custom/Configuration folder
- Copy the XHTML file to the extract/define/applications folder
- Compile all of the src source files into a jar file by the name of the connector
- Copy the jar file into the extract/WEB-INF/lib-connectors folder
- Add an entry to the connectorVsConnectorBundleMapping.json file
    - This requires reading the entry from the manifest.xml file
    - And the list of jar files
    - Also requires parsing the json file and writing it back

Once done the build-init method has to be executed again so that all of the added XML files can be added to the sp.init-custom.xml file

The changes will be found in the build/extract/WEB-INF/lib-connector folder, with some additional artifacts in the build/connectors folder.

Example folder structure:




Under connectorsrc is a lib folder and each connector's folder structure.

Under define/applications is the xhtml file for the connector

Under install/Configuration is a ConnectorRegistry merge file for the connector

Under src is the java code that you would have put under the main src folder.

In the main folder is the manifest.xml file

Example manifest.xml:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<Connector displayName="WAND Connector" minSystemVersion="7.1"
           name="WANDConnector" version="1.0.0">
  <Attributes>
    <Map>
      <entry key="connectorName" value="fairview.integration.WANDConnector"/>
      <entry key="jar1" value="connector-bundle-webservices.jar"/>
      <entry key="jar2" value="WANDConnector.jar"/>
    </Map>
  </Attributes>
</Connector>
```

The code in the scripts file takes the jars, in the order presented using the numbering, and adds them to the WEB-INF/lib-connectors/connectorVsConnectorBundleMapping.json file as follows:

"fairview.integration.WANDConnector":"connector-bundle-webservices.jar, WANDConnector.jar",

Not every connector requires the connector-bundle-webservices.jar file and you will need to test to determine if the line for your connector requires more jar files than just your connector.