Latest update on: 10/18/2016

Tutorial – Implementing YANG modules on ODL

1. **Get Ubuntu OS with an IDE and all requirements running on VirtualBox**
   a. Make sure to assign enough resources to your VM (e.g. 4 CPUs, 8GB Ram, at least 30GB disk space)
   b. Set up a shared folder and clipboard via settings of the VM and the following commands inside the Terminal of the running VM (restart VM afterwards):
      i. `sudo apt-get install virtualbox-guest-dkms` (clipboard)
      ii. `sudo mount -t vboxsf share ~/host` (folder; follow tutorial on https://forums.virtualbox.org/viewtopic.php?t=15868)
   c. Install latest JDK, Eclipse (or your preferred IDE) and the YANG IDE for Eclipse (https://github.com/xored/yang-ide/wiki/Installing).
   d. Download latest release of ODL (or whatever release you need) and set your `JAVA_HOME` in your `.bashrc` (`export JAVA_HOME="/usr/lib/jvm/java-8-openjdk-amd64/"`).
   e. Optional: You might want to install OpenDaylight User Interface (DLUX) by `feature:install odl-dlux-all` while ODL is running and `libcontainer` (http://libcontainer.sourceforge.net/) to add multiple JARs to your Eclipse projects.
   f. Install GIT and Maven using `sudo apt-get install git` and `sudo apt-get install git`.
   g. Get the right `settings.xml` for maven into your folder `~/.m2/` by:

```
wget -q -O - https://raw.githubusercontent.com/opendaylight/odlparent/master/settings.xml
> ~/.m2/settings.xml
```

Maven command to create a new empty project (check for latest archetype versions in the catalog)

```
mvn archetype:generate -DarchetypeGroupId=org.opendaylight.controller -DarchetypeArtifactId=openda
ylight-startup-archetype \

-DarchetypeRepository=http://nexus.opendaylight.org/content/repositories/opendaylight.snapshot/ \

-DarchetypeCatalog=http://nexus.opendaylight.org/content/repositories/opendaylight.snapshot/archet
ype-catalog.xml \

-DarchetypeVersion=1.3.0-SNAPSHOT
```

```
NETCONF testtool verifying:
ssh admin@localhost -p (PORT NO; e.g. 17830) -s -oHostKeyAlgorithms=+ssh-dss netconf
```

Link to MD-SAL:Startup Project Tutorial:

https://wiki.opendaylight.org/view/OpenDaylight_Controller:MD-SAL:Startup_Project_Archetype

Link to Application Development Tutorial:

Note: If you're following this tutorial, make sure, you won't grep an old Snapshot-Version.
Don't remove any auto-generated comments in the beginning of Java-Classes.

https://wiki.opendaylight.org/view/Controller_Core_Functionality_Tutorials:Application_Development_Tutorial#Setup_Development_Environment

Link to walkthrough for that Tutorial:

https://www.youtube.com/watch?v=2wTEuNyxspY&index=13&list=PL8F5jrwEpGAiJG252ShQudYeodGSsks2l

How to Debug Karaf:

Latest update on: 10/18/2016

https://youtu.be/EfK-_NA7jqU?t=30m50s

How to work on existing ODL projects:

Search on git/github for the module you are looking for: https://git.opendaylight.org/gerrit/p/ or https://github.com/opendaylight

```
git clone https://git.opendaylight.org/gerrit/yang-push
cd yang-push/
mvn clean install
```

Then go to Eclipse and import as existing Maven project. After some time, errors inside workspace may occur that will even after a 'mvn clean install' persist. To fix this issue, delete your project from eclipse (do not remove it from your disk) and then re-import it again as existing Maven project.

Data Store:

https://youtu.be/yDTiL8R-PAw?t=2m3s

Ignores some style checks like 'is the following statement less than 40 characters'

No testing will be done

mvn clean install -Dcheckstyle.skip=true -DskipTests

Maybe you have to run it a few times, for no reason.

no-snapshot-updates

mvn –nsu clean install

How to remove Java home warning on karaf startup:
Add *export JAVA_HOME=*"*/usr/lib/jvm/java-8-openjdk-amd64/*" (or your individual path) to */home/{user}/.bashrc*

ODL-Toaster Tutorial is currently offline:

Jun 14, 2016 8:13 AM: `'Remove toaster, as this is an outdated version, latest is actually still in controller.'`

In case there are some BUILD FAILURES due to imports try:

- ⇨ delete the folders below ~/.m2/
- ⇨ delete projects (e.g. 'hello')
- ⇨ get new repository (`mvn archetype:generate…`)

Latest update on: 10/18/2016

Maven Import of YANG Projects in Eclipse:

- Import all POM-files end edit the java files in the separate projects, so that Eclipse interprets them as Java-projects.

Shows installed features called 'hello'

ODL Commands

```
feature:list -i | grep hello
```

```
feature:info odl-hello-ui
```

To test your own provider, use following command from time to time:

```
log:display | grep yangpushserver
```

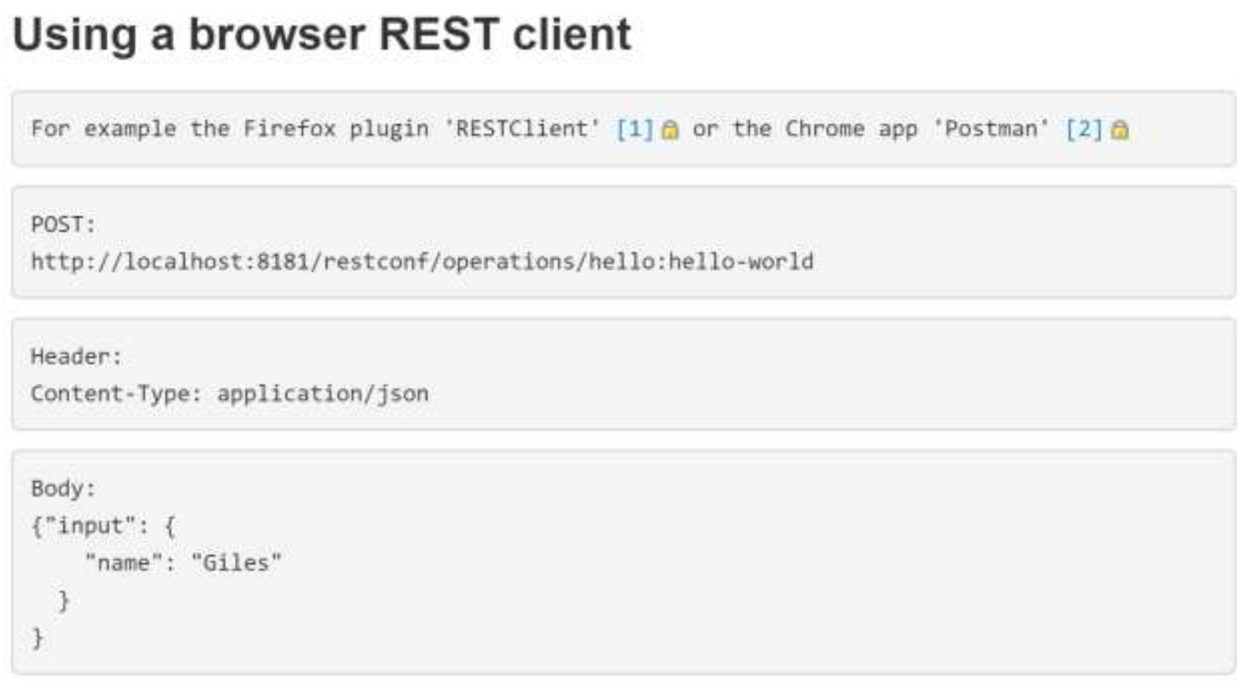
and look for something like following:

```
yangpushserverProvider Session initiated.
```

This will provide more detailed log (As usual also stored in karaf.log):

```
log:set TRACE
```

Test 'hello world' RPC using **POSTMAN**:



Using a browser REST client

For example the Firefox plugin 'RESTClient' [1] or the Chrome app 'Postman' [2]

```
POST:
http://localhost:8181/restconf/operations/hello:hello-world
```

```
Header:
Content-Type: application/json
```

```
Body:
{"input": {
    "name": "Giles"
  }
}
```

Helpful links:

Latest update on: 10/18/2016

https://maven.apache.org/guides/introduction/introduction-to-dependency-mechanism.html

https://nexus.opendaylight.org/content/repositories/public

https://github.com/opendaylight/controller/tree/stable/boron

How to add features to your ODL distribution (e.g. `odl-netconf-mdsal` aka. NETCONF northbound support **+** `odl-netconf-connector-all` for `odl-inventory`)

1. Go to `features/src/main/features/features.xml` in your projects directory
2. Look for your desired feature on https://nexus.opendaylight.org/content/repositories/opendaylight.snapshot/org/opendaylight/ and add it as repository to your features.xml as follows:
```
<repository>mvn:org.opendaylight.netconf/features-
netconf/{{VERSION}}/xml/features</repository>
<repository>mvn:org.opendaylight.netconf/features-netconf-connector/1.2.0-
SNAPSHOT/xml/features</repository>
```
3. Go to your `features/pom.xml` and add a version for your new feature under `<properties>` like:
```
<netconf.version>1.2.0-SNAPSHOT</netconf.version>
```
4. Now just add the `dependency` for your new feature under `<dependencyManagement>` like:
```
      <dependency>
        <groupId>org.opendaylight.netconf</groupId>
        <artifactId>netconf-artifacts</artifactId>
        <version>${netconf.version}</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
      <dependency>
        <groupId>org.opendaylight.netconf</groupId>
        <artifactId>sal-netconf-connector</artifactId>
        <version>1.5.0-SNAPSHOT</version>
        <type>pom</type>
        <scope>import</scope>
      </dependency>
```
And under `<dependencies>` further below like:
```
    <dependency>
      <groupId>org.opendaylight.netconf</groupId>
      <artifactId>features-netconf</artifactId>
      <classifier>features</classifier>
      <type>xml</type>
      <scope>runtime</scope>
    </dependency>
    <dependency>
      <groupId>org.opendaylight.netconf</groupId>
      <artifactId>features-netconf-connector</artifactId>
      <classifier>features</classifier>
      <type>xml</type>
      <scope>runtime</scope>
    </dependency>
```

Latest update on: 10/18/2016

Restconf: Change event notification subscription (https://wiki.opendaylight.org/view/OpenDaylight_Controller:MD-SAL:Restconf:Change_event_notification_subscription) required feature in ODL is called `odl-mdsal-remoterpc-connector`.

You probably have to upload huge files (>100MB && <1GB) to github, in case you want to collaborate with other people. This short video may help a lot:

https://www.youtube.com/watch?v=uLR1RNqJ1Mw

More basics for github:

https://help.github.com/articles/adding-an-existing-project-to-github-using-the-command-line/

Opendaylight Logfile:

Normally saved in the file 'karaf.log' under /PROJECTNAME/karaf/target/assembly/data/log

---

Try to connect to NETCONF md-sal northbound SSH server, you can do it like this:

```
ssh -oHostKeyAlgorithms=+ssh-dss admin@127.0.0.1 -p 2830 -s netconf
```

You send your hello message to server. This should be just fine:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<hello xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
      <capabilities>
      <capability>urn:ietf:params:netconf:base:1.0</capability>
      </capabilities>
</hello>]]>]]>
```

And then finally you will send your RPC. So you type:

```
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" message-id="101">
      <establish-subscription xmlns="urn:ietf:params:xml:ns:yang:ietf-event-notifications">
            <encoding>encode-xml</encoding>
            <stream>push-update</stream>
            <period xmlns="urn:ietf:params:xml:ns:yang:ietf-yang-push">30</period>
      </establish-subscription>
</rpc>]]>]]>
```

When using NCClient to communicate with ODLs Netconf northbound server with settings like above make sure the payload of the messages you send include `]]>]]>` **as well.** (Delimiter to indicate end of message; usually already provided by NCClient?)

Latest update on: 10/18/2016

To register change listeners for MD-SALs data store do the following:

```
// Inside of onSessionInitiated of your Provider
DOMDataBroker db = session.getService(DOMDataBroker.class);
// Getting the actual service that allows to register for data tree change events
DOMDataTreeChangeService changeService = (DOMDataTreeChangeService)
db.getSupportedExtensions().get(DOMDataTreeChangeService.class);

changeService.registerDataTreeChangeListener(new
DOMDataTreeIdentifier(LogicalDatastoreType.OPERATIONAL, yiid), this);
```

How to check what OpenDaylight version your distribution is using:

Type `version` inside your running karaf.