ThingWorx binding to OpenHAB 2.0 distro

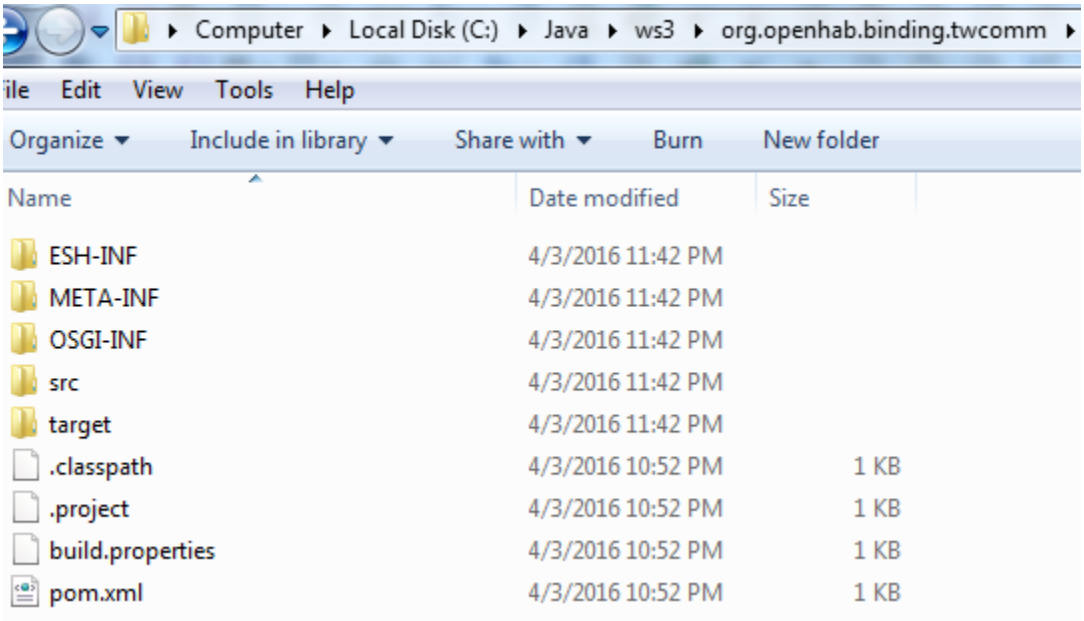
I tried to create TWcomm binding by following here
http://docs.openhab.org/developers/development/bindings.html
Developing a New Binding for openHAB 2
http://docs.openhab.org/developers/development/bindings.html#setup-and-run-the-binding

I downloaded addons source files
https://github.com/openhab/openhab2-addons
and extracted to openhab2-addons-master folder
and enter into addons\binding folder.

Then executed
.\create_openhab_binding_skeleton.cmd TWcomm twcomm
I had to correct JAVA_HOME environmental variable.

Then the org.openhab.binding.twcomm folder (skeleton) is created.

I got this folder.



with some files starting with twcomm…

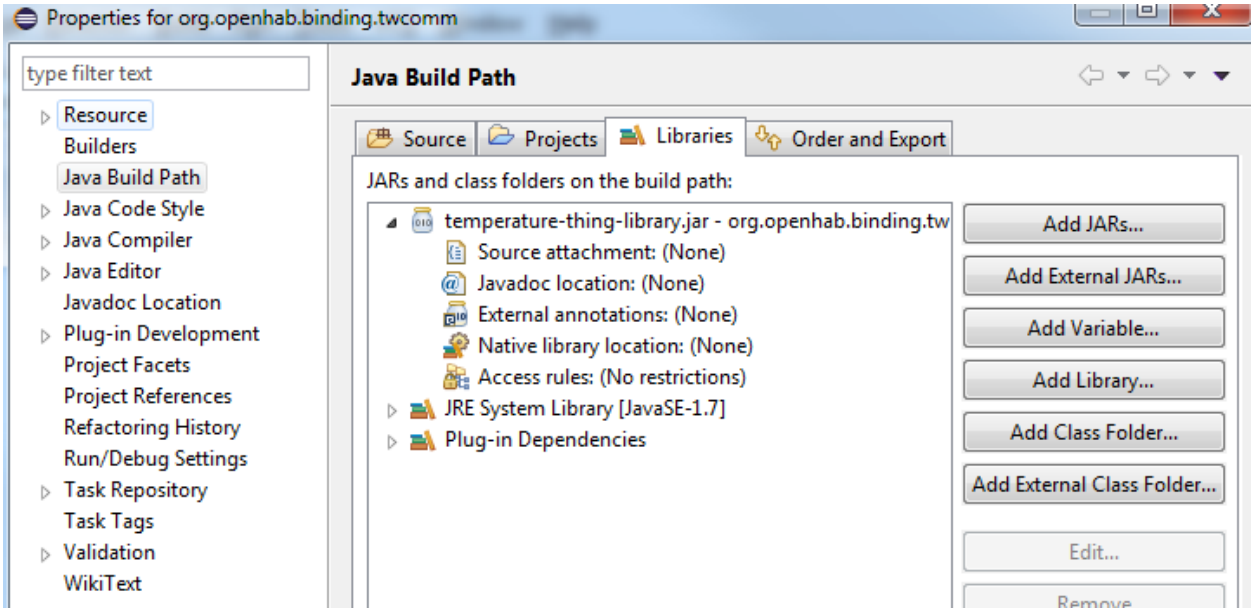
Name	Path	Size
org.openhab.binding.twcomm	C:₩Java₩ws3₩.metadata₩.plugins₩org.eclipse.core.resources₩.projects₩	
twcomm	C:₩Java₩ws3₩org.openhab.binding.twcomm₩src₩main₩java₩org₩openhab₩binding₩	
twcomm	C:₩Java₩ws3₩org.openhab.binding.twcomm₩target₩classes₩org₩openhab₩binding₩	
org.openhab.binding.twcomm	C:₩Java₩ws3₩	
twcomm_xx_XX.properties	C:₩Java₩ws3₩org.openhab.binding.twcomm₩ESH-INF₩i18n₩	465 Bytes
TWcommHandlerFactory.class	C:₩Java₩ws3₩org.openhab.binding.twcomm₩target₩classes₩org₩openhab₩binding₩twcomm₩internal₩	1.65 KB
TWcommBindingConstants.class	C:₩Java₩ws3₩org.openhab.binding.twcomm₩target₩classes₩org₩openhab₩binding₩twcomm₩	744 Bytes
TWcommHandlerFactory.xml	C:₩Java₩ws3₩org.openhab.binding.twcomm₩OSGI-INF₩	719 Bytes
TWcommHandlerFactory.java	C:₩Java₩ws3₩org.openhab.binding.twcomm₩src₩main₩java₩org₩openhab₩binding₩twcomm₩internal₩	1.56 KB
TWcommBindingConstants.java	C:₩Java₩ws3₩org.openhab.binding.twcomm₩src₩main₩java₩org₩openhab₩binding₩twcomm₩	928 Bytes
TWcommHandler.class	C:₩Java₩ws3₩org.openhab.binding.twcomm₩target₩classes₩org₩openhab₩binding₩twcomm₩handler₩	1.62 KB
TWcommHandler.java	C:₩Java₩ws3₩org.openhab.binding.twcomm₩src₩main₩java₩org₩openhab₩binding₩twcomm₩handler₩	2.52 KB

There are other files in the folders.

When I import TWcomm, the following file is modified
C:\Java\openhab-distro\launch\openHAB_Runtime.launch
in this field
<stringAttribute key="selected_workspace_plugins" value="……,
org.openhab.binding.twcomm@default:default, ……/>

I put “temperature-thing-library.jar” in the folder of “org.openhab.binding.twcomm”

Then in the Java Build Path for twcomm, for Libraries, I added the temperature-thing-library.jar with Add JARs… button (as shown below)



In thing-type.xml file
Now checking with terms: channel_id, channel_type_id, thing_id, thing_type_id, config-description, parameter-group, parameter, in addition to binding-id.
parameter-group names: identification, device, binding

parameter names: room (room number or name), name (e.g., device name), comfortTemp, ecoTemp
channel type IDs: valve, battery, mode, actual_temp, set_temp, temperature, power, switch
channel IDs: valve, battery_low, temperature, power, switch1
(thing-type IDs: weather, a system or product name,
thing IDs: location, a node, a thing)

The syntax for a channel link is { channel = "<binding-id>:<thing-type-id>:<thing-id>:<channel-id>" }

{ channel = “twcomm:tw_comm:mynode:temperature” }
{ channel = “twcomm:tw_comm:mynode:humidity” }

What is channel-type?
channel type IDs: valve, battery, mode, actual_temp, set_temp, temperature, power, switch
channel IDs: valve, battery_low, temperature, power, switch1

When finally all are running properly, in ThingWorx, need to create a Thing and a Mashup.
There are many tutorials for this, so I skip.

Controlling a device from property change or from mashup

One way of triggering a service is by using event/subscription.
For example, a property can be used to trigger a service as follows.
https://community.thingworx.com/thread/39369 (very helpful)

```
Below is skeleton of Subscription :
if(isConnected == true)
{
     //call remote service on sdk
     var result = me.checkIsConnected("test");
}
```

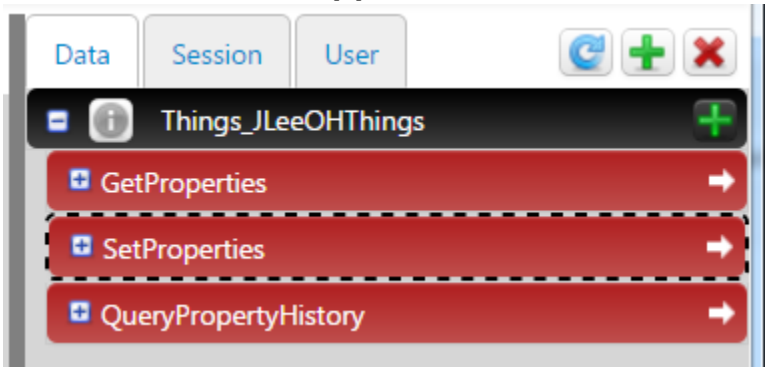
```
Below is skeleton of SDK :
@ThingworxServiceDefinition(name = "checkIsConnected", description = "send data to sdk")
@ThingworxServiceResult(name = CommonPropertyNames.PROP_RESULT, description = "Result", baseType =
"BOOLEAN")
public bool checkIsConnected(@ThingworxServiceParameter(name = "test", description = "Value 1", baseType =
"STRING") String str) throws Exception {

        log.info("\nService Start "+ str  );
        try {
                //called thingworx service to get data
        } catch (Exception e) {
                log.error("Error while processing the request ", e);
        }
        log.info("Ending remote method ++++++++++++++++++++++++++");
        return true;
}
```

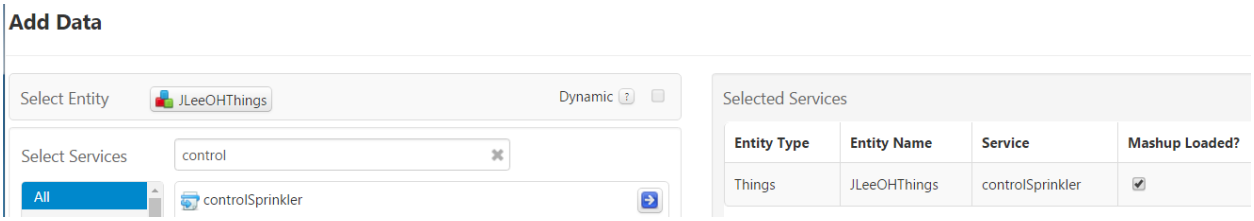
Another way is to trigger the service from mashup.
Try to create a button to invoke a service and control something.

https://community.thingworx.com/thread/2217 (helpful for mashup)

Before binding to any component in a mashup, you must add *service* data (e.g., controlSprinkler) in the list of Data as follows. Click [+] below.



Then add controlSprinkler service item.



Helpful tips to add Services, Properties, and Events, follow here
http://support.ptc.com/cs/help/thingworx_hc/thingworx_edge/index.jspx?id=thingworx15&action=show

1. Open the **SteamSensor1** thing.
2. Select the **Properties** tab.
3. Refresh the properties to verify that the **isConnected** property is **true**.
4. If the page is not in Edit mode, click the **Edit** button to enter Edit mode.
5. Click the **Manage Bindings** button to browse for the remote properties.
6. In the **Manage Property Bindings** page, click the **Remote** tab to view the remote properties.
7. Click the **Add All Above Properties** button to add all of the properties.
8. Click **Done**.
9. Click **Save**.
10. Refresh the properties and verify that they are changing.
11. Select the **Services** tab.
12. Click the **Browse Remote Services** button.
13. Click the **Add All Above Services** button to add the services.
14. Click **Done**.
15. Click **Save**.
16. Click the **Test** button next to the **AddNumbers** service and verify it is working.
17. Select the **Events** tab.
18. Click **Browse Remote Events**.

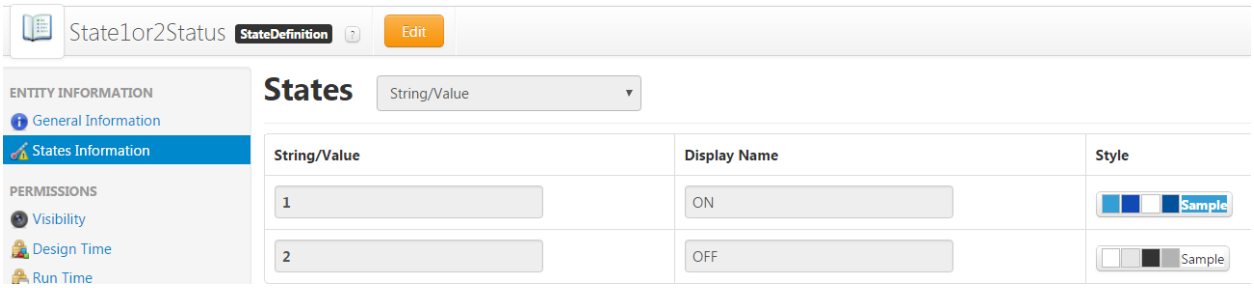
19. Click **Add All Above Events**.
20. Click the **Create** button next to the **SteamSensor.Fault** Data Shape if it has not already been created.
21. Click **Save**.
22. Select the **Subscriptions** tab.
23. Click **Add My Subscriptions**.
24. From the **Select an Event** drop down menu, select **SteamSensorFault**.
25. Select the **Enabled** check box.
26. Enter the following for the Script text that will write an event when there is a steam sensor fault:

```
logger.error("Steam Sensor 1 Fault: " + eventData.message);
```

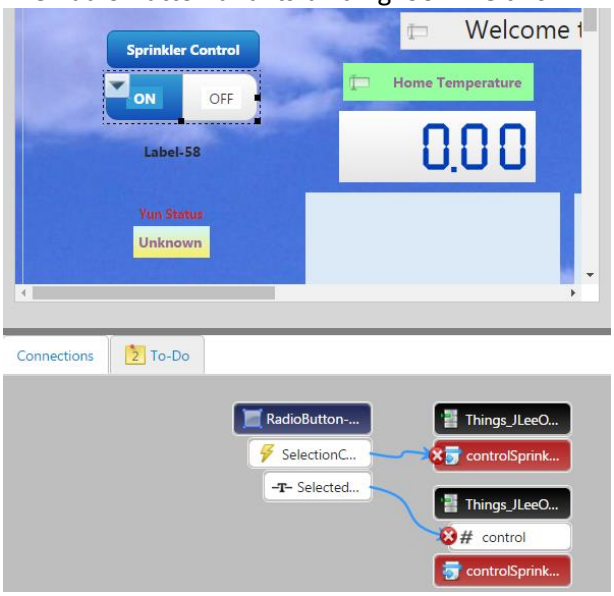
27. Click **Done**.
28. Click **Save**.

To pass large properties, one can use `InfoTable`.

In the meantime, I added a Radio Button, but because I want to have two selection, for ButtonStates, I had to create a Status. I created State1or2Status and edited as follows.



The Radio Button and its binding look like this.

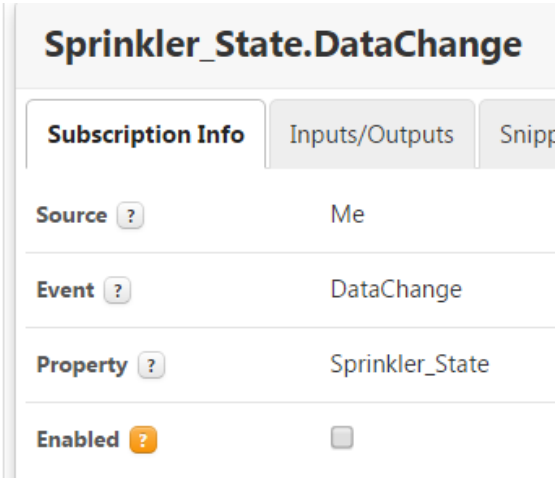


As you can see, I also create a button named Sprinkler Control.

Now, if I click ON button, the *controlSprinkler* service is invoked with *control* value 1, and if I click OFF button, the *controlSprinkler* service is invoked with *control* value 2.

I also added state display as follows but it displayed true or false, which I have to fix.

Now I try to display the status of sprinkler with string “ON” or “OFF” instead of check box.
For this I had to create a property called On_Off_text, whose value will be changed when Sprinkler_State changes.
Then I created a subscription which does it.



Also wrote Script as follows.

```
if(eventData.newValue.value==true) {
    me.On_Off_text = "On";
    me.On_Off_image = "/Thingworx/MediaEntities/SubsystemRunning";
} else if (eventData.newValue.value==false) {
```

```
    me.On_Off_text = "Off";
    me.On_Off_image = "/Thingworx/MediaEntities/StatusBusy";
}
```

Then I added the following properties: Sprinkler_State_Image, Yun_State_Image, and Dehumid_State_Image.

Added code for each source: Sprinkler_State, Prop_YunAlive, and Base_Dehumid.

```
if(eventData.newValue.value==true) {
    me.Yun_State_Image = "/Thingworx/MediaEntities/SubsystemRunning";
} else if (eventData.newValue.value==false) {
    me.Yun_State_Image = "/Thingworx/MediaEntities/DeleteIcon";
}
```

I also added Front_Motion code, so that I can trigger an email when front motion is detected.

Now I am adding Cam2OnOff control.
I added the following service code in TempAndHumidityThing.java.

```
    @ThingworxServiceDefinition(name = "controlCam2", description = "Turn IP Camera 2 On or Off")
    @ThingworxServiceResult(name = CommonPropertyNames.PROP_RESULT, description = "Result", baseType = "BOOLEAN")
    public boolean controlCam2(
            @ThingworxServiceParameter(name = "control", description = "on or off", baseType = "NUMBER") Double on_off)
            throws Exception {
        try {
            if (on_off == 1) {
                TWcommHandlerFactory.sendCommand("Cam2OnOff", "ON");
                return true;
            } else if (on_off == 2) {
                TWcommHandlerFactory.sendCommand("Cam2OnOff", "OFF");
                return true;
            } else {
                return false;
            }
        } catch (Exception e) {
            LOG.error("Service execution for controlCam2 caused exception" + e);
        }
        return false;
    }
```

Then, added a remote service.

I also created several panels (79~81) and grouped some Buttons and ValueDisplays into each panel. This move can be done by selecting them and click CUT icon and then select the panel to which they will be moved and click PASTE icon.
As can be seen I added Camera 1 state, Camera 2 state and control, Arduino Yun state, Dehumidifier state.

I wanted TW to send an email when front motion is detected.
But, if I use Alert type of Event in Subscriptions, then it seems like I must create an Alert in the property.

See other files.
Good Luck IoTing^^