Eventing

by SuperBonBon

1. UPNP# Events

You can basically receive 2 types of events messages from an UPNP# device :

- Hello or good bye messages sent by the UPNP# device when a device is joining or leaving the network.
- Messages sent to registered clients when a state variable changes

UPNPLib provides two classes to handle this kind of events.

2. Discovery Alive and Bye Bye events handling

You can use the *net.sbbi.upnp.DiscoveryAdvertisement* class to register objects coupled with objects implementing the net.sbbi.upnp.DiscoveryEventHandler interface. You can specify to listen for specific type of devices or you can use the NT "upnp:rootdevice" to listen to messages sent by every kind of devices on your network.

Here is a small example of a class that listen for all new devices spotted on the network :

```
package net.sbbi.upnp.samples;

import java.net.MalformedURLException;
import java.net.URL;
import java.util.HashMap;
import java.util.Map;

import net.sbbi.upnp.DiscoveryAdvertisement;
import net.sbbi.upnp.DiscoveryEventHandler;
import net.sbbi.upnp.devices.UPNPRootDevice;

public class MyDiscoveryEventsHandler
  implements DiscoveryEventHandler {

  private Map devices = new HashMap();

  public void eventSSDPAlive( String usn, String udn,
                              String nt, String maxAge,
                              URL location ) {
    System.out.println( "Device " + usn + " at " +
                        location + " of type " +
                        nt + " alive" );
    if ( devices.get( usn ) == null ) {
      // let's create the device
      UPNPRootDevice device = null;
      try {
        device = new UPNPRootDevice( location, maxAge );
        devices.put( usn, device );
        System.out.println( "Device " + usn + " added" );
        // and now let's play with the device..
      } catch ( MalformedURLException ex ) {
        // should never happen unless the UPNP devices
        // sends crappy URLs
      }
    }
  }

  public void eventSSDPByeBye( String usn, String udn,
                               String nt ) {
    if ( devices.get( usn ) != null ) {
      devices.remove( usn );
      System.out.println( "Device " + usn + " leaves" );
    }
  }

  public static void main( String[] args ) {
    // let's look for all root devices joining the network
    // ( "upnp:rootdevice" ) and set the events handler thread
    // as a non deamon thread so that the JVM does not stop
    // when the main static methods ends
    DiscoveryAdvertisement instance = DiscoveryAdvertisement.getInstance();
    MyDiscoveryEventsHandler handler = new MyDiscoveryEventsHandler();
    instance.setDaemon( false );
    instance.registerEvent( DiscoveryAdvertisement.EVENT_SSDP_ALIVE,
                            "upnp:rootdevice", handler );
  }
}
```

3. State variables events

You can also receive events every time a state variable changes on the UPNP device. You'll need to use the *net.sbbi.upnp.ServicesEventing* to register/unregister for specific device services state variable changes. The registered object need to implement the *net.sbbi.upnp.ServiceEventHandler* interface

Here is a small example to do it :

```
package net.sbbi.upnp.samples;

import java.io.IOException;
```

```java
import net.sbbi.upnp.Discovery;
import net.sbbi.upnp.ServicesEventing;
import net.sbbi.upnp.ServiceEventHandler;
import net.sbbi.upnp.devices.UPNPDevice;
import net.sbbi.upnp.devices.UPNPRootDevice;
import net.sbbi.upnp.services.UPNPService;

public class MyStateVariableEventsHandler
  implements ServiceEventHandler {

  public void handleStateVariableEvent( String varName, String newValue ) {
    System.out.println( "State variable " + varName +
                        " changed to " + newValue );
  }

  public static void main( String[] args ) {

    ServicesEventing instance = ServicesEventing.getInstance();
    MyStateVariableEventsHandler handler = new MyStateVariableEventsHandler();
    instance.setDaemon( false );
    // let's find a device
    UPNPRootDevice[] devices = null;
    try {
      devices = Discovery.discover();
    } catch ( IOException ex ) {
      ex.printStackTrace( System.err );
    }
    if ( devices != null ) {
      UPNPDevice firstDevice = (UPNPDevice)devices[0].getChildDevices()
                                                     .iterator().next();
      UPNPService firstService = (UPNPService)firstDevice.getServices()
                                                         .iterator().next();
      try {
        int duration = instance.register( firstService, handler, -1 );
        if ( duration != -1 ) {
          System.out.println( "State variable events registered for " + duration + " ms
        }
      } catch ( IOException ex ) {
        ex.printStackTrace( System.err );
        // comm error during registration with device such as timeoutException
      }
    } else {
      System.out.println( "Unable to find devices" );
    }
  }
}
```