LG U+ T-SDN

Network element Plug-in 개발 안내서

version 0.6.0

목 차

사전에 준비해야 할 것들

1. Operating System
   - Ubuntu 14.04 LTS

2. Java Development Kit
   - Oracle JDK Standard Edition 8 이상 버전 설치

3. Apache Maven
   - Apache Maven 3.3.9 이상 버전 설치

4. Eclipse IDE
   - Eclipse Neon 이상 버전 설치

5. 아래 내용들에 대한 최소한의 지식을 가지고 있어야 합니다.
   - Ubuntu OS 사용법
   - JDK를 이용한 Java Programming
   - Maven을 이용한 Java 빌드, 패키징 등...
   - Eclipse IDE 사용법

이 문서는 LG U+ Plug-in 개발 안내서로 위 Operating System, JDK, Maven, Eclipse IDE 설치에 필요한 안내를 하지 않음을 알립니다. Ubuntu, JDK, Maven, Eclipse IDE에 대한 정보는 인터넷 상에 광범위하게 소개 되어 있기 때문에 관련 내용을 찾는데 어려움은 없을 것 입니다.

문서 내용 중 버전 정보(예)tsdn-plugin-api-0.6.0.zip에서 0.6.0) 에 대해 `스크린 캡처 이미지 내용과 문서 텍스트의 내용이 다를 수 있지만 문맥 상 스크린 캡처 상의 버전은 문서의 버전과 동일함을 예상해야 합니다.

제공되는 리소스들

1. settings.xml
   - Maven settings.xml

2. LGU+ Plug-in Manager Opendaylight Components
   - Plug-in 개발용 Plug-in Manager 컴포넌트
   - tsdn-plugin-manager-base-0.6.0.jar
   - tsdn-plugin-manager4vendor-0.6.0.jar
   - lgup.plugin.manager.cfg

3. LG U+ Plug-in API Component source
   - tsdn-plugin-api-0.6.0.zip

4. Plug-in 구현 샘플
   - tsdn-plugin-sample-0.6.0.zip

Plug-in 개발 환경 구축 방법 1/4

1. settings.xml 설치

- $ cp settings.xml ~/.m2/settings.xml

2. Opendaylight 설치

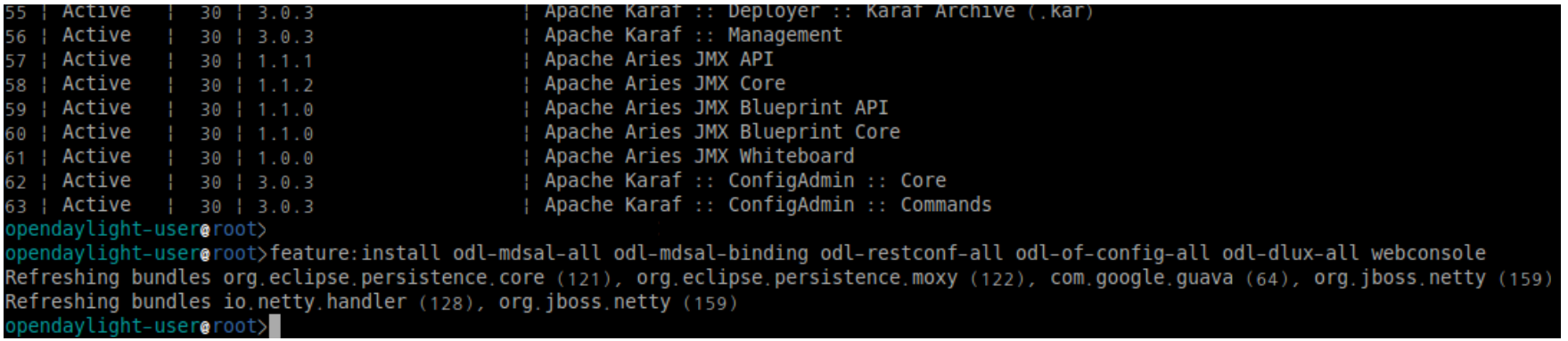
- $ mkdir ~/Applications
- $ cd ~/Applications
- $ wget https://nexus.opendaylight.org/content/repositories/opendaylight.release/org/opendaylight/integration/distribution-karaf/0.4.4-Beryllium-SR4/distribution-karaf-0.4.4-Beryllium-SR4.zip
- $ unzip distribution-karaf-0.4.4-Beryllium-SR4.zip

3. Opendyalight 실행

- $ cd ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4
- $ bin/karaf
- 콘솔 창이 아래와 같이 출력됨.

  opendaylight-user@root>
- Opendaylight feature들을 설치한다.

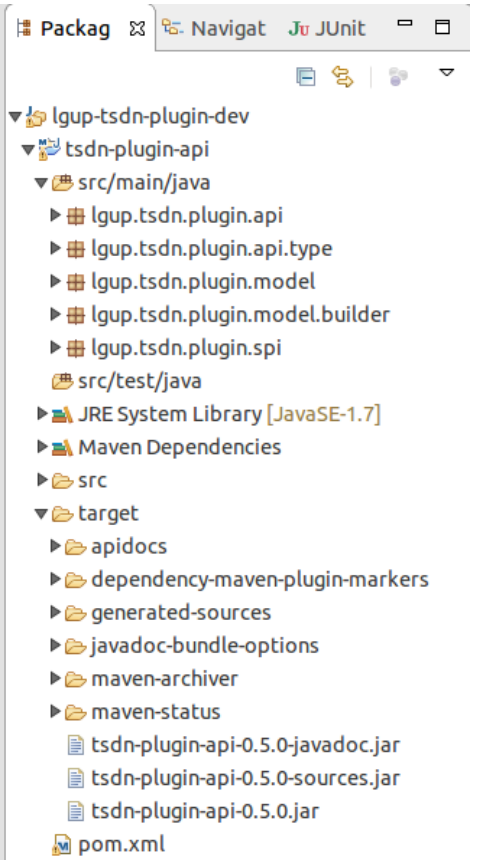
  opendaylight-user@root>feature:install odl-mdsal-all odl-mdsal-binding odl-restconf-all odl-of-config-all odl-dlux-all webconsole



Plug-in 개발 환경 구축 방법 2/4

4. Import Plug-in API Maven Project

- $ mkdir ~/lgup_tsdn_projects
- $ cp tsdn-plugin-api-0.6.0.zip ~/lgup_tsdn_projects
- $ unzip tsdn-plugin-api-0.6.0.zip
- Eclipse 실행
- File >> Import … >> Select >> Existing Maven Projects >> Root Directory
  ~/lgup_tsdn_projects/tsdn_plugin_api 디렉토리 선택



5. Build Plug-in API Project

- $ cd ~/lgup_tsdn_projects/tsdn_plugin_api
- $ mvn clean install -DskipTests -Dcheckstyle.skip=true
- $ ls -la target/

```
drwxrwxr-x 3 kty kty   4096 12월  8 14:04 apidocs
drwxrwxr-x 4 kty kty   4096 12월  8 14:04 classes
drwxrwxr-x 2 kty kty   4096 12월  8 14:04 dependency-maven-plugin-markers
drwxrwxr-x 3 kty kty   4096 12월  8 14:04 generated-sources
drwxrwxr-x 2 kty kty   4096 12월  8 14:04 javadoc-bundle-options
drwxrwxr-x 2 kty kty   4096 12월  8 14:04 maven-archiver
drwxrwxr-x 3 kty kty   4096 12월  8 14:04 maven-status
-rw-rw-r-- 1 kty kty 522498 12월  8 14:04 tsdn-plugin-api-0.5.0-javadoc.jar
-rw-rw-r-- 1 kty kty  43302 12월  8 14:04 tsdn-plugin-api-0.5.0-sources.jar
-rw-rw-r-- 1 kty kty 103755 12월  8 14:04 tsdn-plugin-api-0.5.0.jar
```

6. Deploy Plug-in API Component

- $ cp target/tsdn-plugin-api-0.6.0.jar ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4/deploy
- 설치 확인

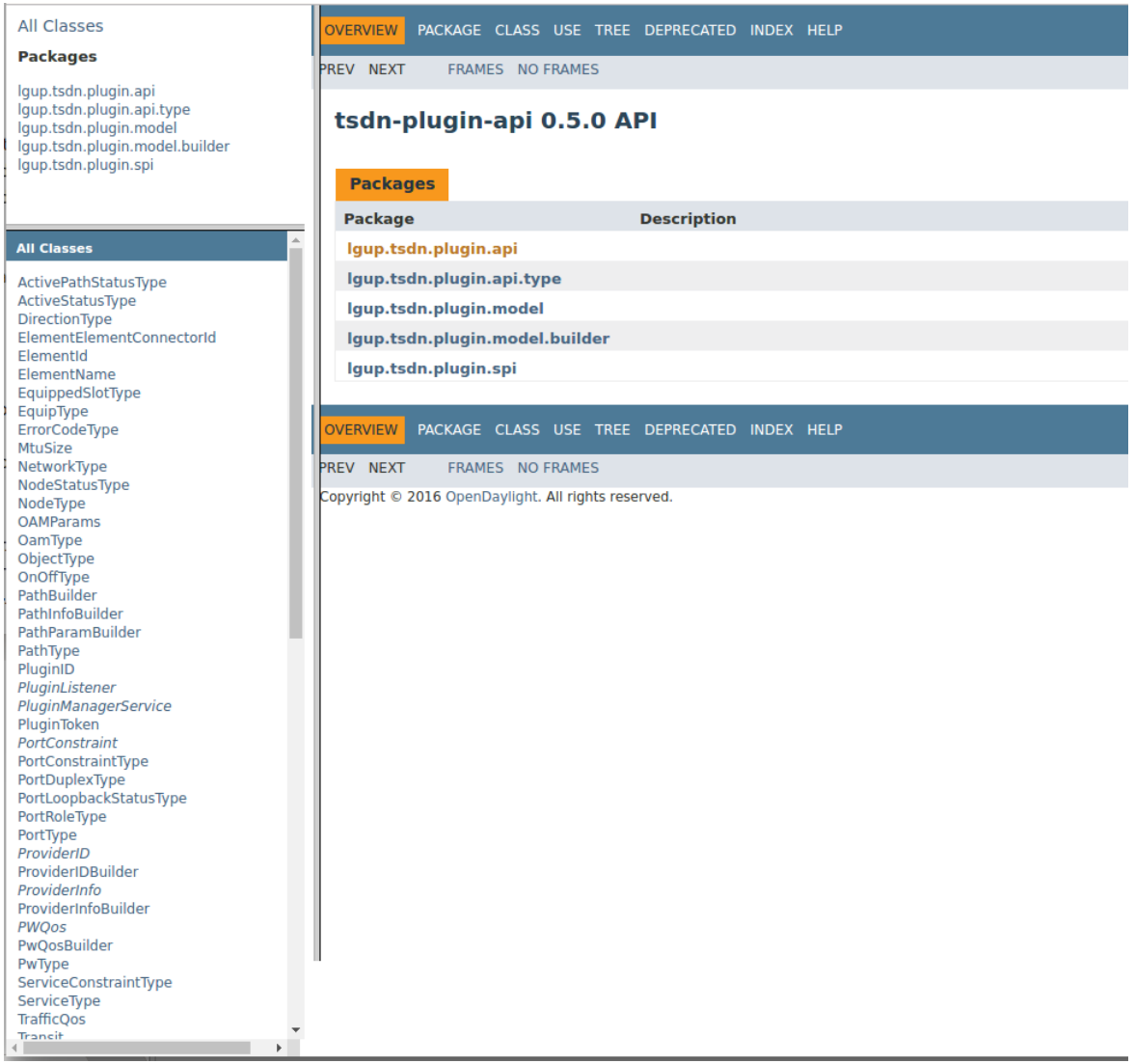
opendaylight-user@root> list

```
304 | Active   |  80 | 0.3.4.Beryllium-SR4               | dlux.yangvisualizer
305 | Resolved |  30 | 3.0.3                             | Apache Karaf :: Web Console :: Branding, Hosts: 306
306 | Active   |  30 | 3.0.3                             | Apache Karaf :: Web Console :: Console, Fragments: 305
307 | Active   |  30 | 3.0.3                             | Apache Karaf :: Web Console :: Instance Plugin
308 | Active   |  30 | 3.0.3                             | Apache Karaf :: Web Console :: Features Plugin
309 | Active   |  30 | 3.0.3                             | Apache Karaf :: Web Console :: Gogo Plugin
310 | Active   |  30 | 3.0.3                             | Apache Karaf :: Web Console :: HTTP Plugin
311 | Active   |  80 | 0.5.0                             | tsdn_plugin_api
opendaylight-user@root>
```

Plug-in 개발 환경 구축 방법 3/4

7. Show Plug-in API Java Documentations

- “~/lgup_tsdn_projects/tsdn_plugin_api/target/apidocs/index.html” 파일을 웹 브라우저로 불러오기.



Plug-in 개발 환경 구축 방법 4/4

8. Plug-in 개발용 Plug-in Manager 설치

- 아래 순서로 복사해야 함.
- $ cp tsdn-plugin-manager-base-0.6.0.jar ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4/deploy
- $ cp tsdn-plugin-manager4vendor-0.6.0.jar ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4/deploy
- $ cp lgup.plugin.manager.cfg ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4/etc
- 설치 확인

opendaylight-user@root> list

```
300 | Active   |  80 | 0.3.4.Beryllium-SR4              | dlux.node
301 | Active   |  80 | 0.3.4.Beryllium-SR4              | dlux.yangui
302 | Active   |  80 | 0.3.4.Beryllium-SR4              | dlux.common.yangutils
303 | Active   |  80 | 0.3.4.Beryllium-SR4              | dlux.common.sigmatopology
304 | Active   |  80 | 0.3.4.Beryllium-SR4              | dlux.yangvisualizer
305 | Resolved |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Branding, Hosts: 306
306 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Console, Fragments: 305
307 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Instance Plugin
308 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Features Plugin
309 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Gogo Plugin
310 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: HTTP Plugin
311 | Active   |  80 | 0.5.0                            | tsdn_plugin_api
312 | Active   |  80 | 0.5.0                            | tsdn_plugin_manager_base
313 | Active   |  80 | 0.5.0                            | tsdn_plugin_manager4vendor
opendaylight-user@root>
```

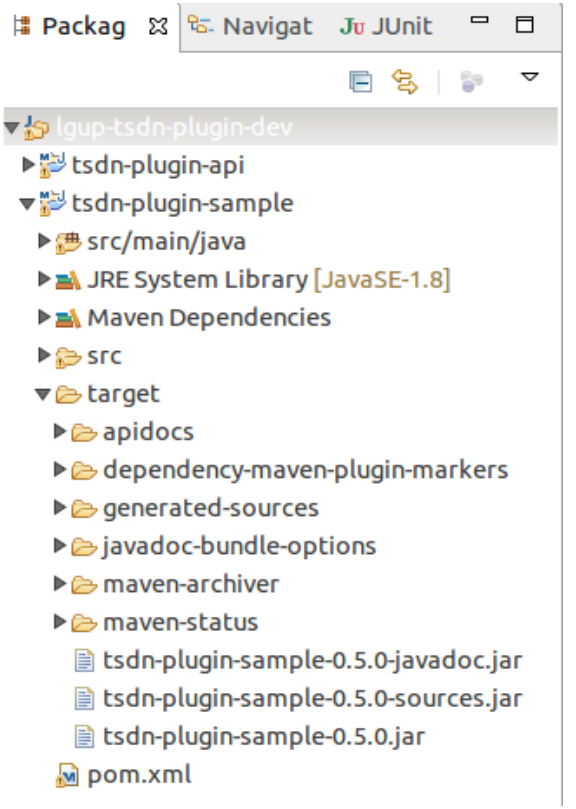
opendaylight-user@root> config:property-list -p lgup.plugin.manager

```
  plugin.1.id = 0
  plugin.1.provider.1.id = 201
  plugin.1.provider.1.password = bar
  plugin.count = 1
  felix.fileinstall.filename = file:/home/kty/Applications/distribution-karaf-0
  service.pid = lgup.plugin.manager
  plugin.1.provider.2.url = 192.168.1.21:5050
  plugin.1.provider.1.url = 192.168.1.20:5050
  plugin.1.provider.2.id = 202
  plugin.1.provider.count = 2
  plugin.1.provider.1.userName = foo
pendaylight-user@root>
```

Plug-in 구현 샘플 1/5

1. Import Plug-in Sample Maven Project

- $ cp tsdn-plugin-sample-0.6.0.zip ~/lgup_tsdn_projects
- $ unzip tsdn-plugin-sample-0.6.0.zip
- Eclipse File >> Import … >> Select >> Existing Maven Projects >> Root Directory ~/lgup_tsdn_projects/tsdn_plugin_sample 디렉토리 선택



2. Build Plug-in Sample Project

- $ cd ~/lgup_tsdn_projects/tsdn-plugin-sample
- $ mvn clean install -DskipTests -Dcheckstyle.skip=true
- $ ls -la target/

```
drwxrwxr-x 9 kty kty  4096 12월  8 14:17 .
drwxrwxr-x 5 kty kty  4096 12월  8 14:17 ..
drwxrwxr-x 3 kty kty  4096 12월  8 14:17 apidocs
drwxrwxr-x 4 kty kty  4096 12월  8 14:17 classes
drwxrwxr-x 2 kty kty  4096 12월  8 14:17 dependency-maven-plugin-markers
drwxrwxr-x 5 kty kty  4096 12월  8 14:17 generated-sources
drwxrwxr-x 2 kty kty  4096 12월  8 14:17 javadoc-bundle-options
drwxrwxr-x 2 kty kty  4096 12월  8 14:17 maven-archiver
drwxrwxr-x 3 kty kty  4096 12월  8 14:17 maven-status
-rw-rw-r-- 1 kty kty 24767 12월  8 14:17 tsdn-plugin-sample-0.5.0-javadoc.jar
-rw-rw-r-- 1 kty kty  5622 12월  8 14:17 tsdn-plugin-sample-0.5.0-sources.jar
-rw-rw-r-- 1 kty kty 20557 12월  8 14:17 tsdn-plugin-sample-0.5.0.jar
```

3. Deploy Plug-in Sample Component

- $ cp ~/lgup_tsdn_projects/tsdn_plugin_sample/target/tsdn-plugin-sample-0.6.0.jar ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4/deploy/
- opendaylight-user@root> list

```
304 | Active   |  80 | 0.3.4.Beryllium-SR4              | dlux.yangvisualizer
305 | Resolved |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Branding, Hosts: 306
306 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Console, Fragments: 305
307 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Instance Plugin
308 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Features Plugin
309 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: Gogo Plugin
310 | Active   |  30 | 3.0.3                            | Apache Karaf :: Web Console :: HTTP Plugin
311 | Active   |  80 | 0.5.0                            | tsdn_plugin_api
312 | Active   |  80 | 0.5.0                            | tsdn_plugin_manager_base
313 | Active   |  80 | 0.5.0                            | tsdn_plugin_manager4vendor
314 | Active   |  80 | 0.5.0                            | tsdn_plugin_sample
opendaylight-user@root>
```

4. Test Plug-in Sample Component

- opendaylight-user@root> lgup:discovery 0-1 devices
- $ tail -f ~/Applications/distribution-karaf-0.4.4-Beryllium-SR4/data/log/karaf.log

```
2016-12-08 15:14:22,949 | INFO  | l for user karaf | TsdnRPCImpl                      | 314 - lgup.tsdn.plugin.tsdn-plugin-sample - 0.5.0 | discov
eryDevices:ProviderID[id=0-1]
2016-12-08 15:14:23,449 | INFO  | pool-66-thread-4 | TsdnNotificationImpl             | 313 - lgup.tsdn.plugin.tsdn-plugin-manager4vendor - 0.5.0
| deviceConnected ProviderID[id=0-1], triggerType:Discovery, ElementId[[ni-0]0-1]
2016-12-08 15:14:23,949 | INFO  | pool-66-thread-4 | TsdnNotificationImpl             | 313 - lgup.tsdn.plugin.tsdn-plugin-manager4vendor - 0.5.0
| deviceConnected ProviderID[id=0-1], triggerType:Discovery, ElementId[[ni-1]0-1]
2016-12-08 15:14:24,449 | INFO  | pool-66-thread-4 | TsdnNotificationImpl             | 313 - lgup.tsdn.plugin.tsdn-plugin-manager4vendor - 0.5.0
| deviceConnected ProviderID[id=0-1], triggerType:Discovery, ElementId[[ni-2]0-1]
```

- opendaylight-user@root> stop 314

```
310 | Active   |  30 | 3.0.3                                | Apache Karaf :: Web Console :: HTTP Plugin
311 | Active   |  80 | 0.5.0                                | tsdn_plugin_api
312 | Active   |  80 | 0.5.0                                | tsdn_plugin_manager_base
313 | Active   |  80 | 0.5.0                                | tsdn_plugin_manager4vendor
314 | Resolved |  80 | 0.5.0                                | tsdn_plugin_sample
opendaylight-user@root>
```

- opendaylight-user@root> lgup:discovery 201 devices

```
Error executing command: There is no registered plugin. pluginID:PluginID[id=0]
opendaylight-user@root>
```

- opendaylight-user@root> start 314
- opendaylight-user@root> lgup:discovery 201 devices

Plug-in 구현 샘플 3/5

5. lgup.tsdn.plugin.spi.TsdnRPC.java (tsdn-plugin-api)

```
public interface TsdnRPC {
        /**
         * 이 플러그인의 ID를 리턴한다. 이 ID 값은 LG U+ Plug-in 관리자로 부터 이메일 또는 공식적인 방법으로 발급받는다.
         * @return 발급 받은 plugin id
         */
        PluginID getPluginID();

        /**
         * 이 메소드는 호출 후 바로 리턴하도록 구현해야 한다.(비동기 방식 구현 요구됨).<br/>
         * 장치를 발견하고 device-connected 이벤트로 발견한 장치 정보들을 알려준다.
         *
         * @param providerId       providerId
         * @throws UnKnownProviderException       UnKnownProviderException
         */
        void discoveryDevices(ProviderID providerId)throws UnKnownProviderException;

    …..
```

Plug-in 구현 샘플 4/5

6. lgup.tsdn.plugin.sample.TsdnRPCImpl.java

```
class TsdnRPCImpl implements TsdnRPC{
	private static final Logger LOG = LoggerFactory.getLogger(TsdnRPCImpl.class);

	private PluginID pluginId = PluginID.build(0); //ID값은 lgup.plugin.manager.cfg 파일의 plugin.1.id=0 속성과 함께 수정해야 함.
	private TsdnNotification notification;
	private Set<ProviderInfo> providers;
	private Set<ProviderID> providerIDs = new HashSet();

	private java.util.concurrent.ExecutorService exeService =
			java.util.concurrent.Executors.newFixedThreadPool(5);

	@Override
	public PluginID getPluginID() {
		return pluginId;
	}
	@Override
	public void discoveryDevices(ProviderID providerId) throws UnKnownProviderException{
		checkItsMyProvider(providerId);

		exeService.execute(new Runnable() {
			@Override
			public void run() {
				for( int i=0; i<3; i++ ){
					try {
						Thread.sleep(500);
					} catch (InterruptedException e) { }
					notification.deviceConnected(providerId, TriggerType.Discovery,
							ElementId.build("[ni-"+i+"]"+providerId.id()));
				}
			}
		});

		LOG.info("discoveryDevices:"+providerId);
	}

  ……
```

Plug-in 구현 샘플 5/5

7. lgup.tsdn.plugin.sample.PluginActivator.java

```
public class PluginActivator extends AbstractBrokerAwareActivator implements BindingAwareProvider {
	private static final Logger LOG = LoggerFactory.getLogger(PluginActivator.class);

	private ServiceRegistration<TsdnRPC> rpcRegi;
	private ServiceReference<PluginManagerService> pluginManagerServiceRef;
	private TsdnRPCImpl rpcImpl;

	private PluginToken pluginToken;
	private File rootDir;

	@Override
	public void startImpl(BundleContext context) {
		rpcImpl = new TsdnRPCImpl();

		rpcRegi = context.registerService(TsdnRPC.class, rpcImpl, new Hashtable<String,String>());

		pluginManagerServiceRef = context.getServiceReference(PluginManagerService.class);
		PluginManagerService service = context.getService(pluginManagerServiceRef);

		pluginToken = service.register(rpcImpl.getPluginID(), new PluginListener() {
			@Override
			public void start(TsdnNotification notification, Set<ProviderInfo> providers) {
				rpcImpl.start(notification, providers);
			}
			@Override
			public void shutdown() {
				rpcImpl.shutdown();
			}
			@Override
			public void providerRemoved(ProviderInfo provider) {
				rpcImpl.removed(provider);
			}
			@Override
			public void providerAdded(ProviderInfo provider) {
				rpcImpl.added(provider);
			}
		});
		rootDir = service.rootDir(pluginToken);

		LOG.info("startImpl");
	}
...
```

Sample Data Generator 사용법 1/3

1. tsdn-plugin-sample project

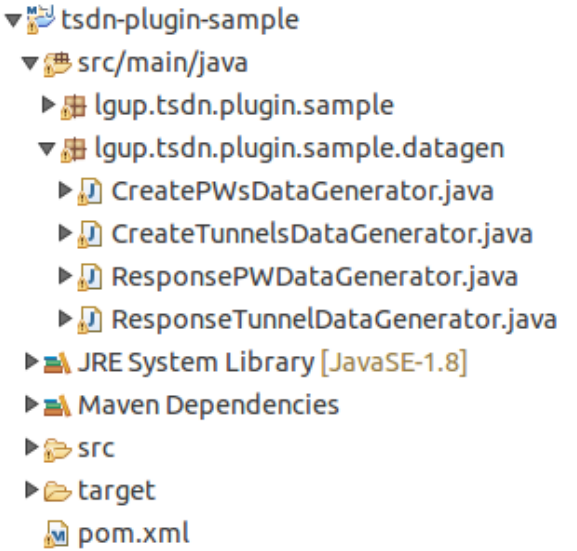
- 앞선 문서의 내용 중 Plug-in 구현 Sample Project의 lgup.tsdn.plugin.sample.dataget 패키지에 Generator 구현 클래스 파일들이 존재함.

- 이 클래스들은 자바 객체를 생성하고 해당 객체를 json 문자열로 출력하도록 구현됨.

- 플러그인 개발자가 필요에 따라 이 코드들을 수정해 활용할 수 있음.

- CreatePWsDataGenerator.java : 슈도와이어 생성 시 필요한 파라미터 객체들을 생성하는 코드가 구현됨.

- CreateTunnelsDataGenerator.java : 터널 생성 시 필요한 파라미터 객체들을 생성하는 코드가 구현됨.



- ResponsePWDataGenerator.java : 슈도와이어 정보를 Notification 하기위해 생성해야 하는 객체들을 생성하는 코드가 구현됨.

- ResponseTunnelDataGenerator.java : 터널정보를 Notification 하기위해 생성해야 하는 객체들을 생성하는 코드가 구현됨.

- 위 클래스들은 java의 “public static void main(String[] args)”메소드를 가진 자바 어플리케이션 코드이기 때문에 Eclipse IDE에서 해당 자바 파일을 선택 후 Java Application으로 실행할 수 있음.

Sample Data Generator 사용법 2/3

1. CreatePWsDataGenerator.java 예시

```
public class CreatePWsDataGenerator {

	public static void main(String[] args) throws JsonProcessingException{
	List<TsdnPWParams> objList = new ArrayList();

	TsdnPWParamsImpl testObj = new TsdnPWParamsImpl();
	testObj.setServiceType(ServiceType.ELine);
	testObj.setPwDesc("pwDesc");
	testObj.setCbs(22);
	testObj.setCir(23);
	testObj.setPbs(24);
	testObj.setPir(25);
	testObj.setAcId("acId-1");
	List<PortConstraint> portConstraintList = new ArrayList();
	List<Long> vlanIdList = new ArrayList();
	vlanIdList.add(Long.valueOf(100));
	vlanIdList.add(Long.valueOf(101));
	portConstraintList.add( new PortConstraintBuilder()
			.setLlcfFlag(Boolean.FALSE)
			.setPortConstraintType(PortConstraintType.PortConstraintOuterVid)
			.setVlanIdList(vlanIdList)
			.build() );
...
	testObj.setSource(new TsdnPWParamsSourceBuilder()
			.setSrcNodeTpId(ElementElementConnectorId.build(ElementId.build("srcNodeId"),
					ElementId.build("srcTpId")))
			.setWestPwId(ElementId.build("pw-22"))
			.setWestTunnelId(ElementId.build("t-22"))
			.setInLabel("inLabel")
			.setOutLabel("outLabel")
			.setPortConstraintList(portConstraintList)
			.build());
	tunnelIdList = new ArrayList();
	tunnelIdList.add(ElementId.build("t-121"));
	tunnelIdList.add(ElementId.build("t-122"));
	testObj.setTunnelIdList(tunnelIdList);
	objList.add(testObj);

	String json = new ObjectMapper().enable(SerializationFeature.INDENT_OUTPUT).writeValueAsString(objList);

	System.out.println(json);
	}
}
```

Sample Data Generator 사용법 3/3

1. CreatePWsDataGenerator가 출력한 json 예시


```
[ {
  "cir" : 23,
  "pir" : 25,
  "cbs" : 22,
  "pbs" : 24,
  "oamParams" : {
    "localId" : "localId",
    "remoteId" : "remoteId",
    "groupName" : "groupName",
    "messageInvervalTime" : 34567
  },
  "acId" : "acId-1",
  "direction" : "Uni",
  "source" : {
    "rootNode" : true,
    "portConstraintList" : [ {
      "portConstraintType" : "PortConstraintOuterVid",
      "llcfFlag" : false,
      "vlanIdList" : [ 100, 101 ]
    } ],


...

      "id" : "t-21"
    }
  },
  "tunnelIdList" : [ {
    "id" : "t-121"
  }, {
    "id" : "t-122"
  } ],
  "serviceType" : "ELine",
  "pwDesc" : "pwDesc",
  "pwname" : {
    "name" : "pw-name-2"
  }
} ]
```

Test Command 사용법 1/3

1. Karaf shell command

- karaf shell 프롬프트 에서 키보드의 tab 키를 클릭하면 현재 프롬프트 상에서 사용할 수 있는 명령 목록을 조회할 수 있다.
- 아래와 같이 키보드의 tab 키를 클릭 후 키보드 y 키를 클릭한다.

```
opendaylight-user@root>Display all 319 possibilities? (y or n)
*:bundle                        *:config                        *:dev                           *:exit
*:feature                       *:help                          *:http                          *:instance
*:jaas                          *:kar                           *:lgup                          *:log
*:odl                           *:package                       *:region                        *:service
*:shell                         *:ssh                           *:system                        *:web
addbundle                       addfilter                       addregion                       alias
bundle                          bundle:capabilities             bundle:classes                  bundle:diag
bundle:dynamic-import           bundle:find-class               bundle:headers                  bundle:id
bundle:info                     bundle:install                  bundle:list                     bundle:load-test
bundle:refresh                  bundle:requirements             bundle:resolve                  bundle:restart
bundle:services                 bundle:start                    bundle:start-level              bundle:stop
bundle:tree-show                bundle:uninstall                bundle:update                   bundle:watch
cancel                          capabilities                    cat                             cl
classes                         clear                           clone                           completion
config                          config:cancel                   config:delete                   config:edit
config:list                     config:property-append          config:property-delete          config:property-lis
config:property-set             config:update                   connect                         create
date                            delete                          destroy                         dev
dev:dump-create                 diag                            discovery                       display
dump-create                     dynamic-import                  each                            echo
edit                            env                             exception-display               exec
exit                            exports                         feature                         feature:info
feature:install                 feature:list                    feature:repo-add                feature:repo-list
feature:repo-refresh            feature:repo-remove             feature:uninstall               feature:version-lis
find-class                      framework                       get                             grep
```

2. Test command 목록

- opendaylight-user@root> lgup <tab key 클릭>

```
opendaylight-user@root>lgup:
lgup:create           lgup:delete           lgup:discovery        lgup:request-info
opendaylight-user@root>lgup:
```

Test Command 사용법 2/3

3. discovery command

- TsdnRPC의 discoveryDevices, discoveryTunnels, discoveryPWs 메소드를 호출하는 명령.
- opendaylight-user@root> lgup:discovery --help

```
DESCRIPTION
        lgup:discovery

        Discovery commands

SYNTAX
        lgup:discovery providerId command

ARGUMENTS
        providerId
                providerId
        command
                The command argument : devices | tunnels | pws
```

- 위 내용 처럼 argument를 2개를 주고 실행함. 첫째 argument는 providerId, 둘째는 devices 또는 tunnels 이거나 pws를 입력할 수 있음. 각각 discoveryDevices, discoveryTunnels, discoveryPWs 메소드 호출 요청을 의미함.

4. request-info command

- TsdnRPC의 requestDeviceInfo, requestTunnelInfo, requestPWInfo 메소드를 호출하는 명령.
- opendaylight-user@root> lgup:request-info --help

```
DESCRIPTION
        lgup:request-info

        request info commands

SYNTAX
        lgup:request-info providerId command elementId

ARGUMENTS
        providerId
                providerId
        command
                The command argument : device | tunnel | pw
        elementId
                elementId
```

- command는 각각 TsdnRPC의 requestDeviceInfo, requestTunnelInfo, requestPWInfo 메소드 호출 요청을 의미함.
- elementId는 요청하는 command에 따라 nodeElementId 또는 tunnelElementId 이거나 pwElementId를 입력할 수 있음.

Test Command 사용법 3/3

5. create command

- TsdnRPC의 createTunnel, createPW 메소드를 호출하는 명령.
- opendaylight-user@root> lgup:create --help

```
DESCRIPTION
        lgup:create

        Create commands

SYNTAX
        lgup:create providerId command jsonFile

ARGUMENTS
        providerId
                providerId
        command
                The command argument : tunnels | pws
        jsonFile
                json file path
```

- command는 각각 TsdnRPC의 createTunnel, createPW 메소드 호출 요청을 의미함.
- jsonFile은 로컬 파일 시스템의 json file 경로를 의미함.
  (앞선 문서의 Sample Data Generator 설명 중 CreatePWsDataGenerator 를 이용해 출력 저장한 json을 이용.)

5. delete command

- TsdnRPC의 deleteTunnel, deletePW 메소드를 호출하는 명령.
- [illegible]

```
DESCRIPTION
        lgup:delete

        Delete commands

SYNTAX
        lgup:delete providerId command idList

ARGUMENTS
        providerId
                providerId
        command
                The command argument : tunnels | pws
        idList
                id list. comma separated. cf) id1,id2,id3
```

- command는 각각 TsdnRPC의 deleteTunnel, deletePW 메소드 호출 요청을 의미함.
- idList는 요청하는 command에 따라 tunnelElementId 이거나 pwElementId의 목록을 콤마(“,”)로 구분하여 입력할 수 있음.

Plug-in SDK 변경 사항

1. Test command 추가
   - create, delete

2. Sample Data Generator 추가

3. Plug-in API TsdnRPC 변경
   - discoveryLinks 삭제 → 기존 discoveryDevices의 호출로 의미를 포함 시킴
   - device-disconnected, link-detected, link-vanished Notification 호출 상황을 discoveryDevices 처리에 포함 시킴.
   - tunnel-vanished Notification 호출 상황을 discoveryTunnels 처리에 포함 시킴.
   - pw-vanished Notification 호출 상황을 discoveryPWs 처리에 포함 시킴.

4. Plug-in API TsdnNotification 변경
   - tunnelCreated 메소드 추가
   - pwCreated 메소드 추가
   - tunnelChanged 메소드 추가
   - pwChanged 메소드에 TriggerType 추가
   - TriggerType.Discovyer Enum을 TriggerType.RequestDiscovery로 변경
   - TriggerType에 RequestChange, RequestDelete enum 추가. 각각 변경 또는 삭제 요청에 의해 발생한 경우 사용함.

Plug-in SDK 추가 예정 내용

현재 없음