

# Syslog 서버 개발 예제

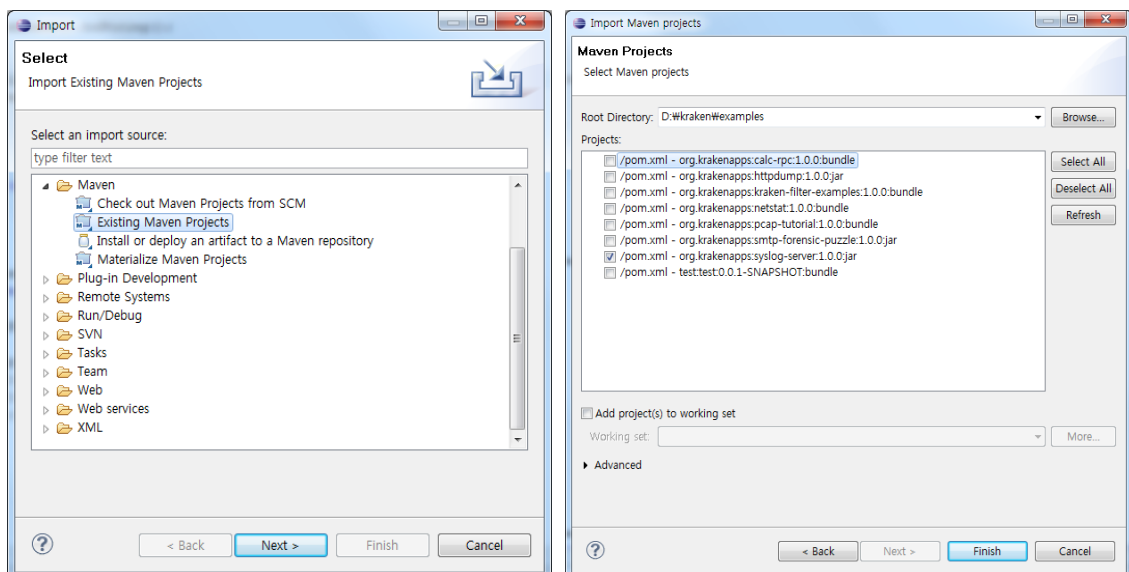
## 1. New project

new.bat 파일을 d:\kraken\examples에 복사하고 명령 프롬프트에서 다음과 같이 실행

```
new [artifact id] [package]
..or..
mvn archetype:generate
-DarchetypeGroupId=org.krakenapps
-DarchetypeArtifactId=kraken-template
-DarchetypeVersion=1.0.1
-DgroupId=org.krakenapps
-DartifactId=syslog-server
-Dversion=1.0.0 -Dpackage= org.krakenapps.examples.syslogserver
```

```
D:\kraken\examples>new syslog-server org.krakenapps.examples
D:\kraken\examples>mvn archetype:generate -DarchetypeGroupId=org.krakenapps -DarchetypeArtifactId=
[INFO] Scanning for projects...
[INFO]
[INFO] -----
[INFO] Building Maven Stub Project (No POM) 1
[INFO] -----
[INFO]
[INFO] >>> maven-archetype-plugin:2.0:generate (default-cli) @ standalone-pom >>>
[INFO]
[INFO] <<< maven-archetype-plugin:2.0:generate (default-cli) @ standalone-pom <<<
[INFO]
[INFO] --- maven-archetype-plugin:2.0:generate (default-cli) @ standalone-pom ---
[INFO] Generating project in Interactive mode
[INFO] Archetype repository missing. Using the one from [org.krakenapps:kraken-template:1.0.1 -> ]
[INFO] Using property: groupId = org.krakenapps
[INFO] Using property: artifactId = syslog-server
[INFO] Using property: version = 1.0.0
[INFO] Using property: package = org.krakenapps.examples
Confirm properties configuration:
groupId: org.krakenapps
artifactId: syslog-server
version: 1.0.0
package: org.krakenapps.examples
Y: : y
```

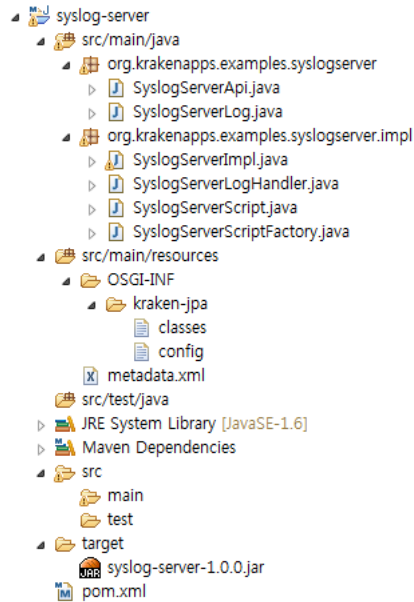
## 2. eclipse에서 File > import



## Sources 설명

### 1. Workspace

모델 클래스와 세부사항을 구현하고 있는 구현 클래스를 각각 다른 패키지로 구분한다.



### 2. Source details

#### ➤ org.krakenapps.examples.syslogserver

1) SyslogServerApi: 각 Script 메소드에서 호출될 메소드에 대한 인터페이스

```
public interface SyslogServerApi {  
    List<SyslogServerLog> getSyslogList(Date d);  
}
```

2) SyslogServerLog: DB 스키마에 매핑될 JPA 엔티티 클래스

```
@Entity  
@Table(name = "syslog_server_logs")  
public class SyslogServerLog {  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    private int id;  
  
    @Column(name = "remote_ip", nullable = false)  
    private String remoteIp;  
  
    @Column(name = "facility", nullable = false)  
    private int facility;  
  
    @Column(name = "severity", nullable = false)  
    private int severity;  
  
    @Column(name = "message", nullable = false)  
    private String message;  
  
    @Column(name = "created_at", nullable = false)  
    private Date created;
```

@Entity: JPA 엔티티를 정의한다. 또한 OSGI-INF/kraken-jpa/classes 설정 파일에 클래스 이름을 적어줘야 한다.

@Table: 테이블 이름을 지정할 수 있다. 여기에서는 syslog\_server\_logs로 한다.

@Id, @GeneratedValue: 각 row가 생성될 때마다 Id를 자동으로 부여한다.

@Column: 자동 설정되는 것 외에 직접 컬럼 이름을 지정하거나, 컬럼 속성을 지정하려고 할 때 사용한다. NULL 허용여부는 nullable 속성으로 지정할 수 있다.

➤ org.krakenapps.examples.syslogserver.impl

### 1) SyslogServerImpl : SyslogServerApi 인터페이스의 구현

```
@Component(name = "syslog-server-api")
@Provides
@JpaConfig(factory = "syslog-server")
public class SyslogServerImpl implements SyslogServerApi {

    @Requires
    private ThreadLocalEntityManagerService entityManagerService;

    @Transactional
    @Override
    public List<SyslogServerLog> getSyslogList(Date d) {
        EntityManager em = entityManagerService.getEntityManager();
        Calendar c = Calendar.getInstance();
        c.setTime(d);
        c.add(Calendar.DAY_OF_MONTH, 1);
        Date begin = d;
        Date end = c.getTime();

        String query = "from SyslogServerLog s where s.created >= ? and s.created < ? order by s.id desc ";
        return em.createQuery(query).setParameter(1, begin)
            .setParameter(2, end).getResultList();
    }
}
```

@Component: syslog-server-api 이름으로 iPOJO 컴포넌트를 정의한다.

metadata.xml 파일에서 컴포넌트 인스턴스를 생성할 때 동일한 이름을 쓰게 된다. 만약 name을 지정하지 않으면 클래스 이름으로 등록된다.

@JpaConfig: syslog-server라는 이름으로 등록된 JPA Entity Manger Factory를 사용한다는 의미이다. 만약 해당 이름으로 JPA Entity Manager Factory가 등록되지 않으면 이 컴포넌트 인스턴스는 무효화되어 동작하지 않는다.

@Provides: 이 컴포넌트가 구현하는 인터페이스를 OSGi 서비스로 등록한다.

@Provides(specifications = { Runnable.class }) 와 같이 쓰면 specifications 에 명시된 인터페이스만 노출한다. @Provides만 쓰면 해당 컴포넌트가 구현하는 모든 인터페이스가 OSGi 서비스로 노출된다.

@Transactional: 선언적 트랜잭션으로, 메소드가 시작될 때 트랜잭션이 시작되고,

메소드를 빠져나갈 때 자동으로 커밋된다. 만약 도중에 예외가 발생하면 롤백된다. 만약 이 어노테이션 없이 EntityManager를 사용한다면 쿼리를 요청할 때 예외가 발생한다.

2) SyslogServerScriptFactory: ScriptFactory 인터페이스를 구현하고 Script 객체를 생성한다.

```
@Component(name = "syslog-server-factory")
@Provides
public class SyslogServerScriptFactory implements ScriptFactory{
    @SuppressWarnings("unused")
    @ServiceProperty(name = "alias", value = "syslog-server")
    private String alias;

    @Requires
    private SyslogServerApi syslogServerApi;

    @Override
    public Script createScript() {
        SyslogServerScript s = new SyslogServerScript();
        s.setSyslogServerApi(syslogServerApi);
        return s;
    }
}
```

@ServiceProperty(name = "alias", value = "syslog-server")

alias라는 이름으로 서비스 프로퍼티를 정의한다. alias는 명령어의 접두어로 사용된다. (예: syslog-server.list)

3) SyslogServerScript : Script 인터페이스를 구현한다. 각 메소드 이름은 명령어로 사용된다.

```

public class SyslogServerScript implements Script {
    private ScriptContext context;
    private SyslogServerApi api;

    public SyslogServerScript() {
    }

    @Override
    public void setScriptContext(ScriptContext context) {
        this.context = context;
    }

    public void setSyslogServerApi(SyslogServerApi api) {
        this.api = api;
    }

    @ScriptUsage(description = "show list of syslog ",arguments =
        @ScriptArgument(name = "date", description = "20101210"))
    public void list(String[] args) {
        context.println("Syslog List");
        context.println("-----");

        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyyMMdd");
        Date d = null;

        try {
            d = dateFormat.parse(args[0]);
        } catch (ParseException e) {
            context.println("wrong date format.");
            return;
        }

        for (SyslogServerLog slog : api.getSyslogList(d)) {
            context.println(slog.toString());
        }
    }
}

```

@ScriptUsage: 필수적으로 사용해야 하는 어노테이션은 아니지만 명령어의 사용 방법을 설명하거나, 매개변수의 타입이나 개수를 강제하려고 할 때 사용한다.

@ScriptArgument: Script와 함께 입력 받을 매개변수의 명세를 적어준다. 만약 arguments에 정의된 필수 매개변수 개수만큼 입력되지 않으면 스크립트가 실행되지 않고 description에 기술된 사용 방법이 출력된다.

4) SyslogServerLogHandler: syslog로 들어오는 메시지를 처리한다.

```

@Validate
public void start() {
    registry.addEventListener(this);
    logger.info("syslog-server started.-----");

    if (registry.contains("syslog-server")) {
        SyslogServer server = registry.getServer("syslog-server");
        server.addListener(this);
    }
}

```

iPOJO 컴포넌트로 LogHandler가 생성되고 유효화 되면서 start() 메소드가 호출된다. 위에서는 SyslogServerRegistry에 LogHandler를 EventListener로 등록하고,

만약 SyslogServerRegistry에 syslog-server라는 이름의 서버가 있으면, LogHandler가 Syslog를 수신할 수 있도록 자기 자신을 SyslogServer에 콜백으로 등록한다.

```
@Invalidate
public void stop() {
    if (registry == null)
        return;

    registry.removeEventListener(this);

    SyslogServer server = registry.getServer("syslog-server");
    if (server == null)
        return;

    server.removeListener(this);
}
```

컴포넌트가 무효화 될 때 stop() 메소드가 호출된다. 레지스트리와 Syslog 서버에 등록했던 콜백을 해제한다.

```
@Override
public void syslogServerAdded(String name, SyslogServer server) {
    if (name.equals("syslog-server")) {
        server.addListener(this);
    }
}

@Override
public void syslogServerRemoved(String name, SyslogServer server) {
    if (name.equals("syslog-server")) {
        server.removeListener(this);
    }
}
```

레지스트리에서 syslog-server라는 이름의 서버가 등록/해제되었다는 이벤트가 전달 되었을 때 Syslog 수신에 사용되는 콜백을 등록하거나 해제한다.

```

@Override
public void onReceive(Syslog syslog) {
    logger.info(
        "-----syslog-server: [{}] log received [{}]",
        syslog.getRemoteAddress(), syslog.getMessage());
    try {
        handle(syslog);
    } catch (Exception e) {
        logger.error("syslog-server: cannot receive log", e);
    }
}

@Transactional
private void handle(Syslog syslog) {
    EntityManager em = entityManagerService.getEntityManager();

    SyslogServerLog slog = new SyslogServerLog();
    slog.setRemoteIp(syslog.getRemoteAddress().toString());
    slog.setFacility(syslog.getFacility());
    slog.setSeverity(syslog.getSeverity());
    slog.setMessage(syslog.getMessage());
    slog.setCreated(new Date());

    em.persist(slog);
}

```

syslog를 받아 파싱하고 저장하는 메소드이다. handle() 메소드는 Syslog를 SyslogServerLog 모델에 맞게 변환해서 DB에 저장한다.

@Validate: 번들이 시작되면서 해당 클래스가 iPOJO의 인스턴스로 정상적으로 등록되면, Validate 어노테이션이 붙어있는 메소드가 호출된다.

@Invalidate: 번들을 정지하면 iPOJO에 등록되어 있던 인스턴스가 무효화 되면서 해당 어노테이션이 붙어있는 메소드가 호출된다.

## 설정파일 및 빌드

### 1. OSGI 설정 파일

src/main/resources/OSGI-INF/kraken-jpa 폴더를 만들고 config, classes 파일 생성

- classes: Entity Class 추가

org.krakenapps.examples.syslogserver.SyslogServerLog
--

- config: connection 정보 등 추가

```
hibernate.connection.driver_class = com.mysql.jdbc.Driver
hibernate.connection.url =
jdbc:mysql://localhost/syslog_server??useUnicode=true&characterEncoding=utf8
hibernate.connection.username = syslog_server
hibernate.connection.password = password
hibernate.dialect = org.hibernate.dialect.MySQLDialect
hibernate.current_session_context_class = thread
hibernate.cache.provider_class =
org.hibernate.cache.NoCacheProvider
hibernate.show_sql = false
hibernate.hbm2ddl.auto = update
hibernate.autoReconnect = true
hibernate.autoReconnectForPools = true
hibernate.is-connection-validation-required = true
```

2. pom.xml 파일 수정

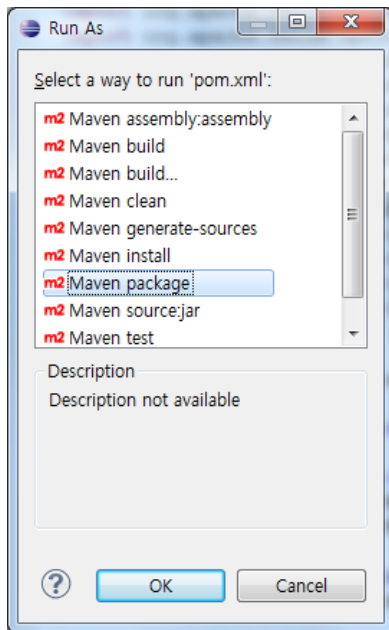
```
<Import-Package>
    javassist.util.proxy,
    org.hibernate.proxy,
    org.hibernate.exception,
    org.hibernate,
    org.hibernate.dialect,
    javax.persistence,*
</Import-Package>
<Export-Package>
    org.krakenapps.examples.syslogserver
</Export-Package>
<Private-Package>
    org.krakenapps.examples.syslogserver.impl
</Private-Package>
```

3. metadata.xml 설정

```
<ipojo
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="org.apache.felix.ipojo
http://felix.apache.org/ipojo/schemas/CURRENT/core.xsd"
  xmlns="org.apache.felix.ipojo">
  <instance component="syslog-server-api"/>
  <instance component="syslog-server-factory"/>
  <instance component="syslog-server-log-handler"/>
</ipojo>
```

#### 4. Build

Ctrl + F11 > Maven Package



빌드가 성공하면 target 폴더에 syslog-server-1.0.0.jar 파일이 생성된다.

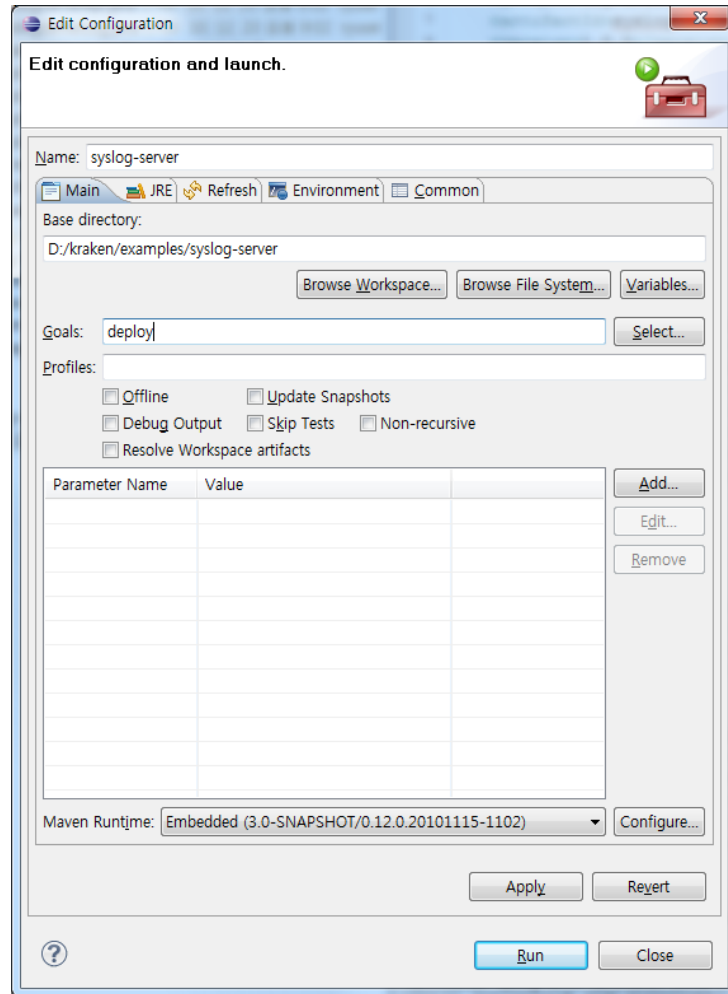
#### 5. 배포하기

##### 1) pom.xml 수정

```
<distributionManagement>  
<repository>  
<id>local</id>  
<url>file:///d:/kraken/examples/syslog-server/repo/</url>  
</repository>  
</distributionManagement>
```

pom.xml 설정에서 배포 파일을 저장할 폴더를 배포 url로 지정한다.

Ctrl+F11을 누르고 "mvn build..."를 선택하면 Edit Configuration 대화상자가 출력된다. 여기에서 Goals에 deploy를 입력하고 Run을 클릭하면, url로 지정된 로컬 경로 하위에 groupId/artifactId 형식으로 org/krakenapps/syslog-server 디렉터리가 생성되고 배포 파일이 저장된다.



- 또는 명령 프롬프트에서 다음과 같이 실행

```
mvn deploy:deploy-file
-Durl=file:///d:/kraken/examples/syslog-server/repo -Drepositoryid=test
-Dfile=target/syslog-server-1.0.0.jar -DgroupId=org.krakenapps
-DartifactId=syslog-server -Dversion=1.0.0 -Dpackaging=jar
```

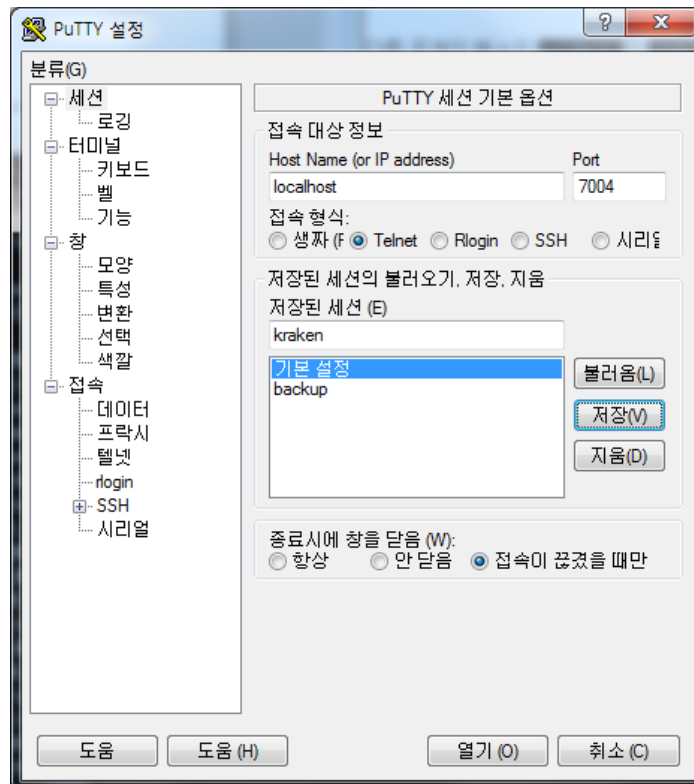
## Kraken 실행

### 1. kraken-core 실행

```
java -jar C:\kraken\kraken-core\target\kraken-core-1.7.0-package.jar
```

### 2. kraken에 접속

Kraken이 시작되면 putty를 열고 localhost:7004, telnet으로 접속



## MySQL DB 및 User 생성

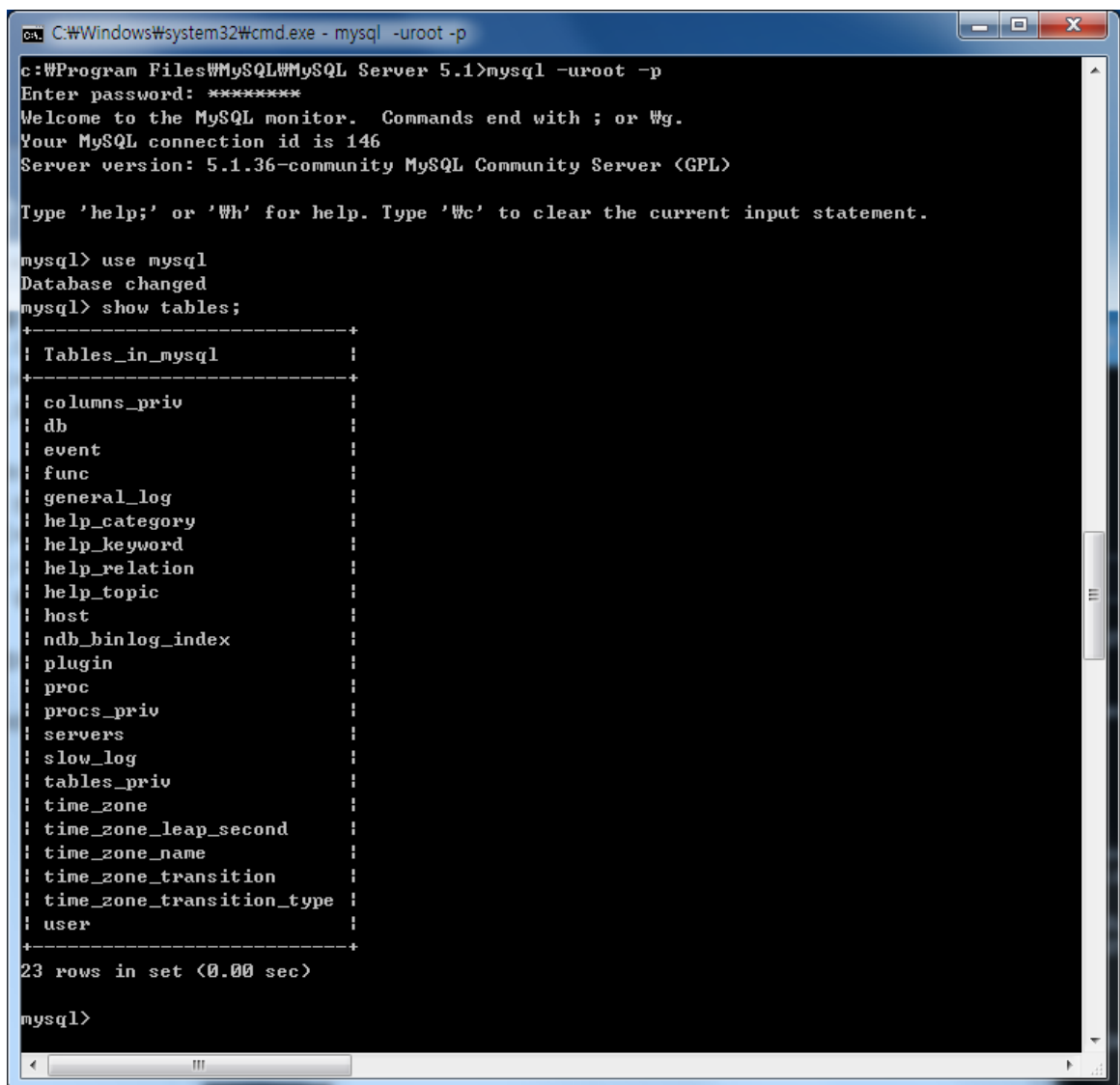
### 1. mysql 접속

- root로 로그인

```
mysql -uroot -p[password]
```

- use [Database name]

```
use mysql;  
show tables;
```



The screenshot shows a Windows command prompt window titled "C:\Windows\system32\cmd.exe - mysql -uroot -p". The prompt is at "c:\Program Files\MySQL\MySQL Server 5.1>mysql -uroot -p". The user enters a password (masked with asterisks). The MySQL monitor displays a welcome message: "Welcome to the MySQL monitor. Commands end with ; or \g. Your MySQL connection id is 146. Server version: 5.1.36-community MySQL Community Server <GPL>". The user enters "mysql> use mysql", and the prompt changes to "mysql>". The user then enters "mysql> show tables;", and the output is a list of 23 tables in the mysql database, including tables\_in\_mysql, columns\_priv, db, event, func, general\_log, help\_category, help\_keyword, help\_relation, help\_topic, host, ndb\_binlog\_index, plugin, proc, procs\_priv, servers, slow\_log, tables\_priv, time\_zone, time\_zone\_leap\_second, time\_zone\_name, time\_zone\_transition, time\_zone\_transition\_type, and user. The output ends with "23 rows in set (0.00 sec)" and the prompt returns to "mysql>".

```
c:\Program Files\MySQL\MySQL Server 5.1>mysql -uroot -p  
Enter password: *****  
Welcome to the MySQL monitor.  Commands end with ; or \g.  
Your MySQL connection id is 146  
Server version: 5.1.36-community MySQL Community Server <GPL>  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
mysql> use mysql  
Database changed  
mysql> show tables;  
+-----+  
| Tables_in_mysql |  
+-----+  
| columns_priv    |  
| db              |  
| event          |  
| func           |  
| general_log     |  
| help_category  |  
| help_keyword   |  
| help_relation  |  
| help_topic     |  
| host           |  
| ndb_binlog_index |  
| plugin         |  
| proc          |  
| procs_priv     |  
| servers        |  
| slow_log       |  
| tables_priv    |  
| time_zone      |  
| time_zone_leap_second |  
| time_zone_name |  
| time_zone_transition |  
| time_zone_transition_type |  
| user          |  
+-----+  
23 rows in set (0.00 sec)  
  
mysql>
```

### 2. 새 사용자 계정 추가

- insert into [tablename](field1, field2, field3..) values('value1', 'value2', 'value3..');

```
desc user;

insert into user(host, user, password, ssl_cipher, x509_issuer, x509_subject)
values('localhost', 'syslog_server', PASSOWRD('password'), "", "", "");
```

### 3. 새 DB 추가

```
desc db;

insert into db values ('localhost', 'syslog_server', 'syslog_server', 'Y', 'Y', 'Y', 'Y',
'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y', 'Y');
```

### 4. DB 접근 권한 등록

```
flush privileges;
```

```
mysql> flush privileges;
Query OK, 0 rows affected (0.08 sec)
```

### 5. 새 DB 생성

```
create database syslog_server;
```

```
mysql> create database syslog_server;
Query OK, 1 row affected (0.00 sec)
```

### 6. Mysql 종료

```
quit
```

### 7. 새로 생성한 syslog\_server(DB) 접속

```
mysql -u[userId] -p[password];
```

```
mysql -usyslog_server -ppassword
```

```
c:\Program Files\MySQL\MySQL Server 5.1>mysql -u syslog_server -p
Enter password: ****
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 125
Server version: 5.1.36-community MySQL Community Server (GPL)

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use syslog_server
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql>
```

jpa에 syslog-server를 아직 등록해주지 않았기 때문에 어떤 table도 없다

## Syslog 서버 동작 화면

### 1. Syslog Server 패키지파일 생성

- Syslog Server 실행에 필요한 번들을 패키지로 설치하기 위해, 패키지 저장소 루트 디렉터리에 패키지 이름으로 된 디렉터를 만들고 다음과 같은 `kraken.package` 파일 생성

```
[description]
Syslog Server

[version]
1.0.0 2010-12-24 15:22:00

[maven repository]
http://download.krakenapps.org/
file:///d:/kraken/examples/syslog-server/repo/
```

description: package 파일 정보를 기술한다.

version: 버전, 변경시각으로 구성된다. 버전이 여러 개 있으면 변경시각이 최근인 버전을 다운로드한다.

maven repository: maven artifact들을 검색할 저장소 URL을 쓴다. `deploy`로 만들어진 `syslog-server` 배포폴더도 추가한다.

- 패키지 디렉토리 하위에 버전과 일치하는 디렉터를 생성하고 `kraken.package` 파일 생성

```
[bundle]
org.krakenapps.examples.syslogserver 1.0.0
org.krakenapps.syslog 1.3.0
org.krakenapps.filter 1.3.0
org.apache.felix.ipojo [1.2.0, 1.4.0]
org.krakenapps.ipojo 1.0.0
org.krakenapps.jpaa 1.5.0
com.springsource.com.mysql.jdbc 5.1.6
com.springsource.org.apache.commons.pool 1.4.0
com.springsource.org.apache.commons.codec 1.3.0
com.springsource.org.apache.commons.collections 3.2.1
com.springsource.org.apache.commons.logging 1.1.1

[start]
org.krakenapps.examples.syslogserver
org.apache.felix.ipojo
org.krakenapps.ipojo
org.krakenapps.syslog
org.krakenapps.filter
org.krakenapps.jpaa
com.springsource.com.mysql.jdbc
com.springsource.org.apache.commons.pool
com.springsource.org.apache.commons.codec
com.springsource.org.apache.commons.collections
com.springsource.org.apache.commons.logging

[maven]
```

```
org.krakenapps syslog-server 1.0.0
org.apache.felix org.apache.felix.ipojo 1.4.0
org.krakenapps kraken-ipojo 1.0.0
org.krakenapps kraken-syslog 1.3.0
org.krakenapps kraken-filter 1.3.0
org.krakenapps kraken-jpa 1.5.0
com.mysql.jdbc com.springsource.com.mysql.jdbc 5.1.6
org.apache.commons com.springsource.org.apache.commons.pool 1.4.0
org.apache.commons com.springsource.org.apache.commons.codec 1.3.0
org.apache.commons com.springsource.org.apache.commons.collections 3.2.1
org.apache.commons com.springsource.org.apache.commons.logging 1.1.1
```

bundle: 번들 버전 요구사항을 기술한다. Bundle-SymbolicName과 Bundle-Version으로 구성되고, 버전의 범위는 [ ]으로 지정한다.

start: 설치 후 기본적으로 시작시킬 번들의 SymbolicName을 지정한다.

maven: maven 저장소에서 다운로드할 아티팩트의 groupId artifactId version의 순서대로 기입한다.

- 패키지 파일 경로 지정

```
pkg.addRepository [alias] [url]
```

```
pkg.addRepository tutorial file:///d:/repo/
```

```
kraken> pkg.addRepository tutorial file:///d:/repo/
repository [tutorial] added
kraken>
```

## 2. syslog-server 패키지 인스톨

```
pkg.install syslog-server
```

```
kraken> pkg.install syslog-server
Resolving {groupId: org.krakenapps, artifactId: syslog-server, version: 1.0.0}
-> trying to download from http://download.krakenapps.org/
-> trying to download from http://repol.maven.org/maven2/
-> trying to download from file:/d:/kraken/examples/syslog-server/repo/
-> resolved
-> installing: org.krakenapps.examples.syslogserver 1.0.0

Resolving {groupId: org.apache.felix, artifactId: org.apache.felix.ipojo, version: 1.4.0}
-> trying to download from http://download.krakenapps.org/
-> resolved
-> installing: org.apache.felix.ipojo 1.4.0

Resolving {groupId: org.krakenapps, artifactId: kraken-ipojo, version: 1.0.0}
-> trying to download from http://download.krakenapps.org/
-> resolved
-> installing: org.krakenapps.ipojo 1.0.0

Resolving {groupId: org.krakenapps, artifactId: kraken-syslog, version: 1.3.0}
-> trying to download from http://download.krakenapps.org/
-> resolved
-> installing: org.krakenapps.syslog 1.3.0

Resolving {groupId: org.krakenapps, artifactId: kraken-filter, version: 1.3.0}
-> trying to download from http://download.krakenapps.org/
-> resolved
-> installing: org.krakenapps.filter 1.3.0

Resolving {groupId: org.krakenapps, artifactId: kraken-jpa, version: 1.5.0}
-> trying to download from http://download.krakenapps.org/
-> resolved
-> installing: org.krakenapps.jpa 1.5.0
```

```
Starting Bundles
-> [OK] org.krakenapps.examples.syslogserver 1.0.0
-> [OK] org.krakenapps.syslog 1.3.0
-> [OK] org.krakenapps.filter 1.3.0
-> [OK] org.apache.felix.ipojo 1.4.0
-> [OK] org.krakenapps.ipojo 1.0.0
-> [OK] org.krakenapps.jpa 1.5.0
-> [OK] com.springsource.com.mysql.jdbc 5.1.6
-> [OK] com.springsource.org.apache.commons.pool 1.4.0
-> [OK] com.springsource.org.apache.commons.codec 1.3.0
-> [OK] com.springsource.org.apache.commons.collections 3.2.1
-> [OK] com.springsource.org.apache.commons.logging 1.1.1

Complete!
```

- 번들 설치 및 상태 확인

```
bundle.list
```

```
kraken> bundle.list
[ ID] Symbolic Name                               Version   Status
-----
[  0] org.apache.felix.framework                   2.0.5    ACTIVE
[180] org.krakenapps.examples.syslogserver        1.0.0    ACTIVE
[181] org.apache.felix.ipojo                       1.4.0    ACTIVE
[182] org.krakenapps.ipojo                         1.0.0    ACTIVE
[183] org.krakenapps.syslog                       1.3.0    ACTIVE
[184] org.krakenapps.filter                       1.3.0    ACTIVE
[185] org.krakenapps.jpa                           1.5.0    ACTIVE
[186] com.springsource.com.mysql.jdbc              5.1.6    ACTIVE
[187] com.springsource.org.apache.commons.pool    1.4.0    ACTIVE
[188] com.springsource.org.apache.commons.codec   1.3.0    ACTIVE
[189] com.springsource.org.apache.commons.collections 3.2.1    ACTIVE
[190] com.springsource.org.apache.commons.logging 1.1.1    ACTIVE
kraken>
```

- JPA에 syslog-server 등록

```
jpa.register [bundleId] [Entity Manager Factory Name]
```

```
jpa.register 1 syslog-server
```

```
kraken> jpa.register 1 syslog-server
syslog-server registered.
kraken>
```

- 등록된 syslog-server 스크립트 사용

```
syslog-server.list
```

```
kraken> syslog-server.list 20101220
Syslog List
-----
kraken>
```

### 3. Syslog 서버 등록

```
syslog.load
syslog receiver name: syslog-server
bind address (default to localhost):
bind port (default to 514):
charset (default to utf-8):
```

```
kraken> syslog.load
syslog receiver name: syslog-server
bind address (default to localhost):
bind port (default to 514):
charset (default to utf-8):
kraken> █
```

- receiver가 등록되었는지 확인

```
syslog.list
```

### 4. Syslog 수신 및 저장 테스트

- syslog를 전송한다.

```
syslog.send localhost 514 message
```

```
syslog.send localhost 514 <PRI>message
```

```
syslog.send localhost 514 <133>hello syslog server test
```

```
kraken> syslog.send localhost 514 <133>hello syslog server test
kraken> █
```

- syslog로 보낸 메시지가 syslog-server DB에 저장된다.

```
mysql> select * from syslog_server_logs;
+----+-----+-----+-----+-----+-----+
| id | created_at      | facility | message                | remote_ip      | severity |
+----+-----+-----+-----+-----+-----+
| 11 | 2010-12-22 15:59:00 | 16      | hello syslog server test | /127.0.0.1:53481 | 5        |
+----+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql>
```