

Developing REST API For ADW Data Access

REST API로 ADW 데이터 접근 방법

SE Hub Interactive Tech Korea Team

Johnson Kim

2019-04-30



Change Record

Date	Author	Version	Change Reference

Reviewer

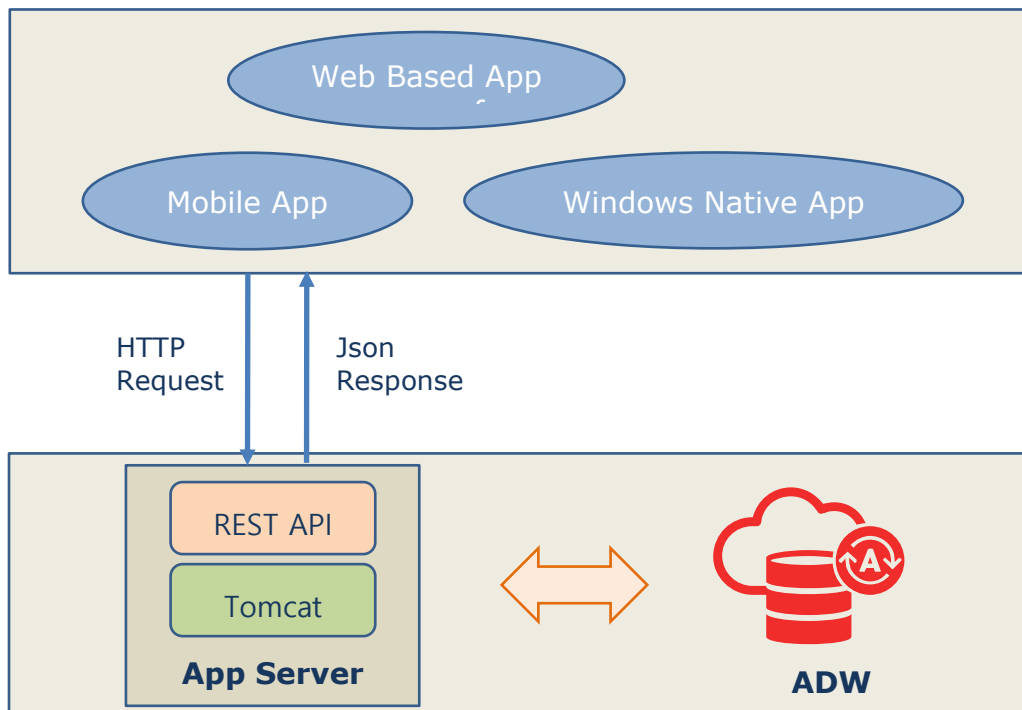
Date	Reviewer	Version	Change Reference



시나리오

REST API로 ADW의 데이터베이스에 접속하여 데이터 관련 작업을 해야 하는 경우가 종종 발생합니다. 본 문서는 널리 사용되고있는 Spring Boot Framework 과 오라클 UCP(Universal Connection Pool)을 이용하여 REST API로ADW의 데이터를 접속하는 과정을 기술하였습니다.

Architecture





목차

1 Table of Contents

1	Rest API 개발을 위한 Pre-Install 작업	5
1.1	JDK 설치	5
1.2	Spring Tool Suite 설치	6
1.3	Maven 설치	6
1.4	수동으로 추가 할 Library resource 파일들을 다운	6
1.5	Postman 설치	7
1.6	ADW credential (Wallet_<database name>.zip) 다운	8
2	Rest API 개발	8
2.1	Spring project 생성	8
2.2	pom.xml 파일에서 dependency 설정	9
2.3	UCP(Oracle Universal Connection Pool) 방식으로 ADW 접속	10
2.4	Spring boot application (Main method)	11
2.5	Rest Controller	11
3	데모	12
3.1	Spring boot application 실행	12
3.2	Rest API로 테이블 생성	13
3.3	Rest API로 테이블 샘플 데이터 입력	13
3.4	Rest API로 테이블 데이터 조회	15



REST API로 ADW Data 접근 방법

1 Rest API 개발을 위한 Pre-Install 작업

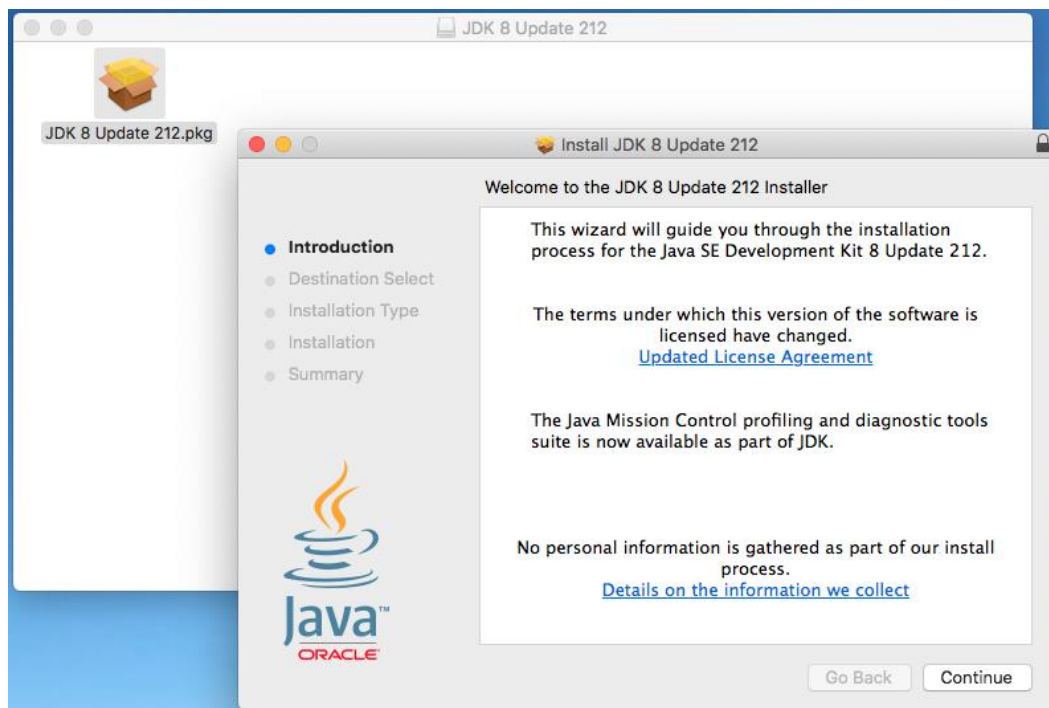
1.1 JDK 설치

아래의 링크를 클릭하여 Java SE Development Kit (JDK) 8u212 혹은 그 이상 버전을 다운 받습니다.

<https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Java SE Development Kit 8u212		
You must accept the Oracle Technology Network License Agreement for Oracle Java SE to download this software.		
<input type="radio"/> Accept License Agreement <input checked="" type="radio"/> Decline License Agreement		
Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	72.86 MB	jdk-8u212-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	69.77 MB	jdk-8u212-linux-arm64-vfp-hflt.tar.gz
Linux x86	174.11 MB	jdk-8u212-linux-i586.rpm
Linux x86	188.92 MB	jdk-8u212-linux-i586.tar.gz
Linux x64	171.13 MB	jdk-8u212-linux-x64.rpm
Linux x64	185.98 MB	jdk-8u212-linux-x64.tar.gz
Mac OS X x64	252.25 MB	jdk-8u212-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	125.06 MB	jdk-8u212-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	88.15 MB	jdk-8u212-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	124.3 MB	jdk-8u212-solaris-x64.tar.Z
Solaris x64	85.41 MB	jdk-8u212-solaris-x64.tar.gz
Windows x86	202.64 MB	jdk-8u212-windows-i586.exe
Windows x64	215.26 MB	jdk-8u212-windows-x64.exe

다운 받은 JDK를 Double 클릭 하여 설치합니다.

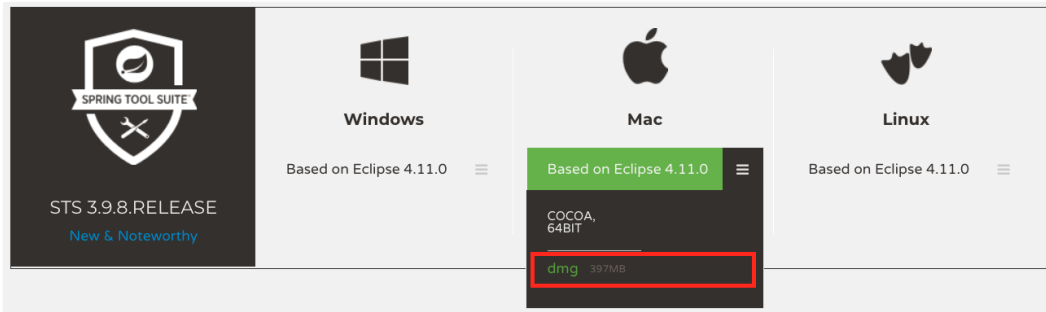


1.2 Spring Tool Suite 설치

Spring tool suite는 Eclipse와 비슷한 Spring boot개발에 편리한 개발 툴입니다.

아래의 링크를 클릭하여 binary zip 파일을 다운 받아 설치합니다.

<https://spring.io/tools3/sts/all/>



1.3 Maven 설치

Maven을 통해 Spring 개발에 필요한 대부분 Library resource들을 자동으로 maven 서버에서 다운 받아 올 수 있습니다. 일부 해당 maven서버에 없는 Library resource는 수동으로 추가할 수 있습니다.

아래의 링크를 클릭하여 binary zip 파일을 다운 받습니다.

<https://maven.apache.org/download.cgi>

	Link
Binary tar.gz archive	apache-maven-3.6.1-bin.tar.gz
Binary zip archive	apache-maven-3.6.1-bin.zip
Source tar.gz archive	apache-maven-3.6.1-src.tar.gz
Source zip archive	apache-maven-3.6.1-src.zip

다운 받은 zip파일을 unzip 하고 .bash_profile에서 PATH에 해당 maven binary 목록을 추가하면 직접 사용할 수 있습니다.

```
$ cd ~
```

```
$ vi .bash_profile
```

```
export PATH=/Users/johnson/apache-maven-3.6.1/bin:$PATH
```

1.4 수동으로 추가 할 Library resource 파일들을 다운

UCP Connection pool 방식으로 wallet을 통해 ADW에 접속하기 위하여 ojdbc8.jar, ucp.jar, oraclepki.jar, osdt_cert.jar, osdt_core.jar Library가 필요하나 이 부분은 default Maven 서버에서




다운 받을 수 없습니다. Maven Repository를 추가하여 자동으로 받도록 할 수도 있으나 아래와 같이 해당 Library들을 다운 받아 수동으로 작업하는 것이 더 쉬울 수 있습니다.

아래의 링크를 클릭하여 다운로드 사이트로 이동하여 ojdbc8-full.tar.gz 파일을 다운 받습니다.

<https://www.oracle.com/technetwork/database/application-development/jdbc/downloads/jdbc-ucp-183-5013470.html>

Oracle Database 18c (18.3) JDBC Driver and UCP Downloads

Get the Zipped JDBC Driver and Companion JARs

 **ojdbc8-full.tar.gz** This archive contains the latest 18.3 JDBC Thin driver (ojdbc8.jar), the Universal Connection Pool (ucp.jar), their Readme(s) and companion jars.
(7,709,868 bytes) - (SHA1: 282a5a52554972c5928112d54bc1818689aa7e03)

다운 받은 후 압축을 풀면 ojdbc8.jar, ucp.jar, oraclepki.jar, osdt_cert.jar, osdt_core.jar 등 jar 파일들이 포함되어 있습니다.

1.5 Postman 설치

Postman을 통해 개발한 REST API을 테스트하고 검증할 수 있습니다.

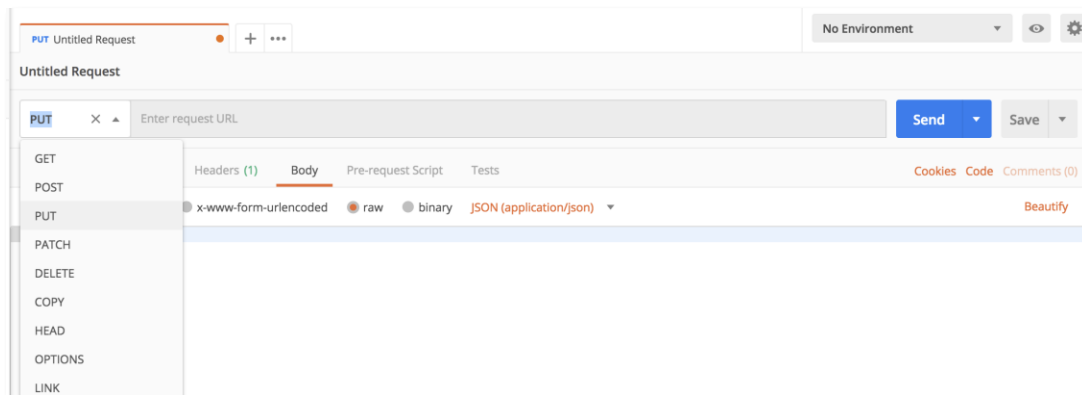
아래의 링크를 클릭하여 앱을 다운 받습니다.

<https://www.getpostman.com/downloads/>



다운 받은 후 Double 클릭하여 앱을 열면 web browser와 비슷한 화면이 보입니다.

URL을 입력하고 GET, POST, PUT과 같은 method들을 결합하여 사용할 수 있습니다.

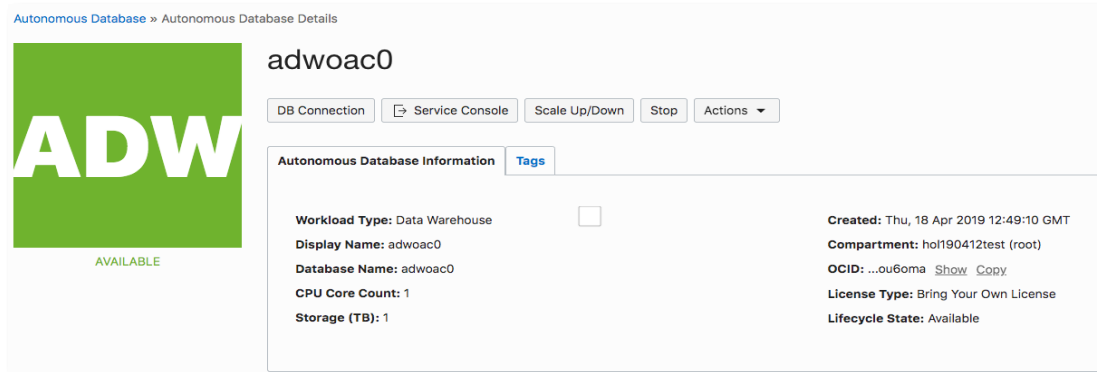




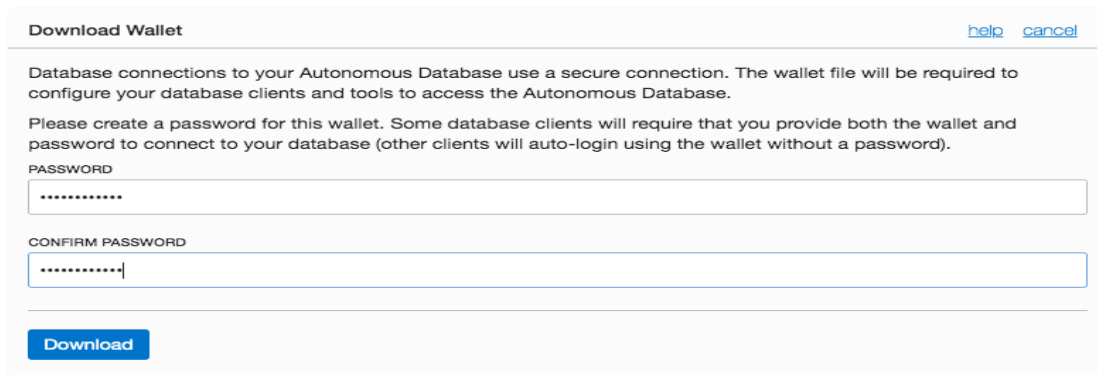
1.6 ADW credential (Wallet_<database name>.zip) 다운

ADW는 일반 Database 접속과 달리 데이터 보안을 강화하기 위하여 접속할 때 Credential을 필요로 합니다.

아래의 ADW 서비스 관리 화면에서 "DB Connection" 버튼을 클릭하여 Wallet을 다운 받습니다.



Wallet을 다운 받기 위하여 해당 wallet 비번 설정이 필요합니다. (예를 들어: Welcome123456)
이 비번은 데이터베이스에 접속하는 유저의 비번이 아닙니다.



2 Rest API 개발

2.1 Spring project 생성

Spring Tool Suite 앱을 열고 아래와 같이 새로운 Spring Project를 생성합니다.

File->New->Spring Starter Project.






2.2 pom.xml 파일에서 dependency 설정

Pom.xml 파일에서 <dependencies>...</dependencies> 내부에 아래의 Dependency들을 추가합니다. 그러면 maven이 해당 dependency library들을 maven 서버에서 자동으로 받아옵니다.

```

<dependency>
    <groupId>com.oracle.jdbc</groupId>
    <artifactId>ojdbc8</artifactId>
    <version>18.3.0.0</version>
</dependency>
<dependency>
    <groupId>com.oracle.jdbc</groupId>
    <artifactId>ucp</artifactId>
    <version>18.3.0.0</version>
</dependency>
<dependency>
    <groupId>com.oracle.jdbc</groupId>
    <artifactId>oraclepki</artifactId>
    <version>18.3.0.0</version>
</dependency>
<dependency>
    <groupId>com.oracle.jdbc</groupId>
    <artifactId>osdt_cert</artifactId>
    <version>18.3.0.0</version>
</dependency>
<dependency>
    <groupId>com.oracle.jdbc</groupId>
    <artifactId>osdt_core</artifactId>
    <version>18.3.0.0</version>
</dependency>
<dependency>
    <groupId>org.json</groupId>
    <artifactId>json</artifactId>
    <version>20180813</version>
</dependency>

```

Maven이 Dependency Library를 maven 서버에서 받아 오지 못할 경우 아래와 같이 로 표기됨을 볼 수 있습니다. 위에서 다운 받은 ojdbc8-full.tar.gz에 해당 Library 파일들이 모두 포함되어 있으므로 직접 아래와 같이 수동으로 추가해 줄 수 있습니다.





예를 들어 ojdbc8-full.tar.gz을 다운 받아 압축을 푼 경로가 "/Users/johnson/Downloads/ojdbc8-full" 일 경우 아래의 명령어를 수행합니다.

```
$ cd /Users/johnson/Downloads/ojdbc8-full /*실제 압축을 푼 폴더의 경로로 바꿔서 작업 하셔야 합니다.*/

$ mvn install:install-file -Dfile=ojdbc8.jar -DgroupId=com.oracle.jdbc -DartifactId=ojdbc8 -Dversion=18.3.0.0 -Dpackaging=jar

$ mvn install:install-file -Dfile=ucp.jar -DgroupId=com.oracle.jdbc -DartifactId=ucp -Dversion=18.3.0.0 -Dpackaging=jar

$ mvn install:install-file -Dfile=osdt_cert.jar -DgroupId=com.oracle.jdbc -DartifactId=osdt_cert -Dversion=18.3.0.0 -Dpackaging=jar

$ mvn install:install-file -Dfile=osdt_core.jar -DgroupId=com.oracle.jdbc -DartifactId=osdt_core -Dversion=18.3.0.0 -Dpackaging=jar
```

다음 pom.xml에서 추가하였던 Dependency를 약간 수정(예를 들어 space하나 추가하고 다시 삭제함)하고 Save 하면 에러 메시지가 사라짐 볼 수 있습니다.

2.3 UCP(Oracle Universal Connection Pool) 방식으로 ADW 접속

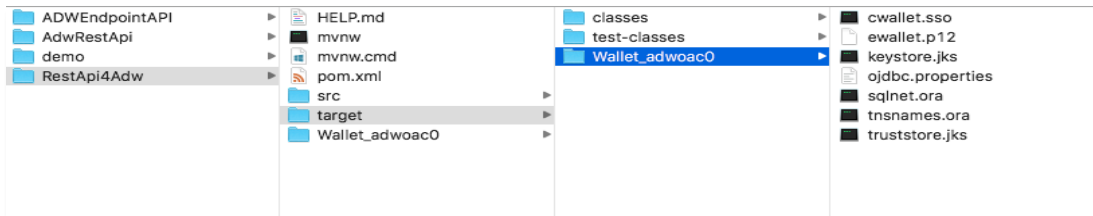
ADWConnection.java라는 파일을 만들고 아래와 같이 UCP 방식으로 connection pool을 만들어 ADW에 접속할 수 있도록 class를 만들어 줍니다.



ADW 는 5 분 이상 inactive 상태인 session 의 connection 을 자동으로 끊어

줌으로 setValidateConnectionOnBorrow(true); 을 추가하여 매번 pool 에서 connection 을 빌려 사용하기 전에 상태를 체크하여 사용 불가능하면 데이터베이스와 다시 connection 을 맺어 사용하도록 합니다.

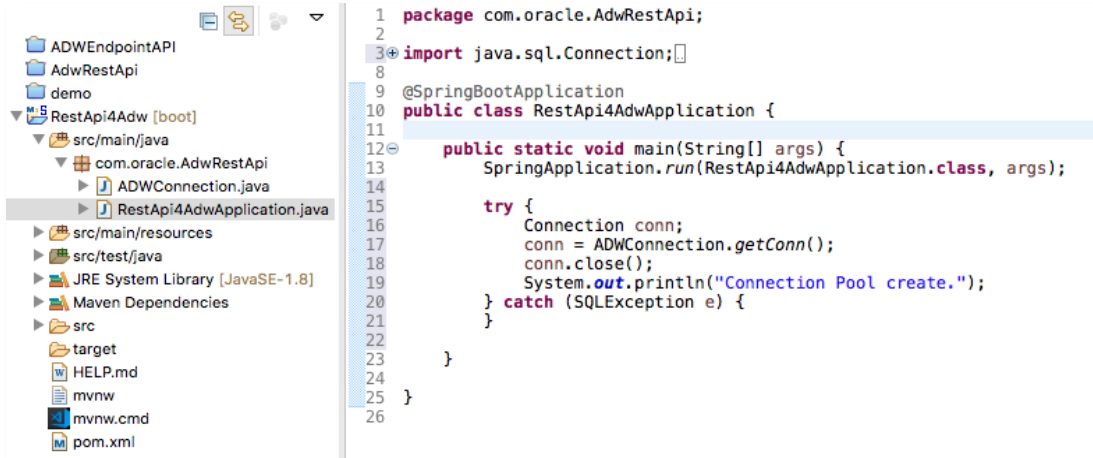
그리고 위에서 다운 받은 Wallet_<database name>.zip 을 unzip 하여 생성된 Wallet_<database name> 폴더를 workspace 와 그 아래 target 폴더에 각각 추가합니다.





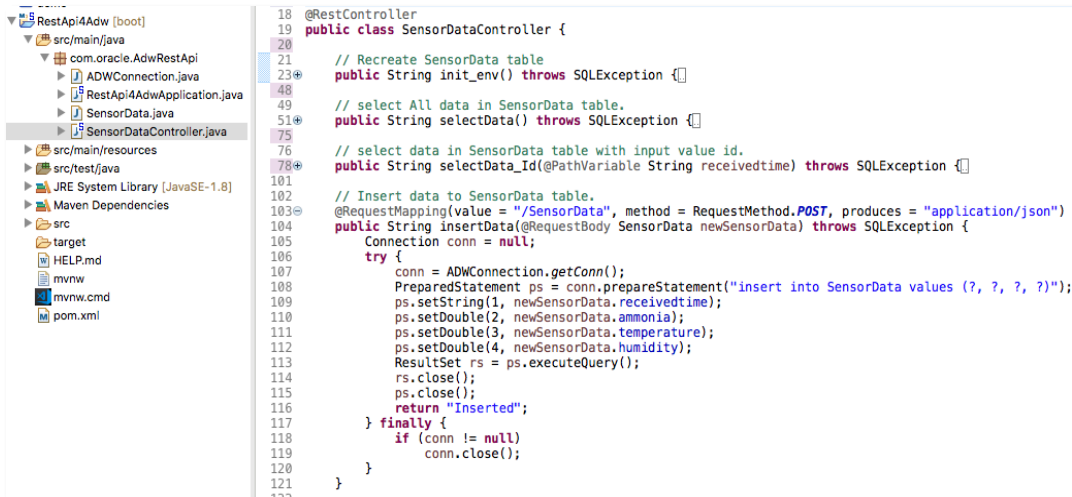
2.4 Spring boot application (Main method)

Connection pool을 생성하는데 시간이 좀 걸리므로 아래와 같이 Application이 시작될 때 connection pool 생성되도록 합니다.



2.5 Rest Controller

아래와 같이 Method: POST, URL: <IP Address>:<Port>/SensorData 를 Insert문과 매핑 시켜주고 Json object 형태로 받은 데이터에서 필요한 정보를 추출하여 테이블에 추가하여 줍니다.



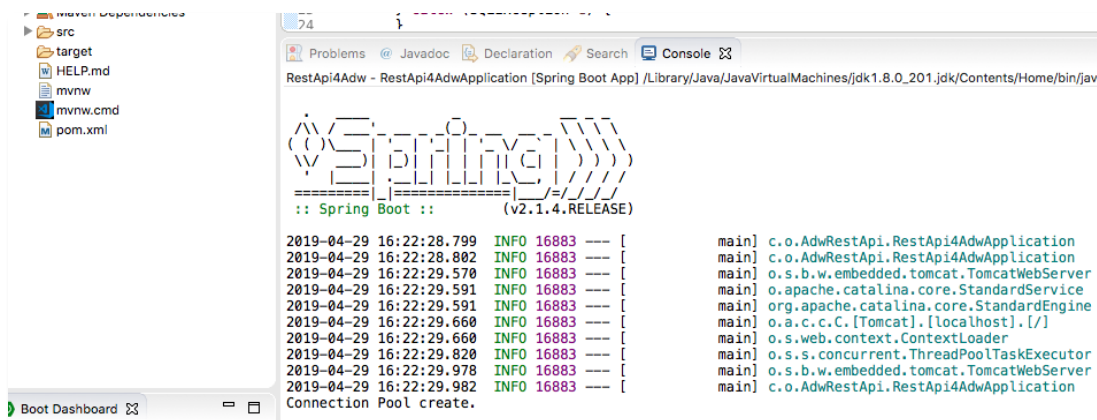
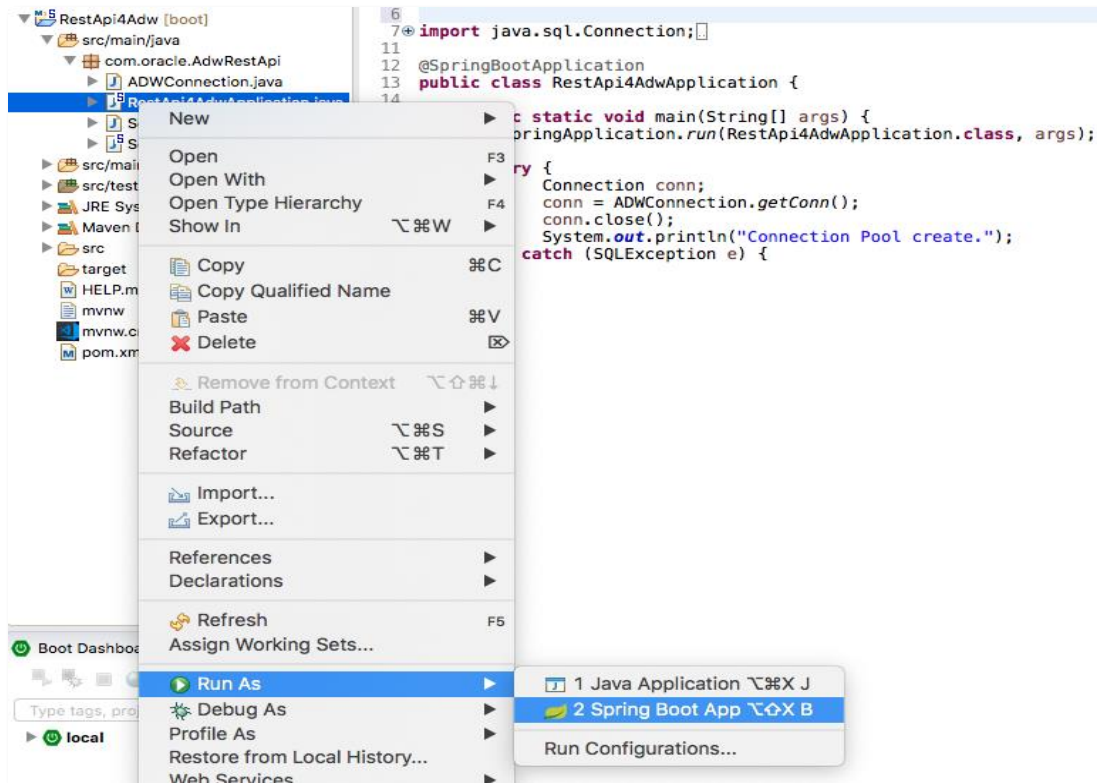


3 데모

3.1 Spring boot application 실행

Web server가 올라가고 Connection pool이 생성되면서 초기 connection이 5개 만들어 질 때까지
즉 "Connection Pool Created." 메시지가 나올 때까지(20초정도) 소요됩니다.

Main class -> right click-> Run As -> Spring boot App.





3.2 Rest API로 테이블 생성

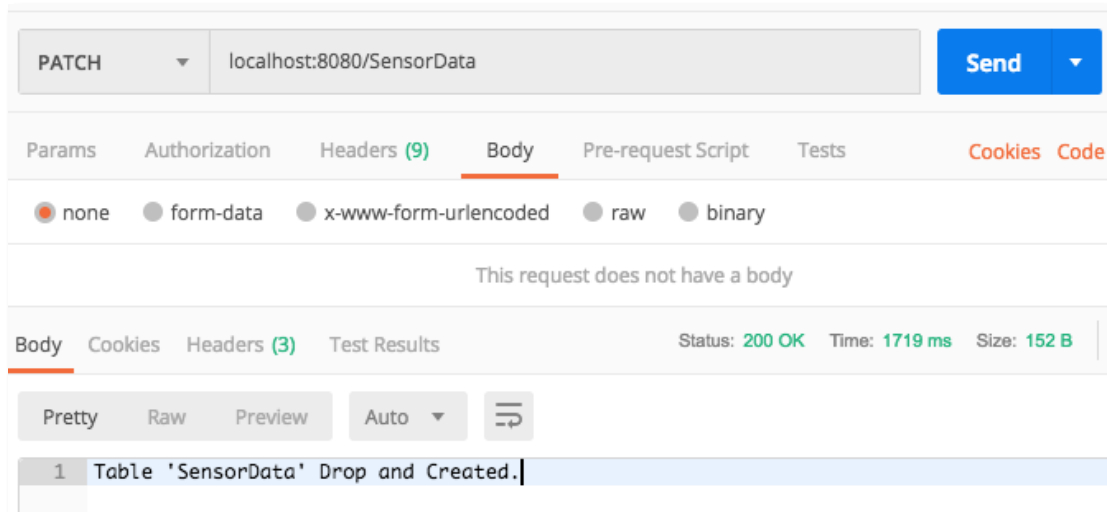
데이터 베이스에 본 데모에서 사용할 유저 테이블이 없으므로 REST API로 SensorData라는 테이블을 생성해 보도록 하겠습니다.

URL: localhost:8080/SensorData

Method: Patch

Body: none

Send 버튼을 클릭



실제 내부에서 수행되는 SQL문:

SQL> create table SensorData (receivedtime varchar2(14), ammonia number, temperature number, humidity number);

3.3 Rest API로 테이블 샘플 데이터 입력

위에서 생성한 테이블에 Jason object 형식으로 데이터 두 건을 각각 입력해 보도록 하겠습니다.

URL: localhost:8080/SensorData

Method: POST

Body: {"receivedtime":"20190425040500","ammonia":100,"temperature":100,"humidity":100}

Send 버튼을 클릭



POST localhost:8080/SensorData Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Cookies Code

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary JSON (application/json)

```
1 {  
2   "receivedtime": "20190425040500",  
3   "ammonia": 100,  
4   "temperature": 100,  
5   "humidity": 100  
6 }
```

Body Cookies Headers (3) Test Results Status: 200 OK Time: 7554 ms Size: 129 B

Pretty Raw Preview JSON

1 Inserted

실제 내부에서 수행되는 SQL문:

SQL> insert into SensorData values (20190425040500, 100, 100, 60);

URL: localhost:8080/SensorData
Method: POST
Body: {"receivedtime":"20190425040530","ammonia":200,"temperature":150,"humidity":60}
Send 버튼을 클릭

POST localhost:8080/SensorData Send

Params Authorization Headers (9) **Body** Pre-request Script Tests Cookies Code

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary JSON (application/json)

```
1 {  
2   "receivedtime": "20190425040530",  
3   "ammonia": 200,  
4   "temperature": 150,  
5   "humidity": 60  
6 }
```

Body Cookies Headers (3) Test Results Status: 200 OK Time: 803 ms Size: 129 B

Pretty Raw Preview JSON

1 Inserted

실제 내부에서 수행되는 SQL문:

SQL> insert into SensorData values (20190425040530, 200, 150, 60);



3.4 Rest API로 테이블 데이터 조회

- 우선 테이블 데이터 전체를 조회해 보도록 하겠습니다.

URL: localhost:8080/SensorData

Method: GET

Body: none

Send 버튼을 클릭

The screenshot shows a REST client interface with the following details:

- Method:** GET
- URL:** localhost:8080/SensorData
- Body:** none
- Status:** 200 OK
- Time:** 6976 ms
- Size:** 279 B
- Response Body (JSON):**

```
[{"receivedtime": "20190425040500", "ammonia": 100, "temperature": 100, "humidity": 100}, {"receivedtime": "20190425040530", "ammonia": 200, "temperature": 150, "humidity": 60}]
```

실제 내부에서 수행되는 SQL문:

SQL> select * from SensorData;

- 다음 receivetime이 "20190425040530"인 데이터만 조회해 보도록 하겠습니다.

URL: localhost:8080/SensorData/20190425040530

Method: GET

Body: none

Send 버튼을 클릭



GET localhost:8080/SensorData/20190425040530 Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Cookies Code

☒ none ☐ form-data ☐ x-www-form-urlencoded ☐ raw ☐ binary

This request does not have a body

Body Cookies Headers (3) Test Results Status: 200 OK Time: 796 ms Size: 197 B

Pretty Raw Preview JSON

```
1 [
2   {
3     "receivedtime": "20190425040530",
4     "ammonia": 200,
5     "temperature": 150,
6     "humidity": 60
7   }
8 ]
```

실제 내부에서 수행되는 SQL문:

SQL> select * from SensorData where receivedtime=20190425040530;



감사합니다