


(ICBMS-3 세부) 사물 가상화, 분산 자율지능 및 데이터  
연계/분석을 지원하는 IoT 기반 플랫폼 기술

# SO 프레임워크 개발 가이드

요약 : SO 프레임워크를 적용하여 개발하는 방법을 설명한다.

 PINEONE	SO 프레임워크 개발가이드		
프로젝트명:	단계:	담당팀:	

## 문서승인

주문서의 부속문서로 승인을 받는 경우 주문서의 서명으로 대체함.

수행사: (주)파인원커뮤니케이션즈

\_\_\_\_\_  
박원욱 책임  
개발팀

\_\_\_\_\_  
Date

주관사: (주)파인원커뮤니케이션즈

\_\_\_\_\_  
김태철 팀장  
개발팀

\_\_\_\_\_  
Date

## 제.개정 이력서

[illegible]

## 목차

1. 개발 가이드 개요	6
2. SO Server 내부 구성 모듈	7
3. Source 빌드 환경 구축	8
3.1 JDK 다운로드 및 설치	8
3.2 STS 다운로드 및 설치	8
3.3 Oasis SO Framework 소스 다운로드 및 압축 해제	10
3.4 SO 프로젝트 빌드 환경 구성	11
3.5 Gradle 을 이용한 Build	13
4. SO 서버 이미지 실행 환경 구축	15
4.1 JDK 다운로드 및 설치	15
4.2 Kafka / 설치 및 실행	15
4.2.1 kafka 설치	15
4.2.2 zookeeper/kafka 실행	16
4.3 MariaDB 다운로드 및 설치	17
4.3.1 MariaDB 다운로드 및 설치안내	17
4.3.2 DB 컬렉션 및 생인 생성	17
4.4 SO Server 실행환경 설정	17
4.5 eclipse 에서 서버 실행	19
4.5.1 서버 실행 환경 설정	19
4.6 배포 환경에서 서버 실행 하기	23
4.6.1 SO 실행 환경 설정	23
4.6.2 SO 실행	24
4.6.3 SO 서버 동작 확인	24
5. SO Framework Server TEST	24
5.1 POSTMAN 으로 생성한 SO(Service Orchestration)를 실행	24
6. Q&A	27

프로젝트명:

단계:

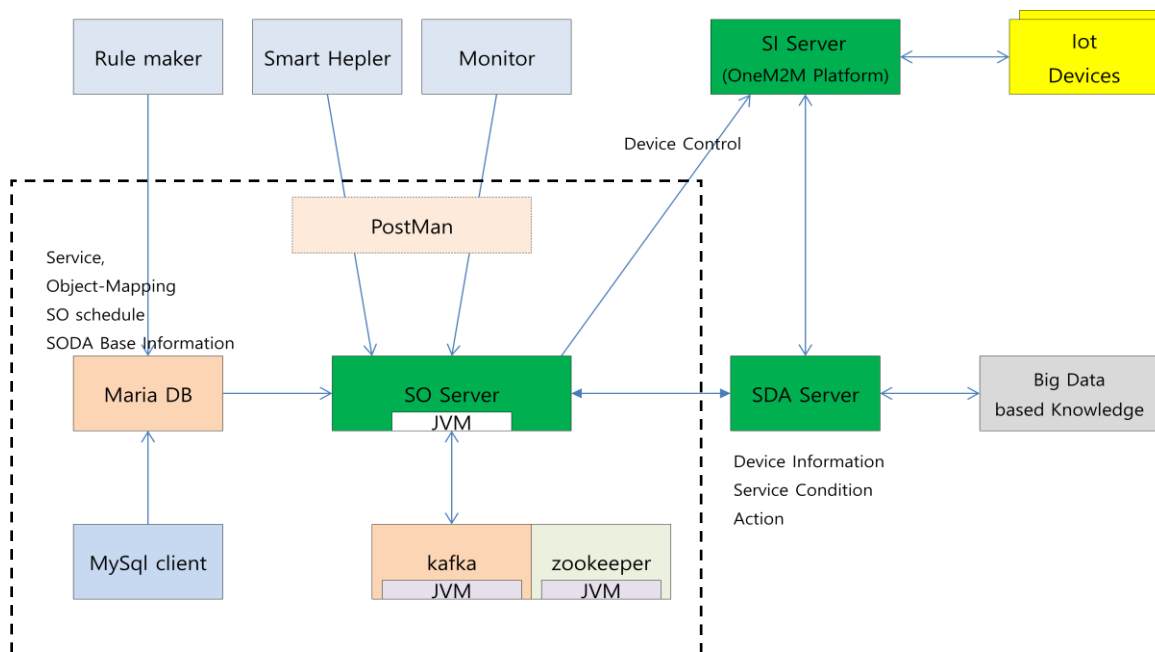
담당팀:

## 1. 개발 가이드 개요

Oasis (Open-source Architecture Semantic lot Service-platform) 프로젝트는 국제 표준을 준용하는 오픈 소스 기반 지능형 사물 인터넷 서비스 플랫폼을 개발하는 것을 목표로 하고 있습니다.

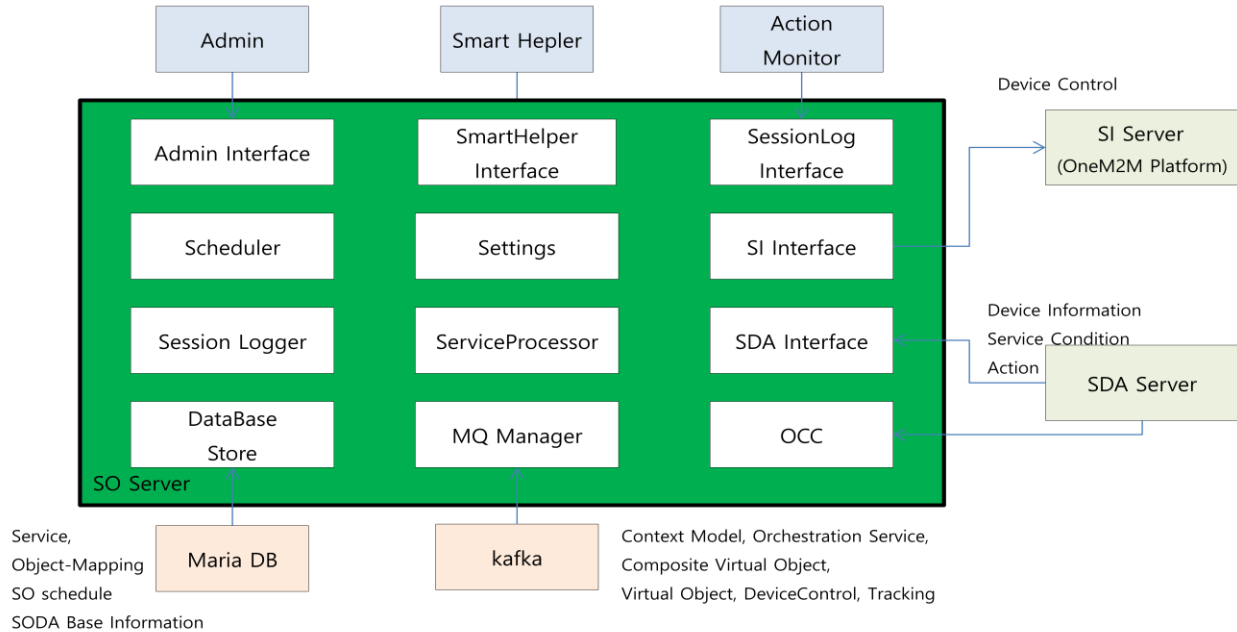
Oasis 프로젝트는 2015 년도 정보통신. 방송 기술개발사업 신규지원 대상과제 "(ICBMS-3 세부) 사물 가상화, 분산 자율지능 및 데이터 연계/분석을 지원하는 IoT 기반 플랫폼 기술 개발" 과제의 결과물로서 오픈소스로 제공됩니다.

Oasis 프로젝트에 사용되는 시스템들은, IOT Devices 들을 제어하고 Device 들로부터의 센서 데이터를 받아 처리하는 SI Server 와 SI Server 로부터 수집된 데이터를 바탕으로 지식 데이터 베이스를 구성하고, 상황 판단에 대한 인터페이스를 제공하는 SDA 서버, SDA 에서 구축된 지식 베이스를 가지고 상황별 서비스를 제공하는 SO 서버, 그리고 각 서버에서 필요로 하는 데이터베이스 서버, 메시지 큐 서버, 서비스 제작 툴 및 서비스 요청 App 으로 구성되어 있습니다.



이 문서는 Oasis SO Framework 를 처음 접하는 분들이 소스를 다운받고 쉽게 개발 환경을 구축할 수 있도록 SO Server 를 중심으로 하여 점선으로 표시된 Box 내의 구성 요소들에 대한 설치와 실행 방법에 대해서 안내합니다.

## 2. SO Server 내부 구성 모듈



- Scheduler : 설정된 스케줄 데이터에 의하여, SDA 서버로부터 CM 발생 여부를 체크
- Settings : 주변 Server 환경 설정값, 서버 동작환경 관리
- SI Interface : IOT Device 제어 데이터를 SI Server 서버로 전달
- SDA Interface : SDA 서버로부터 CM 발생정보, 지식베이스 Device 구성 정보를 획득하는 Interface
- OCC : SDA Server 에 의한 응급 서비스 처리를 위한 Interface
- Service Processor : 저작 데이터를 해석하여 동적으로 Layer 별 Virtual Object 를 하고, 해당 Device 제어 정보를 생성
- MQ Manager : Message Queue Server(KAFKA)에 Message 를 넣거나 획득을 담당
- SessionLogger : 각각의 처리 모듈에서 발생한 로그를 Session 별로 저장하거나 읽는 기능
- Database Store : Database(MariaDB)에 데이터를 저장하거나 읽어내는 기능  
스케줄, CM 상황 별 OS 실행 데이터, OS 별 Virtual Object 제어 데이터, CVO 구성 데이터, 저작시 필요 데이터, CM 목록, Function 목록, Location 목록, Profile 별 의존 관계, 저작 데이터, 세션별 동작 로그 데이터 Read-Write 관리
- Admin Interface : 관리자에 의한 스케줄 관리, 내부 동작 설정 변경을 위한 interface
- SmartHelper Interface : Smart Helper 의 요청한 의한 서비스 처리
- SessionLog Interface : Session 별 서비스 처리 결과를 획득하기 위한 interface

## 3. Source 빌드 환경 구축

SO Server source 를 빌드하기 위해서는 windows 와 JDK 와 STS, SO Framework 소스가 필요합니다.

- linux 의 경우 아래 링크를 참고하여 설치


[JDK 설치안내](#)

이 문서에서는 Windows 64bit 환경에서 아래와 같은 절차로 빌드환경을 구성하는 것에 대하여 설명합니다.

1. JDK 다운로드 및 설치
2. STS 다운로드 및 설치
3. Oasis SO Framework 소스 다운로드
4. SO 프로젝트 빌드 환경 구성
5. 빌드

### 3.1 JDK 다운로드 및 설치

#### 1) 다운로드

- <http://www.oracle.com/technetwork/java/javase/downloads/index.html> 에 접속
-  클릭 (추천 : Java SE 8uxxx 클릭 )
- Accept License Agreement 를 선택 후 OS 에 맞는 link 를 클릭

#### 2) 설치

- 다운로드 된 jdk\_xxx-windows-x64.exe 실행

### 3.2 STS 다운로드 및 설치

#### 1) 다운로드

- <https://spring.io/tools> 접속
- DOWNLOAD STS (3.9.2.RELEASE for Windows) 클릭하여 다운로드

#### 2) 설치

- 다운로드 된 zip 파일 (예:spring-tool-suite-3.9.2.RELEASE-e4.7.2-win32.zip)을 적당한 위치(예: C:\w)에 압축 해제



프로젝트명:

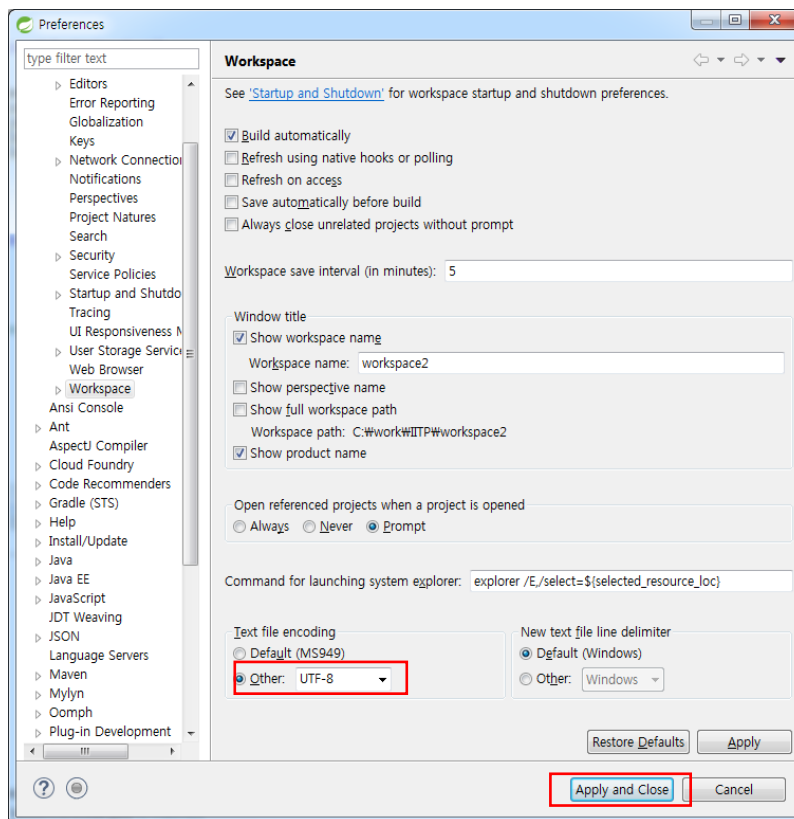
단계:

담당팀:

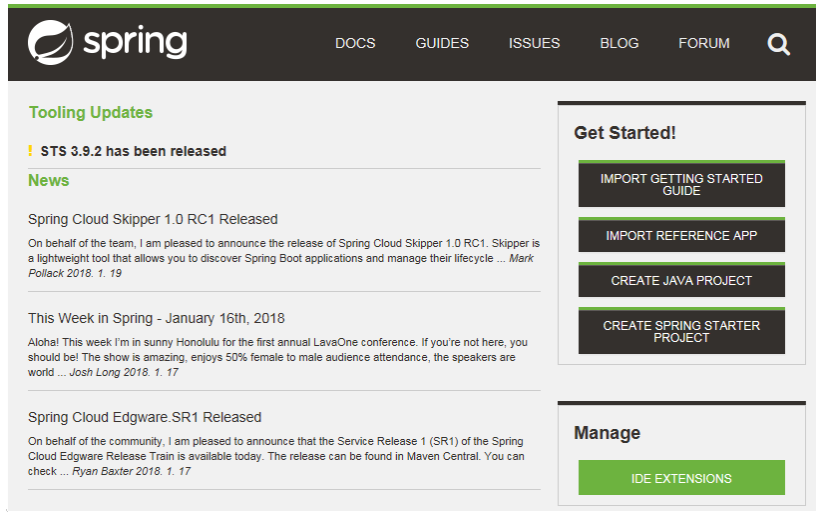
- sts-bundleW sts-3.x.x.RELEASE 폴더에 있는 STS.exe 실행

## 3) STS 환경 설정

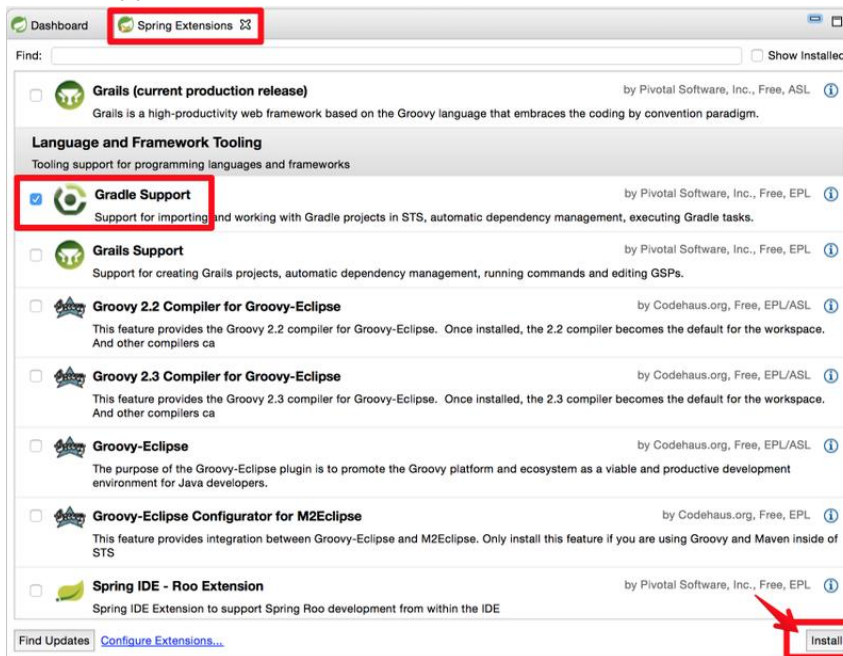
- UTF8 설정
  - Windows 메뉴의 Preferences 클릭
  - General > Workspace 에서 Text file encoding 을 other 로 선택 후 UTF-8 을 입력하고 Apply and Close 버튼을 누른다.



- Gradle 설치
  - Dashboard 창의 IDE EXTENSIONS 버튼을 클릭한다.



Gradle Support 를 선택하고 install 버튼을 클릭한다.



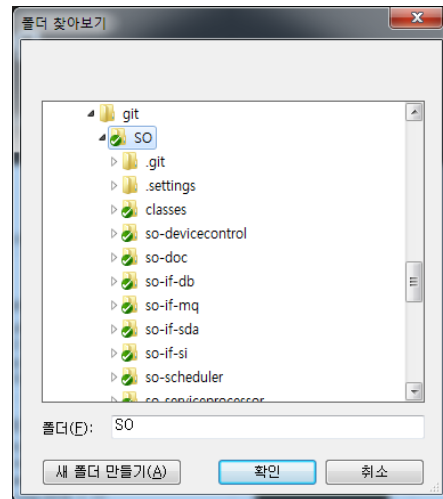
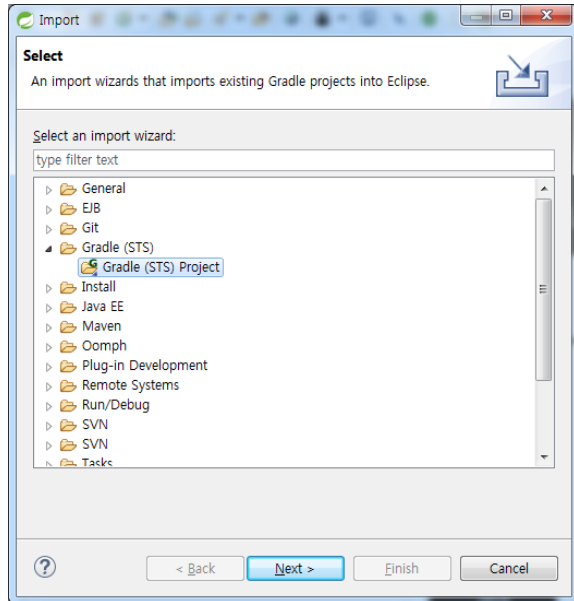
next 를 눌러서 설치를 완료한다.

### 3.3 Oasis SO Framework 소스 다운로드 및 압축 해제

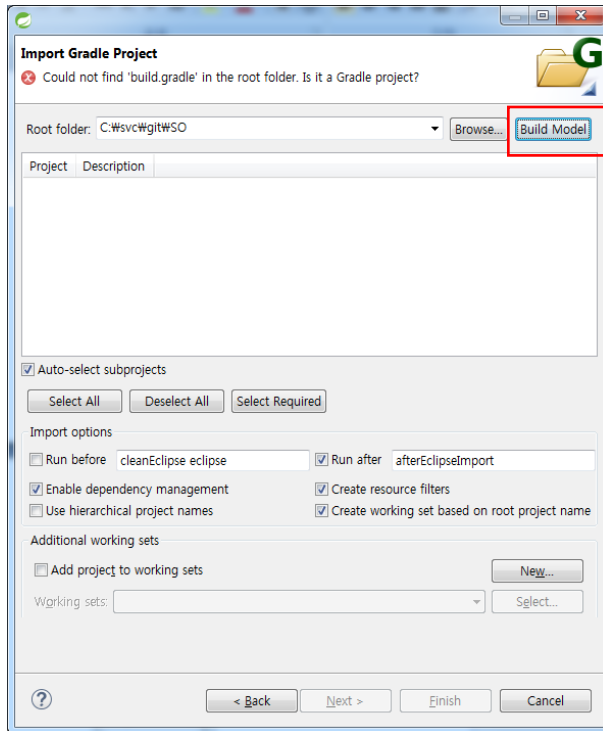
- [릴리즈 페이지](#)에서 SO 소스 및 설치관련 파일을 다운로드
- 다운로드된 파일의 압축 해제

## 3.4 SO 프로젝트 빌드 환경 구성

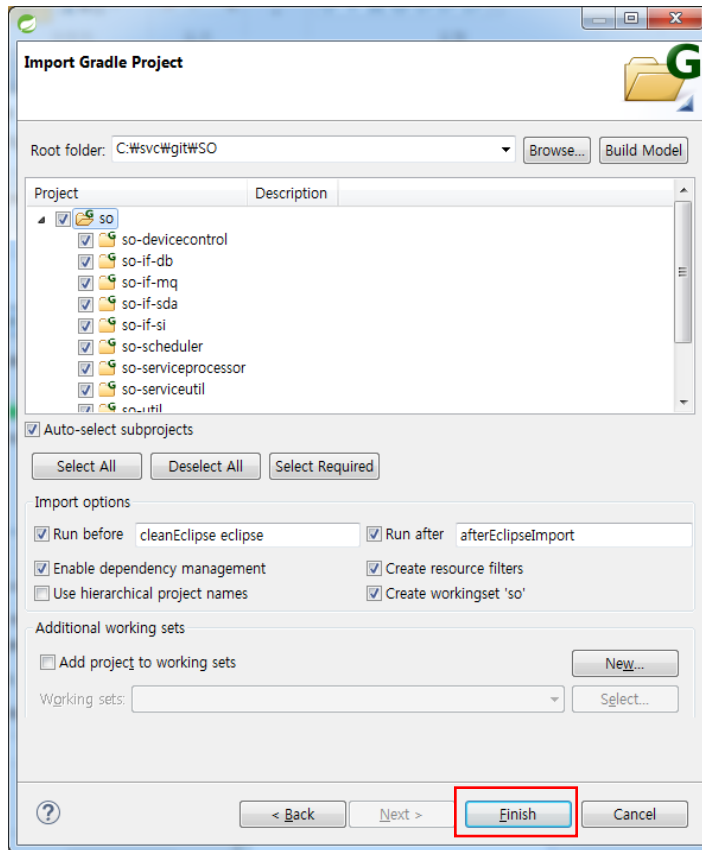
- 메뉴에서 File/Import 메뉴를 선택하여 Import 창을 열어 Gradle(STS) > Gradle(STS) Project 를 선택한 후 SO 소스가 저장된 폴더를 선택한다.



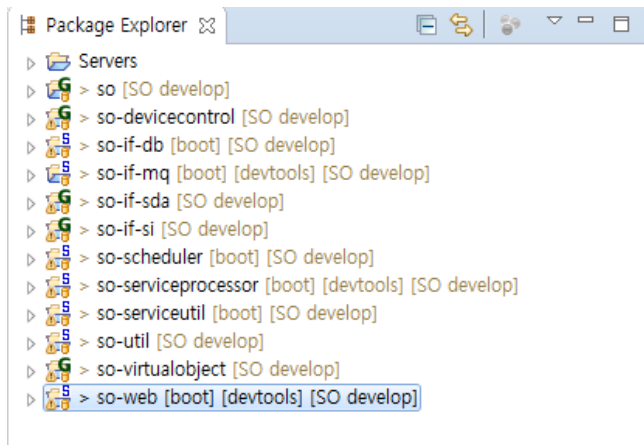
- 아래와 같은 Import Gradle Project 창에서 오른쪽 상단에 위치한 "Build Model" 버튼을 클릭합니다. build.gradle 정보를 생성한다.



- so 디렉토리를 선택한 후 finish 버튼을 누른다.



- Import 된 프로젝트는 아래와 같이 표시됩니다.



## 3.5 Gradle 을 이용한 Build

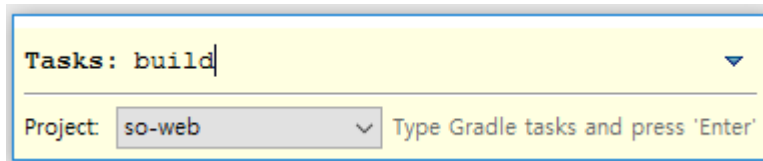
- Ctrl+Alt+Shift+R 로 Gradle Build 실행.

프로젝트명:

단계:

담당팀:

- Project 를 so-web 선택. Tasks 에 build 명령어 입력 후 엔터를 누른다..



Tasks: build ▼

Project: so-web ▼ Type Gradle tasks and press 'Enter'

## 4. SO 서버 이미지 실행 환경 구축

SO Server 실행 환경 구축은 아래의 순서로 진행할 수 있습니다.

1. JDK 다운로드 및 설치
2. Zookeeper /Kafka 설치 및 실행
3. MariaDB 서버 설치 및 실행
4. SO Server 동작 환경 설정
5. SO 프로젝트 빌드 이미지 복사
6. SO Server 실행
7. HTTP 애물레이터(PostMan)를 이용한 시험

Zookeeper, Kafka, SO Server 가 동작하기 위해서는 JDK 가 필요하다. 그러므로 Zookeeper 와 Kafka, SO Server 를 각각 서로 다른 서버에 설치하고자 하는 경우에는, 각 서버마다 JDK 의 설치가 필요합니다.

### 4.1 JDK 다운로드 및 설치

- 3.Source 빌드 환경 구축 때의 JDK 설치 안내를 참고

### 4.2 Kafka / 설치 및 실행

#### 4.2.1 kafka 설치

##### 1) kafka 다운로드

```
# wget http://apache.mirror.cdnetworks.com/kafka/0.11.0.2/kafka_2.11-0.11.0.2.tgz
```

##### 2)압축 해제 및 링크 :

```
# tar xvfz kafka_2.11-0.11.0.2.tgz
# ln -s kafka_2.11-0.11.0.2 kafka
```

3) 릴리즈 페이지에서 받은 SO Framework 소스 중 so-doc 내부의 scripts 폴더를, 다운받은 kafka 폴더가 저장된 폴더로 복사한다.



kafka



scripts

## 4.2.2 zookeeper/kafka 실행

### 1) zookeeper 실행

```
./scripts/1-startup-zookeeper.sh
```

### 2) kafka 실행

```
./scripts/2-startup-kafka.sh
```

### 3) kafka topic 생성

```
▪ ./scripts/create-topics-so.sh
```

### 4) topic 확인

```
▪ ./ scripts/topic-list-so.sh | grep "^Topic"
```

아래와 같은 7 개의 topic 목록이 생성되어 있는지 확인

```
...
Topic:compositevirtualobject PartitionCount:1 ReplicationFactor:1 Configs:
Topic:contextmodel PartitionCount:1 ReplicationFactor:1 Configs:
Topic:devicecontrol PartitionCount:1 ReplicationFactor:1 Configs:
Topic:devicecontrol-seq PartitionCount:1 ReplicationFactor:1 Configs:
Topic:orchestrationservice PartitionCount:1 ReplicationFactor:1 Configs:
Topic:tracking PartitionCount:1 ReplicationFactor:1 Configs:
Topic:virtualobject PartitionCount:1 ReplicationFactor:1 Configs:
```

주의! SO 서버가 실행되는 동안에 zookeeper 와 kafka 는 반드시 실행 중 이여야 한다. (SO 서버에서 Queue 로 사용)

참고 : [kafka 실행 테스트](#)



## 4.3 MariaDB 다운로드 및 설치

### 4.3.1 MariaDB 다운로드 및 설치안내

MariaDB 다운로드 및 설치에 아래 링크를 참조하여 수행한다.

<https://downloads.mariadb.org/>

### 4.3.2 DB 컬렉션 및 색인 생성

[릴리즈 페이지](#)에서 so-doc 폴더 내부의 [MariaDB Script](#) (db\_script.txt)를 다운로드 후 mysql client 툴을 이용하여 실행시켜 기본 컬렉션 및 색인을 생성한다.

```
$ mysql -u root -p
Enter password:
```

```
MariaDB [(none)]> source db_script.txt
```

#### 4.3.2.1 생성된 DB 를 확인 한다.

```
MariaDB [so.open]> show tables;
```

```
+-----+
| Tables_in_so.open |
+-----+
| CM_LocationTemp   |
| composite_virtual_object |
| context_model     |
| cvo_vo            |
| device1121        |
| device_value      |
| functionality     |
| location          |
| noncvo            |
| orchestration_service |
| physical_device_type |
| profile           |
| profile_dep       |
| rule_body         |
| rule_item         |
| session_data      |
| session_data_device |
| session_data_location |
| session_data_vo   |
| smart_helper      |
| tracking          |
| virtual_object    |
| vo_value_type     |
+-----+
23 rows in set (0.00 sec)
```

## 4.4 SO Server 실행환경 설정

설정은 application-xxx.properties 파일로 작성되며 so-web 폴더의 conf 폴더에 존재합니다.

프로젝트명:

단계:

담당팀:

개발용 : application-dev.properties

운영용 : application-product.properties

설정항목별 의미는 아래와 같습니다.

Item	Description
server.port	서버 포트 설정
server.context-path	서버 main-path 설정
logging.config	SO 로그 파일 정보 파일
spring.datasource.url	데이터 베이스 연결 정보
sda.connection.uri	SDA 서버 연결 정보
spring.datasource.username	DB 계정 정보
spring.datasource.password	DB password
spring.jackson.time-zone	서버 시간 설정 (Asia/Seoul 추천)
mq.broker.list	kafka 연결 정보
mq.zookeeper.list	주키퍼 연결 정보
sda.connection.uri	SDA 서버 연결 정보
si.connection.uri	OneM2M SI 서버 연결 정보
lwm2m.connection.uri	LWM2M SI 서버 정보
so.connection.uri	SO 서버 정보

아래는 설정파일 샘플입니다.

```
server.port=10080
server.contextPath=/so
```

```
spring.datasource.url=jdbc:mysql://127.0.0.1:33306/so?useUnicode=true&characterEncoding=utf-8&autoReconnect=true
spring.datasource.username=user
spring.datasource.password=pw
spring.datasource.driverClassName=com.mysql.jdbc.Driver
spring.jackson.time-zone=Asia/Seoul

mq.broker.list=localhost:9092
mq.zookeeper.list=localhost:2181

sda.connection.uri=http://127.0.0.1:30080/sda/ctx/
si.connection.uri=http://127.0.0.1:30081
lwm2m.connection.uri=http://127.0.0.1:50081
so.connection.uri=http://127.0.0.1:${server.port}
```

## 4.5 eclipse 에서 서버 실행

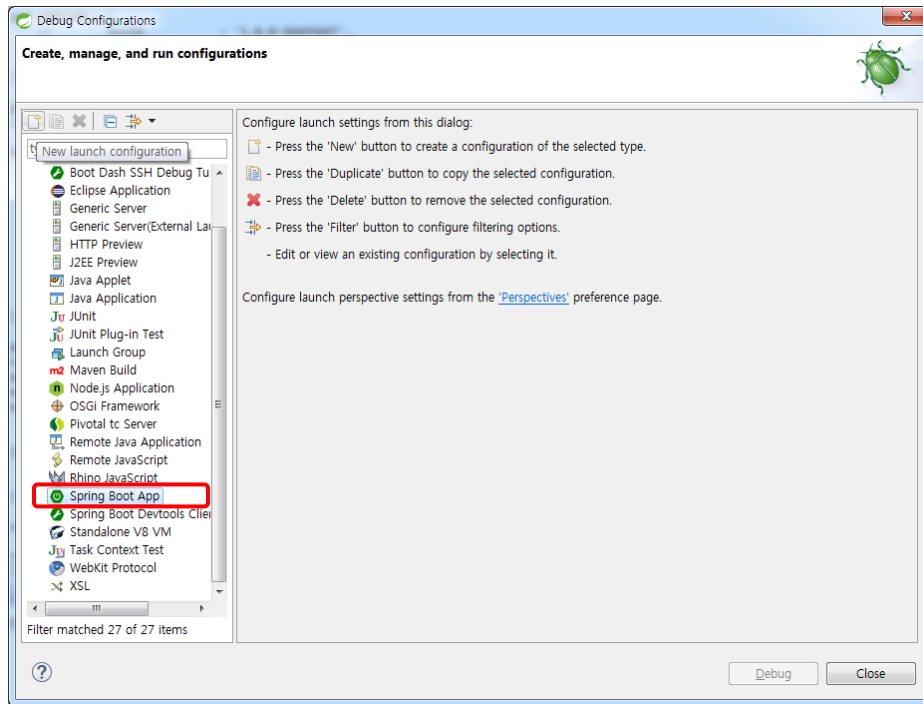
### 4.5.1 서버 실행 환경 설정

- 1) Main Menu 의 Run 서브 메뉴 중 Debug Configuration 을 선택한다.
- 2) Spring Boot App 항목을 선택한 후 New Launch Configuration 버튼을 누르면 (또는 Spring Boot App 항목 더블클릭)

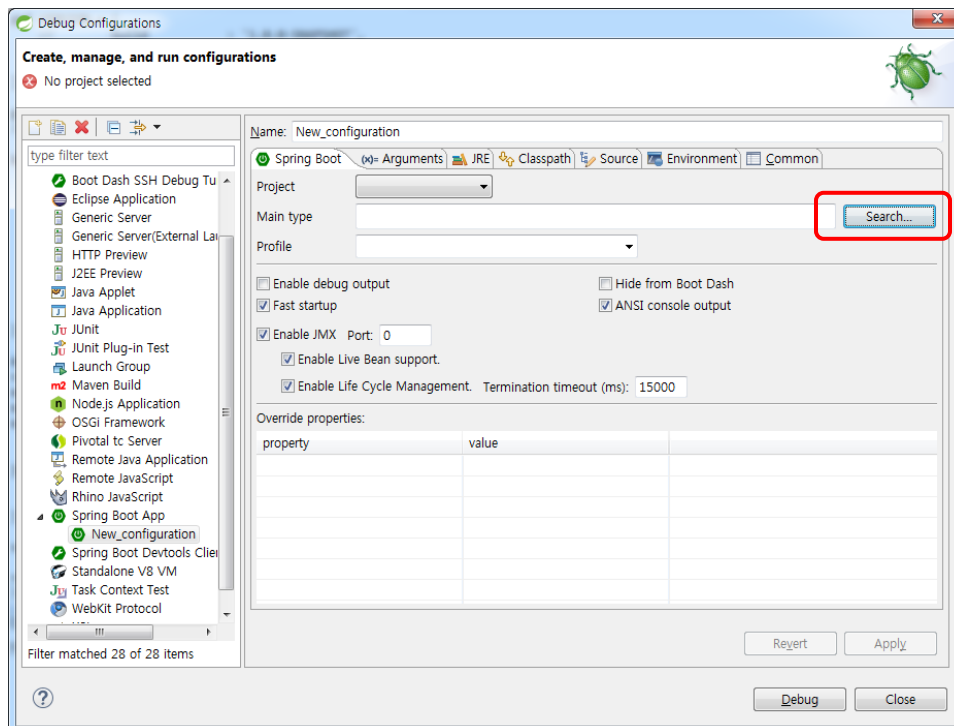
프로젝트명:

단계:

담당팀:



3) 아래와 같은 New configuration 을 지정할 수 있도록 Sub 메뉴가 나타난다.

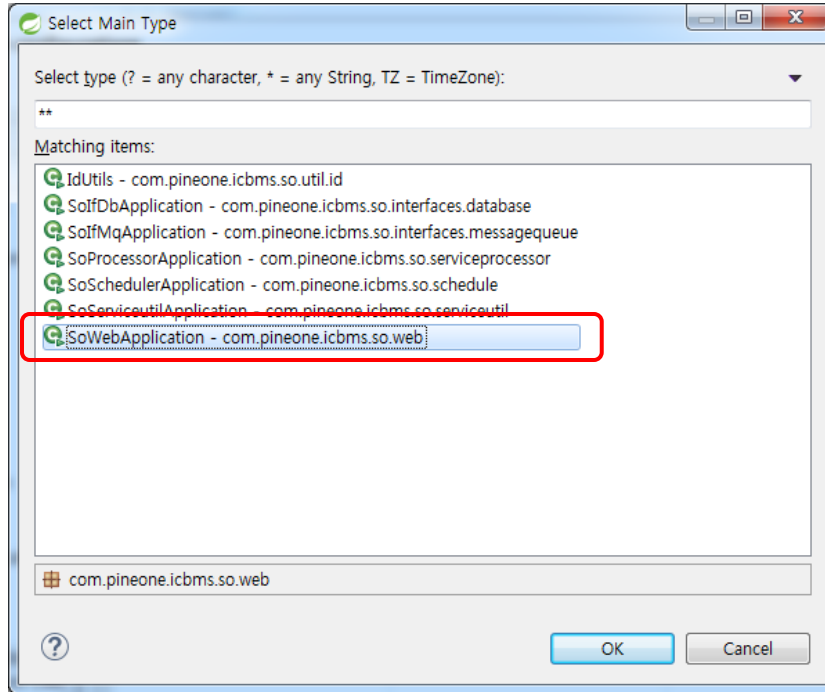


프로젝트명:

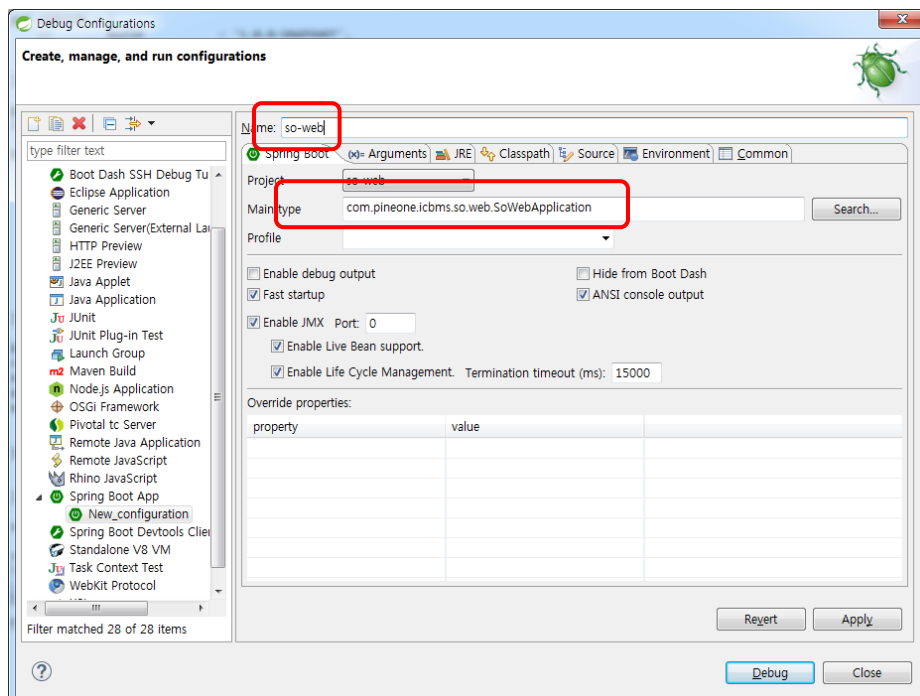
단계:

담당팀:

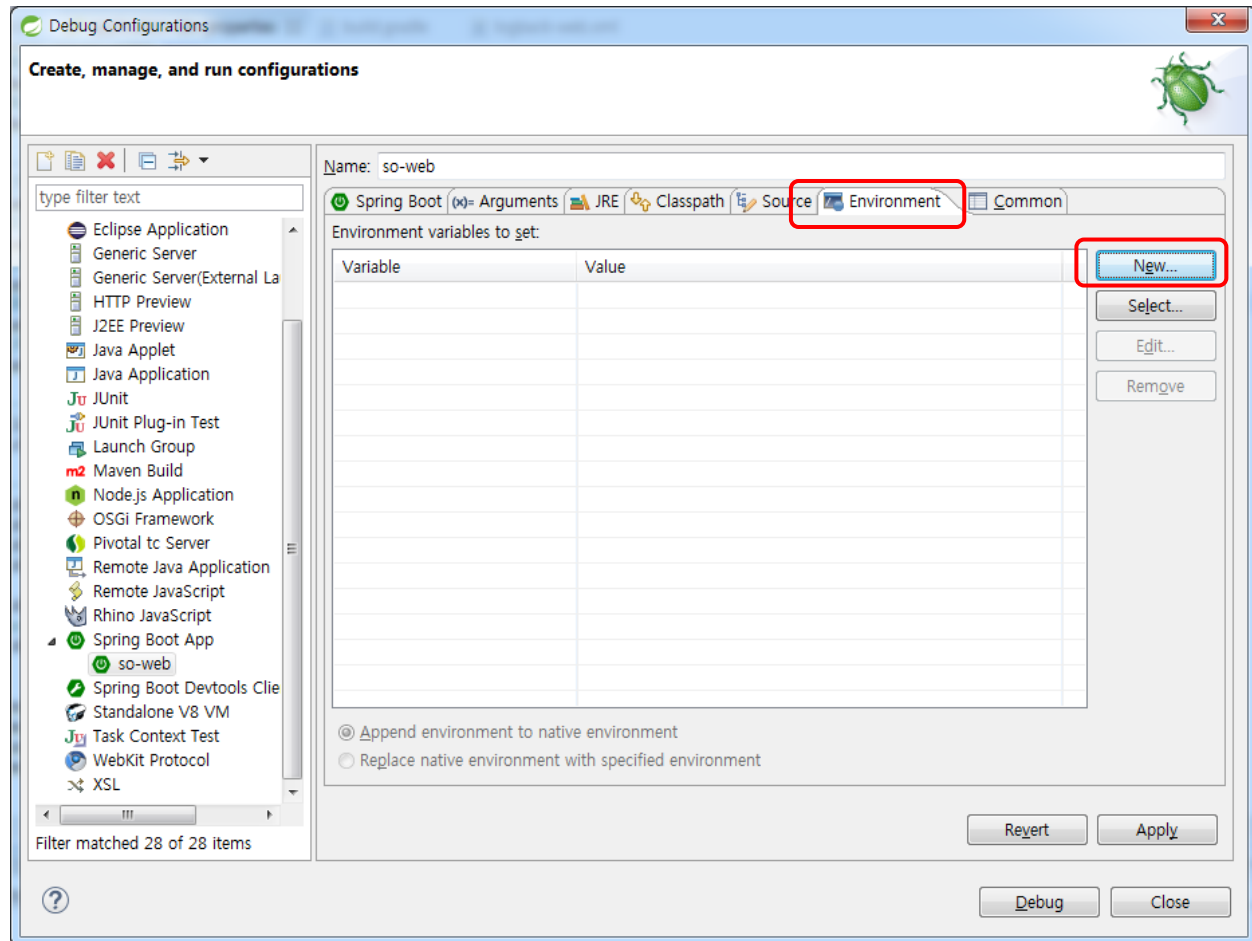
4) Main Type 을 입력하기 위하여 search 버튼을 누른 후 아래와 같은 창이 나타나며, SoWebApplication 을 선택한다.



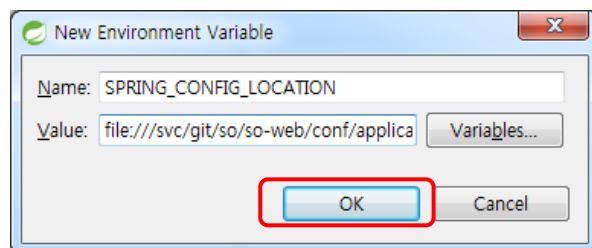
5) Name 항목에 so-web 이라고 입력한다.



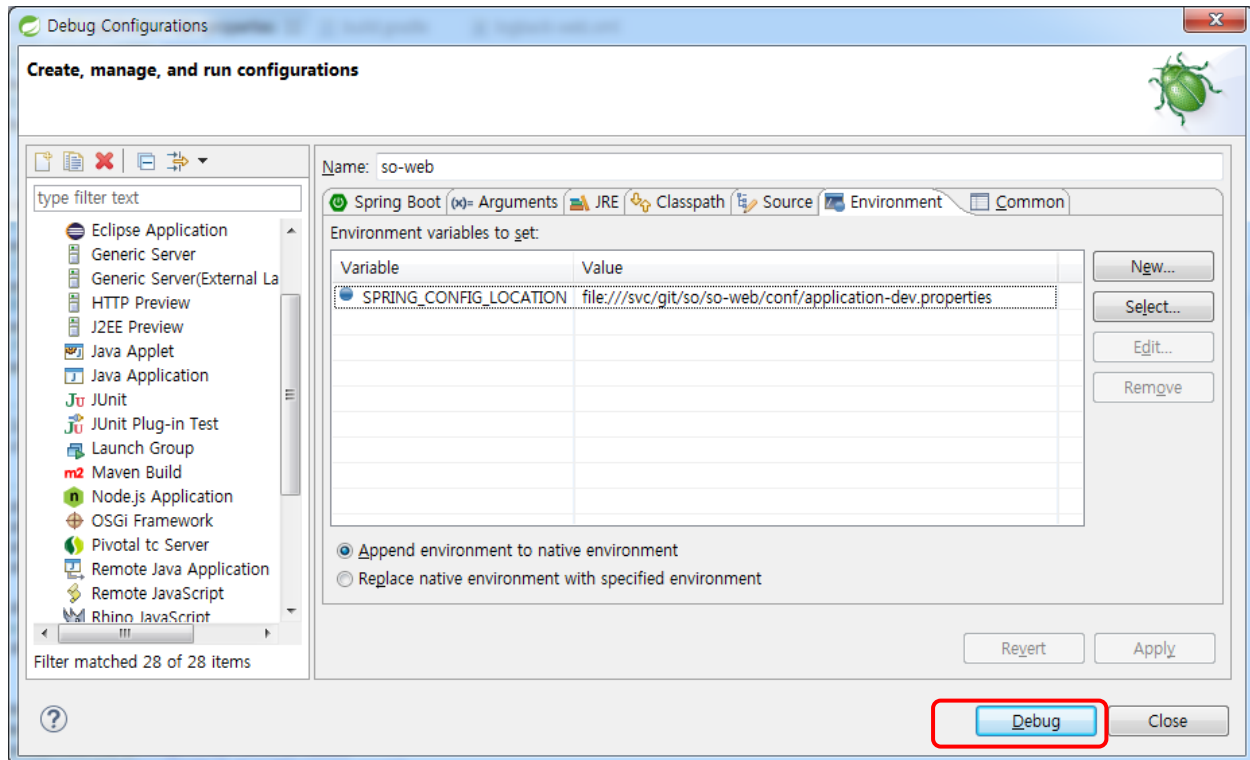
6) Environment 탭을 선택한 후 New 버튼을 누른다.



7) Name 항목에 SPRING\_CONFIG\_LOCATION, Value 항목에 application-dev.properties 가 위치한 file:///svc/git/so/so-web/conf/application-dev.properties (경로가 다를 경우 수정) 을 입력한 후 OK 를 누른다.



8) Debug 버튼을 눌러서 Debug 모드로 서버를 기동한다.



## 4.6 배포 환경에서 서버 실행 하기

### 4.6.1 SO 실행 환경 설정

최초 실행시에 한번만 환경 설정을 하고, 설정 내용이 바뀌지 않으면 so 서버 실행 파일만 교체한다.

- 1) JDK 를 설치한다.
- 2) kafka 와 zookeeper 를 설치하고 실행한 후 topic 을 생성한다.
- 3) MariaDB 를 설치하고 실행한다.
- 4) MariaDB 에 table 과 index 를 생성한다.
- 5) Maria DB 에 SO 서비스에 필요한 Database 를 생성한다.
- 6) 빌드 환경에서 사용했던 환경을 서버로 복사한다.

```
$ cd /home/myhome/
```

```
$ mkdir -p /svc/apps/so/so/conf
```

```
$ cp application-product.properties /svc/apps/so/so/conf/
```

```
$ cp logback-web.xml /svc/apps/so/so/conf/
```

- 7) application-product.properties 와 logback-web.xml 파일을 서버 운영 환경에 맞도록 수정한다.

## 4.6.2 SO 실행

1) 빌드된 target 폴더(so-web/build/libs)에서 so-3.0.0.0.jar 파일을 실행하고자 하는 서버의 위치로 복사를 한다

```
$ cp so-3.0.0.0.jar /svc/apps/so/so
```

2) 서버를 실행한다.

```
$ java -jar -Dspring.profiles.active=dev /svc/apps/ so-3.0.0.0.jar --
spring.config.location=/svc/apps/so/so/conf/application-product.properties &
```

## 4.6.3 SO 서버 동작 확인

- logback-web.xml 에 지정된 로그 디렉토리안에 생성된 로그 파일을 살펴본다.
- database 에 생성된 tracking table 과 session\* 에 생성된 table 의 내용을 참조한다.
- POSTMAN 을 사용하여 session Log 를 살펴본다.
- POSTMAN 시험을 위한 스크립트 파일은 release 페이지에서 다운받을 수 있습니다.
- POSTMAN 을 활용한 SO 서버 시험에 관한 자세한 내용은 [Test 페이지](#)에서 확인할 수 있습니다.

## 5. SO Framework Server TEST

다운 받은 SO Framework 서버를 테스트 하기 위한 방법을 설명 합니다.

### 5.1 POSTMAN 으로 생성한 SO(Service Orchestration)를 실행

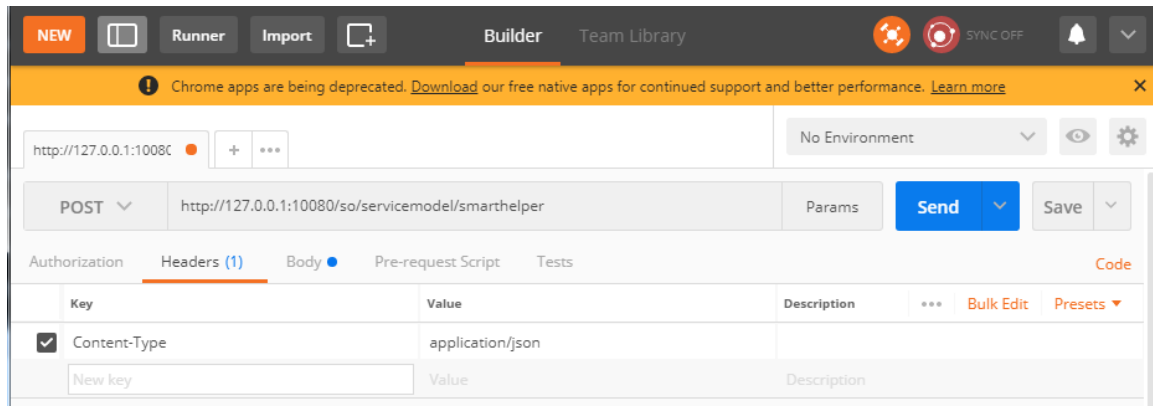
- Method 를 POST 로 바꿈.
- url 에 **http://127.0.0.1:10080/so/servicemodel/smarthelper** 입력  
(remote 인 경우에는 remote 서버 주소 입력)
- Headers 를 선택하고 Key 와 Value 위치에 **Content-Type** 과 **application/json** 을 각각 입력



프로젝트명:

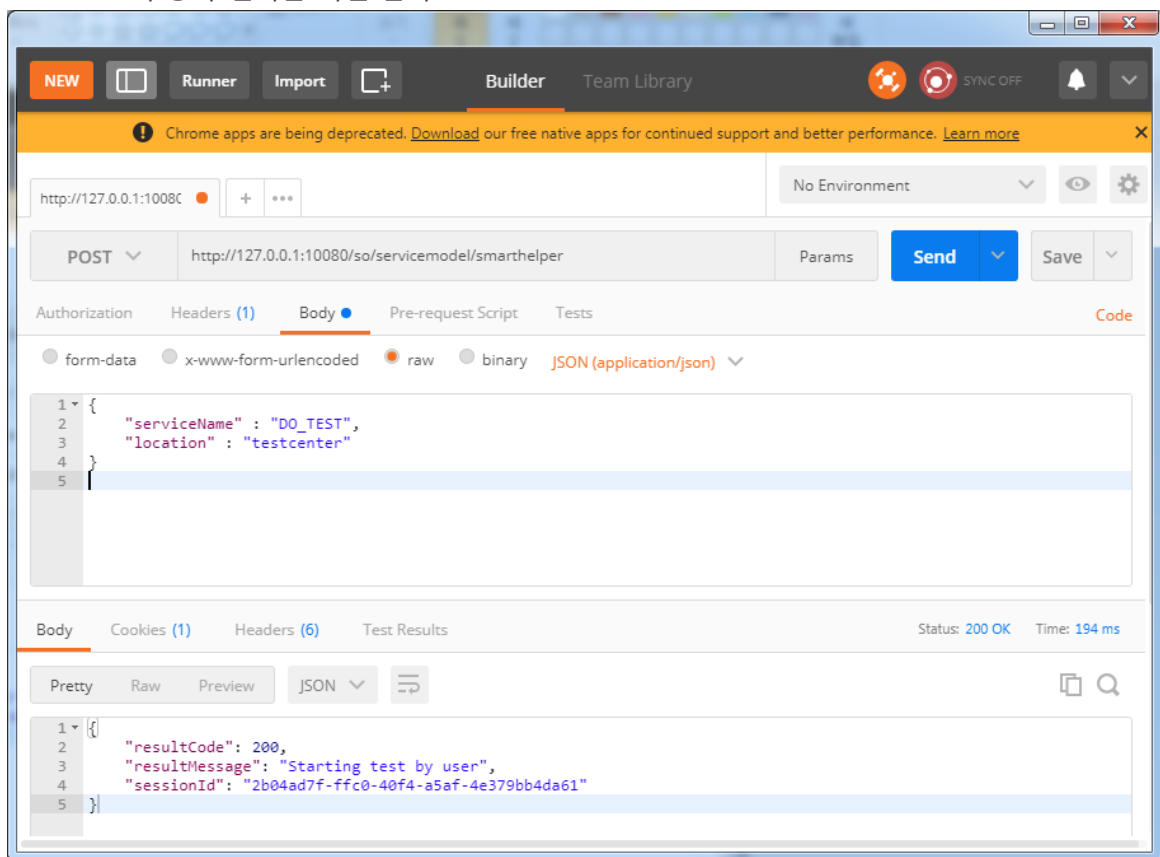
단계:

담당팀:

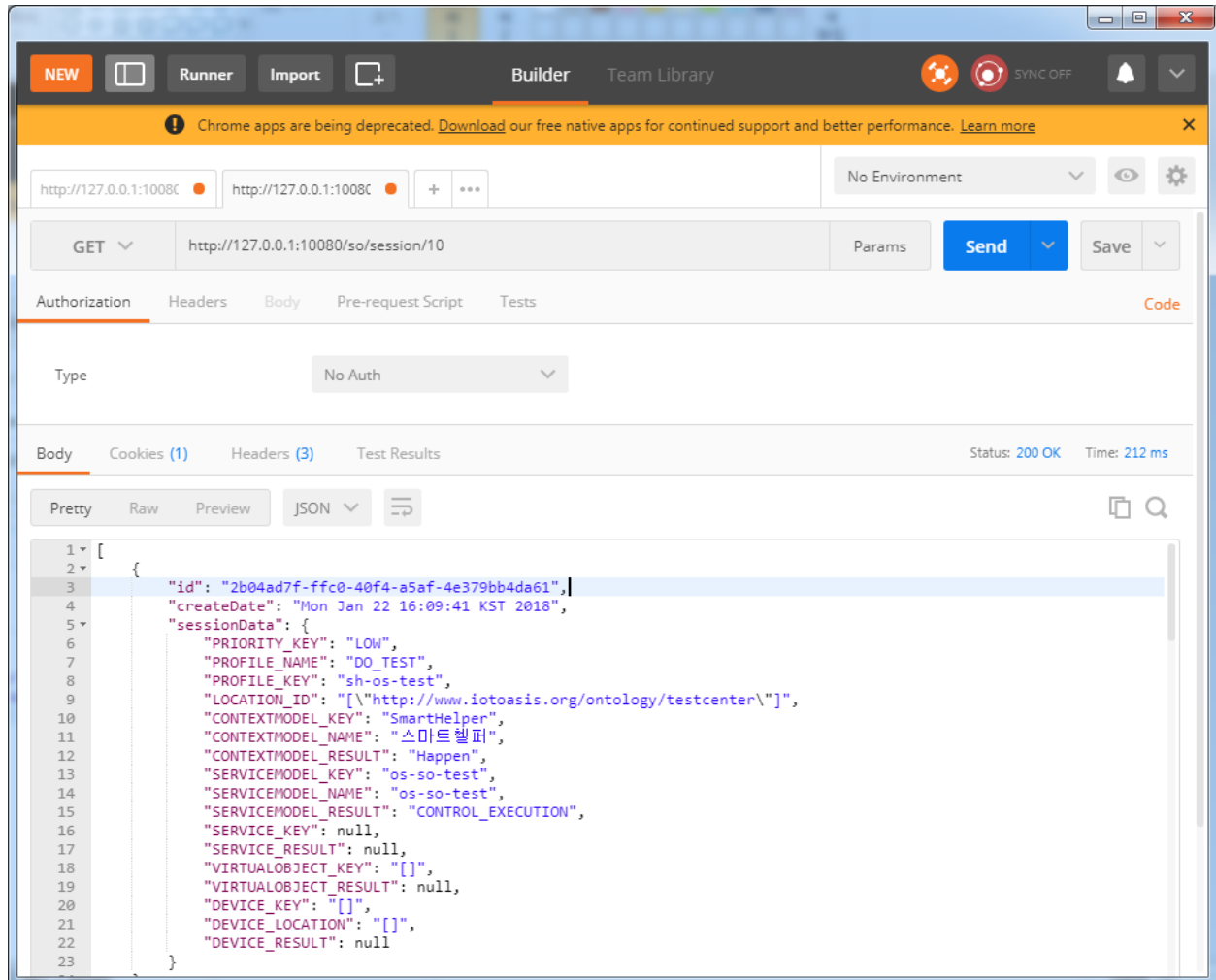


- Body 를 선택한 후, raw 를 선택 하고, 아래와 같은 내용 입력  

```
{
  "serviceName" : "DO_TEST",
  "location" : "testcenter"
}
```
- Send 버튼 클릭
- SO Server 의 동작 결과를 확인 한다.



- 위의 명령이 성공하면 +버튼을 눌러 새로운 탭을 하나 만든 후 URL 에 `http://127.0.0.1:10080/so/session/10` 입력하고
- Send 버튼 클릭
- 결과 창에서 다음과 같은 session Log 를 확인한다.



## 6. Q&A