Apache Storm

김남협

2019.08.07

Data & Knowledge Engineering Lab.

Department of Computer Science, Kangwon National University

Contents

- Apache Storm Outline
- Apache Storm Concept
- Apache Storm Install
- Apache Storm Example

Apache Storm Outline

Storm 개요

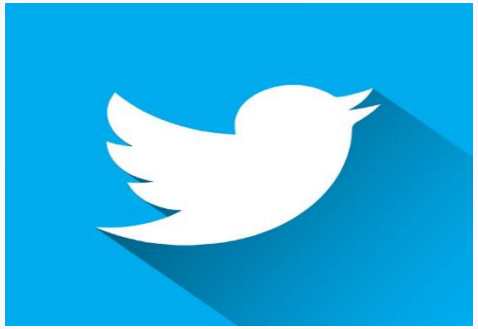
- 분산 연산 프레임워크
- 목적
  - **분산 스트림 처리**, 실시간 데이터 처리
  - 데이터 증가에 따른 처리 시간 지연을 없애려고 하는 것이 분산 병렬 처리 기술
  - 데이터 처리 지연을 최소화하기 위해 데이터를 먼저 처리하고 저장하는 방식의 데이터 처리 모델이 스트림 처리 기술
  - 분산 병렬 처리 기술 + 스트림 처리 기술 = 분산 스트림 처리
- 역사
  - BackType팀의 Nathan Marz가 최초 개발
  - 트위터에 인수합병 후 트위터의 실시간 분석과 최적화, 안티 스팸 등에 활용
  - Storm이라는 이름의 프로젝트를 오픈소스로 공개
- 주 개발 언어
  - JAVA: 주요 인터페이스
  - CLOJURE: 간결성을 위한 핵심 로직
    - Rich Hickey가 만든 LISP 프로그래밍 언어의 방언으로 범용 함수용 언어
    - 멀티스레드 프로그램의 개발에 용이
    - JVM 바이트코드로 컴파일
    - 별다른 복장한 과정 없이 JVM 환경에 배포 가능

Storm 특징

- 쉬운 프로그래밍 모델
  - 맵리듀스가 병렬처리 프로세스 구현의 복잡도를 낮춰주는 것과 같이 Storm 또한 분산 real-time processing 구현의 복잡도를 낮춤
- 다양한 언어로 구현 가능
  - Clojure, Java, Ruby, Python 등 사용자가 익숙한 언어 사용 가능
- 장애 대응성
  - Zookeeper를 통해서 전체 클러스터의 운영 상태를 감시 및 장애를 복구
- 메시지 전달 보장
  - 장애 여부와 상관없이 메시지의 유실 없이 최소한 한 번 메시지가 처리 될 수 있게 지원
    - 하지만 반대로 한 번 이상 같은 메시지가 중복 처리 될 수 있음
- 빠른 속도
  - Netty 혹은 ZeroMQ를 통해 메시지를 빠르게 처리할 수 있도록 설계

Storm의 활용

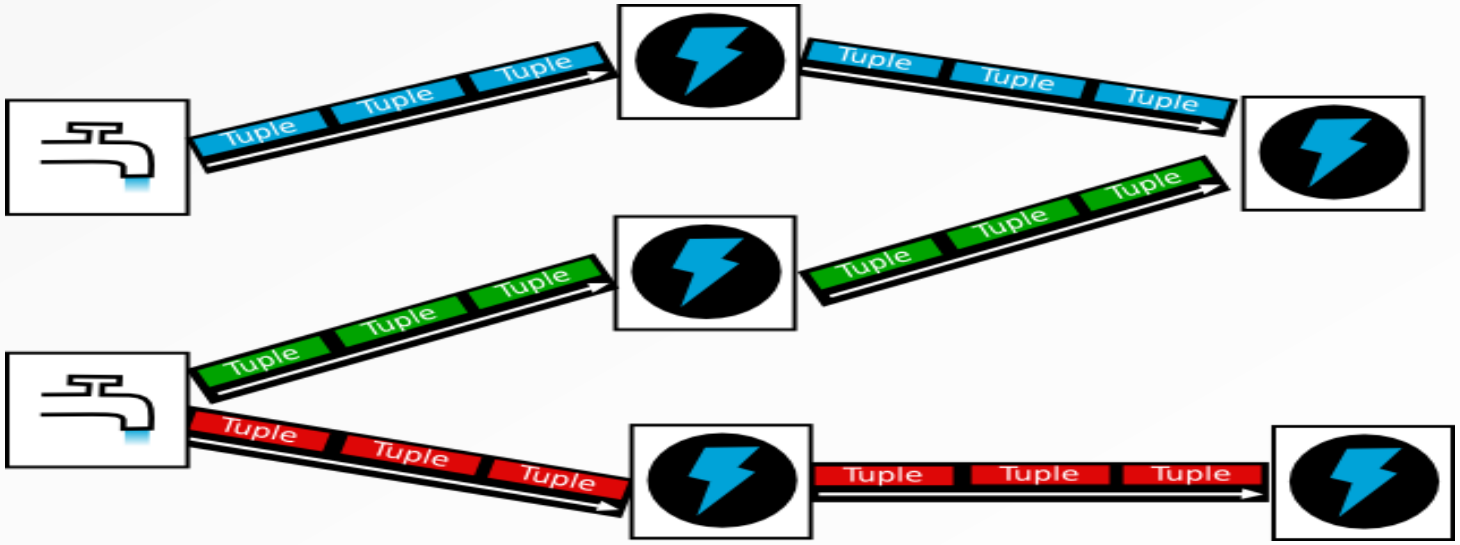
- Twitter
  - Storm은 검색, 실시간 분석, 개인화, 검색, 수익 최적화 등의 응용 프로그램에 이르는 광범위한 시스템을 Twitter 시스템을 지원
- Yahoo
  - 현재 저 지연 프로세싱의 컨버전스를 지원하는 차세대 플랫폼 개발 중 hadoop은 일괄 처리만 가능할 뿐 실시간 처리가 불가능 해서 사용자 이벤트, 콘텐츠 피드 및 응용 프로그램 등의 기술 활용이 부족하여 Storm 사용
- Spotify
  - 스트리밍 음악 플랫폼으로 4천만 명의 활성 사용자에게 음악 추천, 모니터링, 분석 및 광고 티켓팅을 포함한 광범위한 실시간 기능을 제공



Apache Storm Concept

Apache Storm의 기본 구성 요소

- Topology
  - Storm의 분산 처리 구조
  - 구성
    - Data Stream, Spout(데이터의 입력), Bolt(데이터의 처리)



↑Topology

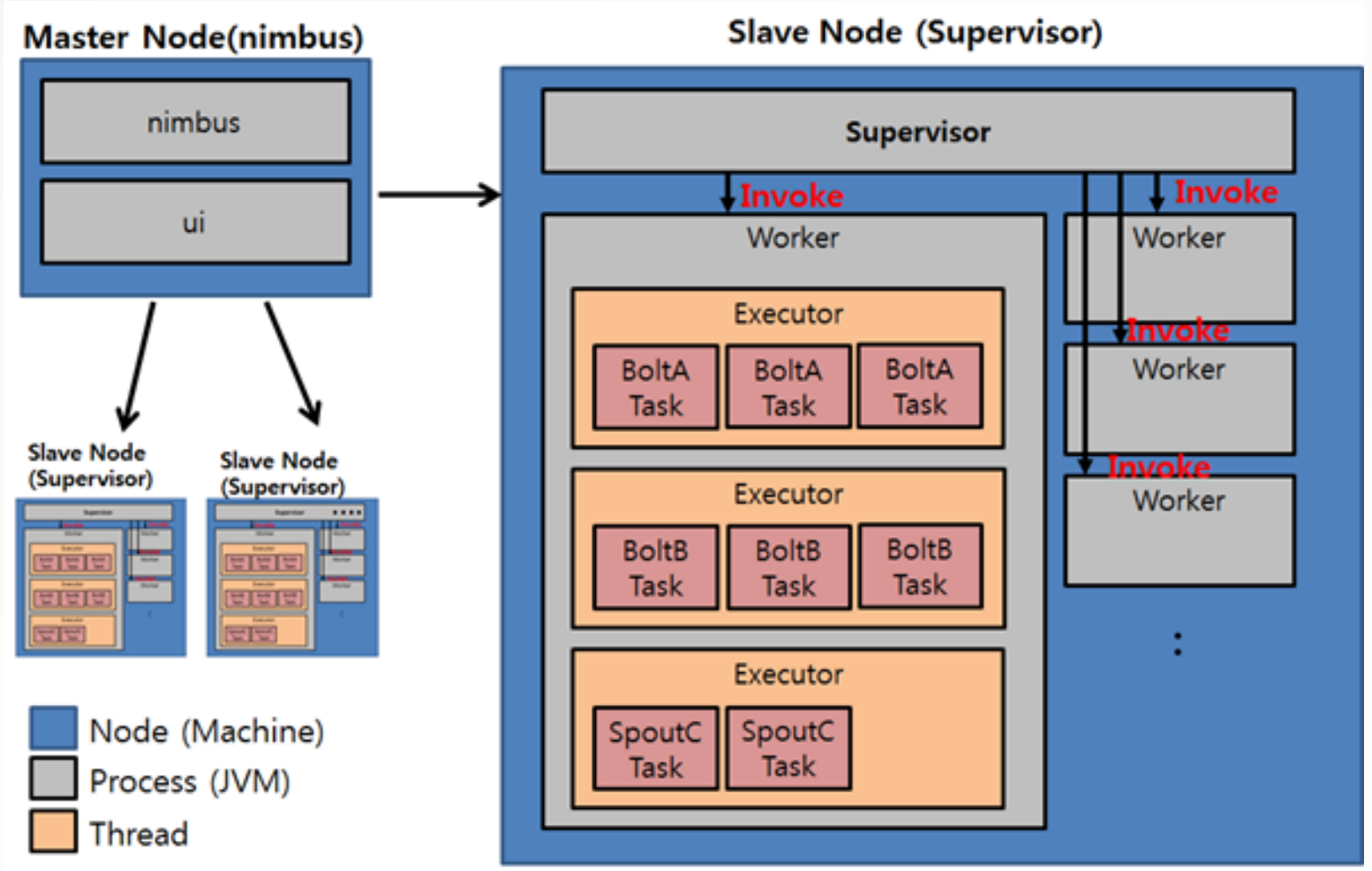
Apache Storm의 기본 구성 요소

- Stream
  - 연속된 Tuple의 흐름
    - Tuple은 Key/value 쌍의 목록으로서 Storm의 기본 데이터 구조체
- Spout
  - 데이터가 Topology 안으로 들어오는 입구
  - 데이터를 Tuple로 변환하여 Stream으로 내보냄
- Bolt
  - 실시간 연산의 연산자나 함수
  - 다수의 스트림을 입력 받아 데이터를 처리
  - 선택적으로 하나 이상의 스트림을 내보냄

Apache Storm의 병렬처리를 이해하기 위한 개념(1/2)

- Storm의 병렬 처리를 이해하기 위한 네 가지 개념
  - Node
    - Node 내부는 Nimbus와 Supervisor로 구성
    - Nimbus와 Supervisor는 프로세스가 작동하는 물리적인 서버
    - Nimbus는 전체 Node 중에서 하나의 프로세스에서만 작동
    - 여러 대를 작동할 수 있지만 Supervisor의 역할 자체가 해당 Node를 관리하는 역할이기 때문에 하나의 Node에 여러 개의 Supervisor를 작동할 필요는 없음
  - Worker
    - Supervisor가 작동하는 Node 안에서는 실질적으로 Spout과 Bolt를 실행시키는 역할 수행
    - MasterNode가 SlaveNode와 연결이 되어있으면 SlaveNode 안에 Supervisor를 두어 Worker 들을 동작시키는 형태이다
  - Executor
    - Worker내에서 수행되는 하나의 자바 Thread를 의미
  - Task
    - Task는 Bolt나 Spout의 객체를 의미
    - Task는 Executor(Thread)에 의해서 수행
    - 각 SlaveNode는 SlaveNode가 제대로 작동하는 등의 통제를 위한 Supervisor 프로세스가 하나씩 존재한다

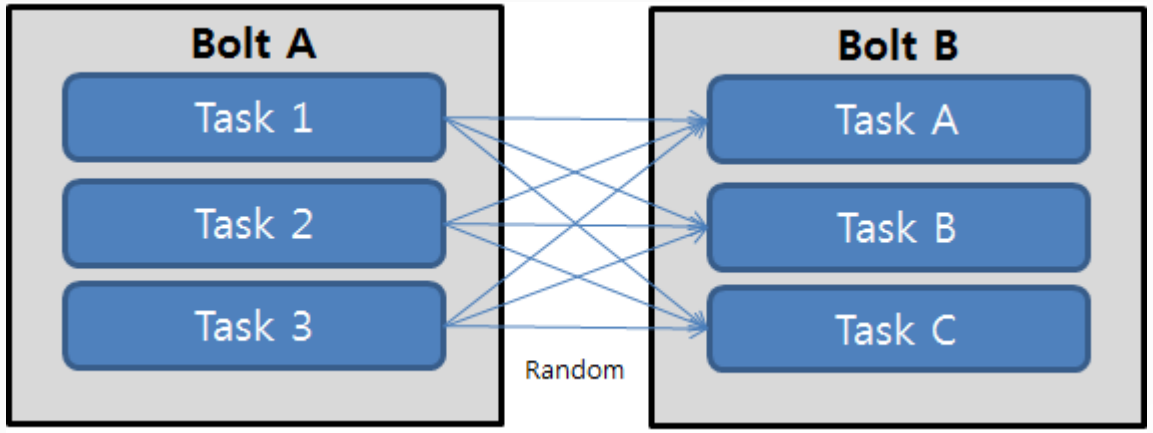
Apache Storm의 병렬처리를 이해하기 위한 개념(2/2)



Grouping(1/3)

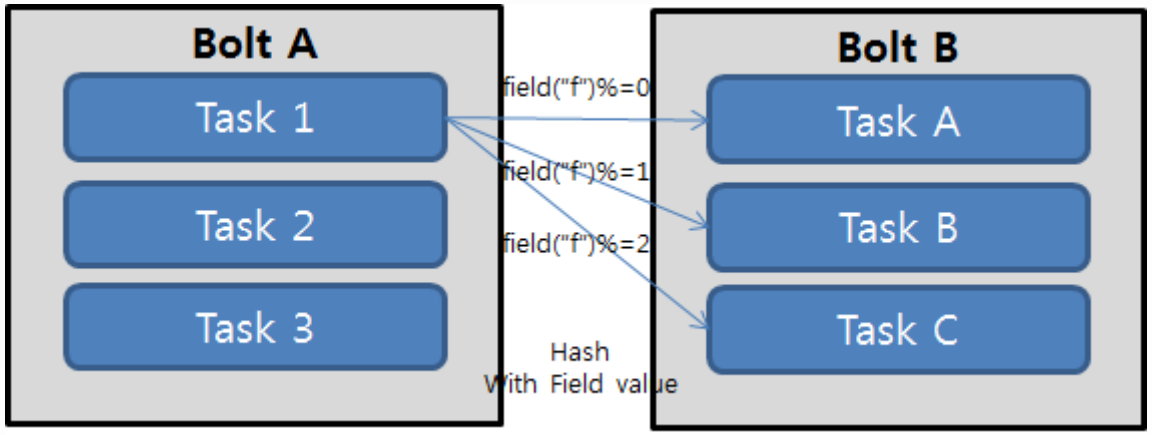
Shuffling

- Bolt A에서 Bolt B로 라우팅을 한다고 가정 하였을 때 Bolt A안에 있는 Task들이 Bolt B에 있는 Task중 아무 Task로 라우팅을 하는 방식
- 이 때 Bolt A의 Task1이 Bolt B의 Task A로 라우팅을 하였을 때 Task2가 Bolt B의 Task A로 라우팅이 가능
- Random 라우팅이 가능한 방식



Field

- Bolt A와 Bolt B에서 라우팅을 한다 가정하였을 때 볼트 내부에서 규칙성을 가지고 라우팅을 하는 방식
- 규칙을 이용하여 나온 계산 값에 의하여 계산 값과 일치하는 Task로 연결시키는 방식
- Bolt에서 로컬 캐쉬를 사용하거나 일치하는 데이터가 나왔을 경우 캐쉬의 Hit을 높이는데 유용하게 사용이 가능



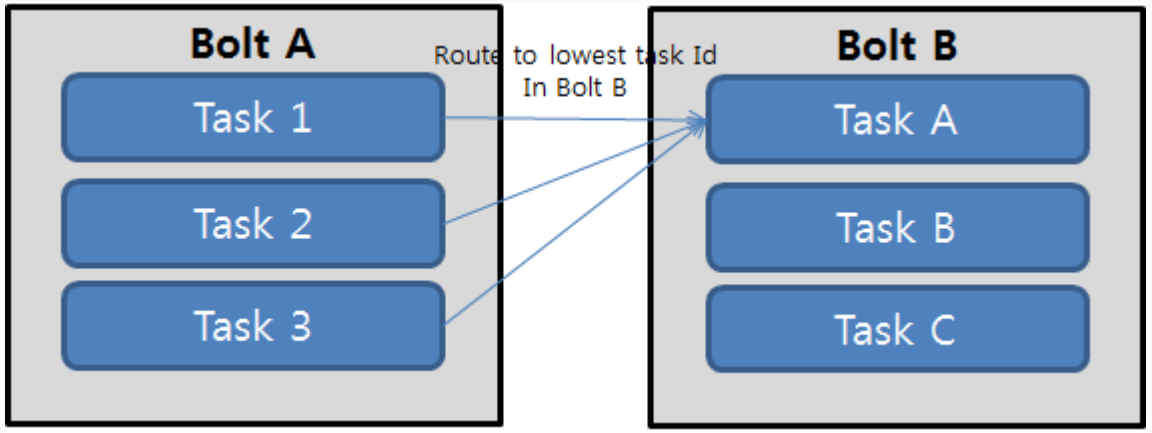
Grouping(2/3)

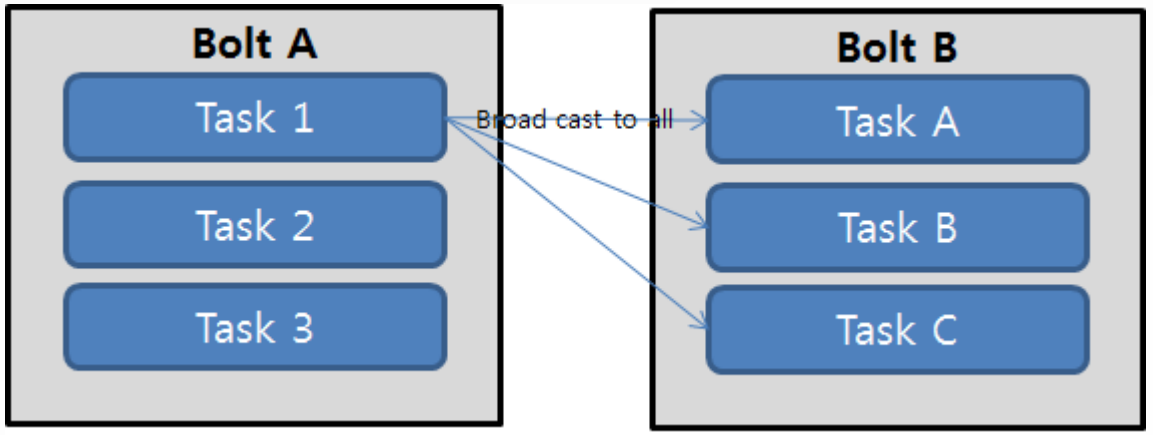
Global

- 볼트 내부의 모든 개념을 모아서 전달하는 Grouping
- 예를 들어 Bolt A에서 Task1,Task2,Task3의 정보를 모두 모아서 Bolt B의 Task A로 정보를 전달하는 개념
- 분산해서 연산한 값을 모두 모아서 합계를 하는 등의 방식으로 사용

All

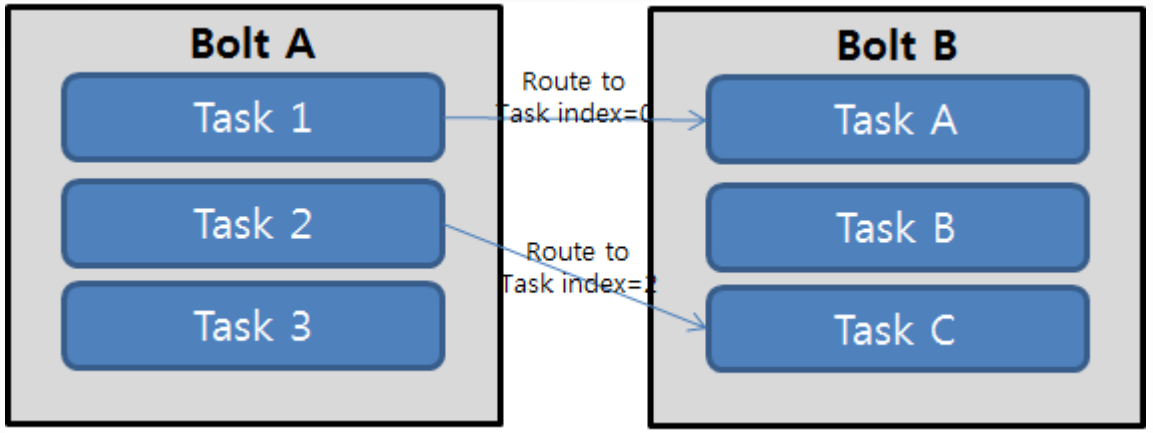
- All 그룹핑은 일종의 Broadcasting개념으로 Bolt A의 하나의 Task가 Task 메시지를 전송하 Bolt B의 모든 메시지를 받는 형태
- 각 Task들에 설정 변경 등에 유용





Grouping(3/3)

- **Direct**
  - 사용자가 직접 라우팅 하는 방식을 지정하는 형식
  - Bolt A의 Task와 Bolt B의 특정 Task의 라우팅을 지정할 수 있는 그룹핑
  - 이 때 주의해야 할 점이 Task의 ID가 아닌 Index로 값을 넘김



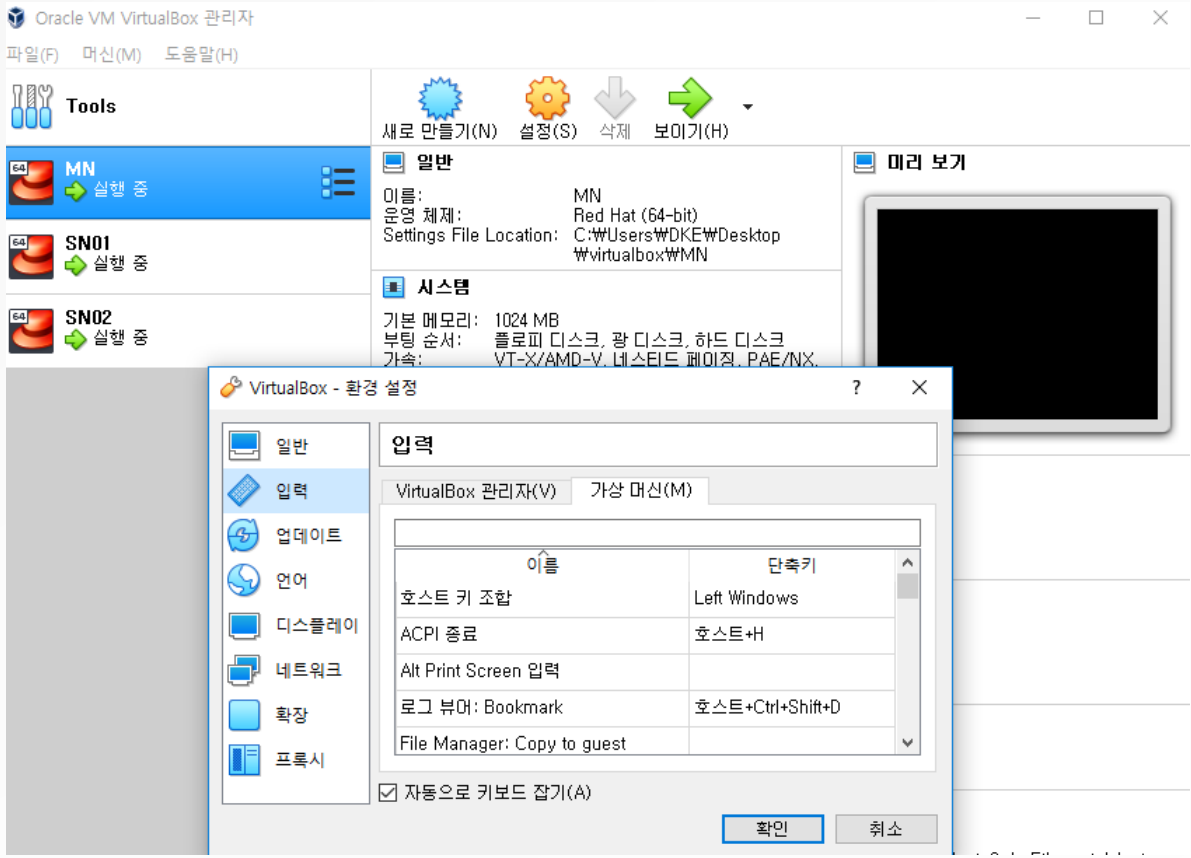
Apache Storm 설치

Apache Storm 설치 주의사항

- Oracle VM VirtualBox의 설치는 버전에 상관 없이 공식 홈페이지에 접속 후 다운
  - https://www.virtualbox.org/
- CentOs 설치시 버전에 따라 오류가 발생할 수 있으므로 CentOs-7-x86_64-DVD-1810.iso 권장
- 이하 버전 사용시 설치는 되나 여러가지 오류가 발생
  - 네트워크 연결 불가
  - 게이트웨이 인터페이스 사용은 가능하나 실질적인 사용이 불가능
  - 이하 버전 다운 할 경우 시간 소모가 크므로 주의할 것

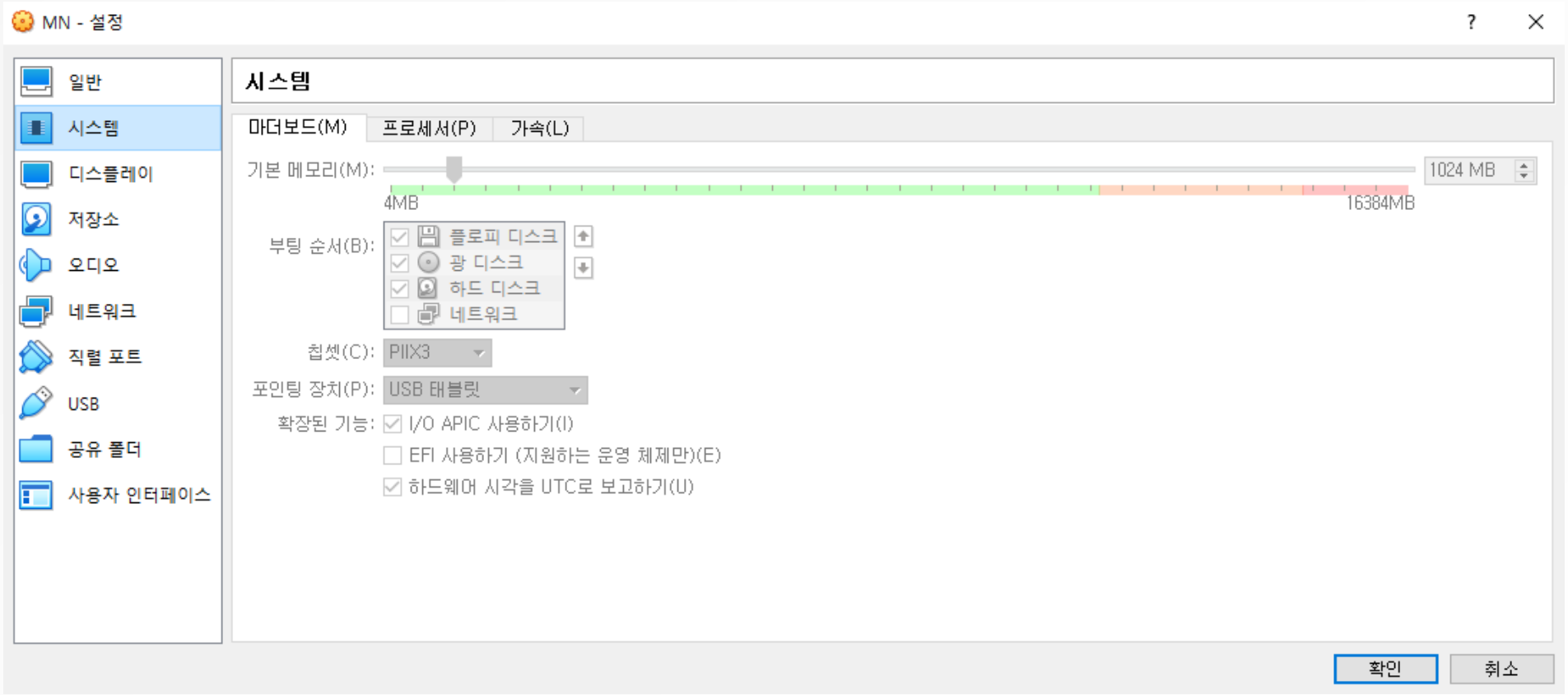
VirtualBox 기본설정(1/2)

- 파일→환경설정→입력→가상머신
  - 호스트 키 조합 Left Windows
  - 가상머신에서 local로 나오는 설정 키(미설정시 shift사용의 제한이 걸림)



VirtualBox 기본설정(2/2)

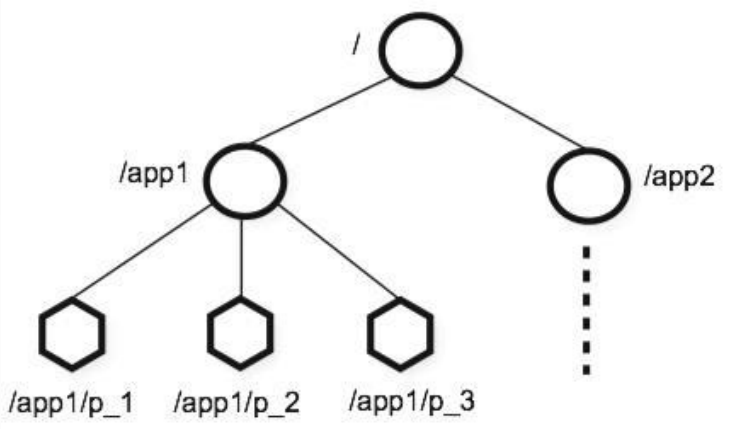
- MasterNode→설정→포인팅 장치→USB 태블릿 변경
- MasterNode→설정→칩셋→PIIX3 변경
  - 마우스 사용을 위한 설정



Coordination Service

- 분산 시스템을 설계하는 과정에서 여러 가지 이슈와 문제점들이 발생
  - 분산 시스템 간의 정보를 어떻게 공유할 것인가?
  - 클러스터에 있는 서버들의 상태를 어떻게 확인할 것인가?
  - 해결을 위한 Coordination Service 등장

- Coordination Service
  - 분산 시스템 내에서 중요한 상태 정보나 설정 정보 등을 유지하는 역할을 하는 시스템
  - 이중화를 통한 안전 고리 역할
  - 정보들의 보안, 설정 등의 유지 보수 형성
  - 이러한 성격에 알맞는 Zookeeper 사용



Zookeeper 설치

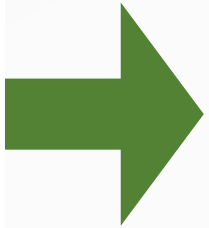
- Zookeeper
  - 설치
  - $wget http://apache.tt.co.kr/zookeeper/zookeeper-3.4.14.tar.gz
    - wget은 웹에서 파일을 다운받는 명령어
    - 버전 존재 안할 시 http://apache.tt.co.kr/zookeeper 접속하여 버전 확인후 최신 버전 설치
  - $tar xzf zookeeper-3.4.14.tar.gz (압축해제)
  - $ln –s zookeeper-3.4.10 zookeeper (명칭 설정)

```
lrwxrwxrwx.   1 hadoop hadoop       18  7월   9 15:22 storm -> apache-storm-1.1.2
lrwxrwxrwx.   1 hadoop hadoop       16  7월   8 19:14 zookeeper -> zookeeper-3.4.14
drwxr-xr-x.  16 hadoop hadoop     4096  7월  15 16:13 zookeeper-3.4.14
```

  - 환경설정 (Zookeeper로 이동)
    - $cp ./conf/zoo_sample.cfg ./conf/zoo.cfg
      - ✓ zoo.cfg vi로 수정(21page 참조)
      - ✓ Zookeeper/bin의 zkServer.sh를 vi로 수정(22page 참조)
    - $scp –r zookeeper SN계정@SN IP :~/ (scp 명령어로 MasterNode에 설정했던것들을 SlaveNode들에게도 전파)
    - $mkdir data (MasterNode,SlaveNode 모두 data directory생성)
    - MasterNode에는 $echo #1 >myid
    - SlaveNode에는 $echo #2 > myid (SlaveNode 수가 증가할 수록 숫자도 증가)
  - Zookeeper 실행
    - $./zookeeper/bin/zkServer.sh start (stop은 종료)

Zookeeper 설치(zoo.cfg)

```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
dataDir=/tmp/zookeeper
# the port at which the clients will connect
clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1
```



```
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
dataDir=/home/hadoop/zookeeper/data
# the port at which the clients will connect
clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
# http://zookeeper.apache.org/doc/current/zookeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1

server.1=MN:2888:3888
server.2=SN01:2888:3888
server.3=SN02:2888:3888

# Enable regular purging of old data and transaction logs every 24 hours
autopurge.purgeINterval=24
autopurge.snapRetainCount=3
```

Zookeeper 설치(zkServer.sh)

- Zoo_Log_DIR=/home/Hadoop/zookeeper/logs 추가
  - log를 저장할 위치를 정의하지 않으면 쉘을 실행한 위치에 생성

```
# If this scripted is run out of /usr/bin or some other system bin directory
# it should be linked to and not copied. Things like java jar files are found
# relative to the canonical path of this script.
#

ZOO_LOG_DIR=/home/hadoop/zookeeper/logs

# use POSTIX interface, symlink is followed automatically
ZOOBIN="${BASH_SOURCE-$0}"
ZOOBIN="$(dirname "${ZOOBIN}")"
ZOOBINDIR="$(cd "${ZOOBIN}"; pwd)"
```

Storm 설치

- Storm
  - 다운로드
    - $wget http://mirror.navercorp.com/apache/storm/apache-storm-1.2.3/apache-storm-1.2.3.tar.gz
      - ✓ 버전 없는 경우 http://mirror.navercorp.com/apache/storm 접속 후에 버전 확인 후 최신 버전 설치
    - $tar xzf apache-storm-1.2.3. tar.gz
    - $ln –s apache-storm-1.2.3 storm
  - 환경설정(apache-storm-1.2.3 이동)
    - Vi conf/storm.yaml 파일 수정(24page 참조)
    - $mkdir data
    - $scp –r storm SN계정@SN IP :~/
  - Storm 실행
    - systemctl stop firewalld(포트 설정을 할 경우 안 해줘도 됨)
    - MasterNode, SlaveNode zookeeper 우선 실행
    - systemctl stop firewalld(포트 설정을 할 경우 안 해줘도 됨)
    - $nohup storm daemon_name > daemon_name.log 2>&1 & (실행 명령어)
    - MasterNode(daemon_name): nimbus, ui, logviewer 순으로 실행
    - SlaveNode(daemon_name): supervisor, logviewer 순으로 실행

Storm설치(Storm.yaml)

```
########### These MUST be filled in for a storm configuration
# Local directory path
storm.local.dir: "/home/hadoop/storm/data"

# Zookeeper Nodes
storm.zookeeper.servers:
- "MN"
- "SN01"
- "SN02"

#     - "server1"
#     - "server2"
#

#Nimbus host
nimbus.seeds: ["MN"]

supervisor.slots.ports:
- 6701
- 6702
- 6703
- 6704
```

Storm 실행(1/4)

- 방화벽 해제
  - 포트 설정을 해주지 않을 경우 방화벽을 해제해줘야 함
  - $systemctl stop firewalld (방화벽 해제)
  - $systemctl status firewalld(방화벽 상태 inactive 확인)

```
[hadoop@SN01 data]$ systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; enabled; vendor preset: enabled)
   Active: inactive (dead) since 금  2019-08-02 18:08:58 KST; 7h ago
     Docs: man:firewalld(1)
  Process: 2753 ExecStart=/usr/sbin/firewalld --nofork --nopid $FIREWALLD_ARGS (code=exited, status=0/SUCCESS)
 Main PID: 2753 (code=exited, status=0/SUCCESS)

8월  02 18:01:12 SN01 systemd[1]: Starting firewalld - dynamic firewall daemon...
8월  02 18:01:13 SN01 systemd[1]: Started firewalld - dynamic firewall daemon.
8월  02 18:08:58 SN01 systemd[1]: Stopping firewalld - dynamic firewall daemon...
8월  02 18:08:58 SN01 systemd[1]: Stopped firewalld - dynamic firewall daemon.
[hadoop@SN01 data]$
```

  - MasterNode, Slavenode 전부 방화벽 해제

Storm 실행(2/4)

- MasterNode, SlaveNode zookeeper 실행
  - $zkServer.sh start
  - $jps(QuorumPeerMain 작동 확인)

```
[hadoop@MN ~]$ cd zookeeper
[hadoop@MN zookeeper]$ cd bin
[hadoop@MN bin]$ ll
합계 52
-rwxr-xr-x. 1 hadoop hadoop  232  3월   7 01:50 README.txt
-rw-rw-r--. 1 hadoop hadoop 1281  7월   9 15:15 index.html
-rwxr-xr-x. 1 hadoop hadoop 1937  3월   7 01:50 zkCleanup.sh
-rwxr-xr-x. 1 hadoop hadoop 1056  3월   7 01:50 zkCli.cmd
-rwxr-xr-x. 1 hadoop hadoop 1534  3월   7 01:50 zkCli.sh
-rwxr-xr-x. 1 hadoop hadoop 1759  3월   7 01:50 zkEnv.cmd
-rwxr-xr-x. 1 hadoop hadoop 2919  3월   7 01:50 zkEnv.sh
-rwxr-xr-x. 1 hadoop hadoop 1089  3월   7 01:50 zkServer.cmd
-rwxr-xr-x. 1 hadoop hadoop 6812  7월   9 15:11 zkServer.sh
-rwxr-xr-x. 1 hadoop hadoop  996  3월   7 01:50 zkTxnLogToolkit.cmd
-rwxr-xr-x. 1 hadoop hadoop 1385  3월   7 01:50 zkTxnLogToolkit.sh
-rw-rw-r--. 1 hadoop hadoop 3878  7월   8 19:16 zookeeper.out
[hadoop@MN bin]$ ./zkServer.sh start
```

Storm 실행(3/4)

- Storm 실행
  - $nohup storm daemon_name > daemon_name.log 2>&1 & (실행 명령어)
    - MasterNode(daemon_name): nimbus, ui, logviewer 순으로 실행
    - SlaveNode(daemon_name): supervisor, logviewer 순으로 실행
  - MasterNode $jps 실행결과

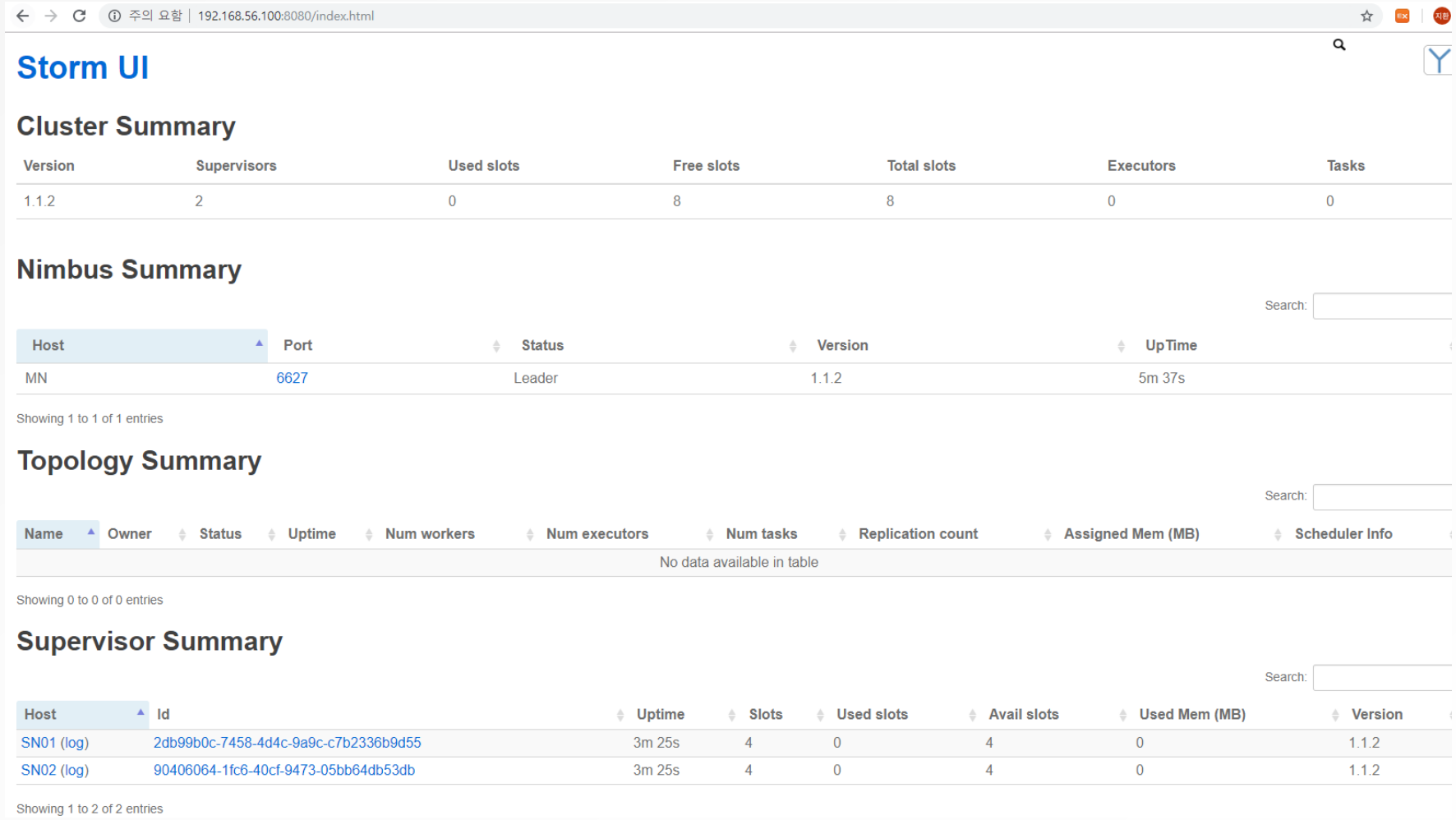
```
[hadoop@MN bin]$ jps
4656 core
4578 nimbus
12420 Jps
4265 QuorumPeerMain
4761 logviewer
```

  - SlaveNode $jps 실행결과

```
[hadoop@SN01 bin]$ jps
4160 Jps
3665 QuorumPeerMain
3877 logviewer
3692 Supervisor
[hadoop@SN01 bin]$
```

Storm 실행(4/4)

- http://{MN IP}:8080 접속 후 작동 확인



Apache Storm Example

WordCount Example

- 글자 데이터를 실시간으로 Spout에 전송하여 단어 수를 세는 예제

```
"my dog has fleas",
"i like cold beveages",
"the dog ate my homework",
"don't have a cow man",
"i don't think i like fleas"
```

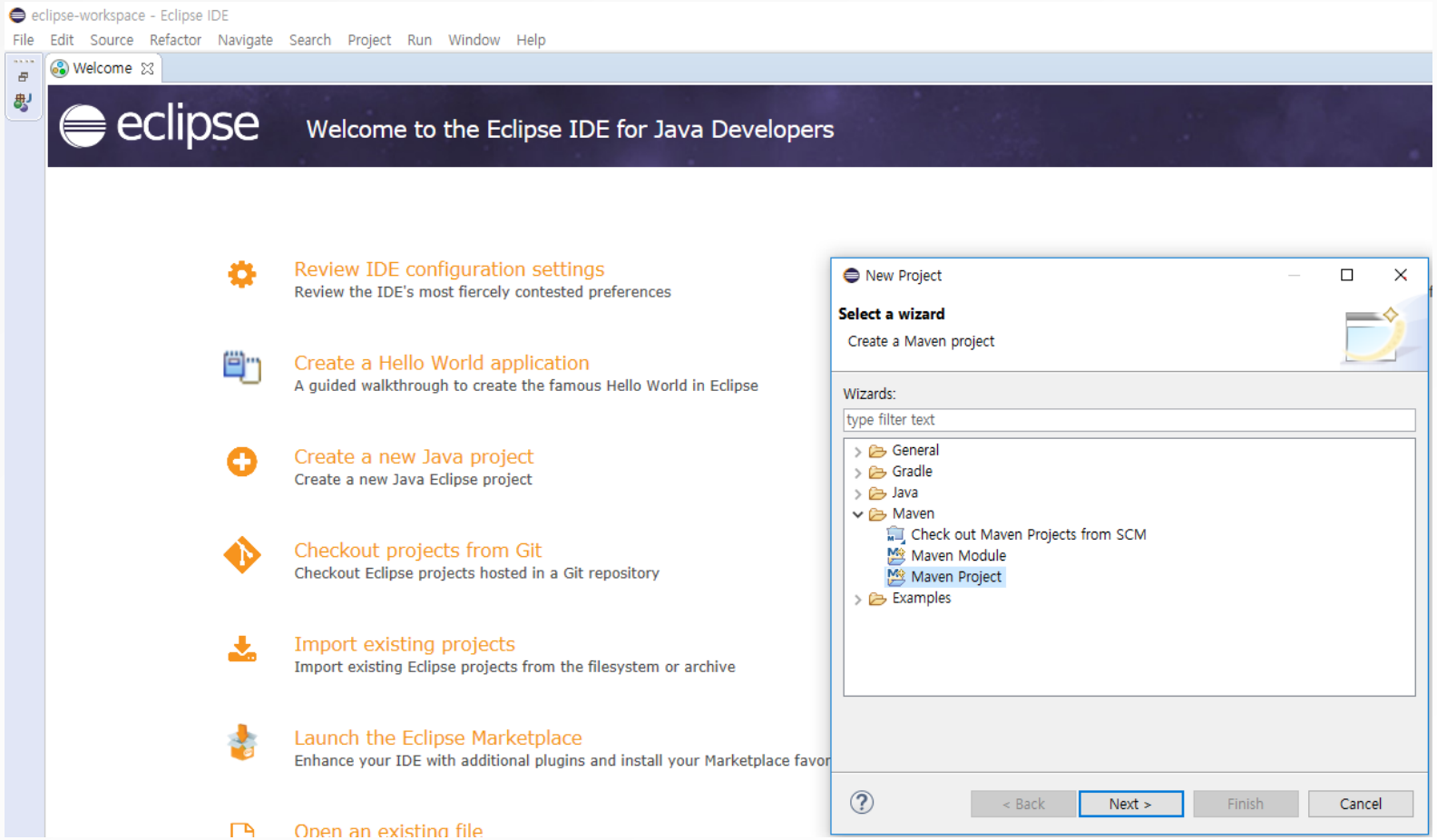
```
41948 [Thread-16-report-bolt-executor[3 3]] INFO  o.a.s.util - Async loop interrupted!
41948 [Thread-15-disruptor-executor[3 3]-send-queue] INFO  o.a.s.util - Async loop interrupted!
------- FINAL COUNT -------
a: 139
think: 139
beverages: 140
like: 280
homework: 139
don't: 279
i: 420
cold: 140
cow: 139
my: 280
the: 140
ate: 139
have: 139
has: 140
man: 139
dog: 280
fleas: 280
--------------------------
41948 [main] INFO  o.a.s.d.executor - Shut down executor report-bolt:[3 3]
41948 [main] INFO  o.a.s.d.executor - Shutting down executor __acker:[1 1]
41948 [Thread-18-__acker-executor[1 1]] INFO  o.a.s.util - Async loop interrupted!
```

Maven

- Maven
  - 자바 프로젝트의 빌드를 자동화 해주는 build tool이다.
    - 자바 소스를 compile하고 package해서 deploy하는 일을 자동화 해주는 것이다.
- Maven이 참조하는 설정 파일
  - Settings.xml
  - Pom.xml

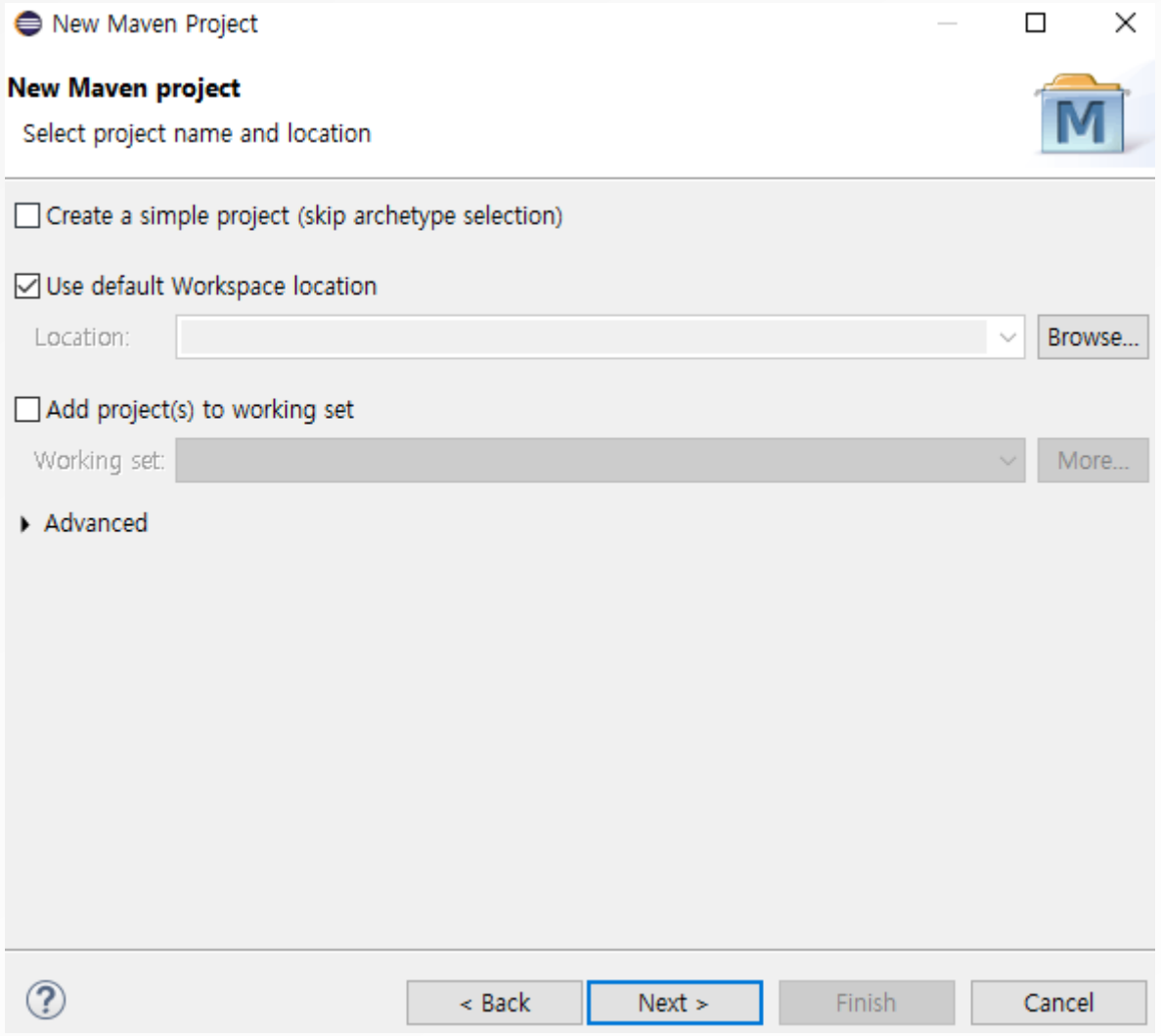
프로젝트 생성(1/6)

- File → New → Project → Maven Project



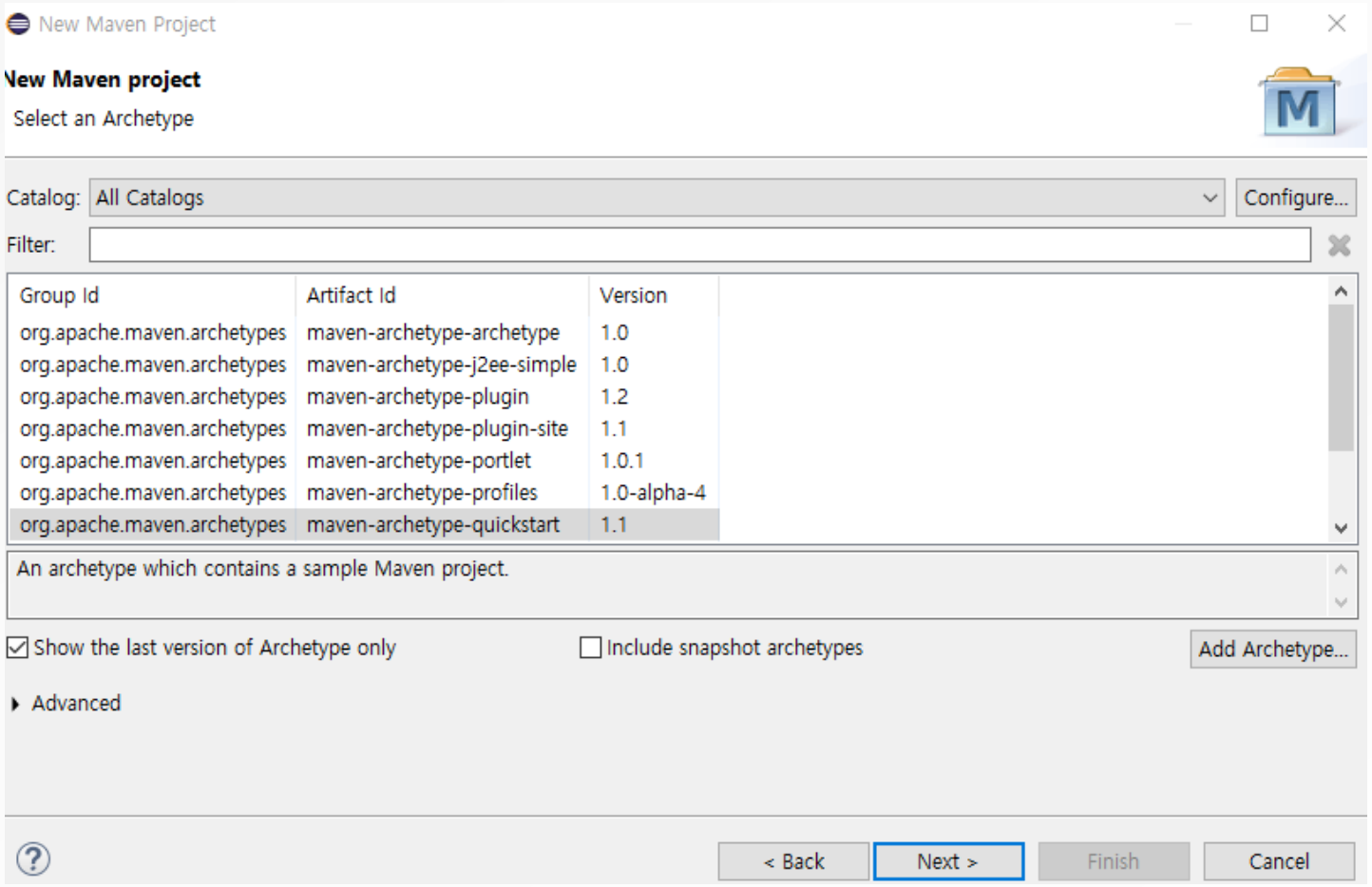
프로젝트 생성(2/6)

- Next…



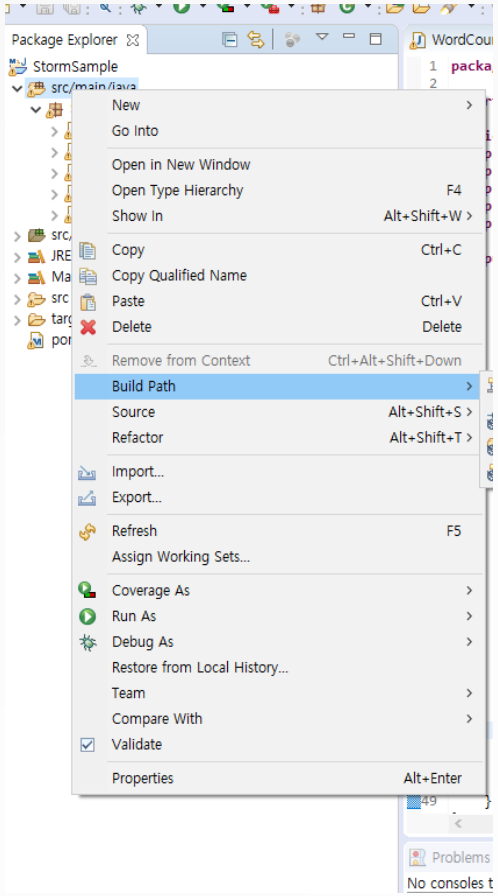
프로젝트 생성(3/6)

- Artifact Id: Maven-archetype-quickstart
- Version: 1.1



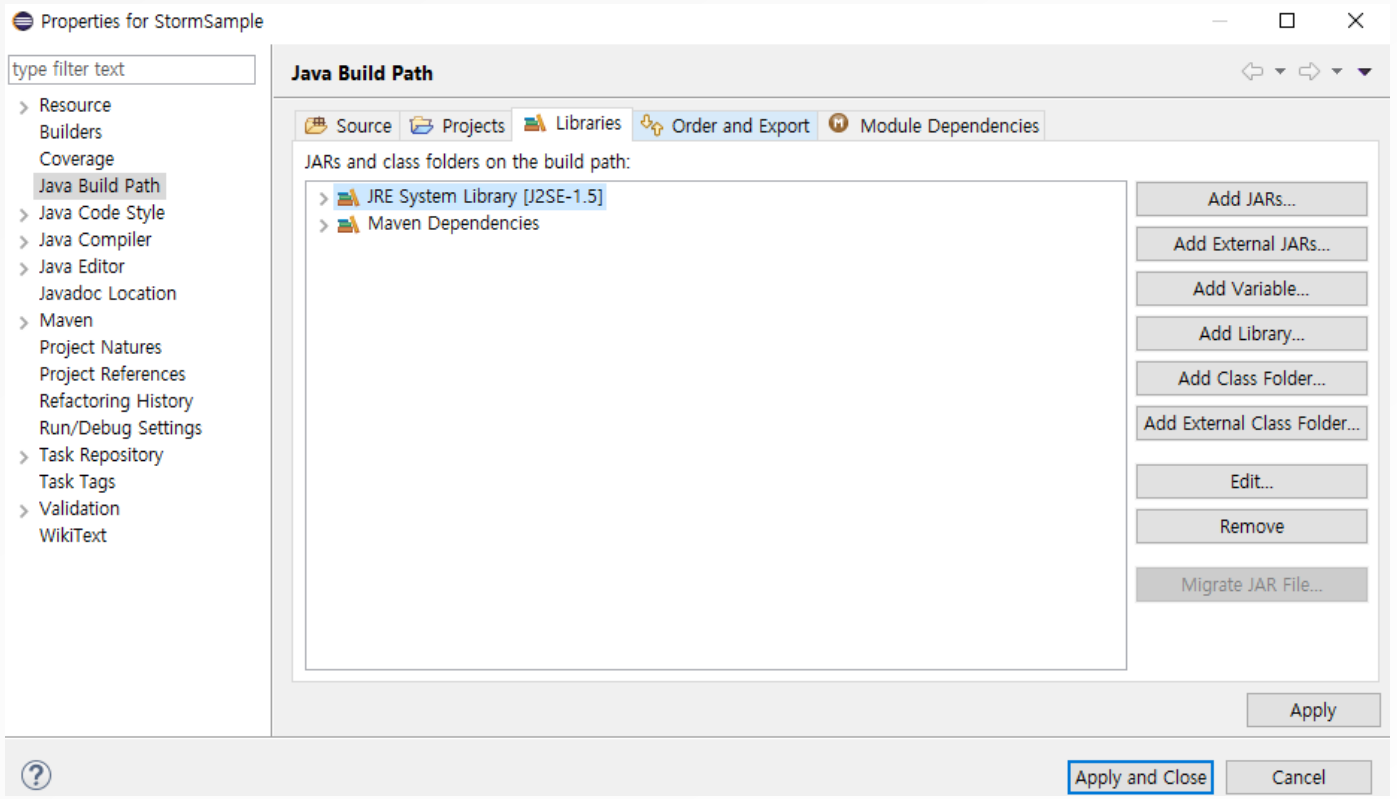
프로젝트 생성(4/6)

- 프로젝트 우클릭 → Build Path → Configure Build Path



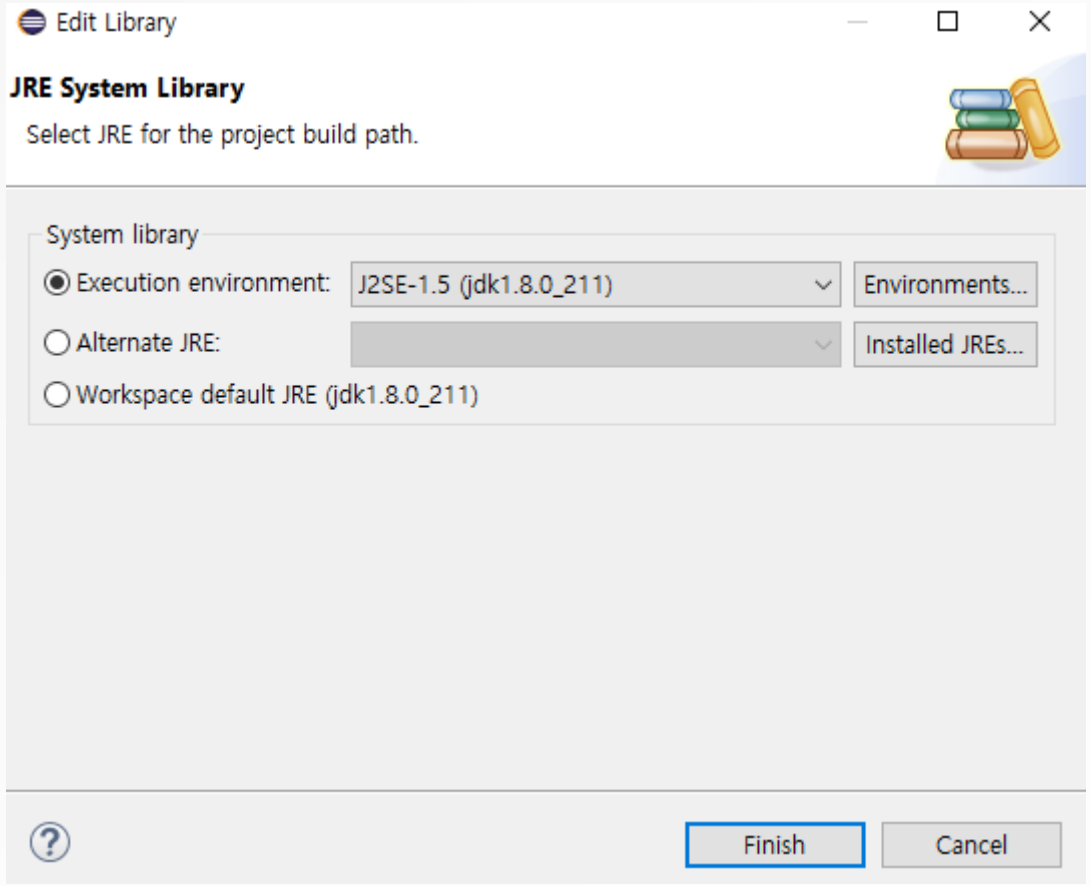
프로젝트 생성(5/6)

- JRE System Library [J2SE-1.5]→Edit



프로젝트 생성(6/6)

- J2SE-1.5(jdk1.8.0_211)→Finish



pom.xml 수정(1/2)

- pom.xml 수정

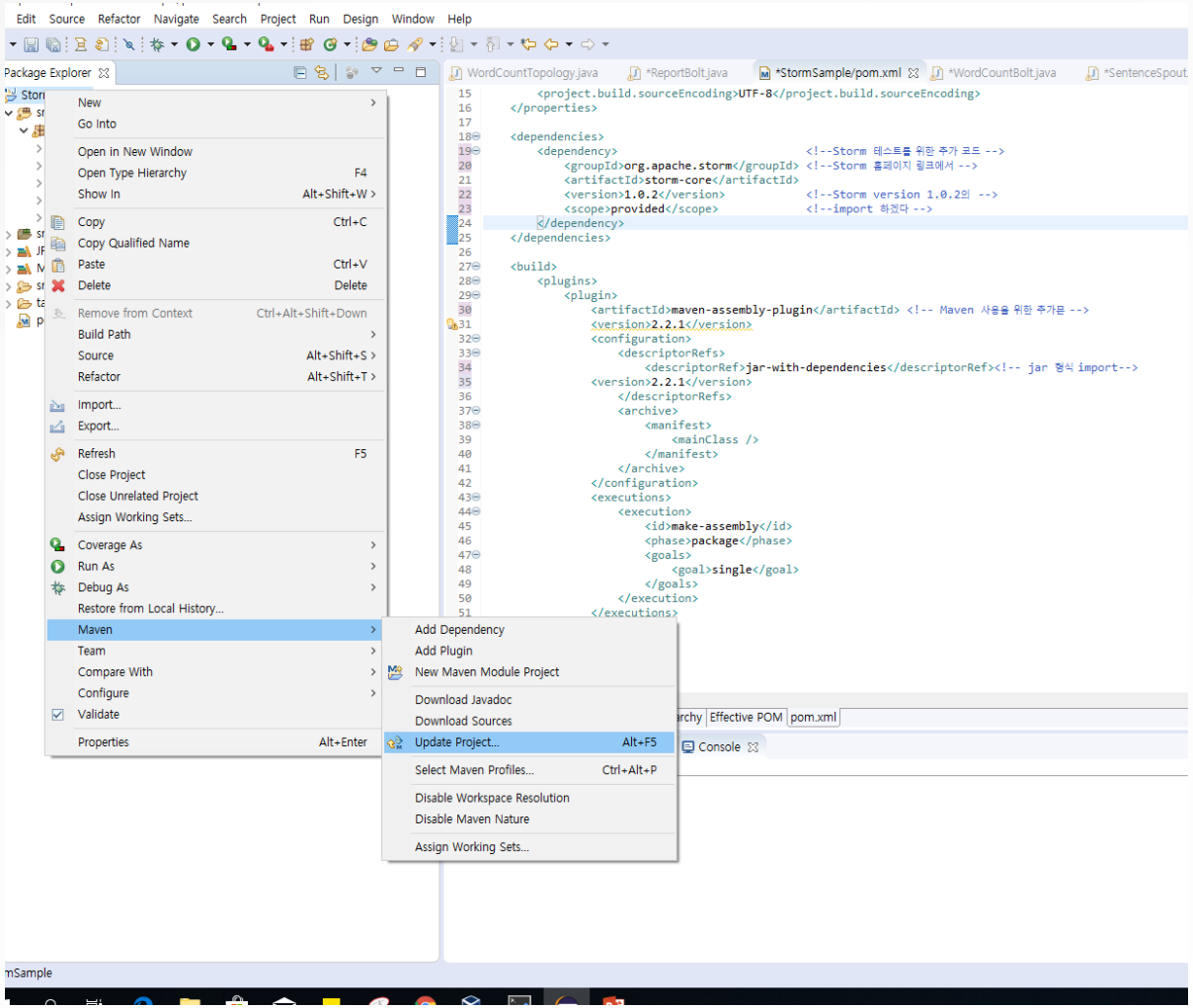
```xml
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>StormSample</groupId>
    <artifactId>StormSample</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>jar</packaging>

    <name>StormSample</name>
    <url>http://maven.apache.org</url>

    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>

    <dependencies>
        <dependency>                            <!--Storm 테스트를 위한 추가 코드 -->
            <groupId>org.apache.storm</groupId> <!--Storm 홈페이지 링크에서 -->
            <artifactId>storm-core</artifactId>
            <version>1.0.2</version>            <!--Storm version 1.0.2의 -->
            <scope>provided</scope>             <!--import 하겠다 -->
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <artifactId>maven-assembly-plugin</artifactId> <!-- Maven 사용을 위한 추가문 -->
                <version>2.2.1</version>
                <configuration>
                    <descriptorRefs>
                        <descriptorRef>jar-with-dependencies</descriptorRef><!-- jar 형식 import-->
                <version>2.2.1</version>
                    </descriptorRefs>
                    <archive>
                        <manifest>
                            <mainClass />
                        </manifest>
                    </archive>
                </configuration>
```

pom.xml 수정(2/2)

```
42                </configuration>
43                <executions>
44                    <execution>
45                        <id>make-assembly</id>
46                        <phase>package</phase>
47                        <goals>
48                            <goal>single</goal>
49                        </goals>
50                    </execution>
51                </executions>
52            </plugin>
53        </plugins>
54    </build>
55</project>
56
```

Maven Update

- 프로젝트 우클릭→Maven→Update Project



Spout 작성(1/2)

- SentenceSpout 작성

```java
package StormSample.StormSample;

import org.apache.storm.spout.SpoutOutputCollector;
import org.apache.storm.task.TopologyContext;
import org.apache.storm.topology.OutputFieldsDeclarer;
import org.apache.storm.topology.base.BaseRichSpout;
import org.apache.storm.tuple.Fields;
import org.apache.storm.tuple.Values;
import java.util.Map;
//SnetenceSpout이 데이터를 계속 들여 오는 Spout임
public class SentenceSpout extends BaseRichSpout{ //여기부터
    private SpoutOutputCollector collector;
    private int index = 0;
    private final String[] sentences = {
            "my dog has fleas",
            "i like cold beveages",
            "the dog ate my bomework",
            "don't have a cow man",
            "i don't think i like fleas"
    };                                          //여기까지 미리 작성한 text를 연속적으로 계속 생산할 것
    public void open(Map map, TopologyContext topologyContext, SpoutOutputCollector spoutOutputCollector) {
        this.collector = spoutOutputCollector;
    } //collector를 세팅

    public void declareOutputFields(OutputFieldsDeclarer outputFieldsDeclarer) { //output 데이터를 설정
        outputFieldsDeclarer.declare(new Fields("sentence"));
    }

```

Spout 작성(2/2)

```
29      public void nextTuple() { //데이터를 계속 들어오게 하는 실질적인 코드, 실제 튜플을 생성하는 메소드
30          this.collector.emit(new Values(sentences[index]));
31          index++;
32          if(index >= sentences.length) {
33              index = 0;
34          }
35          try {
36              Thread.sleep(5);
37              }
38          catch(InterruptedException e)
39          {
40
41          }
42      }
43  }
44
45
```

Bolt 작성(1/3)

- SplitBolt 작성

```
1  package StormSample.StormSample;
2
3  import org.apache.storm.task.OutputCollector;
11  //문장을 받은것을 단어들로 다 짤라주는 볼트
12 public class SplitBolt extends BaseRichBolt { //SentenceSput에 있는 BaseRichBolt를 상속
13     private OutputCollector collector;
14
15     public void prepare(Map map, TopologyContext topologyContext, OutputCollector outputCollector) {
16         this.collector = outputCollector; //출력 데이터 정의
17     }
18
19     public void declareOutputFields(OutputFieldsDeclarer outputFieldsDeclarer) {
20         outputFieldsDeclarer.declare(new Fields("word"));//SentenceSpot에서 받은 Sentence를 Word로 변경
21     }
22
23     public void execute(Tuple tuple) {
24         String sentence = tuple.getStringByField("sentence");
25         String[] words = sentence.split(" ");
26         for (String word: words) {
27             this.collector.emit(new Values(word));
28         }//execute를 통해 입력 받은 tuple을 처리하는 구조  지금 단어 단위로 다 끊어주는 역할을 함
29     }
30 }
```

Bolt 작성(2/3)

- WordCountBolt 작성

```java
package StormSample.StormSample;

import org.apache.storm.task.OutputCollector;
import org.apache.storm.task.TopologyContext;
import org.apache.storm.topology.OutputFieldsDeclarer;
import org.apache.storm.topology.base.BaseRichBolt;
import org.apache.storm.tuple.Fields;
import org.apache.storm.tuple.Tuple;
import org.apache.storm.tuple.Values;
import java.util.HashMap;
import java.util.Map;
 //단어들로 짤라진 데이터를 카운팅하기 위한 Bolt
public class WordCountBolt extends BaseRichBolt {
    private OutputCollector collector;
    private HashMap<String, Long> counter = null; //HashMapdl word를 카운팅한 값을 계속 유지
    public void prepare(Map map, TopologyContext topologyContext, OutputCollector outputCollector) {
        this.collector = outputCollector;
        this.counter = new HashMap<String, Long>();
    }

    public void declareOutputFields(OutputFieldsDeclarer outputFieldsDeclarer) {
        outputFieldsDeclarer.declare(new Fields("word","count")); //count한 값을 실시간으로 계속 출력
    }

    public void execute(Tuple tuple) {
        String word = tuple.getStringByField("word");
        Long count = this.counter.get(word); //word가 들어오면 그 단어를 가져와서
        count = count == null ? 1L : count + 1; //count해서 밑으로 넘겨주고
        this.counter.put(word, count);  //단어에 맞는 숫자 카운터 1증가
        this.collector.emit(new Values(word, count));
    }
}
```

Bolt 작성(3/3)

- ReportBolt 작성

```
1  package StormSample.StormSample;
2
3  import org.apache.storm.task.OutputCollector;
10  //카운팅된 입력 tuple을 받아서 화면에 출력하는 Bolt
11 public class ReportBolt extends BaseRichBolt {
12     private HashMap<String, Long> counter = null;
13  //HashMap에 저장되 있던 값을
14     public void prepare(Map map, TopologyContext topologyContext, OutputCollector outputCollector) {
15         this.counter = new HashMap<String, Long>();
16     }
17
18     public void declareOutputFields(OutputFieldsDeclarer outputFieldsDeclarer) {}
19
20     public void execute(Tuple tuple) {
21         String word = tuple.getStringByField("word");
22         Long count = tuple.getLongByField("count");
23         this.counter.put(word, count);
24     }
25
26     @Override
27     public void cleanup() {                              //Bolt가 정지 시켰을때 실행되는 메소드. 토폴로지의 흐름이 계속 데이터를 받으면서 word를 count하다가
28         System.out.println("------- FINAL COUNT -------");//정지시키면 각 단어들에 count된 값을 가져오는 메소드
29         for (String key: this.counter.keySet()) {
30             System.out.println(key + ": " + this.counter.get(key));
31         }
32         System.out.println("----------------------------");
33     }
34 }
```

Topology 작성(1/2)

- WordCountTopology작성

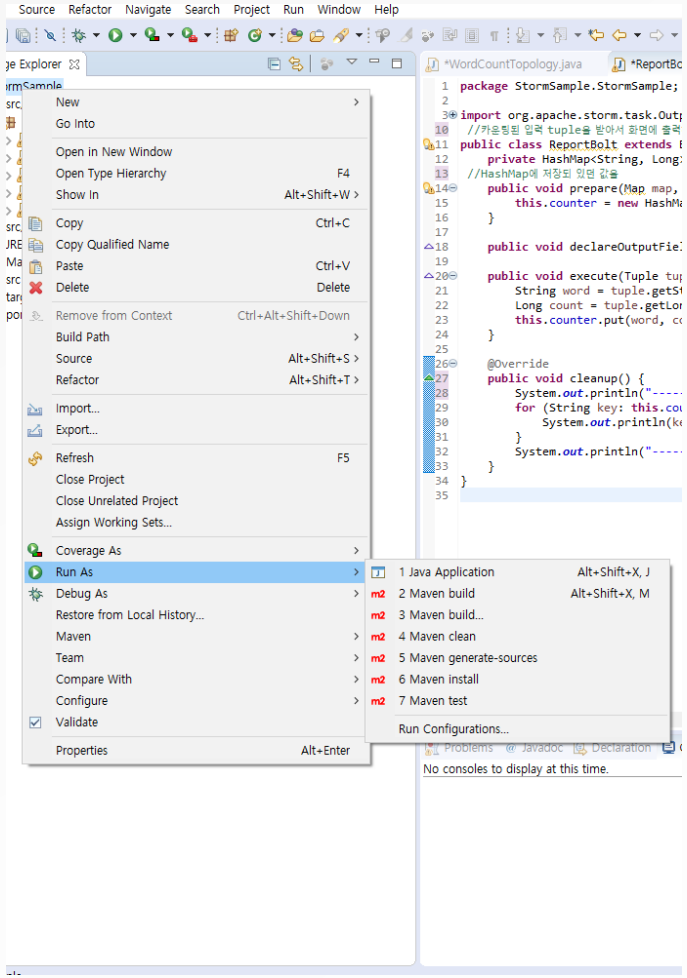
```
1  package StormSample.StormSample;
2
3  import org.apache.storm.Config;
4  import org.apache.storm.StormSubmitter;
5  import org.apache.storm.generated.AlreadyAliveException;
6  import org.apache.storm.generated.AuthorizationException;
7  import org.apache.storm.generated.InvalidTopologyException;
8  import org.apache.storm.topology.TopologyBuilder;
9  import org.apache.storm.tuple.Fields;
10 //만들었던 Bolt와 Spout을 활용하는 Topology(main문)
11 public class WordCountTopology {
12     private static final String SENTENCE_SPOUT_ID = "sentence-spout";
13     private static final String SPLIT_BOLT_ID = "split-bolt";
14     private static final String COUNT_BOLT_ID = "count-bolt";
15     private static final String REPORT_BOLT_ID = "report-bolt";
16     private static final String TOPOLOGY_NAME = "word-count-topology";
17
18     public static void main(String[] args) {
19         SentenceSpout spout = new SentenceSpout(); //SentenceSpout을 가져와서 사용하는것을 선언
20         SplitBolt splitBolt = new SplitBolt();     //SplitBolt를 가져와서 사용하는것을 선언
21         WordCountBolt countBolt = new WordCountBolt();//WordCountBolt를 가져와서 사용하는것을 선언
22         ReportBolt reportBolt = new ReportBolt();//ReportBolt를 가져와서사용하는 것을 선언
23
24         TopologyBuilder builder = new TopologyBuilder();//builder 객체 선언
25
```

Topology 작성(2/2)

```
25
26          builder.setSpout(SENTENCE_SPOUT_ID, spout);//SentenceSpoutdp 있는 데이터를 흐름
27          builder.setBolt(SPLIT_BOLT_ID, splitBolt).shuffleGrouping(SENTENCE_SPOUT_ID);//task에서 task로 정보를 전달하는 방식을 shuffleGrouping을 사용
28          builder.setBolt(COUNT_BOLT_ID, countBolt).fieldsGrouping(SPLIT_BOLT_ID, new Fields("word"));//task에서 task로 정보를 전달하는 방식을 FiledGrouping을 사용
29          builder.setBolt(REPORT_BOLT_ID, reportBolt).globalGrouping(COUNT_BOLT_ID); ////task에서 task로 정보를 전달하는 방식을 GlobalGrouping을 사용
30
31          //local로 사용할 때 사용
32          //        LocalCluster cluster = new LocalCluster();
33          //        cluster.submitTopology(TOPOLOGY_NAME, new Config(), builder.createTopology());
34          //
35          //        try { Thread.sleep(1000 * 30); } catch (InterruptedException e) { }
36          //        cluster.killTopology(TOPOLOGY_NAME);
37          //        cluster.shutdown();
38          Config conf = new Config();
39          conf.setNumWorkers(4); //Works 추가문
40          // Submit topology to cluster
41          try{
42              StormSubmitter.submitTopology(args[0], conf, builder.createTopology());
43          }catch(AlreadyAliveException ae){
44              System.out.println(ae);
45          }catch(InvalidTopologyException ie){
46              System.out.println(ie);
47          } catch (AuthorizationException e) {
48              System.out.println(e);
49          }
50      }
51  }
```

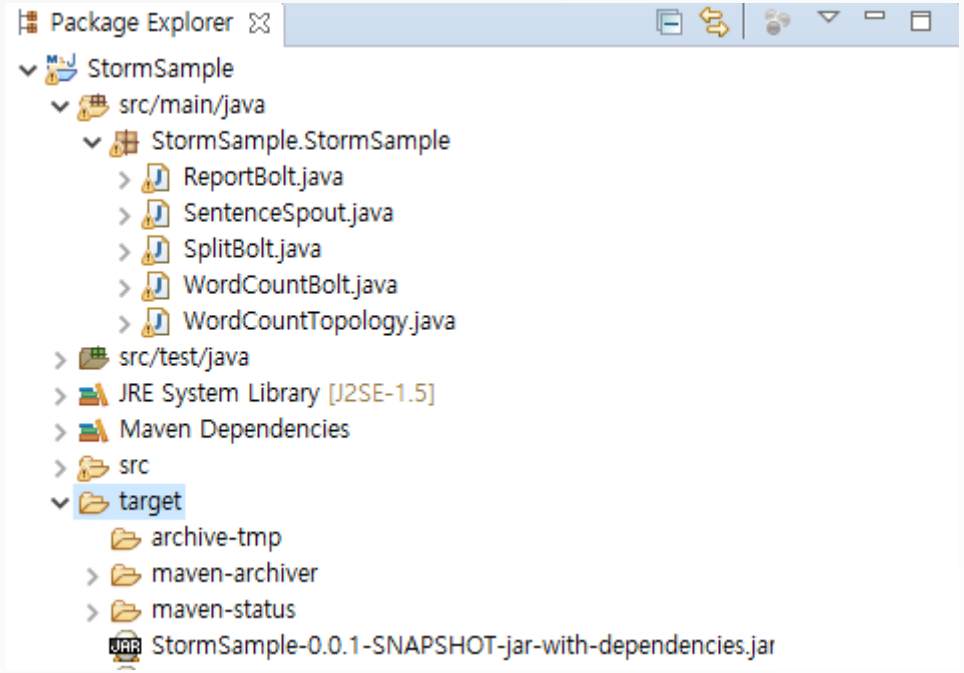
실행(1/4)

- File→Run as→Maven Clean→Maven install



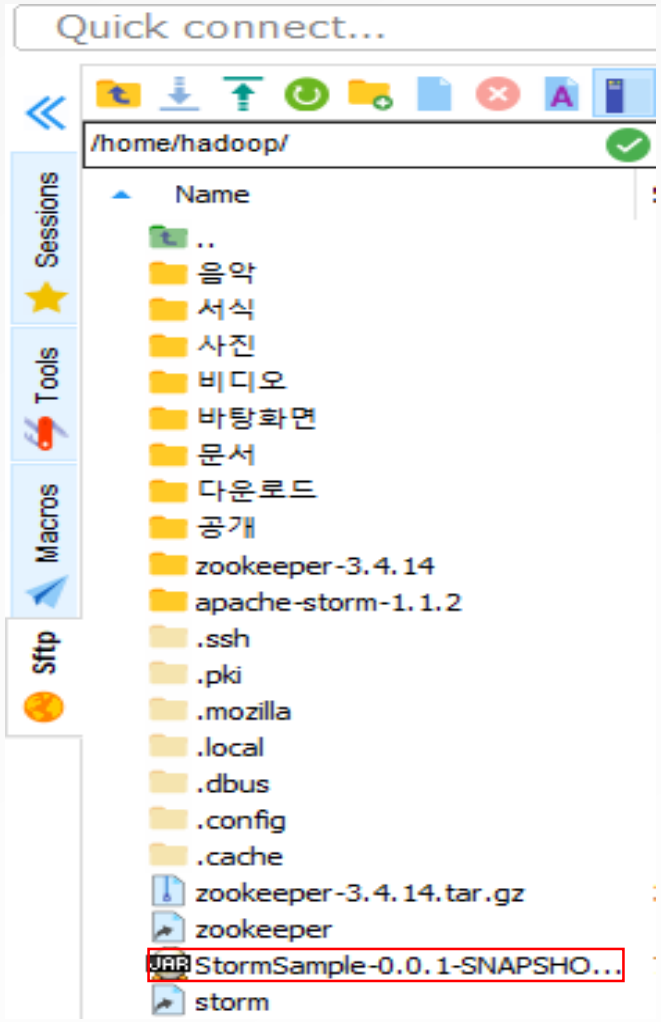
실행(2/4)

- Build 성공 시 패키지기본경로-0.0.1-SNAPSHOT-jar-with-dependencies.jar 생성



실행(3/4)

◈ 패키지기본경로-0.0.1-SNAPSHOT-jar-with-dependencies.jar 가상머신으로 이동



실행(4/4)

- ~/storm/bin/storm jar {생성파일} {패키지기본경로}.{프로젝트명칭}.{Topologyclass명칭} 생성이름
  - Ex: ~/storm/bin/storm jar Storm-jar withdependencies.jar Sample.Sample.Topology WordCount

```
[hadoop@MN storm]$ mv ~/StormSample-0.0.1-SNAPSHOT-jar-with-dependencies.jar .
[hadoop@MN storm]$ ~/storm/bin/storm jar StormSample-0.0.1-SNAPSHOT-jar-with-dependencies.jar StormSample.StormSample.WordCountTopology hello
Running: /usr/lib/jvm/java-1.8.0-openjdk-1.8.0.212.b04-0.el7_6.x86_64/bin/java -client -Ddaemon.name= -Dstorm.options= -Dstorm.home=/home/hadoop/apache-storm-1.1.2 -Dstorm.log.dir=/home/hadoop/apache
-storm-1.1.2/logs -Djava.library.path=/usr/local/lib:/opt/local/lib:/usr/lib -Dstorm.conf.file= -cp /home/hadoop/apache-storm-1.1.2/StormSample-0.0.1-SNAPSHOT-jar-with-dependencies.jar:/home/hadoop/a
pache-storm-1.1.2/lib/storm-core-1.1.2.jar:/home/hadoop/apache-storm-1.1.2/lib/kryo-3.0.3.jar:/home/hadoop/apache-storm-1.1.2/lib/reflectasm-1.10.1.jar:/home/hadoop/apache-storm-1.1.2/lib/asm-5.0.3.j
ar:/home/hadoop/apache-storm-1.1.2/lib/minlog-1.3.0.jar:/home/hadoop/apache-storm-1.1.2/lib/objenesis-2.1.jar:/home/hadoop/apache-storm-1.1.2/lib/clojure-1.7.0.jar:/home/hadoop/apache-storm-1.1.2/lib
/ring-cors-0.1.5.jar:/home/hadoop/apache-storm-1.1.2/lib/disruptor-3.3.2.jar:/home/hadoop/apache-storm-1.1.2/lib/log4j-api-2.8.2.jar:/home/hadoop/apache-storm-1.1.2/lib/log4j-core-2.8.2.jar:/home/had
oop/apache-storm-1.1.2/lib/log4j-slf4j-impl-2.8.2.jar:/home/hadoop/apache-storm-1.1.2/lib/slf4j-api-1.7.21.jar:/home/hadoop/apache-storm-1.1.2/lib/log4j-over-slf4j-1.6.6.jar:/home/hadoop/apache-storm
-1.1.2/lib/servlet-api-2.5.jar:/home/hadoop/apache-storm-1.1.2/lib/storm-rename-hack-1.1.2.jar:StormSample-0.0.1-SNAPSHOT-jar-with-dependencies.jar:/home/hadoop/storm/conf:/home/hadoop/apache-storm-1
.1.2/bin -Dstorm.jar=StormSample-0.0.1-SNAPSHOT-jar-with-dependencies.jar -Dstorm.dependency.jars= -Dstorm.dependency.artifacts={} StormSample.StormSample.WordCountTopology hello
970  [main] WARN  o.a.s.u.Utils - STORM-VERSION new 1.1.2 old null
1025 [main] INFO  o.a.s.StormSubmitter - Generated ZooKeeper secret payload for MD5-digest: -5235194354648995360:-5229729378994607325
1230 [main] INFO  o.a.s.u.NimbusClient - Found leader nimbus : MN:6627
1262 [main] INFO  o.a.s.s.a.AuthUtils - Got AutoCreds []
1266 [main] INFO  o.a.s.u.NimbusClient - Found leader nimbus : MN:6627
1293 [main] INFO  o.a.s.StormSubmitter - Uploading dependencies - jars...
1329 [main] INFO  o.a.s.StormSubmitter - Uploading dependencies - artifacts...
1330 [main] INFO  o.a.s.StormSubmitter - Dependency Blob keys - jars : [] / artifacts : []
1364 [main] INFO  o.a.s.StormSubmitter - Uploading topology jar StormSample-0.0.1-SNAPSHOT-jar-with-dependencies.jar to assigned location: /home/hadoop/storm/data/nimbus/inbox/stormjar-4f0a977c-b307-
4c6f-ba8f-ec95145c60a4.jar
1392 [main] INFO  o.a.s.StormSubmitter - Successfully uploaded topology jar to assigned location: /home/hadoop/storm/data/nimbus/inbox/stormjar-4f0a977c-b307-4c6f-ba8f-ec95145c60a4.jar
1392 [main] INFO  o.a.s.StormSubmitter - Submitting topology hello in distributed mode with conf {"topology.workers":4,"storm.zookeeper.topology.auth.scheme":"digest","storm.zookeeper.topology.auth.p
ayload":"-5235194354648995360:-5229729378994607325"}
1392 [main] WARN  o.a.s.u.Utils - STORM-VERSION new 1.1.2 old 1.1.2
1598 [main] INFO  o.a.s.StormSubmitter - Finished submitting topology: hello
[hadoop@MN storm]$
```

결과(1/3)

Storm UI

Cluster Summary

Version	Supervisors	Used slots	Free slots	Total slots	Executors	Tasks
1.1.2	2	4	4	8	8	8

Nimbus Summary

Search:

Host	Port	Status	Version	UpTime
MN	6627	Leader	1.1.2	1d 0h 35m 45s

Showing 1 to 1 of 1 entries

Topology Summary

Search:

Name	Owner	Status	Uptime	Num workers	Num executors	Num tasks	Replication count	Assigned Mem (MB)	Scheduler Info
hello	hadoop	ACTIVE	6m 25s	4	8	8	1	3328	

Showing 1 to 1 of 1 entries

Supervisor Summary

Search:

Host	Id	Uptime	Slots	Used slots	Avail slots	Used Mem (MB)	Version
SN01 (log)	2db99b0c-7458-4d4c-9a9c-c7b2336b9d55	1d 0h 33m 36s	4	2	2	1664	1.1.2
SN02 (log)	90406064-1fc6-40cf-9473-05bb64db53db	1d 0h 33m 36s	4	2	2	1664	1.1.2

Showing 1 to 2 of 2 entries

결과(2/3)

Topology summary

Name	Id	Owner	Status	Uptime	Num workers	Num executors	Num tasks	Replication count	Assigned Mem (MB)	Scheduler Info
hello	hello-2-1565072872	hadoop	INACTIVE	30s	4	8	8	1	3328	

Topology actions

Activate | Deactivate | Rebalance | Kill | Debug | Stop Debug | Change Log Level

Topology stats

Window	Emitted	Transferred	Complete latency (ms)	Acked	Failed
10m 0s	3400	3400	0		
3h 0m 0s	3400	3400	0		
1d 0h 0m 0s	3400	3400	0		
All time	3400	3400	0		

Spouts (All time)

Search:

Id	Executors	Tasks	Emitted	Transferred	Complete latency (ms)	Acked	Failed	Error Host	Error Port	Last error	Error Time
sentence-spout	1	1	300	300	0.000	0	0				

Showing 1 to 1 of 1 entries

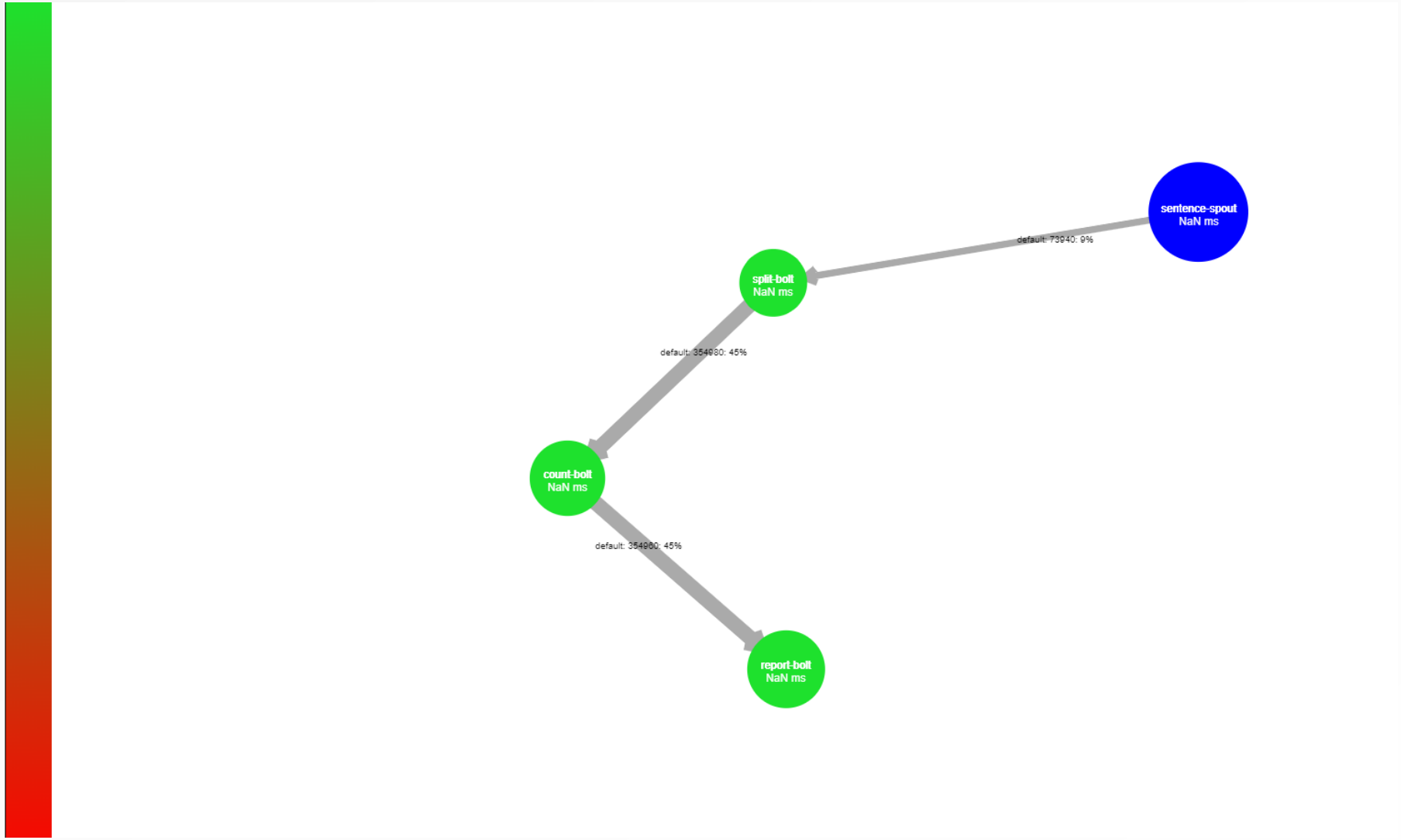
Bolts (All time)

Search:

Id	Executors	Tasks	Emitted	Transferred	Capacity (last 10m)	Execute latency (ms)	Executed	Process latency (ms)	Acked	Failed	Error Host	Error Port	Last error	Error Time
count-bolt	1	1	1520	1520	0.000	0.000	1520	0.000	0	0				
report-bolt	1	1	0	0	0.000	0.000	1540	0.000	0	0				
split-bolt	1	1	1580	1580	0.005	0.063	320	0.000	0	0				

Showing 1 to 3 of 3 entries

결과(3/3)


sentence-spout
NaN ms
default: 73940: 9%
split-bolt
NaN ms
default: 354980: 45%
count-bolt
NaN ms
default: 354960: 45%
report-bolt
NaN ms