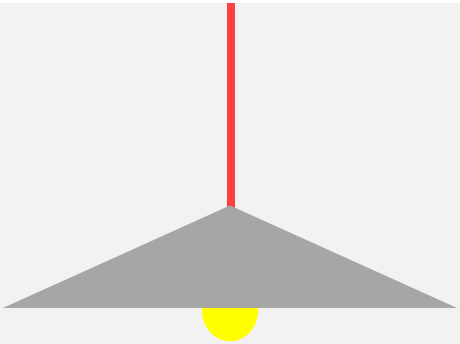


소프트웨어 검증

Junit/Eclipse/빌드환경

T3

박준모 200911391

한종철 200911429

신민용 201111364 *

목차

1 Eclipse

2 JUnit

3 빌드 환경

\+ Q&A

1

Eclipse

1-1 JDK

- JVM ⊂ JRE ⊂ JDK
- JDK 설치
- path 설정
- 설치 확인
- JDK 설치 오류 및 해결방법

1-2 Eclipse

- Eclipse
- Eclipse 설치
- Eclipse 설치 오류 및 해결방법

Eclipse

JDK

$$JVM \subset JRE \subset JDK$$

JVM (Java Virtual Machine, 자바 가상 머신)

- 자바 바이트코드를 읽고, 검증하고, 실행할 수 있는 주체
- 자바 가상 머신 규격에 정의된 대로 자바 바이트코드를 실행
- 실행환경의 규격을 제공 (필요한 라이브러리 및 기타 파일)
- 플랫폼에 의존적

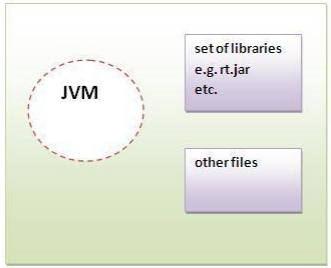
Eclipse

JDK

$$JVM \subset JRE \subset JDK$$

JRE (Java Runtime Environment, 자바 실행 환경)

- JRE는 JVM이 자바 프로그램을 동작시킬 때 필요한 라이브러리 파일들과 기타 파일들을 가지고 있음
- JRE는 JVM의 실행환경을 구현했다고 할 수 있음



JRE

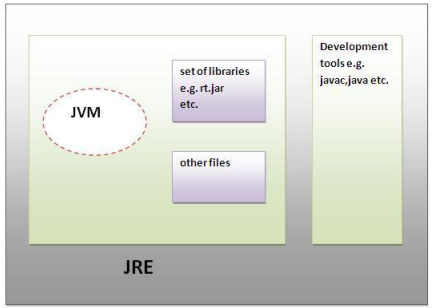
Eclipse

JDK

$JVM \subset JRE \subset JDK$

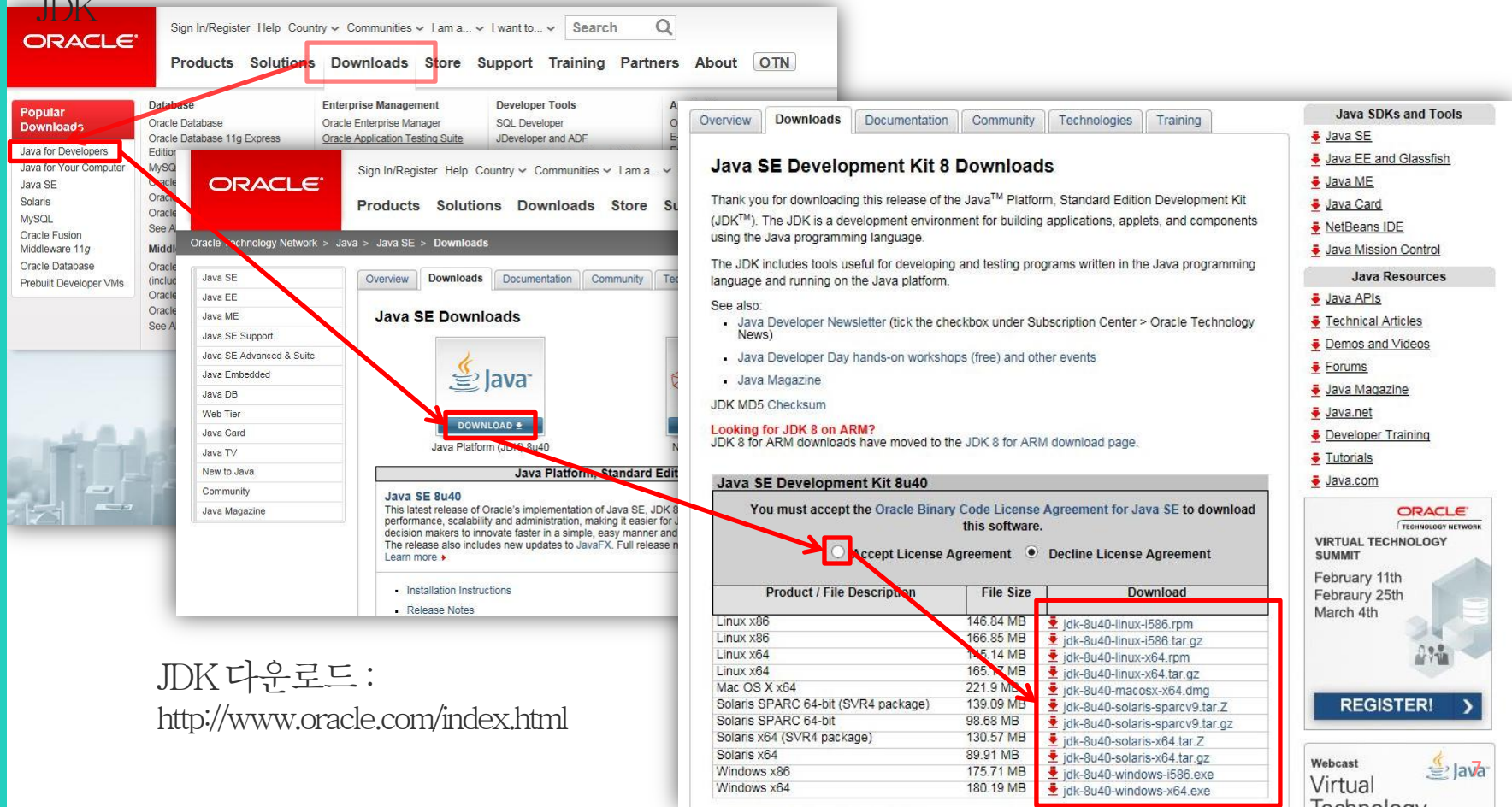
JDK (Java Development Kit, 자바 개발 도구)

- 작성한 소스코드를 컴파일하여 바이트코드를 생성하는 단계까지 제공
- JDK는 개발을 위해 필요한 온갖 API가 정의되어 있어 개발자들이 이용할 수 있음
- 자바 바이트코드는 플랫폼에 독립적



1 Eclipse

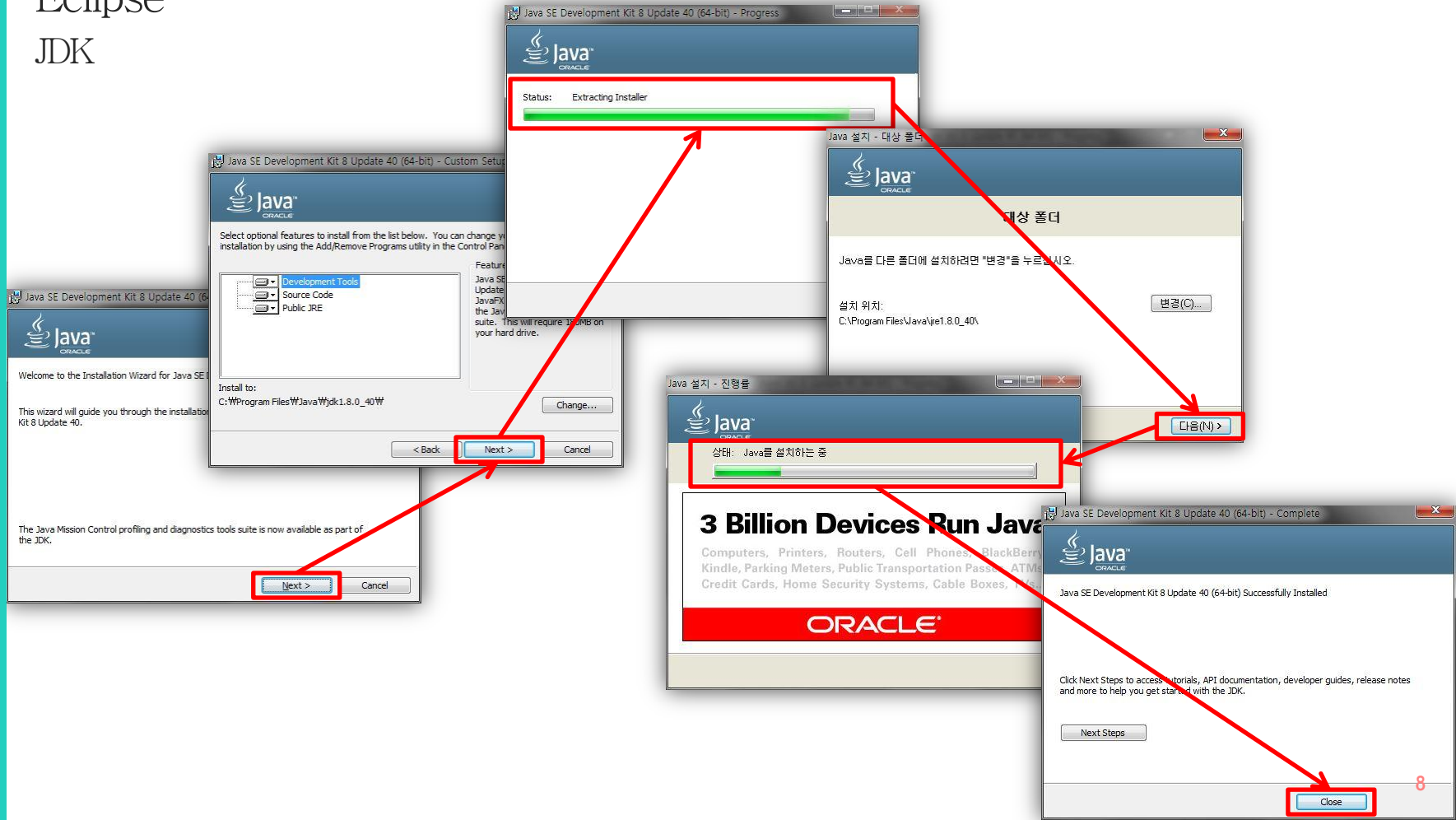
1-1 JDK



JDK 다운로드 :
http://www.oracle.com/index.html

1 Eclipse

1-1 JDK



path 설정

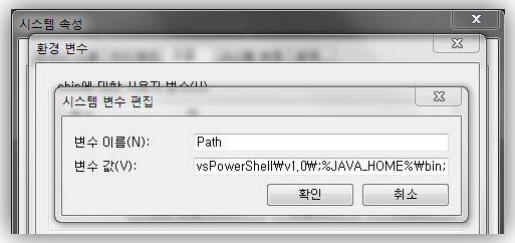
제어판 > 시스템 및 보안 > 시스템 > 고급 시스템 설정> 환경 변수 > 시스템 변수

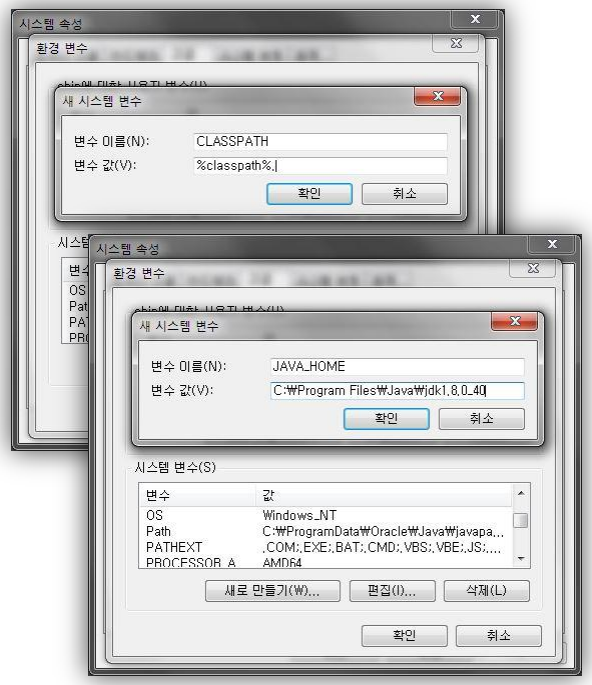
CLASSPATH & JAVA_HOME

- 새로 만들기
- CLASSPATH: %classpath%.
- JAVA_HOME: C:₩Program Files₩Java₩jdk1.8.0_40(JDK가 설치된 위치)

Path

- 편집
- %JAVA_HOME%₩bin; 추가



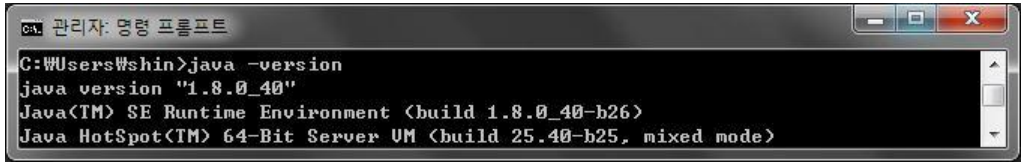


Eclipse

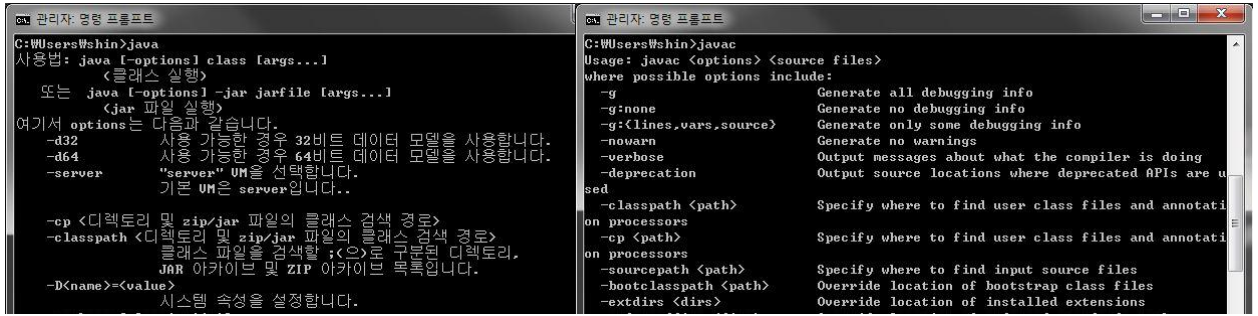
JDK

설치 확인

java -version



java & javac



JDK 설치 오류 및 해결방법

- Windows XP 유저의 경우, JDK 8을 받으면 이클립스가 실행 안 됨(오라클이 Windows XP에 대한 지원을 중단하면서 해당 문제가 처리되지 않음) ➜ **JDK 7 설치**
- JDK 설치 경로에 특수문자나 공백이 있을 경우, 문제 발생할 가능성 있음 ➜ **환경 변수 path 설정 시에 " "(큰따옴표)로 경로를 감싸주거나 JDK 설치 시에 경로에 특수문자나 공백이 없도록 바꿈**

Eclipse

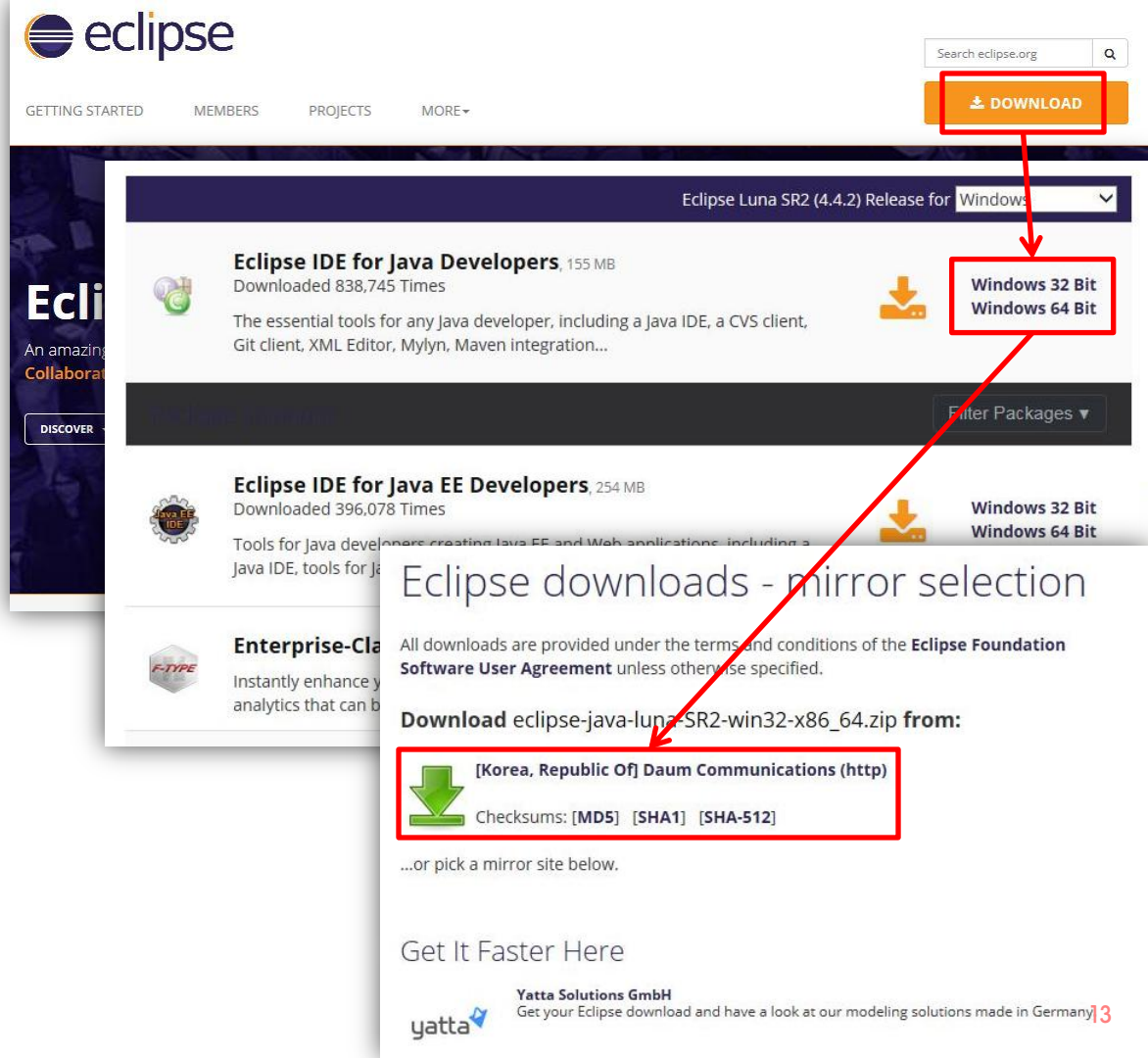
Eclipse

- 비영리 공개 소프트웨어 커뮤니티에 의해서 제공되는 개방형 개발 플랫폼
- 자바를 비롯한 다양한 언어를 지원하는 프로그래밍 통합 개발 환경을 목적으로 시작하였으나, 현재는 OSGi를 도입하여 범용 응용 소프트웨어 플랫폼으로 진화
- 자바로 작성되어 있으며, 자유 소프트웨어이지만 막강한 기능을 자랑

Eclipse

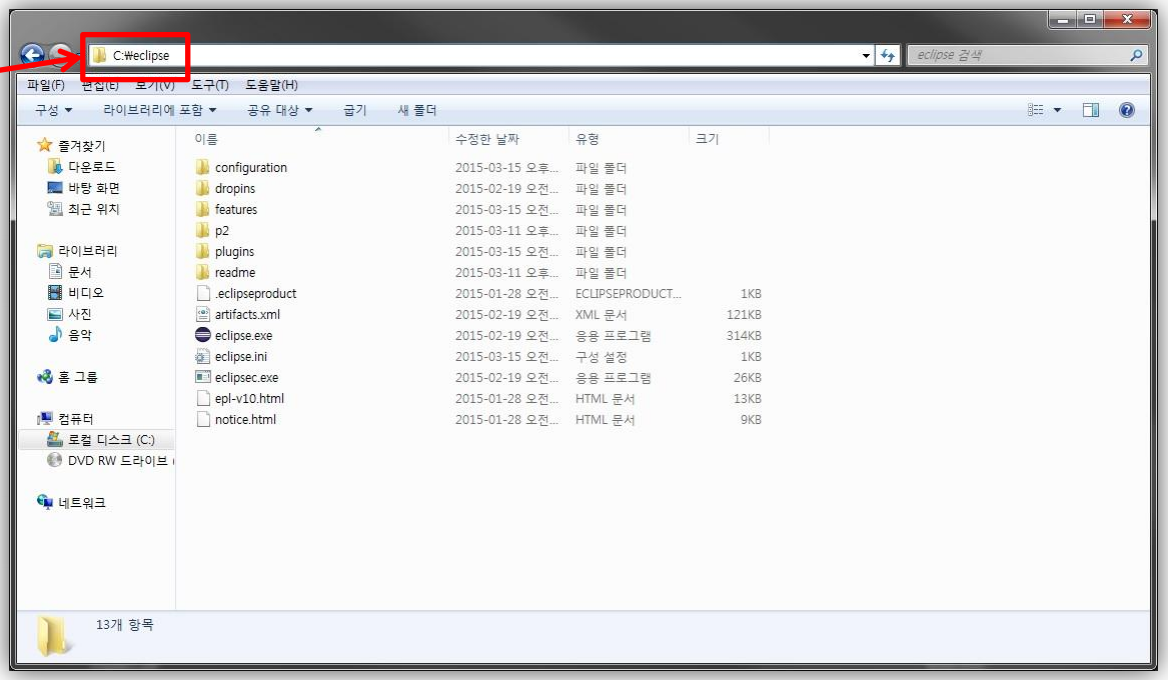
Eclipse 설치

- http://www.eclipse.org/



Eclipse 설치 (계속)

- 압축을 풀기만 하면 됨
- 공백이 없는 위치에...



Eclipse 설치 오류 및 해결방법

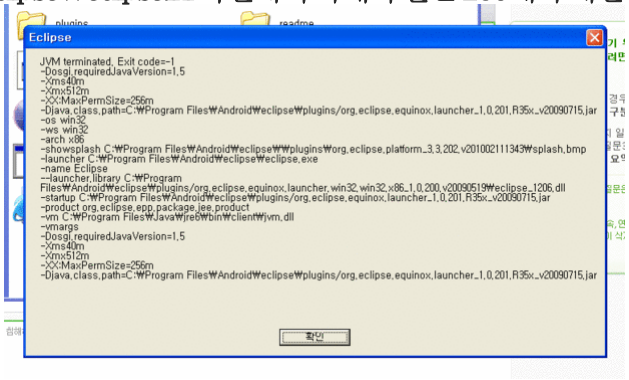
- Failed to create the Java Virtual Machine(Eclipse가 JVM을 인식 못함) ➜ **eclipse₩eclipse.ini 파일 수정**

```
-vm
C:₩Program Files₩Java₩jdk1.8.0_40₩bin₩javaw.exe
```

JDK가 설치된 경로를 확인하여 수정

- JDK와 Eclipse의 버전(32bit/64bit)이 다를 경우 오류 발생 ➜ **같은 버전으로 설치**
- 아래 스크린샷과 같은 경우 ➜ **eclipse₩eclipse.ini 파일에서 아래와 같은 256메가 제한옵션들을 지워줌**

```
--launcher.XXMaxPermSize
256M
```



2

JUnit

2-1 JUnit?

- JUnit이란?
- JUnit 특징
- Unit Test?

2-2 JUnit 사용방법

2-3 JUnit Test Method

2-4 Annotation JUnit

+ 사용 예제

JUnit

JUnit?

JUnit이란?

- Java 에서 사용하는 단위 테스트(Unit Test)를 위한 FrameWork
- 단위 모듈이 정확히 구현되었는지 확인 가능
- 2.1 버전 이후부터는 Eclpise 안에 내장

JUnit

JUnit 특징

주요기능	지원내용
테스트 범위	단위테스트
TDD 환경 지원	지원
코드지원	지원
Test Suite	지원
UI Report	지원
Code Trace	지원, 실시간 추적 가능

JUnit

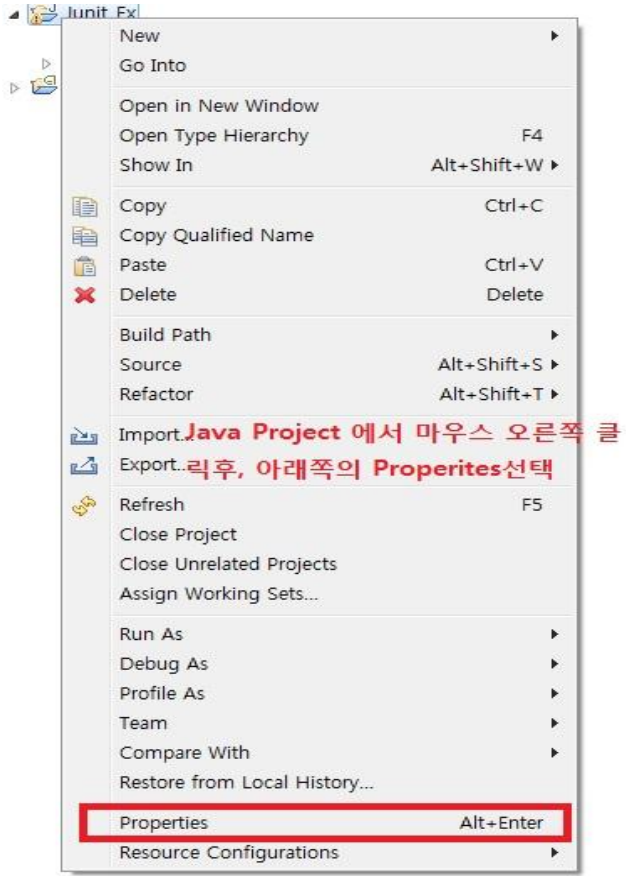
JUnit?

Unit Test?

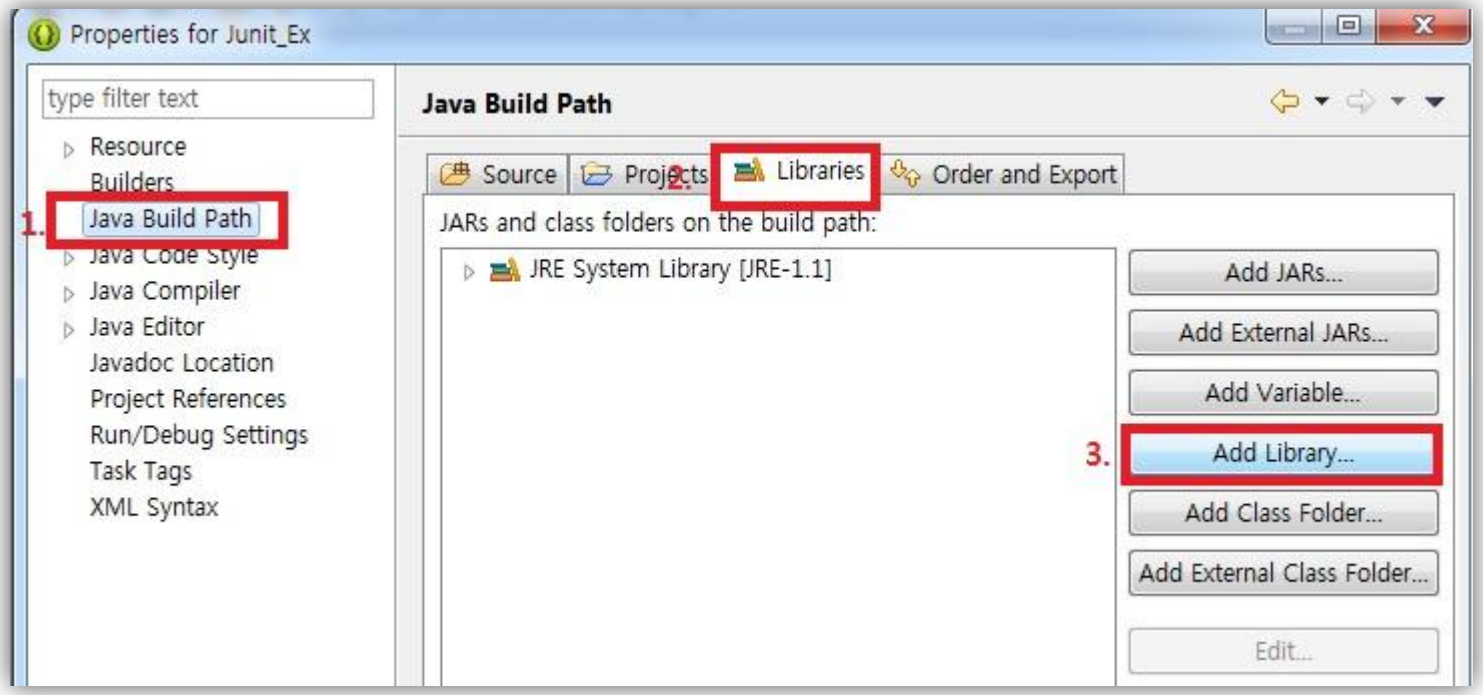
- 프로그램 코드를 개개의 Unit 단위로 쪼개어, 각 Unit 단위 코드가 제대로 작동하는지 확인하는 테스트 방법 중 하나
- 테스트 시 전체 코드를 돌리지 않고 Unit 단위로 돌리기 때문에 실행시간 및 비용 측면에서 이득
- 코드 에러를 빠르게 찾을 수 있음

JUnit 사용방법

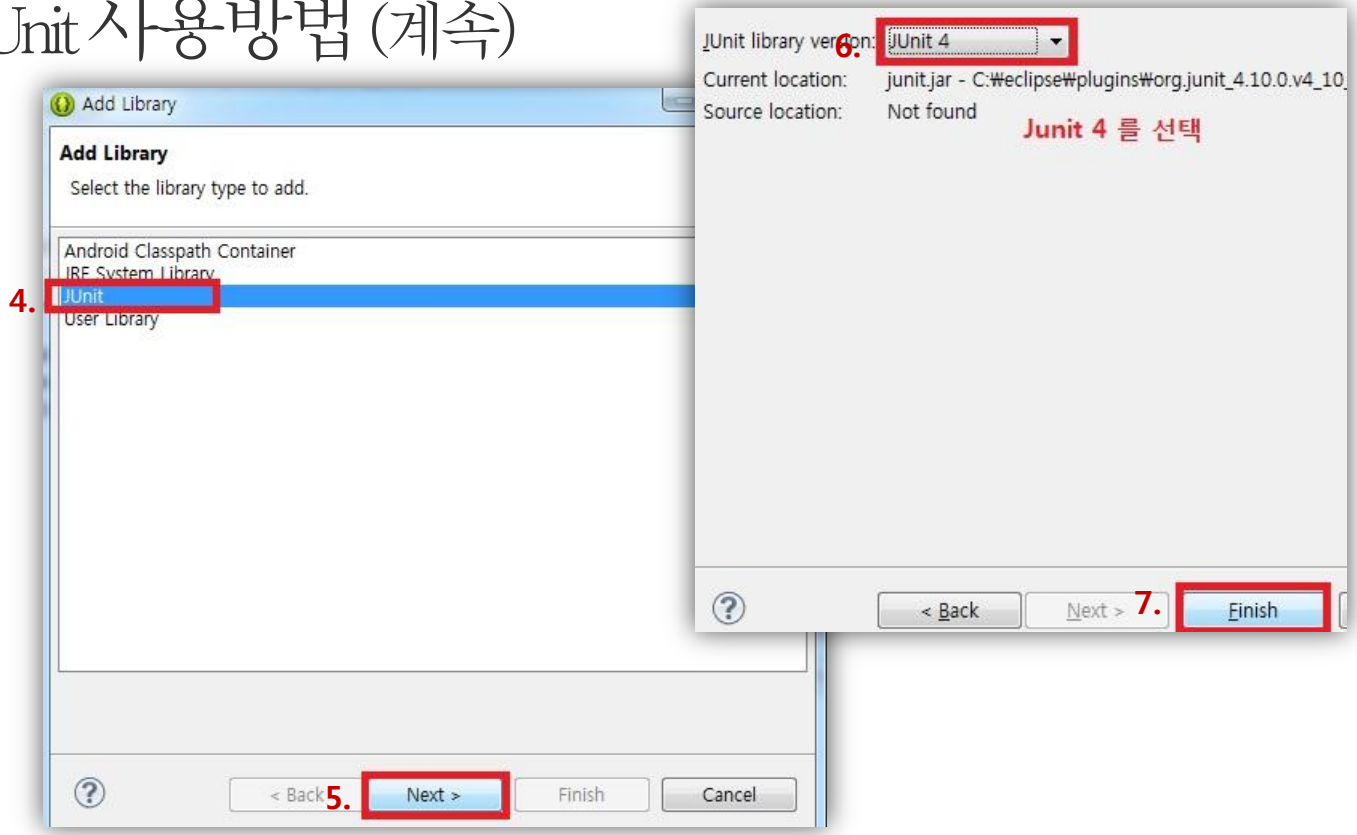
- JUnit은 Eclipse 설치 시 내장되어 있음
- JUnit을 사용할 Java 프로젝트에서 생성



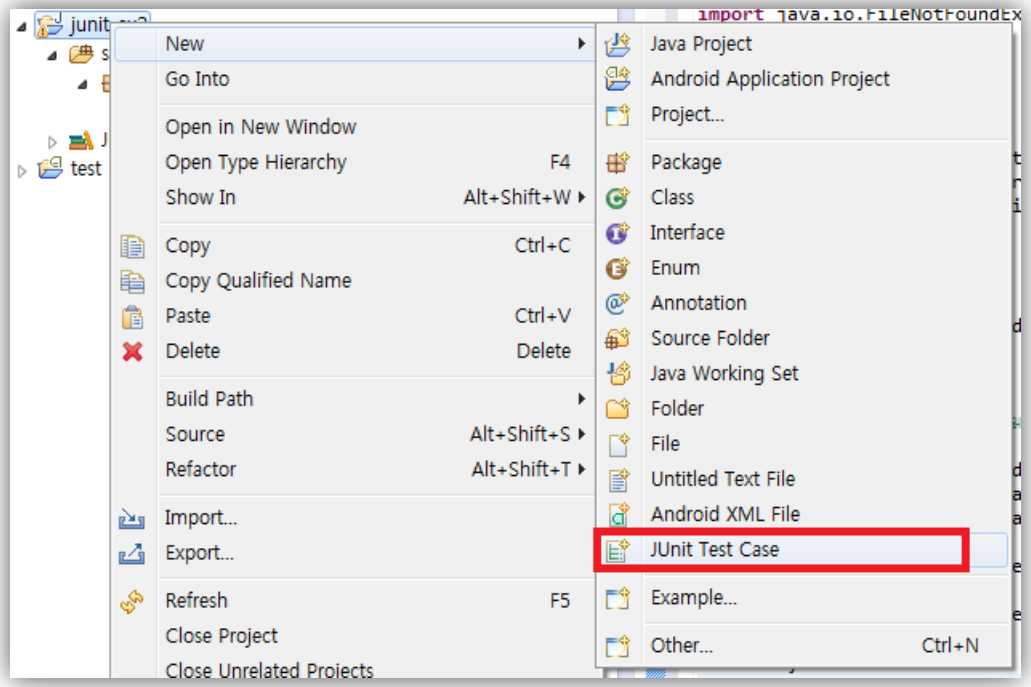
JUnit 사용방법 (계속)



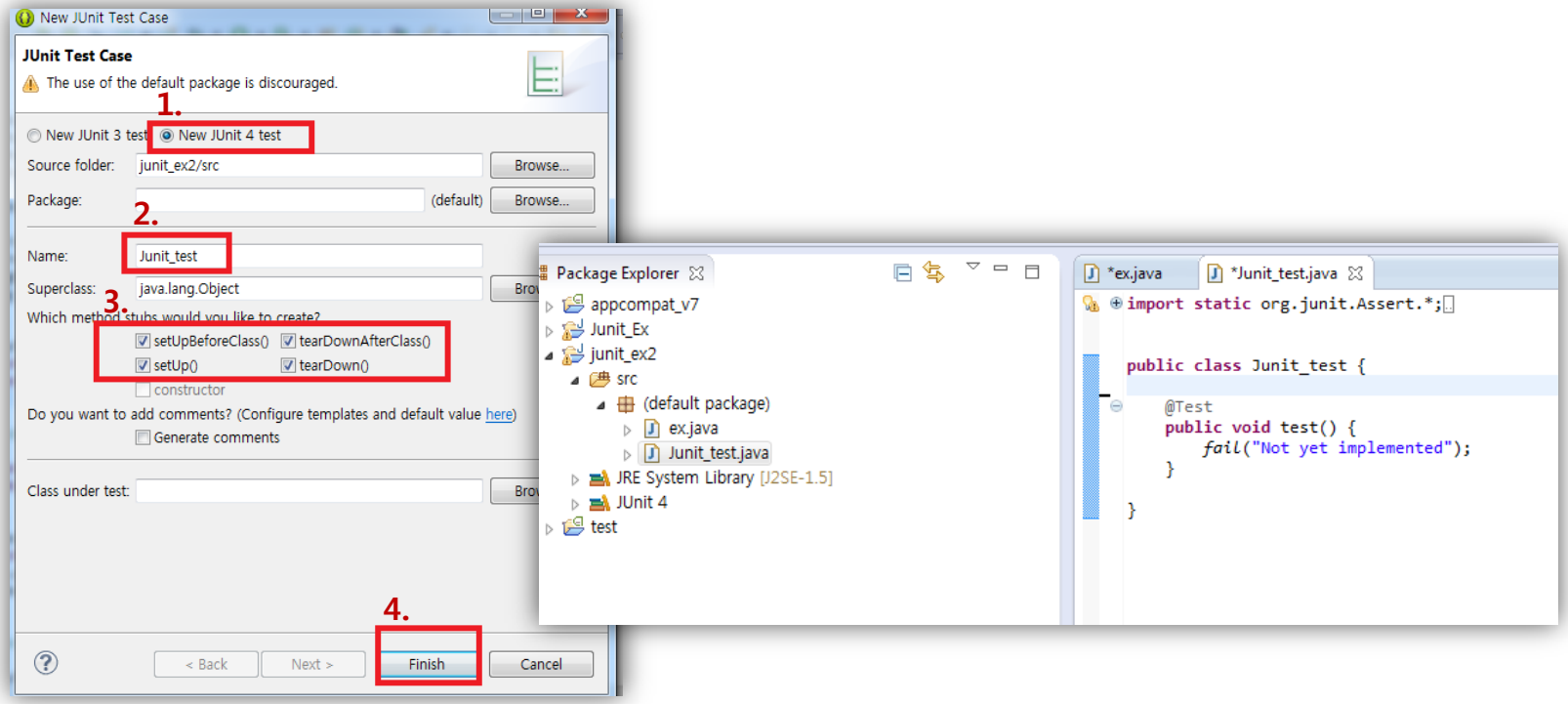
JUnit 사용방법 (계속)



JUnit 사용방법 (계속)



JUnit 사용방법 (계속)



JUnit Test Method

- assertEquals(a, b) : 객체 a,b가 서로 일치하는지 확인
- assertArrayEquals(a,b) : 배열 a,b가 일치하는지 확인(순서포함)
- assertNull(a) : 객체가 Null을 참조하는지 시험
- assertNotNull(a) : 객체가 실존하는 객체를 확인하는지 시험
- assertTrue(a) : 조건식 a가 참인지 확인
- assertFalse(a) : 조건식 a가 False인지 확인
- assertSame(a,b) : a가 참조하는 객체를 b도 참조하는지 시험
- asserNotSame(a,b) :a와 b가 서로 다른 객체를 참조하는지 시험
- fail() : 테스트 결과를 실패로 처리한다.
- 기타 : http://junit.sourceforge.net/javadoc/org/junit/Assert.html

JUnit of Annotation

Annotation	설명
@Test	Unit Test를 수행하는 대상 method
@Before	각 Unit test의 method 실행 전에 실행되는 method
@After	각 Unit test의 method 실행 후에 실행되는 method
@BeforeClass	Class안에 정의된 모든 method에 대해서 Test 전,후에 한번만 호출된다. 객체 생성 등에 사용.
@AfterClass	
@Ignore	테스트를 수행하지 않을 method
@RunWith(value=class)	unit Test 클래스를 실행하기 위한 러너(Runner)를 명시적으로 지정할 수 있다.
@SuiteClasses(value=classes)	보통 여러 개의 Test Class를 수행하기위해 쓰인다. @Runwith를 이용해 Suite.class를 러너로 사용한다.
@Parameters	하나의 method에 대해 다양한 테스트 값을 한꺼번에 실행시키고자 할 때 사용한다.

사용 예제

JUnit 사용 예제 코드

```
public class Testcode {

    public static String connection="off";

    public String getconnect(){
        return connection;
    }

    public void connect(){
        connection="on";
    }

    public void disconnect(){
        connection="off";
    }

    public String Testread(String path) {

        if(getconnect()=="on"){

        BufferedReader bufferedReader = null;    // path로부터 텍스트 파일을 읽을 객체
        StringBuilder textBuilder = new StringBuilder();    // 파일로부터 읽은 텍스트를 중간에 담을 객체
        String text = null;    // 결과 텍스트를 담는 객체입니다.

        try {
            bufferedReader = new BufferedReader(new FileReader(path));
        } catch (FileNotFoundException e) {
            return null;
        }

        // 텍스트 파일로부터 문자들을 읽어들이고, 이를 textBuilder에 추적시킵니다.
        try {
            int characterRead = -1;
            while (-1 != (characterRead = bufferedReader.read())) {
                textBuilder.append((char)characterRead);
            }
            text = textBuilder.toString();
        } catch (IOException e) {
            e.printStackTrace();
```

※예제 코드 :
Text 파일을 읽어오는 간단한 프로그램

사용 예제

JUnit 사용 예제(Annotation)

1.
```
@BeforeClass // 테스트 클래스 시작전에 불러오는 method
public static void Setup() throws Exception{
    textReader = new Testcode();
    System.out.println("@BeforeClass Annontation : Testcode 객체 생성");

    textReader.connect(); //연결
    System.out.println("connection 연결");

}
```

2.
```
@AfterClass
public static void Setoff() throws Exception{
    textReader.disconnect(); //연결해제
    System.out.println("@AfterClass Annontation : 연결 해제");
}
```

3.
```
@Before
public void Setup_path(){
    text_path="test.txt";
    text_path2="nontext.txt";
    System.out.println("@Before Annontation : text path 설정");
}
```

4.
```
@After
public void Setoff_path(){
    text_path=null;
    text_path2=null;
    System.out.println("@After Annontation : text path,2 초기화");
}
```

Result

```
@BeforeClass Annontation : Testcode 객체 생성
connection 연결
@Before Annontation : text path 설정
@After Annontation : text path,2 초기화
@Before Annontation : text path 설정
@After Annontation : text path,2 초기화
@Before Annontation : text path 설정
@After Annontation : text path,2 초기화
@Before Annontation : text path 설정
@After Annontation : text path,2 초기화
@Before Annontation : text path 설정
@After Annontation : text path,2 초기화
@AfterClass Annontation : 연결 해제
```

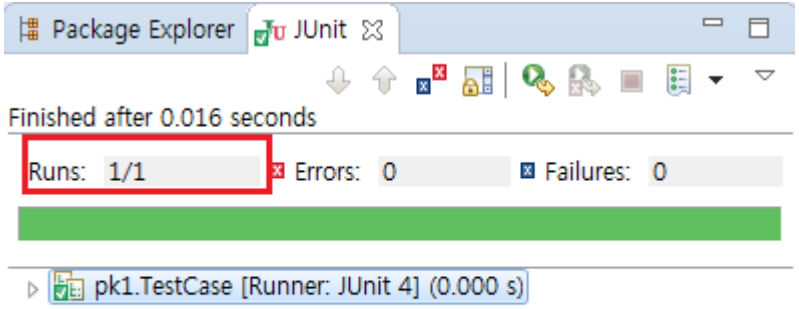
붉은색 : @BeforeClass 결과
검은색 : @AfterClass 결과
파란색 : @Before 결과
초록색 : @After결과

사용 예제

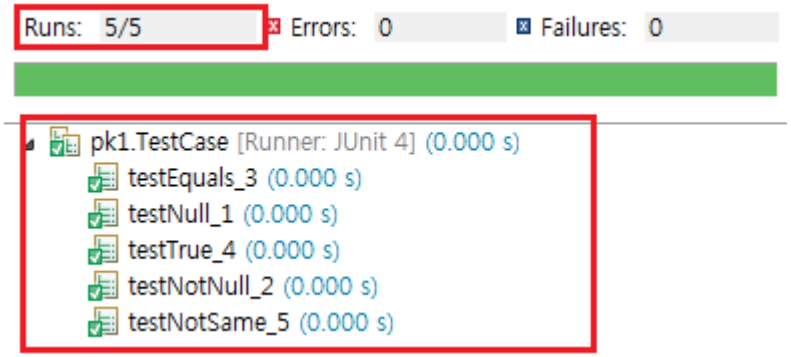
JUnit 사용 예제(Annotation)

5.
```
@Test
public void testNull_1() {
```

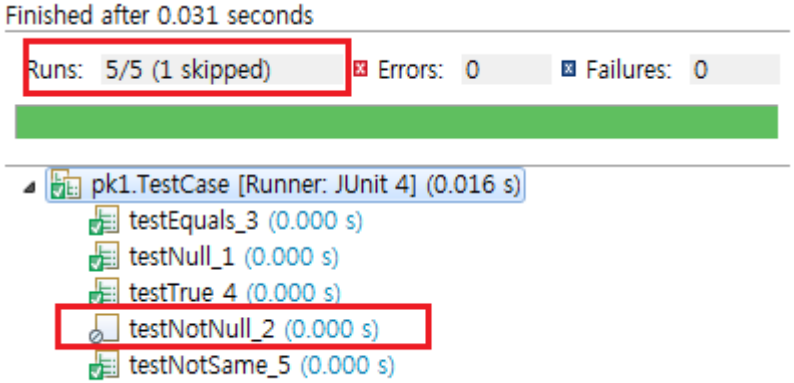
6.
```
@Ignore
public void testNotNull_2(){

    // TestCase 2 : assertNotNull
    text = textReader.Testread(text_path);
    assertNotNull(text);
```



- 5개의 Test case를 중 1case만 @Test를 선언하고 나머지 Case에는 @Test 선언을 제거한 경우



- 5개의 Test case를 정상적으로 실행했을 때



- 5개의 Test case중 testNotNull_2 에 @Ignore선언

사용 예제

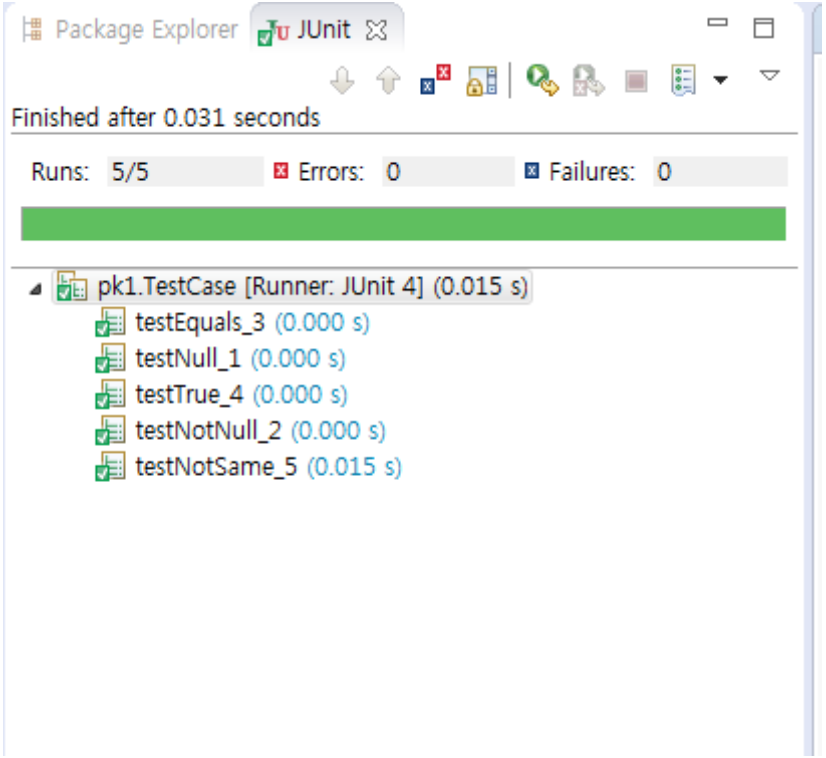
JUnit 사용 예제

```
@Test
public void testNull_1() {

    //Test case 1 : assertNull
    text = textReader.Testread(text_path2);
    assertNull(text);
}

@Test
public void testNotNull_2(){

    // TestCase 2 : assertNotNull
    text = textReader.Testread(text_path);
    assertNotNull(text);
}

@Test
public void testEquals_3(){

    // TestCase 3 : assertEquals 파일의 내용 확인
    text = textReader.Testread(text_path);
    assertEquals("Test!", text);

}
```

```
@Test
public void testTrue_4(){

    //TestCAse 4 : assertTrue 현재 커넥션 상태가 ON인지 확인.
    boolean T=false;
    text = textReader.getconnect();
    if(text=="on")
        T=true;
    else
        T=false;
    assertTrue(T);

}
@Test
public void testNotSame_5(){

    //TestCAse 5 : assertSame
    text_path2="test2.txt";
    text = textReader.Testread(text_path);
    text2 = textReader.Testread(text_path2);

    assertNotSame(text, text2);

}
```

사용 예제

JUnit 사용 예제

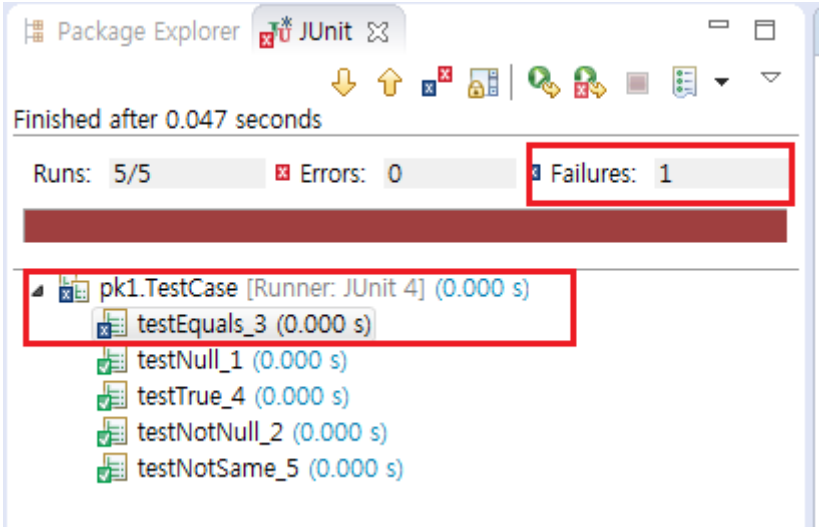


사용 예제

JUnit 사용 예제

```
@Test
public void testEquals_3(){

    // TestCase 3 : assertEquals 파일의 내용 확인
    text = textReader.Testread(text_path);
    assertEquals("Test!", text);

}
```

```
@Test
public void testEquals_3(){

    // TestCase 3 : assertEquals 파일의 내용 확인
    text = textReader.Testread(text_path);
    assertEquals("It's Wrong!", text);
```



3 빌드 환경

3-1 Ant

- Ant
- Ant 설치
- build.xml
- 빌드하는 방법 1
- 빌드하는 방법 2

3-2 Hudson

- Hudson
- Hudson 설치
- 빌드하는 방법

빌드 환경

Ant

Ant(Another Neat Tool)

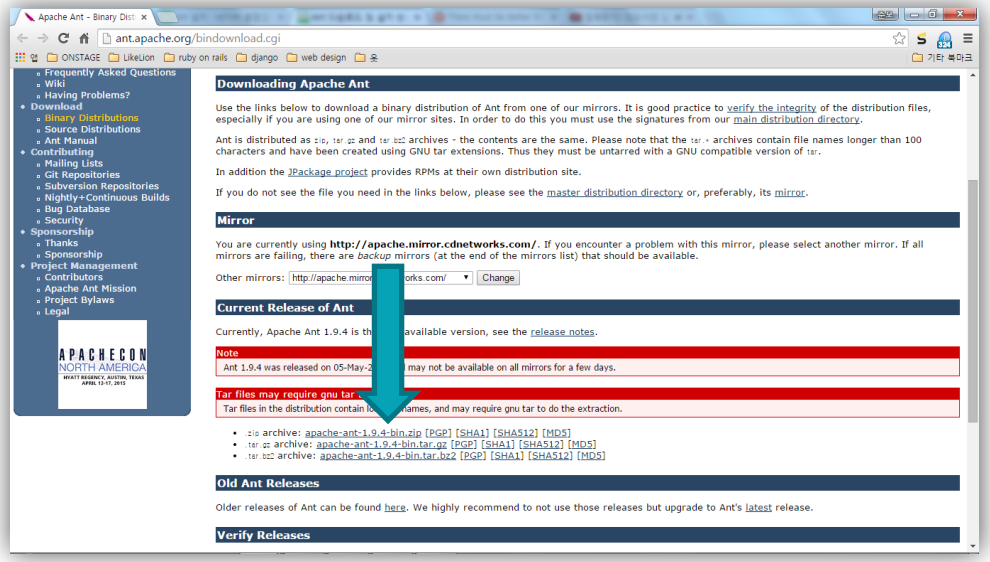
- 자바 기반 자동화 빌드 도구
- 빌드를 위한 설정 및 환경 구성을 XML파일을 사용
- 기본 XML파일은 build.xml

빌드 환경

Ant

Ant 설치

- Eclipse는 기본 플러그인으로 내장되어 있음
- 혹은 http://ant.apache.org/bindownload.cgi에서 다운

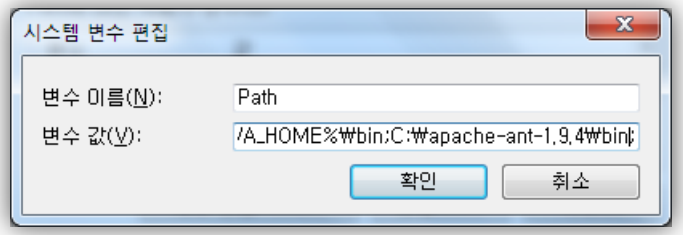


다운로드 받은 후
원하는 곳에 압축을 푼다.

Ant 설치 (계속)

- 시스템 변수 편집

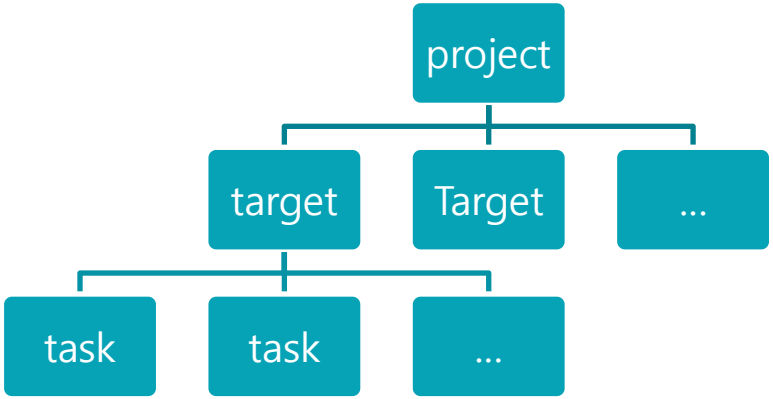
  : 'ant설치폴더₩bin' 경로를 시스템 변수 Path의 값에 추가한다.



빌드 환경

Ant

build.xml




```
<project basedir="." default="build" name="Practice">
    <property environment="env"/>
    <property name="debuglevel" value="source,lines,vars"/>
    <property name="target" value="1.8"/>
    <property name="source" value="1.8"/>
    <path id="Practice.classpath">
        <pathelement location="bin"/>
    </path>
    <target name="init">
        <mkdir dir="bin"/>
        <copy includeemptydirs="false" todir="bin">
            <fileset dir="src">
                <exclude name="**/*.java"/>
            </fileset>
        </copy>
    </target>
    <target name="clean">
        <delete dir="bin"/>
    </target>
    <target depends="clean" name="cleanall"/>
    <target depends="build-subprojects,build-project" name="build"/>
```

build.xml (계속)

task

- 특정한 작업을 수행하는 코드 단위
- java, javac, jar, mkdir, copy, exec 등
- 기본적인 형태

  <task_name attribute="value1" attribute2="value2" ... />

  예) `<copy todir="${ant.library.dir}">`

build.xml (계속)

target

속성	설명	타입	필수
name	타겟 이름	String	yes
depends	의존성이 있는 타겟 목록을 ','(콤마)로 구분	String	no
if	타겟이 실행되려면 설정해야 할 프로퍼티 이름	Property명	no
unless	타겟이 실행되려면 설정되지 말아야 할 프로퍼티 이름	Property명	no
description	타겟의 설명	String	no

```
<target depends="clean" name="cleanall"/>
<target depends="build-subprojects,build-project" name="build"/>
<target name="build-subprojects"/>
<target depends="init" name="build-project">
```

build.xml (계속)

project

속성	설명	타입	필수
name	프로젝트 이름	String	no
default	지정한 타겟이 없을 때 자동으로 실행할 디폴트 타겟 이름	String	yes
basedir	빌드 파일 내에서 경로 지정의 기본이 되는 디렉토리	path	no

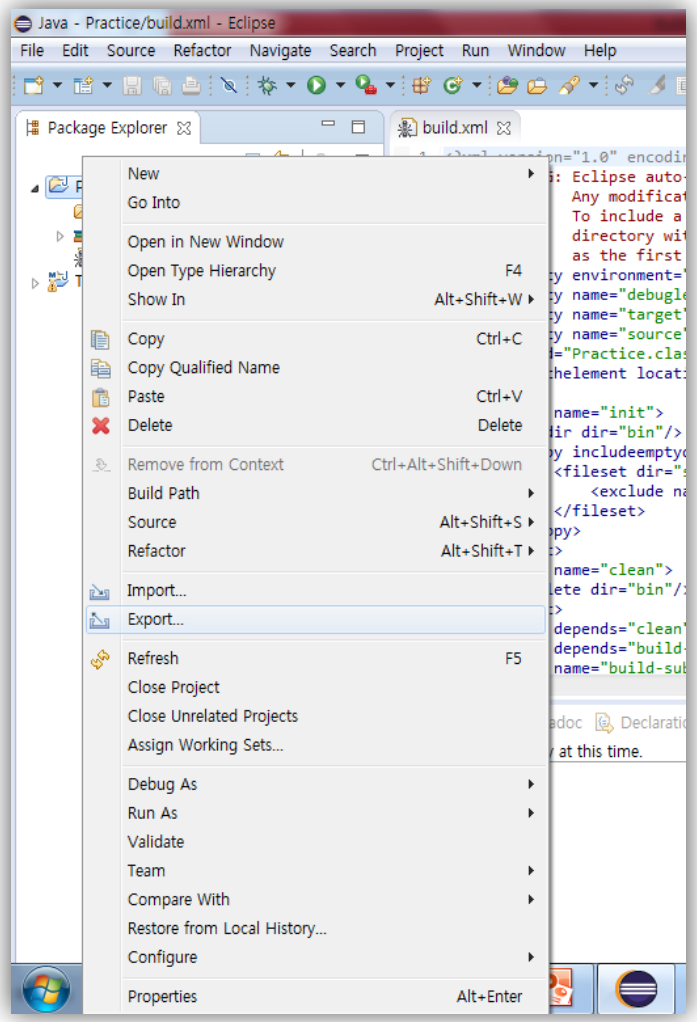
```
<project basedir="." default="build" name="Practice">
```

빌드 환경

Ant

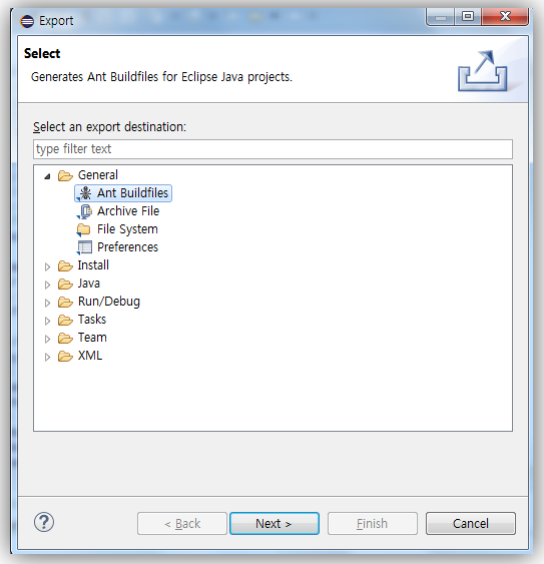
빌드하는 방법 1

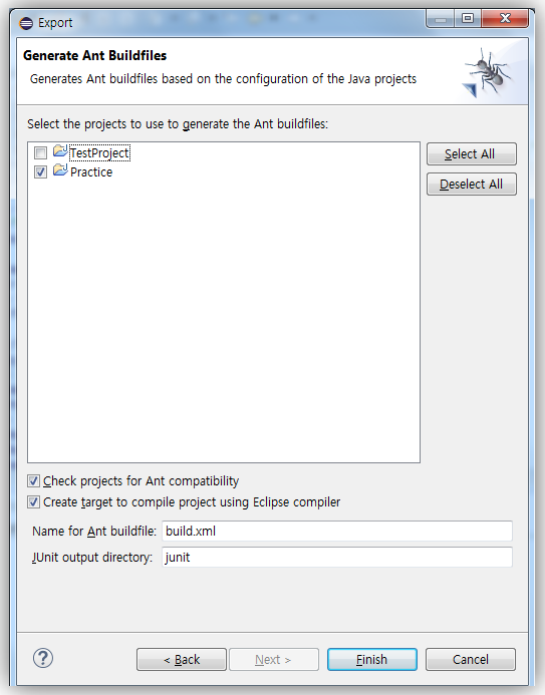
- 프로젝트 생성 후 마우스 오른쪽 클릭 → Export



빌드하는 방법 1 (계속)

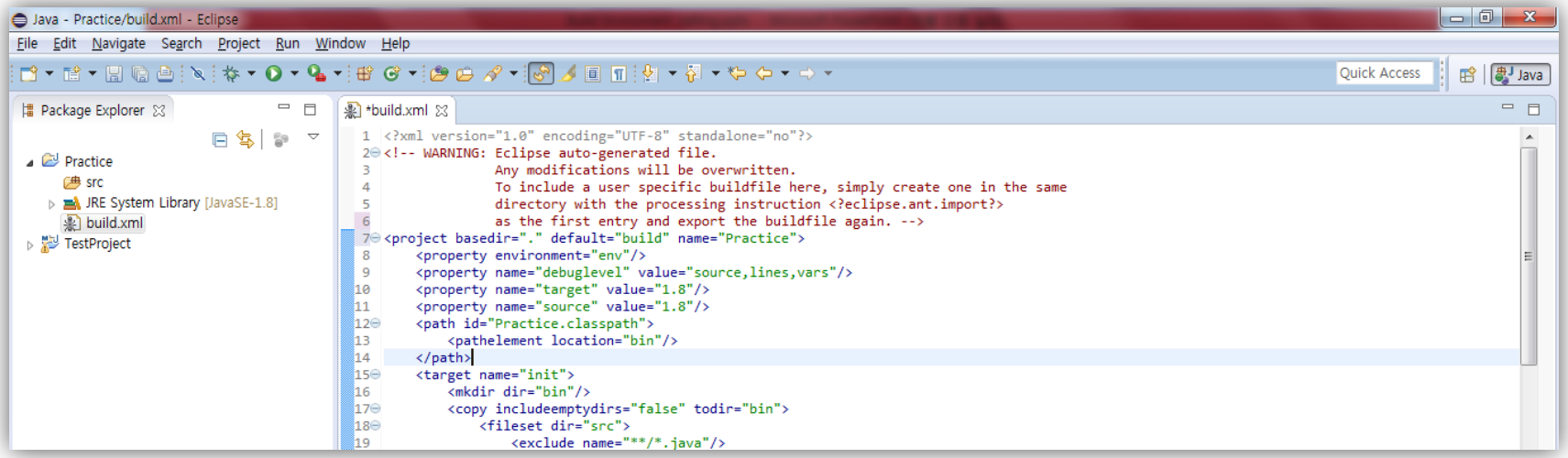
- Export → Ant Buildfiles → Finish





빌드하는 방법 1 (계속)

- build.xml이 프로젝트 내에 생성됨

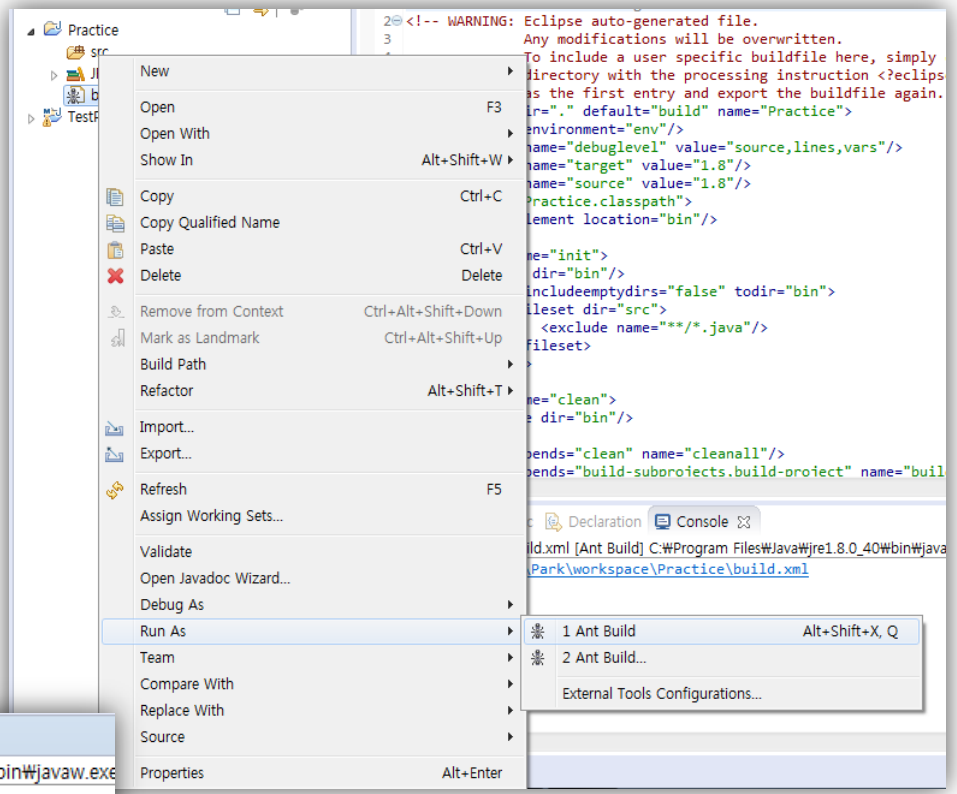


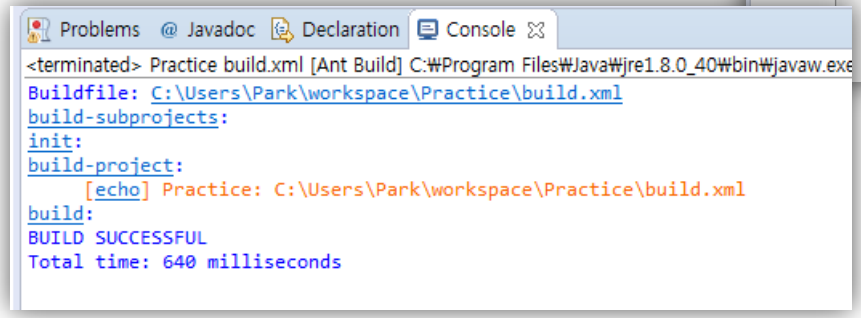
빌드 환경

Ant

빌드하는 방법 1 (계속)

- build.xml 마우스 우클릭 → Run as → Ant Build



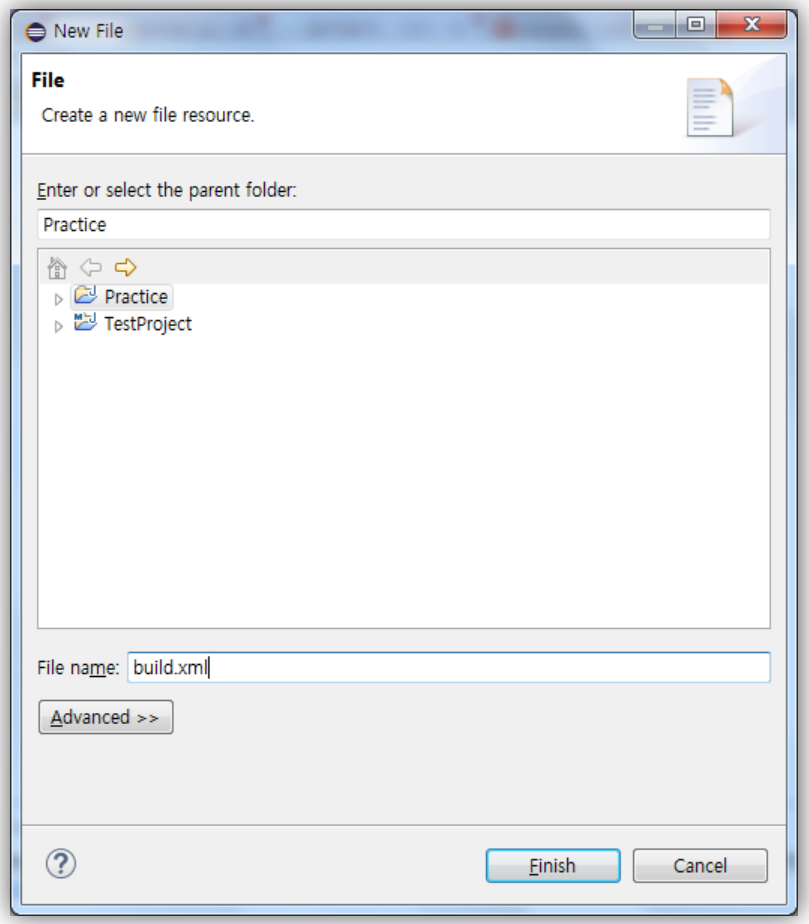


➜ 결과

빌드 환경

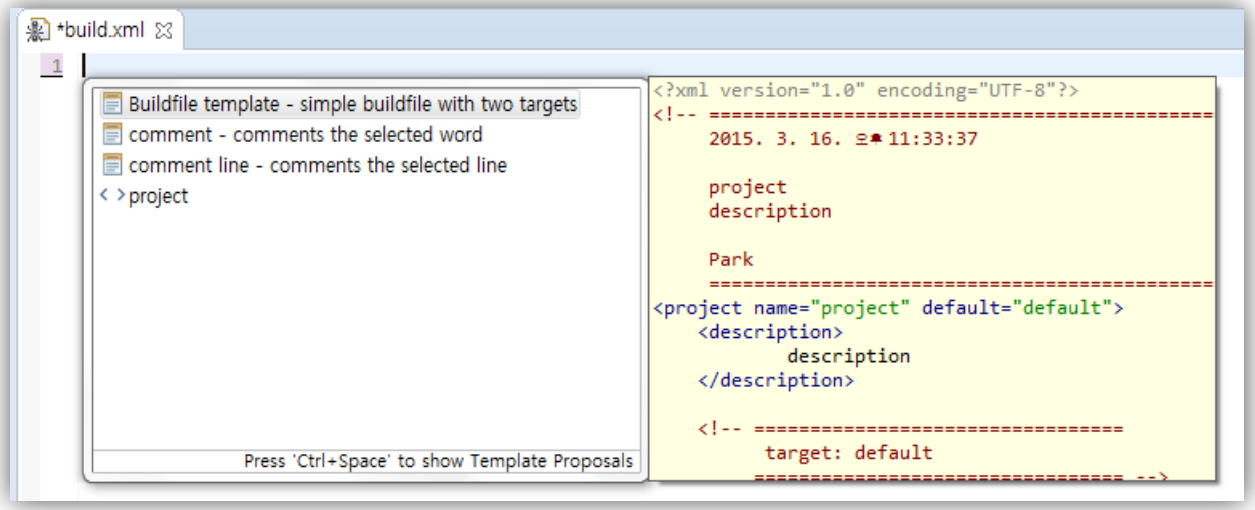
빌드하는 방법 2

- XML파일을 프로젝트 내에 직접 생성



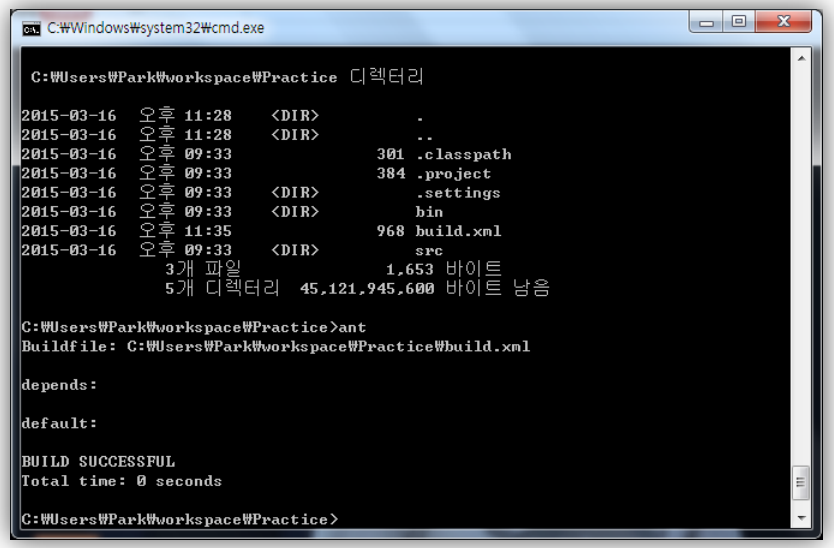
빌드하는 방법 2(계속)

- Ctrl+ space 로 buildfile의 기본 xml 템플릿을 생성
- Project, target, task를 맞게 작성한 다음 Ant Build 실행



빌드하는 방법 3 (계속)

- CMD창에서 ant 입력



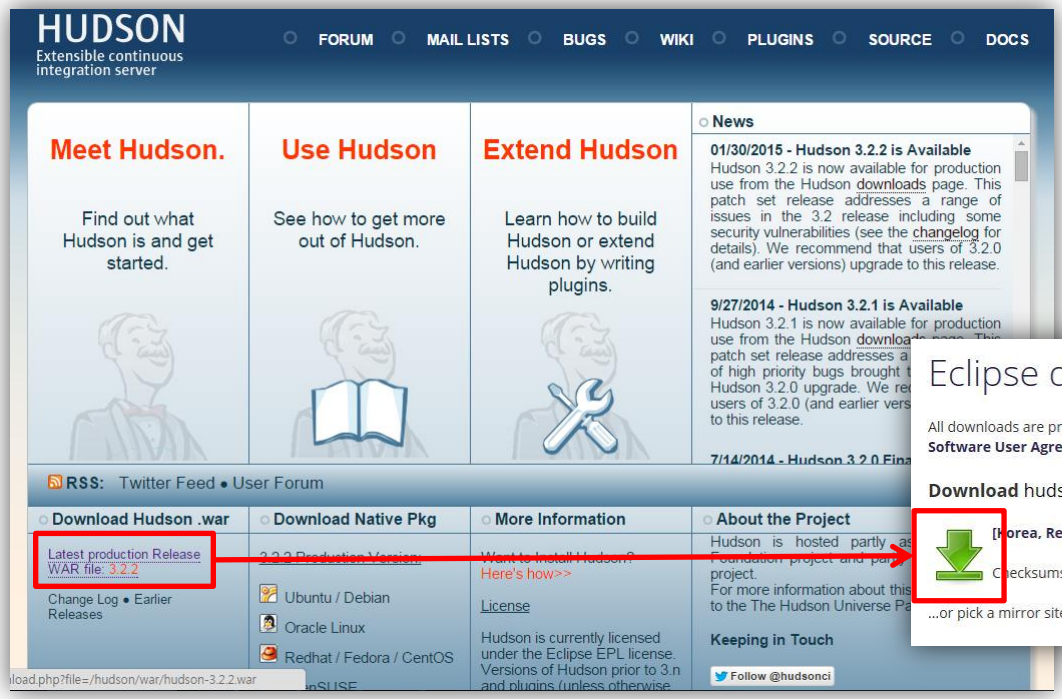
Hudson

- Continuous Integration Tool : 정기적으로 혹은 특정 시기에 자동으로 빌드
- Git과 SourceTree로 버전 관리

빌드 환경

Hudson

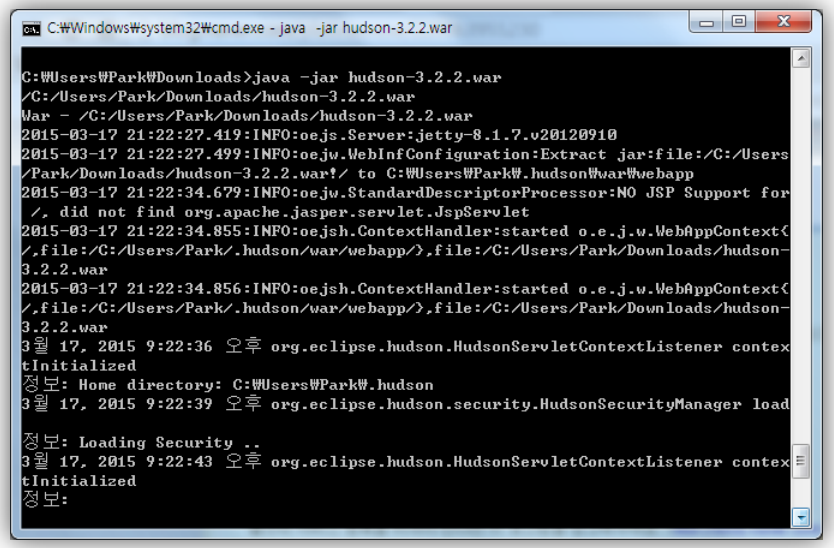
Hudson 설치



Hudson.war 다운로드 :
http://hudson-ci.org/

Hudson 설치 (계속)

- java -jar hudson.war 명령으로 hudson 설치 및 실행

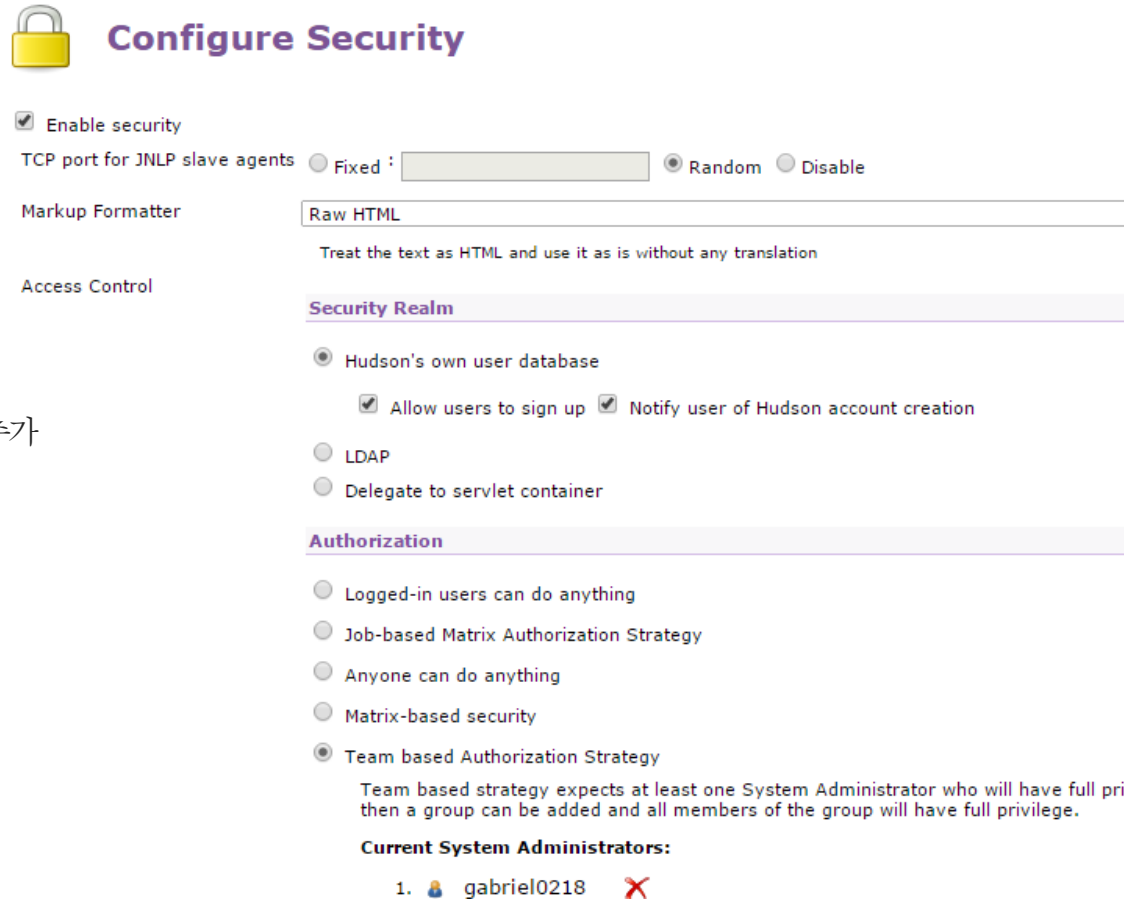


빌드하는 방법

계정 권한 설정

- Admin으로 사용할 계정 생성 후

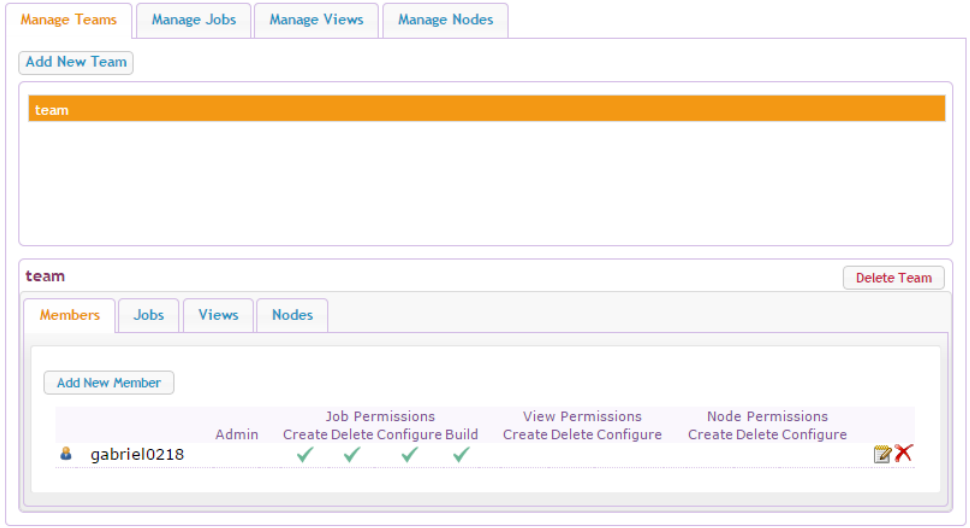
  Manage hudson – configure security에서 추가



빌드하는 방법 (계속)

계정 권한 설정

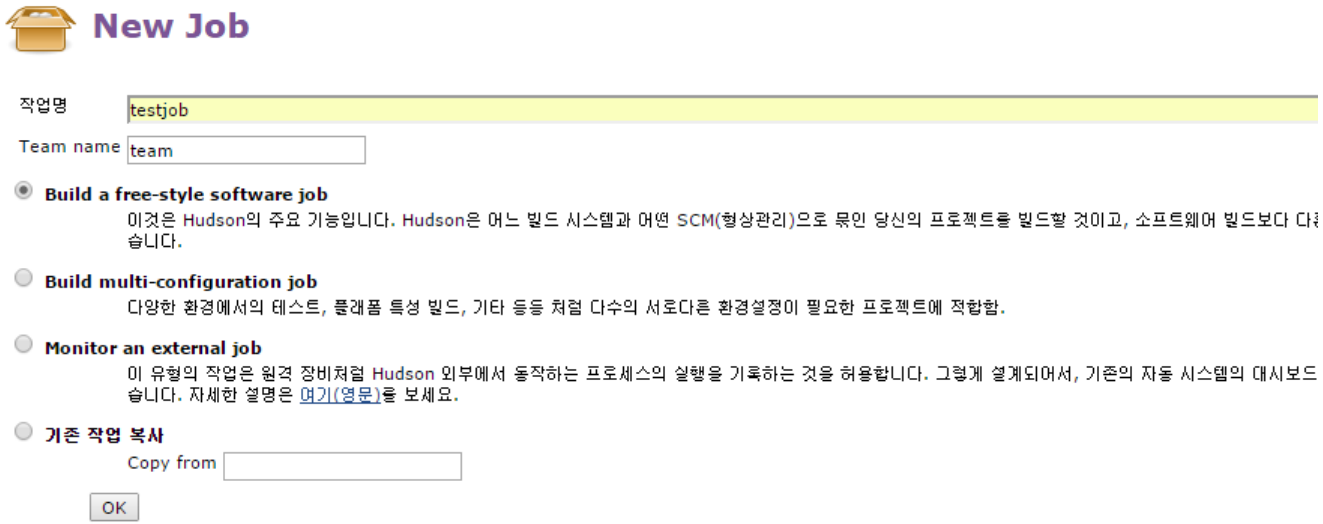
- Manage team에서 작업할 사용자를 추가



빌드하는 방법 (계속)

새 작업 만들기

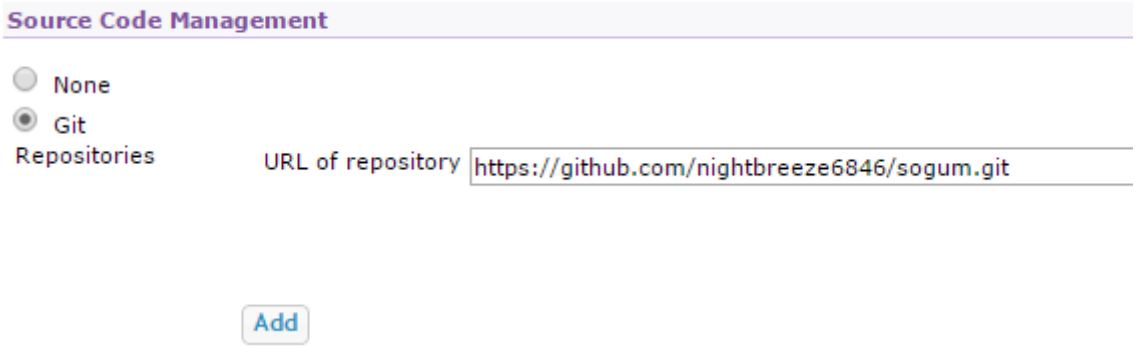
- 새 작업을 만들고 과 작업을 수행할 팀을 지정



빌드하는 방법 (계속)

작업 설정

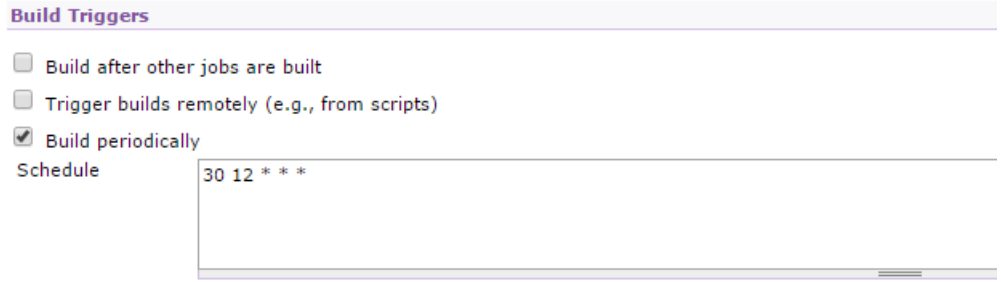
- 작업 설정에서 코드를 저장할 git repository를 설정



빌드하는 방법 (계속)

작업 설정

- Build Triggers – Schedule란에 '분 시 일 월 요일' 순서로 자동 빌드할 시간을 기록
- *는 전부를 뜻함



분	0 ~ 59 사이의 값을 입력합니다.
시	0 ~ 23 사이의 값을 입력합니다.
일	1 ~ 31 사이의 값을 입력합니다.
월	1 ~ 12 사이의 값을 입력합니다.
요일	0 ~ 7 사이의 값을 입력합니다. (0과 7은 일요일입니다.)

빌드하는 방법 (계속)

빌드

- Build now를 클릭하면 즉시 빌드 시작

빌드하는 방법 (계속)

빌드 결과

- 빌드 시간, 코드 변경사항, 에러 등 확인 가능

Q&A

Question and Answer

참고

출처: http://wearehacker.com/index.php?mid=java&category=0&page=2&document_srl=62684
출처: http://blog.naver.com/hakchangs/130110701396
출처: http://ko.wikipedia.org/wiki/%EC%9E%90%EB%B0%94_%EA%B0%80%EC%83%81_%EB%A8%B8%EC%8B%A0

출처: http://blog.naver.com/ksh81850/220020148964
출처: http://goo.gl/YzTZUn
출처:
http://kin.naver.com/qna/detail.nhn?d1id=1&dirId=1040201&docId=67978846&qb=amRrIOqzteuwsQ==&enc=utf8&section=kin&rank=9&search_sort=0&spq=0

출처:
http://kin.naver.com/qna/detail.nhn?d1id=1&dirId=1040202&docId=148199275&qb=7J207YG066a97IqkIOyEpOy5mCDsmKTrpZg=&enc=utf8&section=kin&rank=2&search_sort=0&spq=0
출처: http://bitkorea.net/220179446077
출처:
http://kin.naver.com/qna/detail.nhn?d1id=1&dirId=10402&docId=118493438&qb=7J207YG066a97IqkIOyEpOy5mCDsmKTrpZg=&enc=utf8&section=kin&rank=4&search_sort=0&spq=0

출처:
http://ko.wikipedia.org/wiki/%EC%9D%B4%ED%81%B4%EB%A6%BD%EC%8A%A4_(%EC%86%8C%ED%94%84%ED%8A%B8%EC%9B%A8%EC%96%B4)
출처: http://terms.naver.com/entry.nhn?docId=2454871&cid=42346&categoryId=42346

감사합니다☺