

1 : 2018-03-21

JUnit & IntelliJ 및 빌드 환경

# Software Verification

---

**[2018SV][T1]**

201311263 김민환 201311308 전세진 201411278 서희진 201411317 조민규

1.

IDE

IntelliJ

2.

Unit Test

JUnit

3.

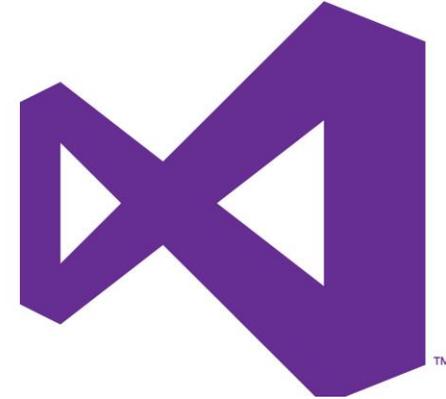
Build Configuration & CI

Gradle with Jenkins

# 1. IDE

## IDE (Integrated Development Environment )

프로그램 개발에 관련된 모든 작업을  
하나의 프로그램안에서 처리하는 환경을 제공하는 **소프트웨어**

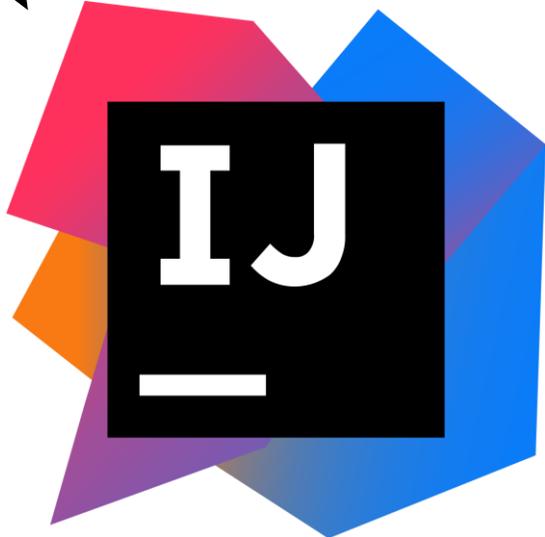


# 1. IDE



# 1. IDE

장점



IntelliJ

Jetbrains 사에서 개발한 Java IDE

## 편리한 설치

설치단계부터 단계별 설정가능

Project 단위로 관리

## 편리한 인터페이스

Workspace 단위로 관리

html,css,javascript,...

## 다양한 언어지원

다른 언어 사용시  
플러그인 설치 필요

## 안정적인 plugin

plugin끼리의 의존성문제

## 안정적인 IDE

호환성 문제 다수..

단점



# 1. IDE

장점

## 무료 사용가능

US \$ 499.00 / 1year

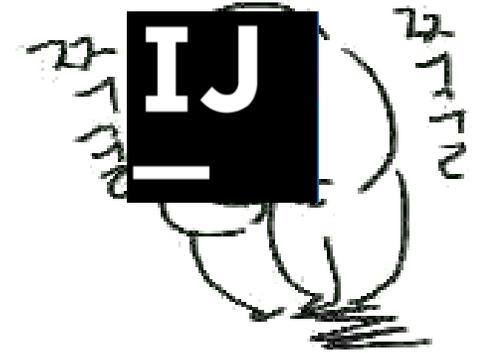
### 파일 수정 시 자동배포 가능

JSP 등 동적로딩이 가능한 파일을 수정하면  
자동으로 배포되지 않음

### Tomcat 구동속도가 빠름

Tomcat 구동속도가 느림

단점



# 1. IDE

무료 사용가능

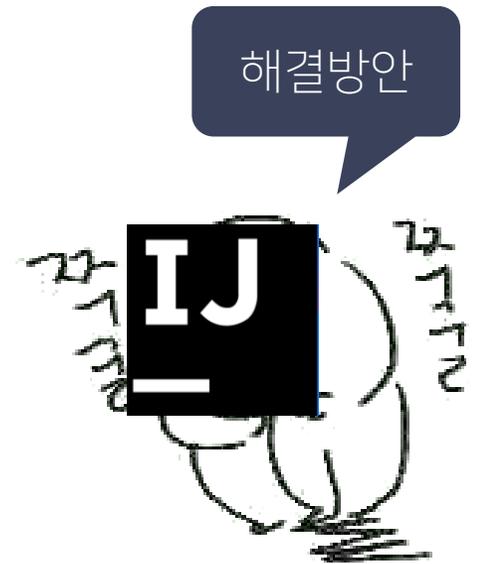
학생용 라이선스 사용!

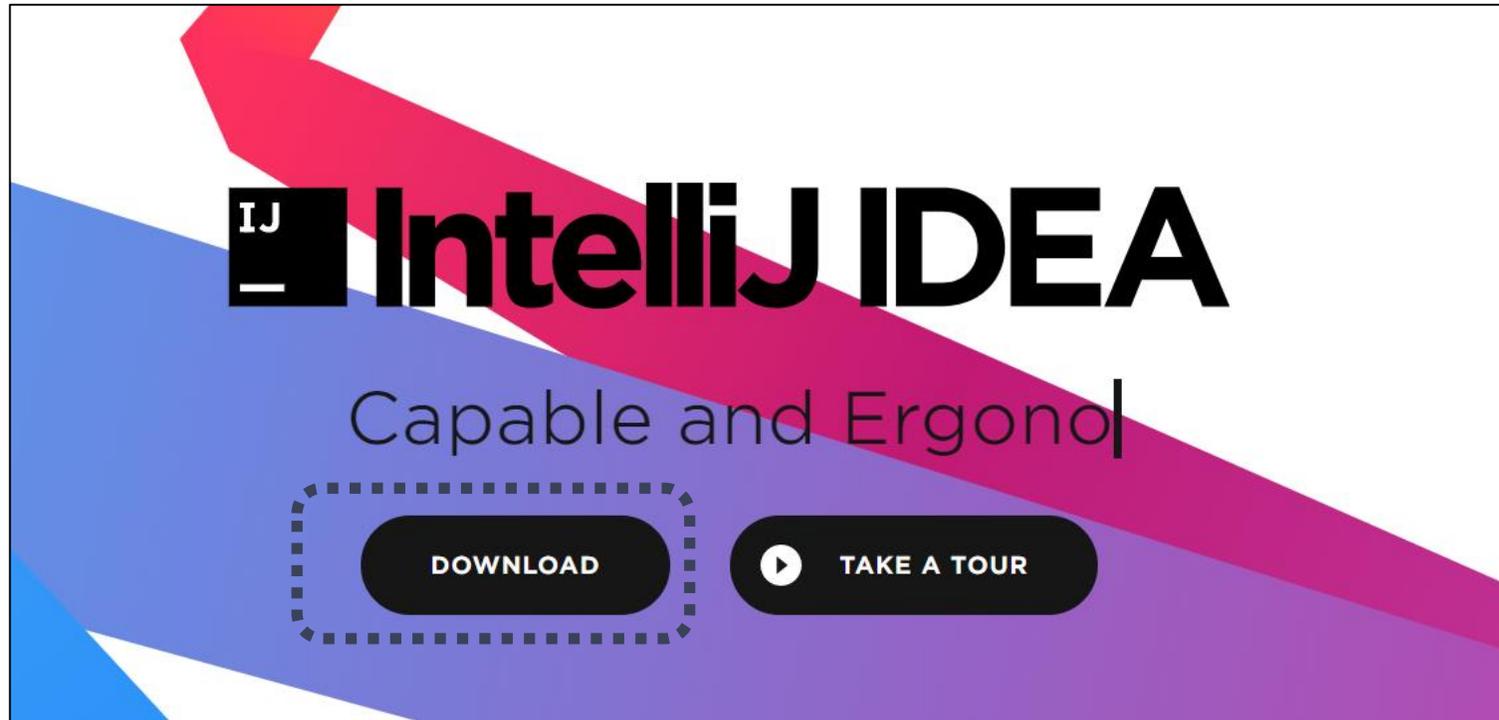
파일 수정 시 자동배포 가능

본 프로젝트에선  
사용할 일 없음

Tomcat 구동속도가 빠름

본 프로젝트에선  
사용할 일 없음





홈페이지 접속

<https://www.jetbrains.com/idea/>

**Download IntelliJ IDEA**

Windows macOS Linux

**Ultimate**  
Web, mobile and enterprise development  
**DOWNLOAD .EXE**  
Free trial

**Community**  
Java, Groovy, Scala and Android development  
**DOWNLOAD .EXE**  
Free, open-source

Version: 2017.3.5  
Build: 173.4674.33  
Released: March 13, 2018  
[Release notes](#)

[System requirements](#)  
[Installation Instructions](#)  
[Previous versions](#)

Ultimate 선택

### 1 License

#### JetBrains Product Pack for Students

[Download](#) ▾

License ID:

23

7

Licensed to: **Heejin Suh**

[Download activation code](#) for offline usage

License

restriction: For educational use only

Valid through: February 20, 2019

Following products included:

- [AppCode](#)
- [CLion](#)
- [DataGrip](#)
- [dotCover](#)
- [dotMemory](#)
- [dotTrace](#)
- [GoLand](#)
- [IntelliJ IDEA Ultimate](#)
- [PhpStorm](#)
- [PyCharm](#)
- [ReSharper](#)
- [ReSharper C++](#)
- [Rider](#)
- [RubyMine](#)
- [WebStorm](#)

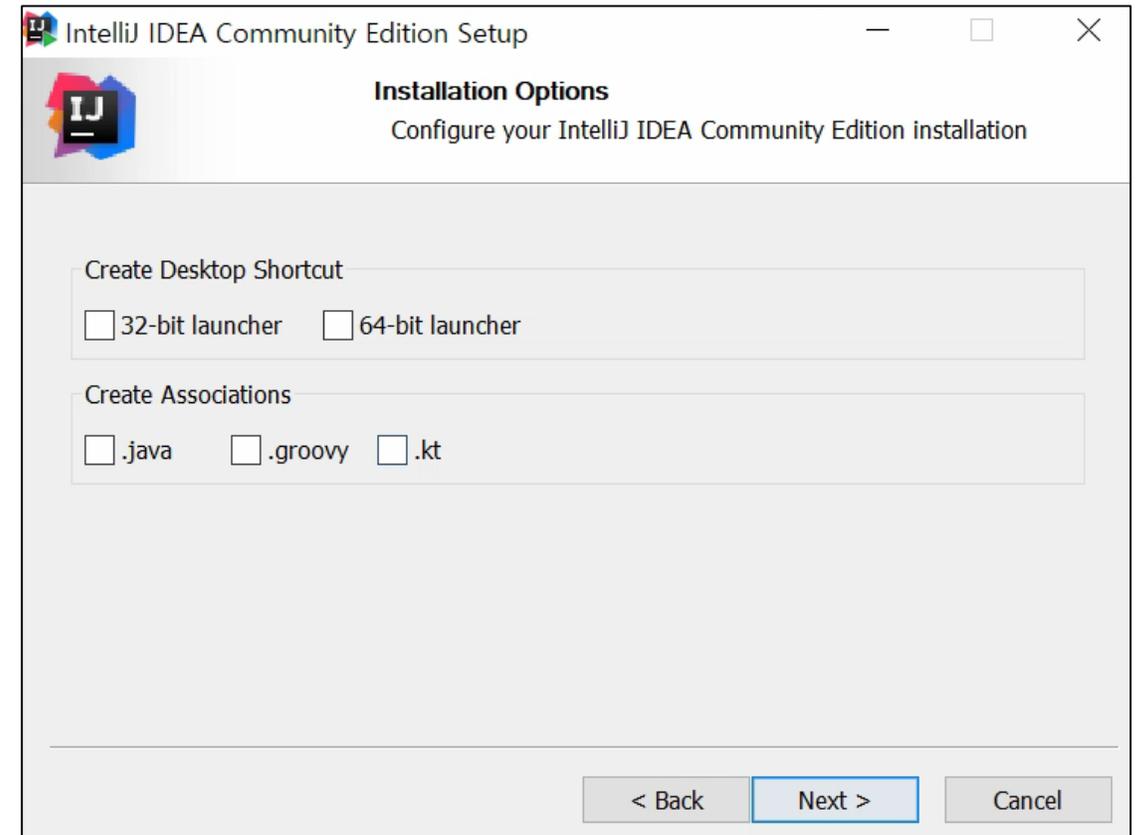
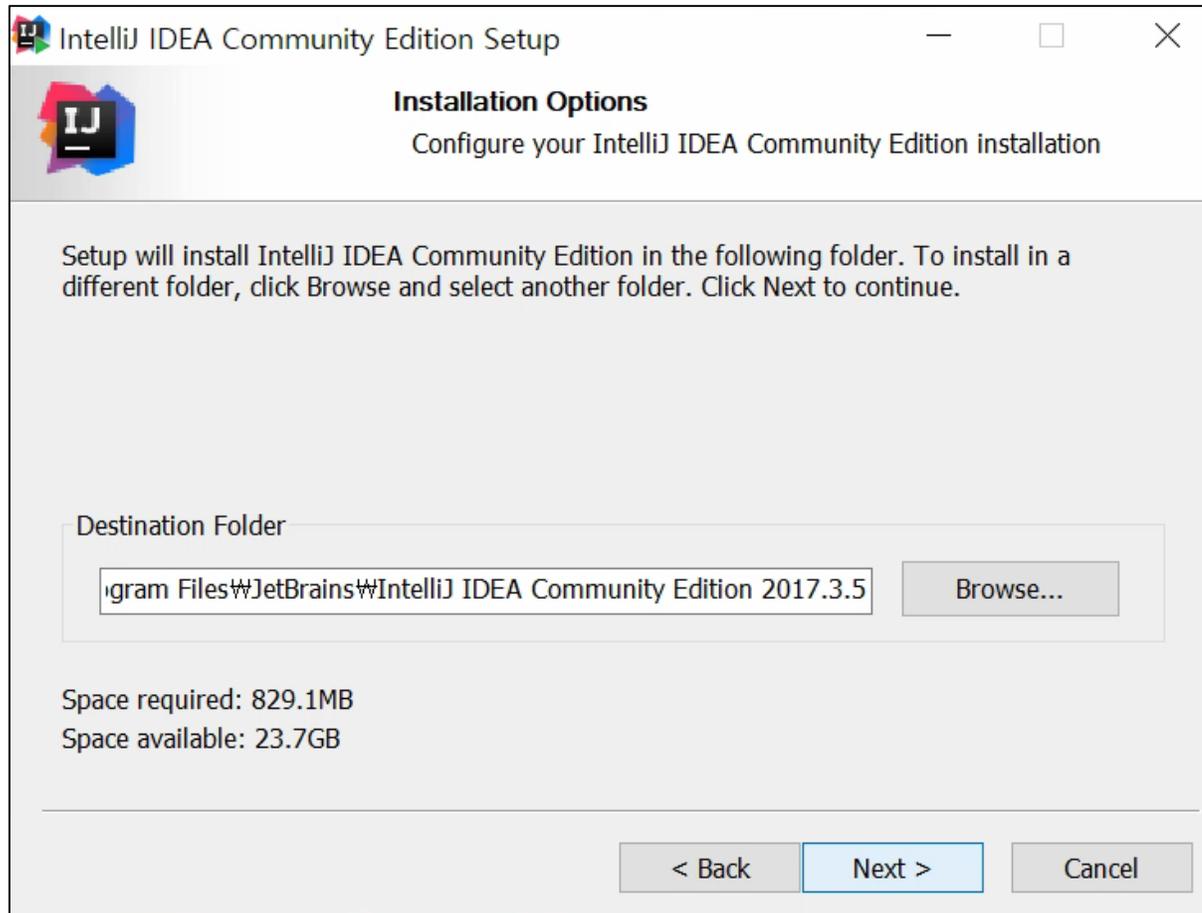
After downloading and installing the software, simply run it and follow the on-screen prompts to sign in with your JetBrains Account.

Can't find your license here? Link your past purchases to your JetBrains Account by [providing a license key or domain](#).

학생용 라이선스

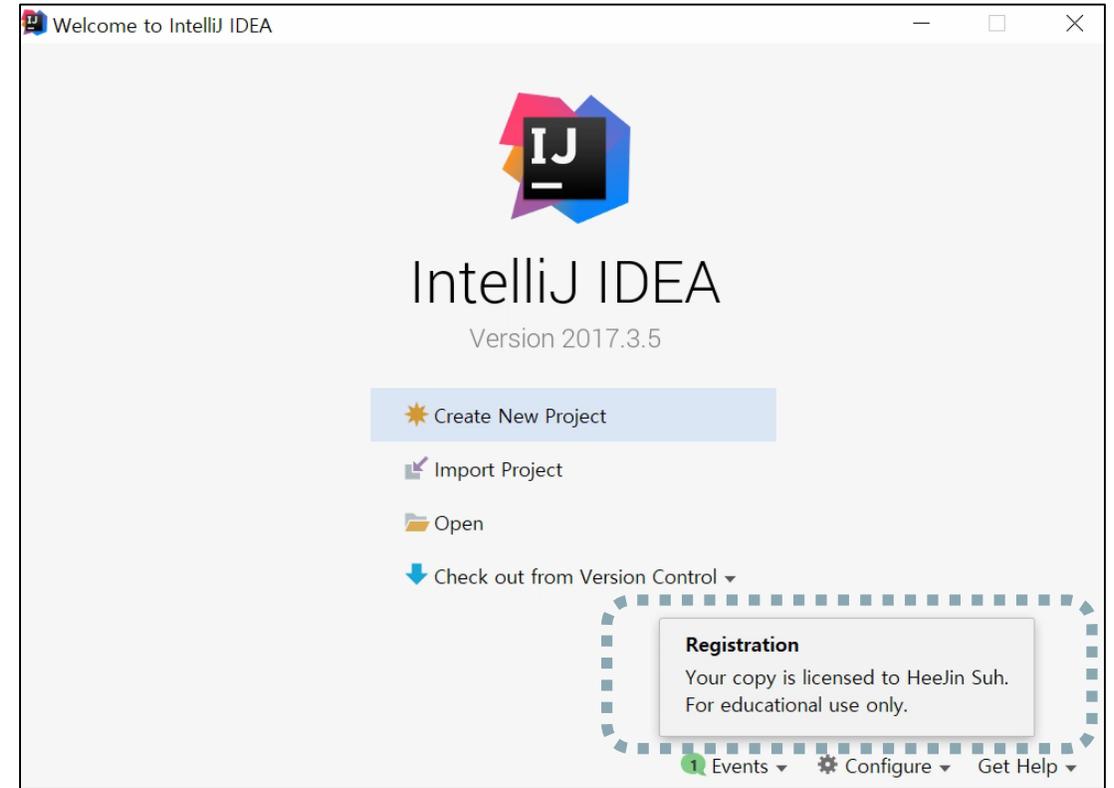
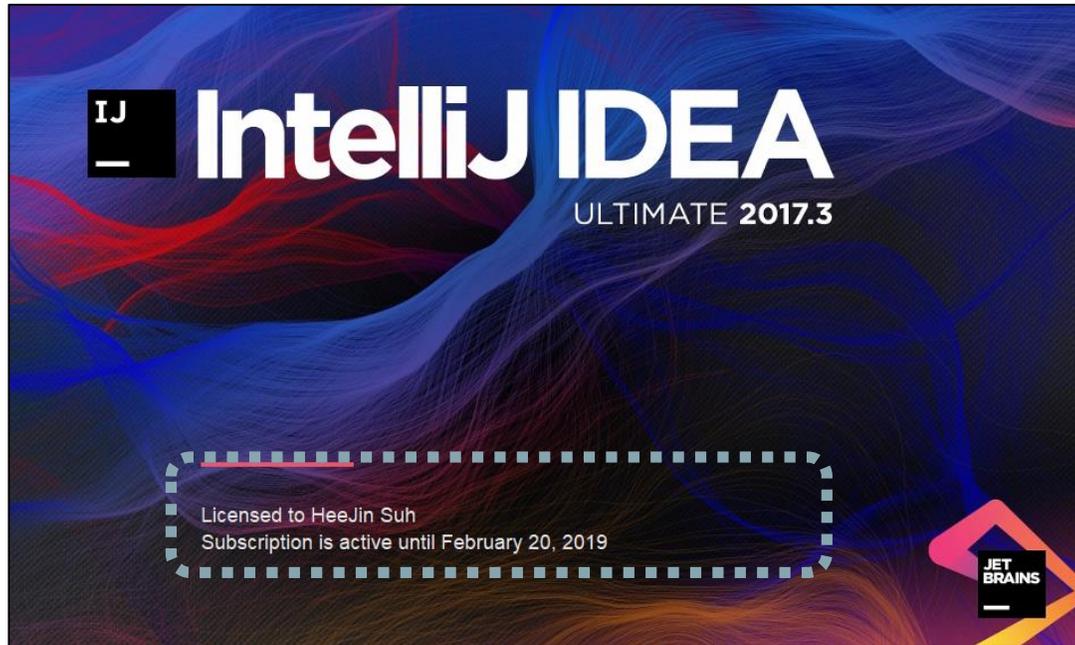
# 1. IDE

## IntelliJ 설치방법



# 1. IDE

## IntelliJ 설치방법



학생용 라이선스 적용 확인

### Java SE Development Kit 8u161

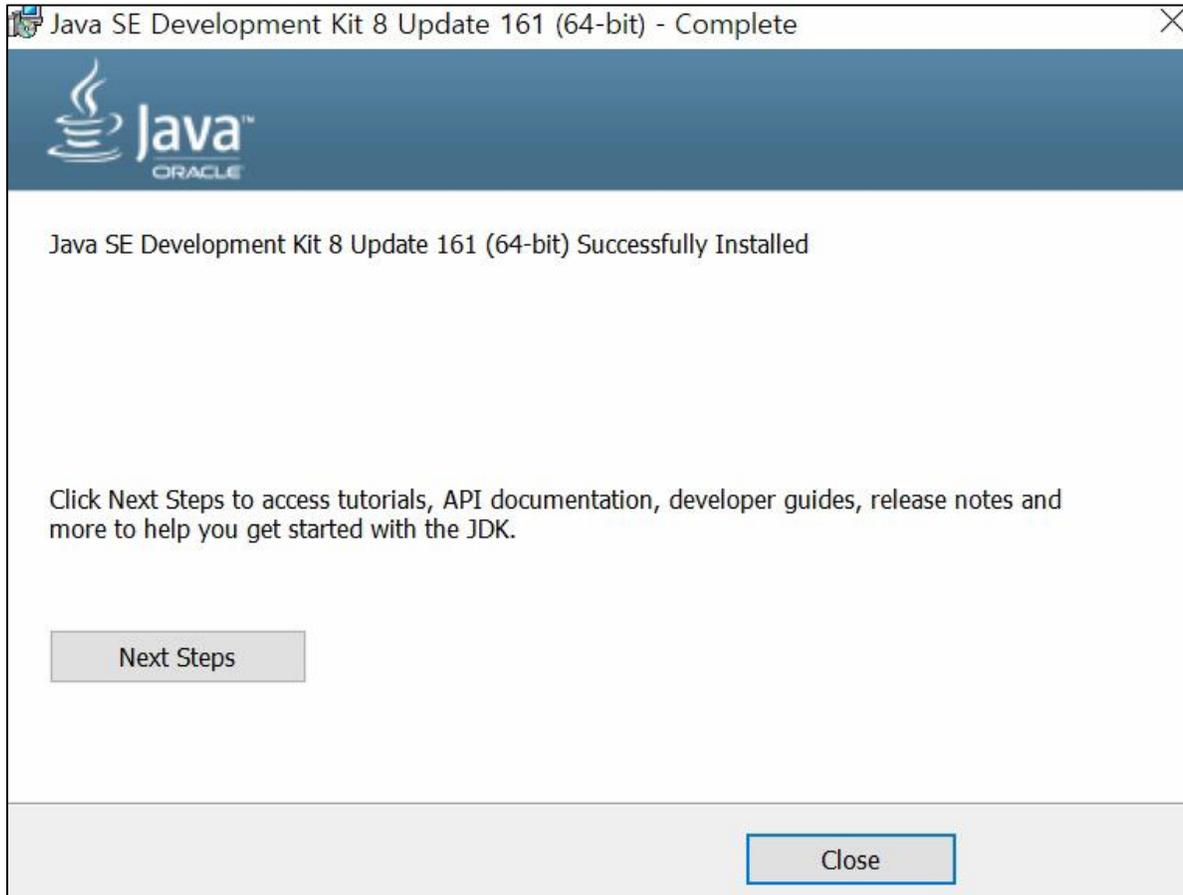
You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

Thank you for accepting the [Oracle Binary Code License Agreement for Java SE](#); you may now download this software.

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.92 MB	<a href="#">jdk-8u161-linux-arm32-vfp-hflt.tar.gz</a>
Linux ARM 64 Hard Float ABI	74.88 MB	<a href="#">jdk-8u161-linux-arm64-vfp-hflt.tar.gz</a>
Linux x86	168.96 MB	<a href="#">jdk-8u161-linux-i586.rpm</a>
Linux x86	183.76 MB	<a href="#">jdk-8u161-linux-i586.tar.gz</a>
Linux x64	166.09 MB	<a href="#">jdk-8u161-linux-x64.rpm</a>
Linux x64	180.97 MB	<a href="#">jdk-8u161-linux-x64.tar.gz</a>
macOS	247.12 MB	<a href="#">jdk-8u161-macosx-x64.dmg</a>
Solaris SPARC 64-bit (SVR4 package)	139.99 MB	<a href="#">jdk-8u161-solaris-sparcv9.tar.Z</a>
Solaris SPARC 64-bit	99.29 MB	<a href="#">jdk-8u161-solaris-sparcv9.tar.gz</a>
Solaris x64	140.57 MB	<a href="#">jdk-8u161-solaris-x64.tar.Z</a>
Solaris x64	97.02 MB	<a href="#">jdk-8u161-solaris-x64.tar.gz</a>
Windows x86	198.54 MB	<a href="#">jdk-8u161-windows-i586.exe</a>
Windows x64	206.51 MB	<a href="#">jdk-8u161-windows-x64.exe</a>

홈페이지 접속

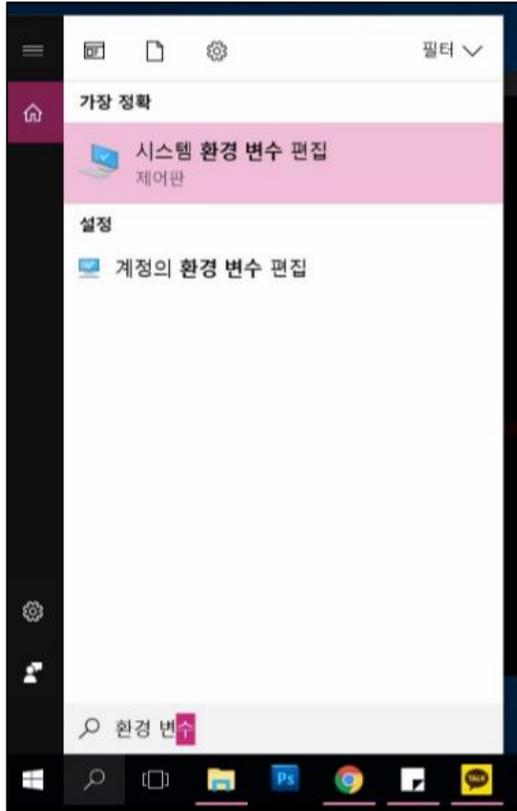
<http://www.oracle.com/technetwork/java/javase/downloads/index.html>



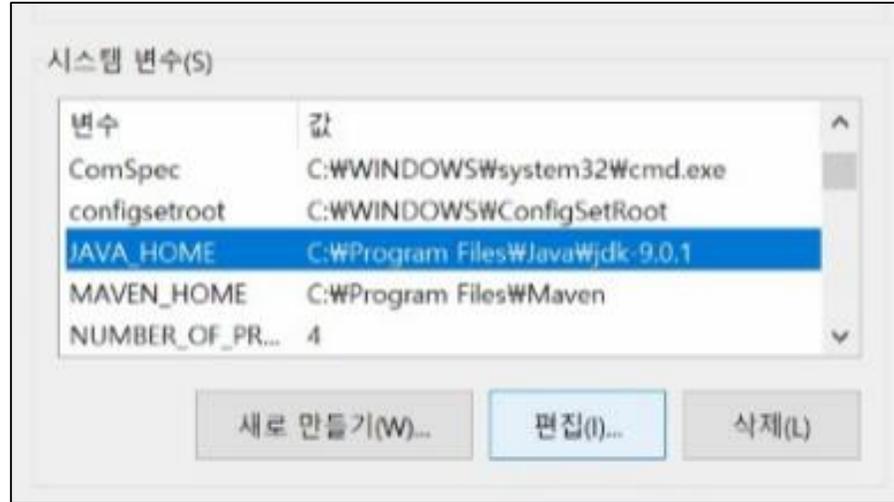
자신의 운영체제에 맞는 파일 다운 & 설치

# 1. IDE

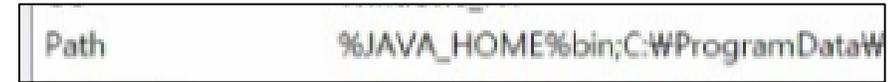
## Java 설치방법



시스템 환경 변수 편집



JAVA\_HOME 변수 설정



Path 변수에  
JAVA\_HOME 추가

## 2. Unit Test



JUnit

Java 기반의 Unit Test를 위한 Framework

Test Case를 생성하여  
Method와 같은 단위 모듈이  
정확하게 구현되었는지 검사

## 2. Unit Test



테스트 주도 개발(TDD)에서  
많이 사용하는 Framework

Method가 Public으로 선언되어야 한다.

Method에 `@Test Annotation`을 붙여준다.

Source Code와 다른 폴더에 저장해서 구분한다.

`Assert`를 사용하여 개발자가 예상한 값과

정확히 일치하면 Success 그렇지 않으면 Fail로 표시

## 2. Unit Test



### 주요 Annotation

@BeforeClass : @Test Method 실행 전에 초기화와 자원 할당 작업 수행

@Before :  
@Test Method 실행 전에 실행되어야 하는 method 정의

@Test :  
Unit Test 대상 Method 정의

@After :  
@Test Method 실행 후에 실행되어야 하는 Method 정의

@AfterClass : @Test Method 실행 후에 초기화와 자원 정리 작업 수행

### 주요 API

`assertArrayEquals(a, b)`: 배열 a와 b가 일치하는지 확인

`assertSame(a, b)`: 객체 a와 b가 같은 객체인지 확인

`assertTrue(a)`: a가 참인지 확인

`assertNotNull(a)`: a 객체가 null이 아님을 확인

## 2. Unit Test

```
1 package com.cal;
2
3 public class Calculator {
4
5     public int add(int a, int b){
6         return a + b;
7     }
8
9     public int min(int a, int b){
10        return a - 2*b ;
11    }
12
13    public int mul(int a, int b){
14        return a * b;
15    }
16
17    public int div(int a, int b){
18        return a / b;
19    }
20 }
```

예제코드

## 2. Unit Test

```
4   import com.cal.Calculator;
5   import static org.junit.Assert.assertEquals;
6
7   public class CalTest {
8
9       @Test
10      public void testAdd(){
11          Calculator cal = new Calculator();
12          assertEquals( expected: 10, cal.add( a: 7, b: 3));
13      }
14
15  }
16
```

CalTest

1 test passed - 2ms

```
"C:\Program Files (x86)\Java\jdk1.8.0_151\bin\java" ...
Process finished with exit code 0
```

테스트결과 - 성공

## 2. Unit Test

```
7 public class CalTest {
8
9     @Test
10    public void testMin(){
11        Calculator cal = new Calculator();
12        assertEquals( expected: 4, cal.min( a: 7, b: 3));
13    }
14
15 }
```

>> 1 test failed - 10ms

```
ns
ns java.lang.AssertionError:
Expected :4
Actual   :1
<Click to see difference>
+ <1 internal call>
+ at org.junit.Assert.failNotEquals(Assert.java:834) <2 internal calls>
+ at com.cal.test.CalTest.testMin(CalTest.java:12) <22 internal calls>
```

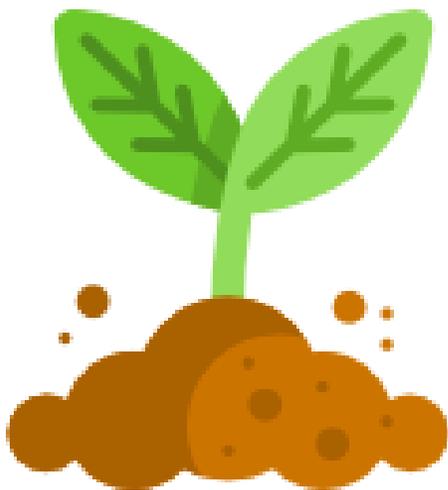
테스트결과 - 실패

Source Code File

↓ Build

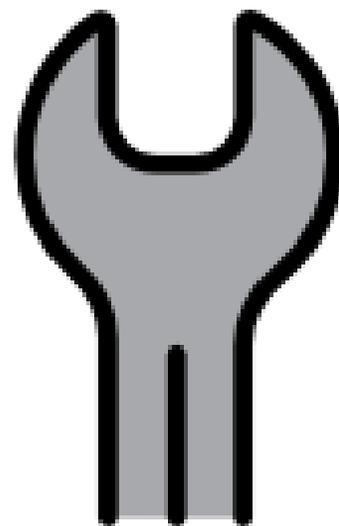
Executable File !

## 관련용어



### Build Environment

: 빌드를 진행하기 위해 구성한 환경



### Build Tool

: Source Code에 대한  
다양한 작업을 지원하는 툴.

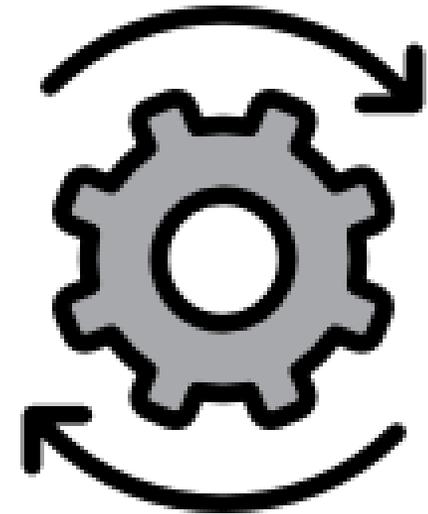


### Build Automation

: 반복적으로 수행하는  
Build를 자동화 시키는 행위.

# Build Automation

- A. Source Code → Binary Code
- B. Packaging
- C. Test
- D. Deploy to Operating System
- E. Create Document



# Build Automation

→ Gradle로 간다!



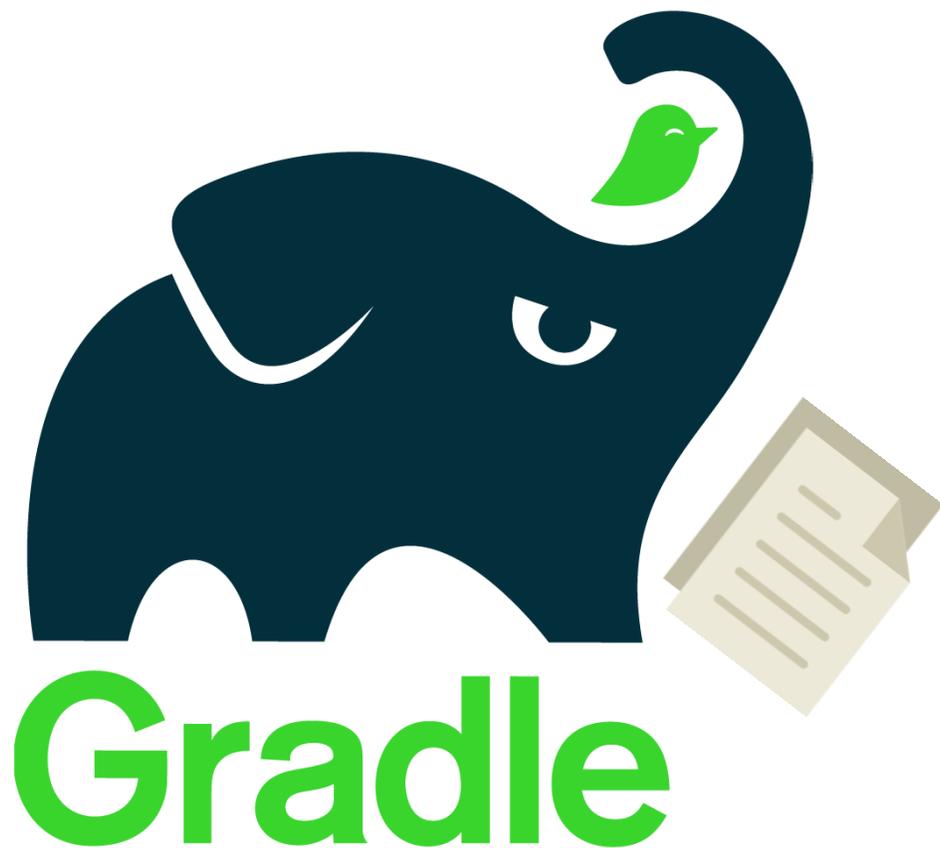
## Gradle의 장점



간결함

Groovy를 사용함  
XML을 사용하지 않음.  
로직 구현이 가능!

## Gradle의 장점



문서화

공식 홈페이지 문서화  
굉장히 잘 되어있음!



## Gradle의 장점

하나의 repository내에  
여러 개의 하위 프로젝트 구성 가능.

상위 프로젝트의 의존성 및 설정을  
하위 프로젝트에서 상속받아 사용가능



# Gradle

유연성 + 확장성

## Gradle의 장점

Groovy 기반 스크립팅을 통해 다양한 기능을 스크립트안에 직접 구현할 수 있음.

직접 Task 구현 및 플러그인 제작가능



## Gradle의 장점

다양한 플러그인 사용 가능!

Ex) checkstyle, pmd, findBugs,  
Sonar, Lint 등..



유연성

## Gradle의 장점

여러가지 언어들에 대한 Build Environment를 제공.

Google은 Android용 공식 빌드 도구로 **Gradle**을 채택!

## Gradle의 장점



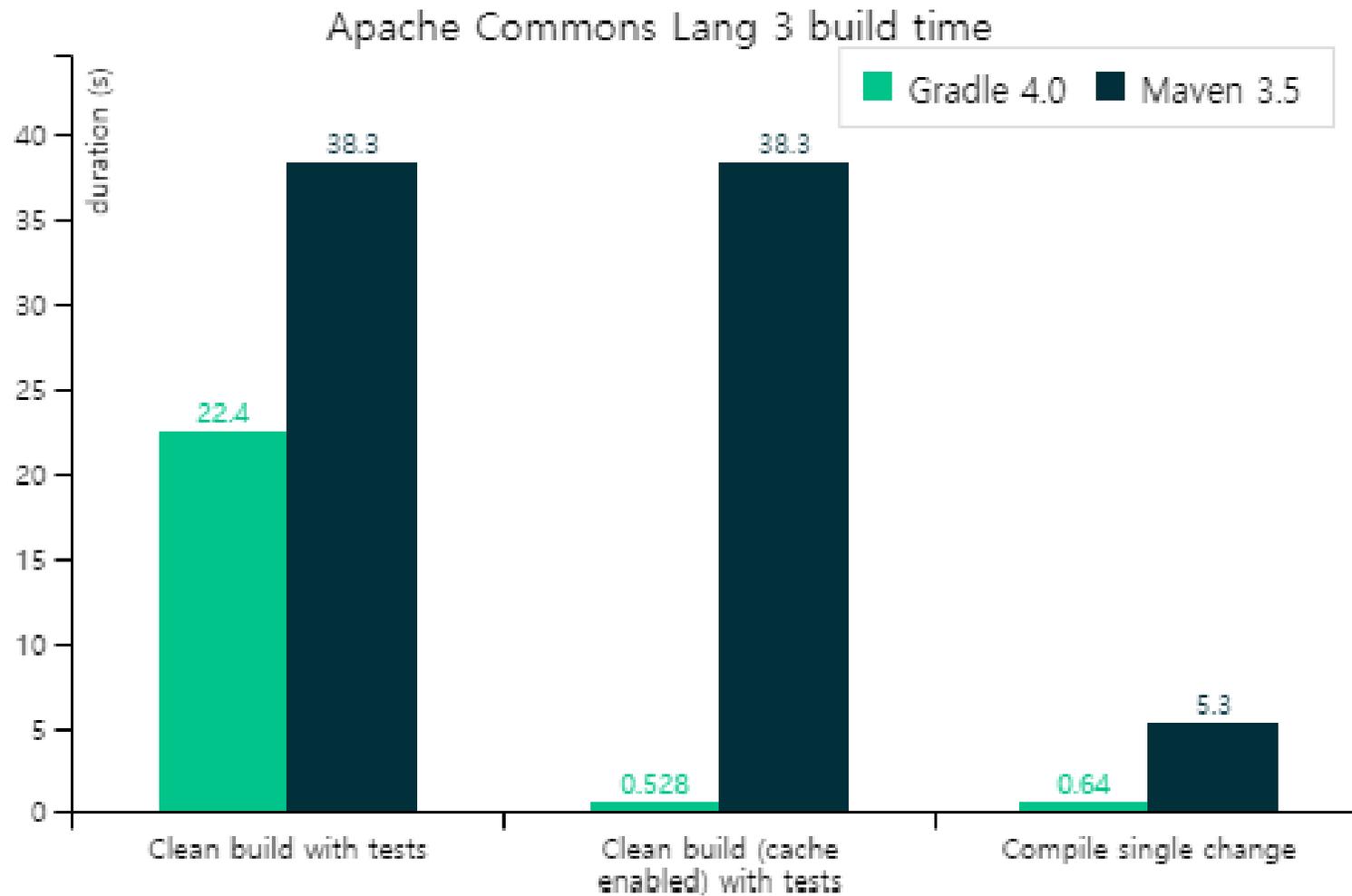
# Gradle

속도

빌드 시스템에서  
빌드 속도 → 개발 생산성

Gradle은 성능 향상을 위한  
다양한 기능들을 지원함.

# 3. Build Conf. & CI



VS

***maven***

# 성능

빌드 시간 향상 → 개발 생산성 증가

# 성능

## Incrementality

: Gradle은 작업의 입력 및 출력을 추적하고 필요한 작업만 실행함.  
가능한 경우 변경된 파일만 처리하여 작업을 방지.

# 성능

## Build Cache

: 빌드의 결과를 Cache에 저장하여 재활용할 수 있다.

# 성능

## Gradle Daemon

: 빌드 정보가 메모리에 최신으로 유지됨.

성능

Build time performance test result  
(normal)

**Gradle Daemon**

**Gradle :  $x$  sec.**

: 빌드 정보가 메모리에 최신으로 유지됨.

**Maven :  $x * 2$  sec.**

성능

Build time performance test result  
(Memcached based)

**Gradle Daemon**

**Gradle :  $x$  sec.**

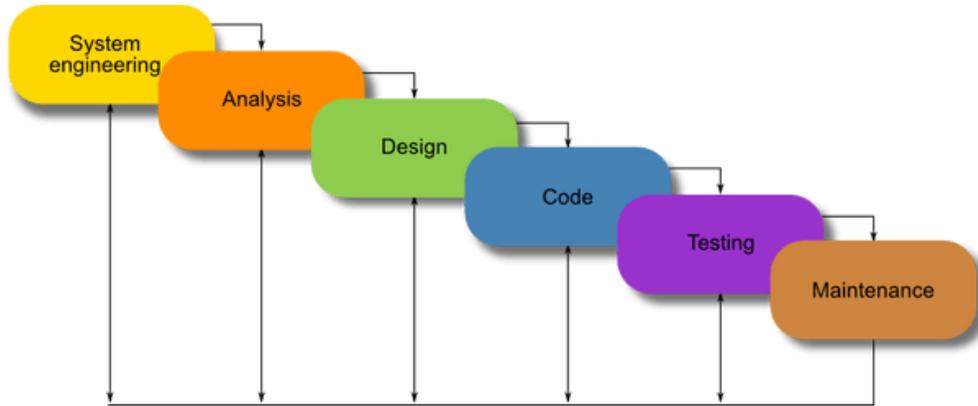
: 빌드 정보가 메모리에 최신으로 유지됨.

**Maven :  $(x * 100) + a$  sec.**

# User Experience

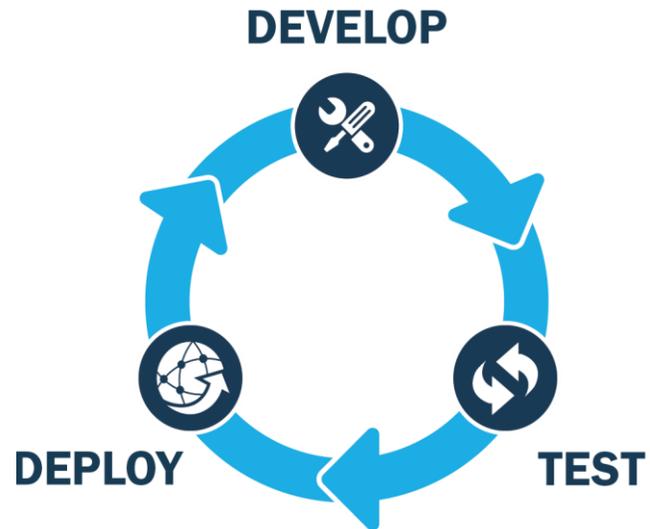
- 향상된 IDE 플러그인  
: Gradle팀과 IDE제작팀과의 Collaboration!
- 최신 CLI 기능들 제공.
- 빌드 디버깅 및 최적화를 위한 Build Scan툴 제공.

### 3. Build Conf. & CI



#### 기존 방식

모든 개발 완료 후 QC 수행.



#### CI(지속적 통합)

자동화를 통해 릴리스 가능한 소프트웨어를  
짧은 기간 반복하여 생산

### 3. Build Conf. & CI

지속적으로 테스트를 실행하고  
테스트가 품질을 보증한다고 신뢰할 수 있다면  
언제, 어디서든지 소프트웨어를 릴리스할 수 있다.



### 3. Build Conf. & CI



Jenkins??

개발 작업을 지원하기 위한 약 1,400가지의 플러그인을 가지고 있는 오픈소스 자동화 서버

## 3. Build Conf. & CI

빌드 자동화

빌드 파이프라이닝

자동화 테스트

정적 코드 분석

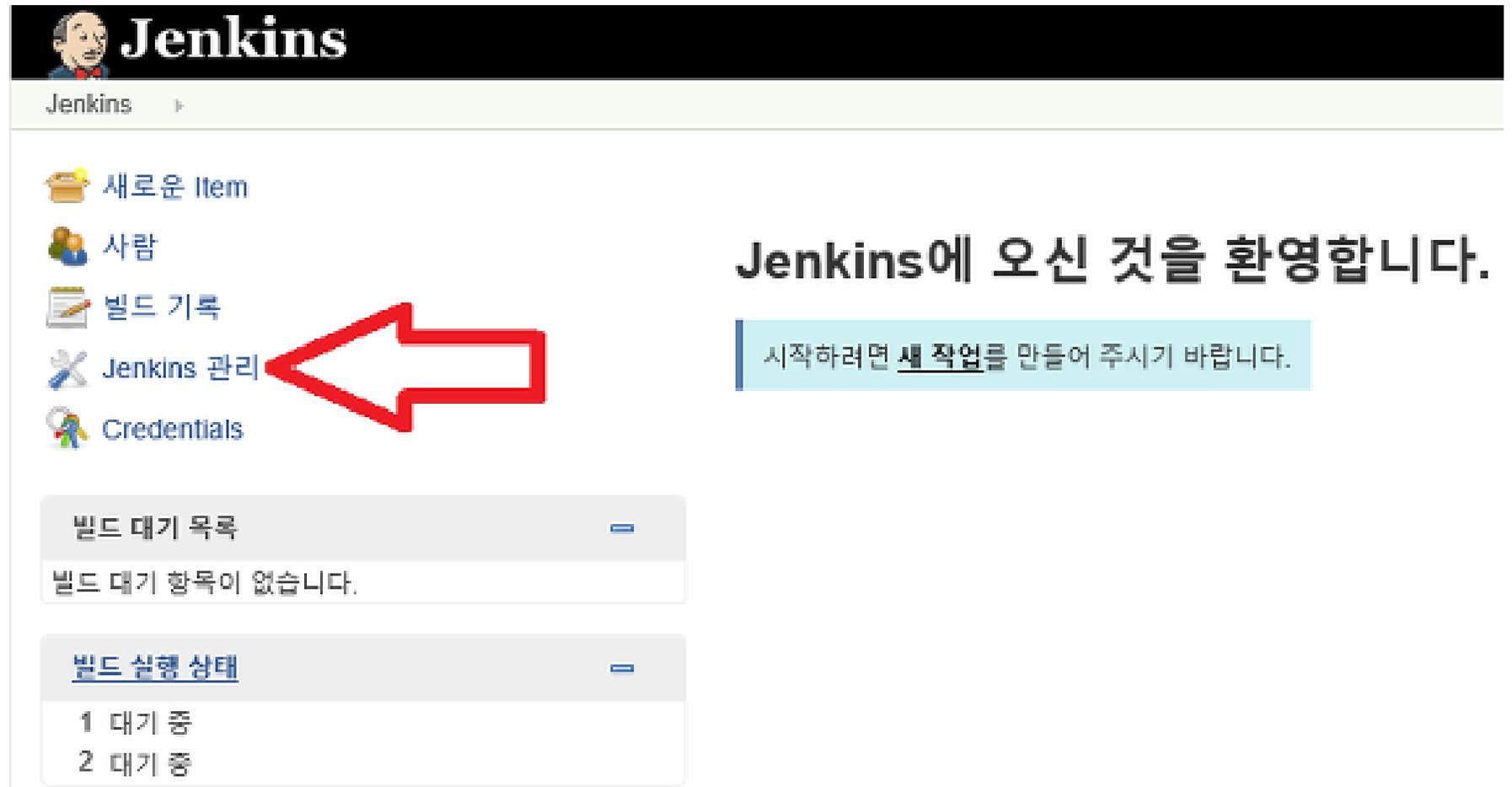
배포 자동화

주요 기능

# 3. Build Conf. & CI

## 빌드 자동화

### 1. Gradle 연동



**Jenkins**

Jenkins ▾

- 새로운 Item
- 사람
- 빌드 기록
- Jenkins 관리 ←
- Credentials

빌드 대기 목록 =

빌드 대기 항목이 없습니다.

빌드 실행 상태 =

- 1 대기 중
- 2 대기 중

**Jenkins에 오신 것을 환영합니다.**

시작하려면 새 작업을 만들어 주시기 바랍니다.

# 3. Build Conf. & CI

## 빌드 자동화

### 1. Gradle 연동

#### Jenkins 관리



##### 시스템 설정

환경변수 및 경로 정보등을 설정합니다.



##### Configure Global Security

Secure Jenkins; define who is allowed to access/use the system.



##### Configure Credentials

Configure the credential providers and types



##### Global Tool Configuration

Configure tools, their locations and automatic installers.



##### Reload Configuration from Disk

Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.



##### 플러그인 관리

Jenkins의 기능을 확장하기 위한 플러그인을 추가, 제거, 사용, 미사용으로 설정할 수 있습니다. **(업데이트 가능함)**



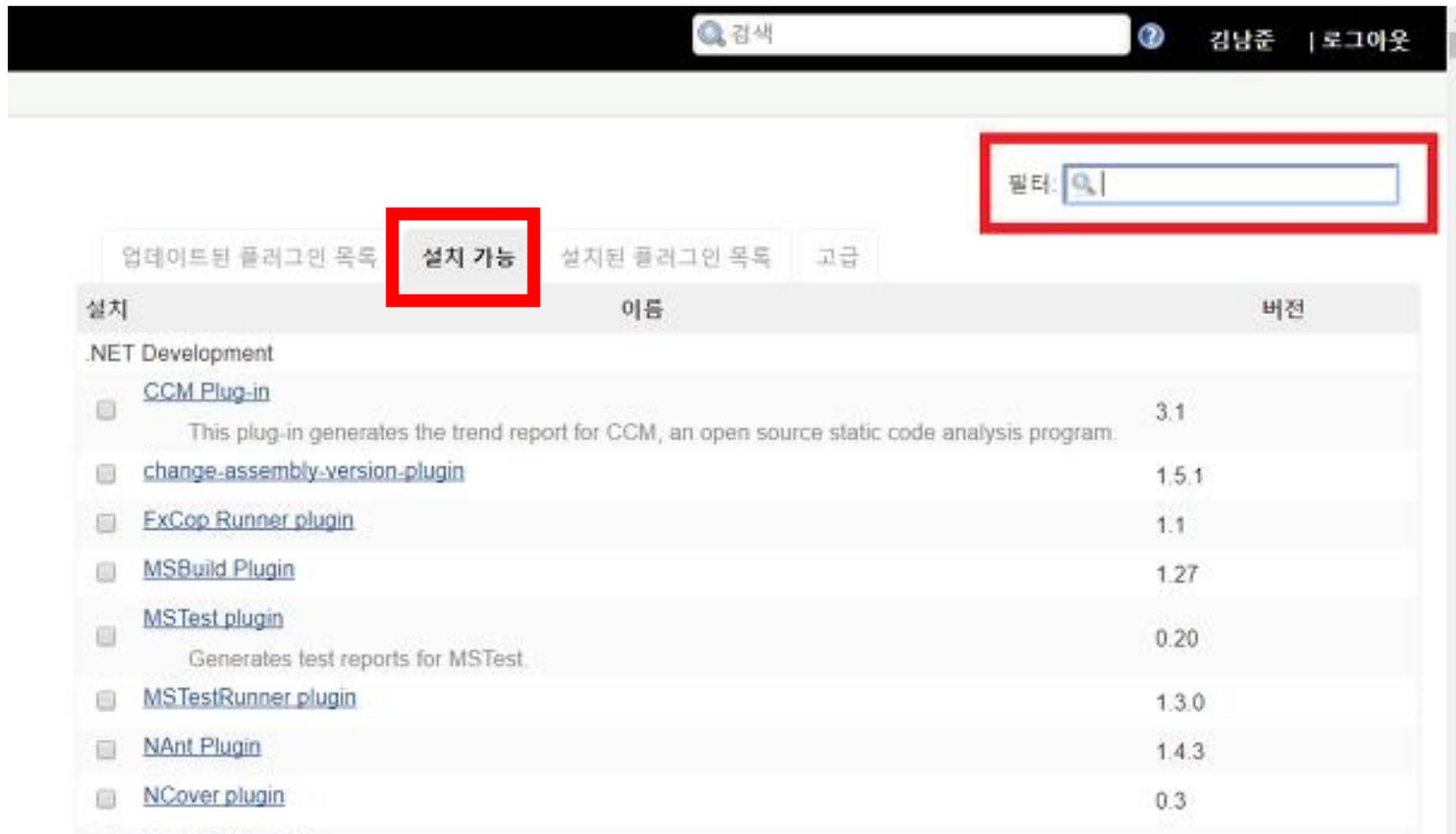
##### 시스템 정보

문제 해결을 돕기위한 다양한 환경 정보를 보여줍니다.

# 3. Build Conf. & CI

## 빌드 자동화

### 1. Gradle 연동



# 3. Build Conf. & CI

## 빌드 자동화

## 2. VCS/빌드 설정



# 3. Build Conf. & CI

## 빌드 자동화

### 2. VCS/빌드 설정

Enter an item name

\* This field cannot be empty, please enter a valid name

-  **Freestyle project**  
이것은 Jenkins의 주요 기능입니다. Jenkins은 어느 빌드 시스템과 어떤 SCM(항상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.
-  **Pipeline**  
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
-  **External Job**  
이 유형의 작업은 원격 장비처럼 Jenkins 외부에서 동작하는 프로세스의 실행을 기록하는 것을 허용합니다. 그렇게 설계되어서, 기존의 자동 시스템의 대시보드로서 Jenkins을 사용할 수 있습니다.
-  **Multi-configuration project**  
다양한 환경에서의 테스트, 플러폼 특성 빌드, 기타 동등 처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.
-  **Folder**  
Contains a container that stores nested items in it. Useful for organizing things together. Unlike view, which is just a filter, a folder creates a separate

# 3. Build Conf. & CI

## 빌드 자동화

## 2. VCS/빌드 설정

The screenshot shows the Jenkins configuration page for Subversion. The page title is "소스 코드 관리" (Source Code Management). The "Subversion" option is selected under "Source Code Management". The "Modules" section is expanded, showing the following fields:

- Repository URL: [Empty text box] with a red error message: "Repository URL is required."
- Credentials: [- none -] with an "Add" button
- Local module directory: [.]
- Repository depth: [infinity]
- Ignore externals: [checked]

Below the modules section, there are buttons for "Add module...", "Add additional credentials...", and a "Check-out Strategy" dropdown menu set to "Use 'svn update' as much as possible". The "Repository browser" is set to "(자동)".

# 3. Build Conf. & CI

## 빌드 자동화

### 3. 빌드 유발

**Build**

**Invoke Gradle script**

- Invoke Gradle
  - Gradle Version:
  - Switches:
  - Tasks:
- Use Gradle Wrapper

Root Build script:

Build File:

Specify Gradle build file to run. Also, [some environment variables are available to the build script](#)

Force GRADLE\_USER\_HOME to use workspace

Pass job parameters as Gradle properties

### 3. Build Conf. & CI

빌드 자동화

### 4. 빌드 후 조치

**빌드 후 조치**

Deploy war/ear to a container

WAR/EAR files:

Context path:

Containers

- Tomcat 7.x
  - Manager user name:
  - Manager password:
  - Tomcat URL:

Add Container

Deploy on failure:

빌드 후 조치 추가

**저장** Apply

1: 2018-03-21

Thanks!

---