

Java기반의 CTIP 환경 설정

소프트웨어 검증

컴퓨터공학부

김미셀, 김은정, 엄준용, 한경미

목차

1. CI & CTIP

2. CTIP 환경설정

3. Hudson Result

목차

1. CI & CTIP

1.1 CI

1.2 CI 사용

1.3 CTIP

1.4 CTIP 구성

2. CTIP 환경설정

3. Hudson Result

CI(Continuous Integration)

- 여러 명으로 구성된 팀이 작업한 것을 통합
- 대규모의 프로젝트 개발에 통합과정의 문제를 해결하기 위해 나온 해결책

CI(Continuous Integration) 사용

1. Source Repository로 부터 최신 소스 다운로드(Check out)
2. 코드를 작성한 후 자신의 개발기에서 정상적으로 동작하는지 충분히 검토
3. 검증이 끝난 후 개발자는 작업 내용을 Source Repository에 업로드(Check in)
4. 개발자(또는 통합 관리자 또는 자동화된 시스템)는 통합 서버에 방금 작업한 코드가 반영된 전체 코드 내용에 대한 빌드 수행
5. 만약 빌드가 실패할 경우, 실패 원인을 분석하고 문제를 해결하여 빌드를 성공시킴.

CTIP (Continuous Test & Integration Platform)

- CI 개념을 Java기반의 개발 프로젝트에 쉽게 적용하기 위한 Platform
- 소스 검토(테스트 및 정적 분석), 빌드, 통합, 배포 및 레포트 기능을 제공하는 개발 지원 Platform

CTIP (Continuous Test & Integration Platform)

- 단일 소스 저장소 관리
- 빌드 자동화
- 자체적으로 테스트 가능한 빌드
- 빠른 빌드 수행
- 운영환경과 유사한 환경 구성
- 최신결과물에 대한 쉬운 접근
- 손쉬운 빌드 상태 모니터링

CTIP 구성

IDE Tool
Eclipse

CI Tool
Hudson

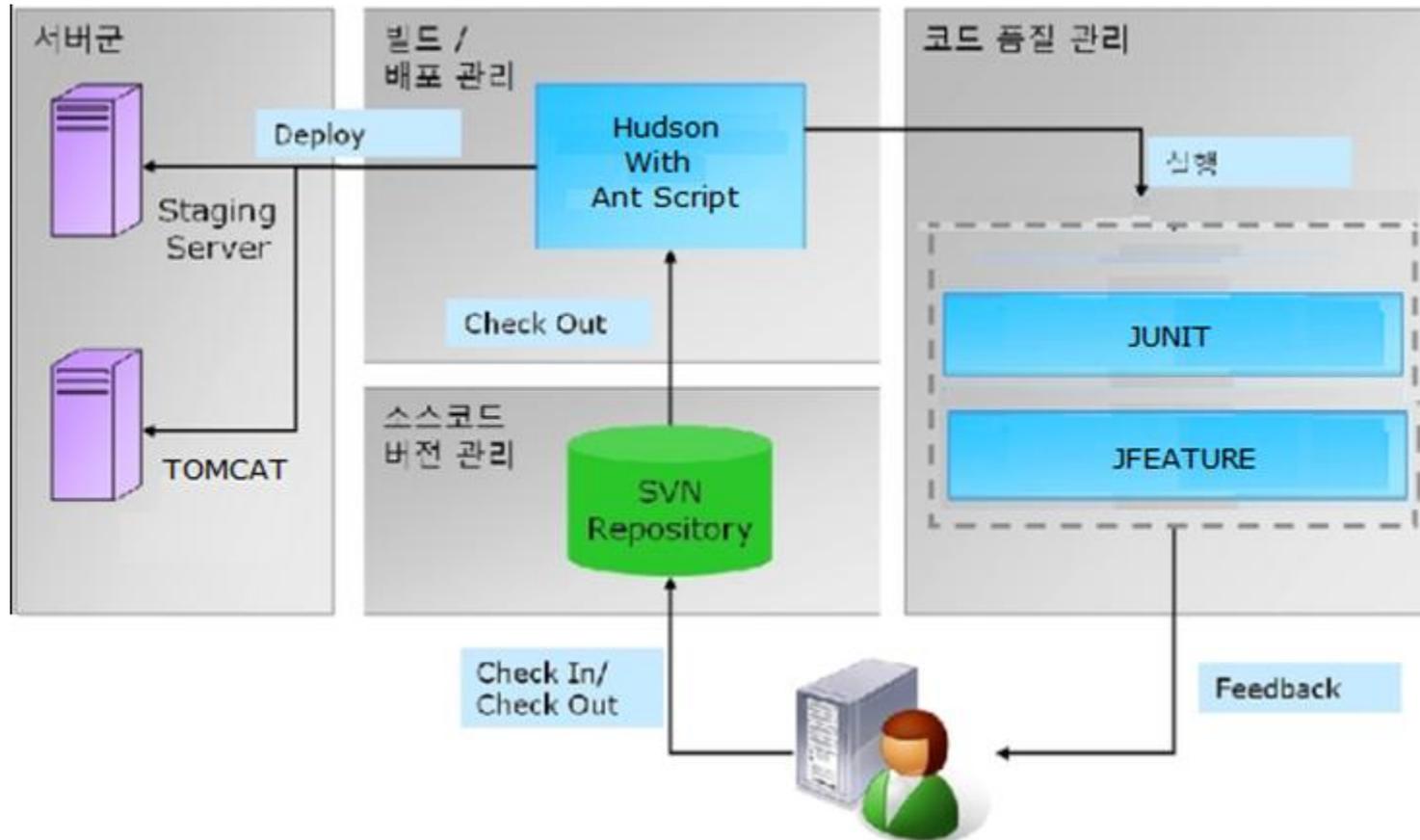
Unit Test
JUnit

WAS
Tomcat 6.0

Build
Tortoise SVN

SCM
Ant

CTIP 구성



목차

1. CI & CTIP

2. CTIP 환경설정

2.1 Tomcat & Hudson 설치

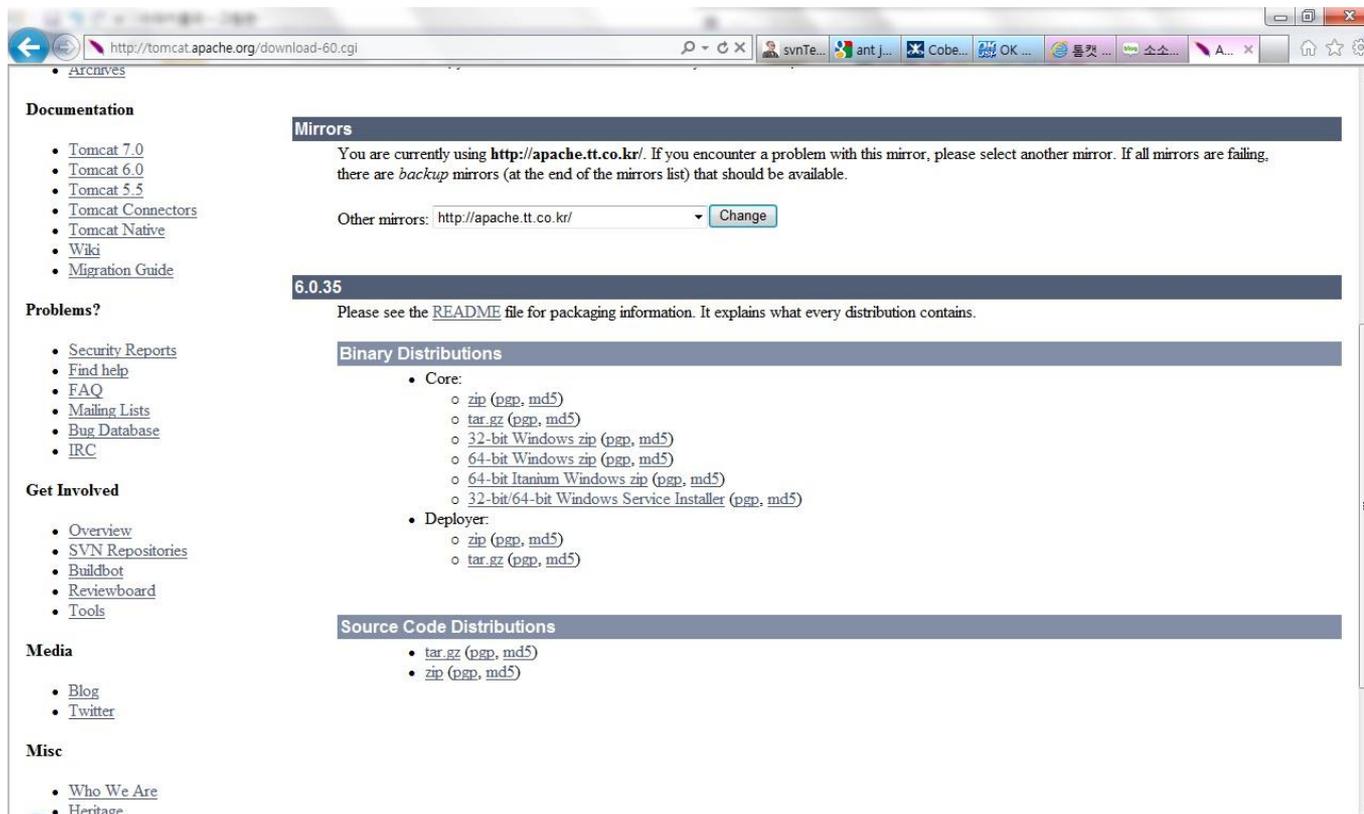
2.2 Ant

2.3 Tortoise SVN

3. Hudson Result

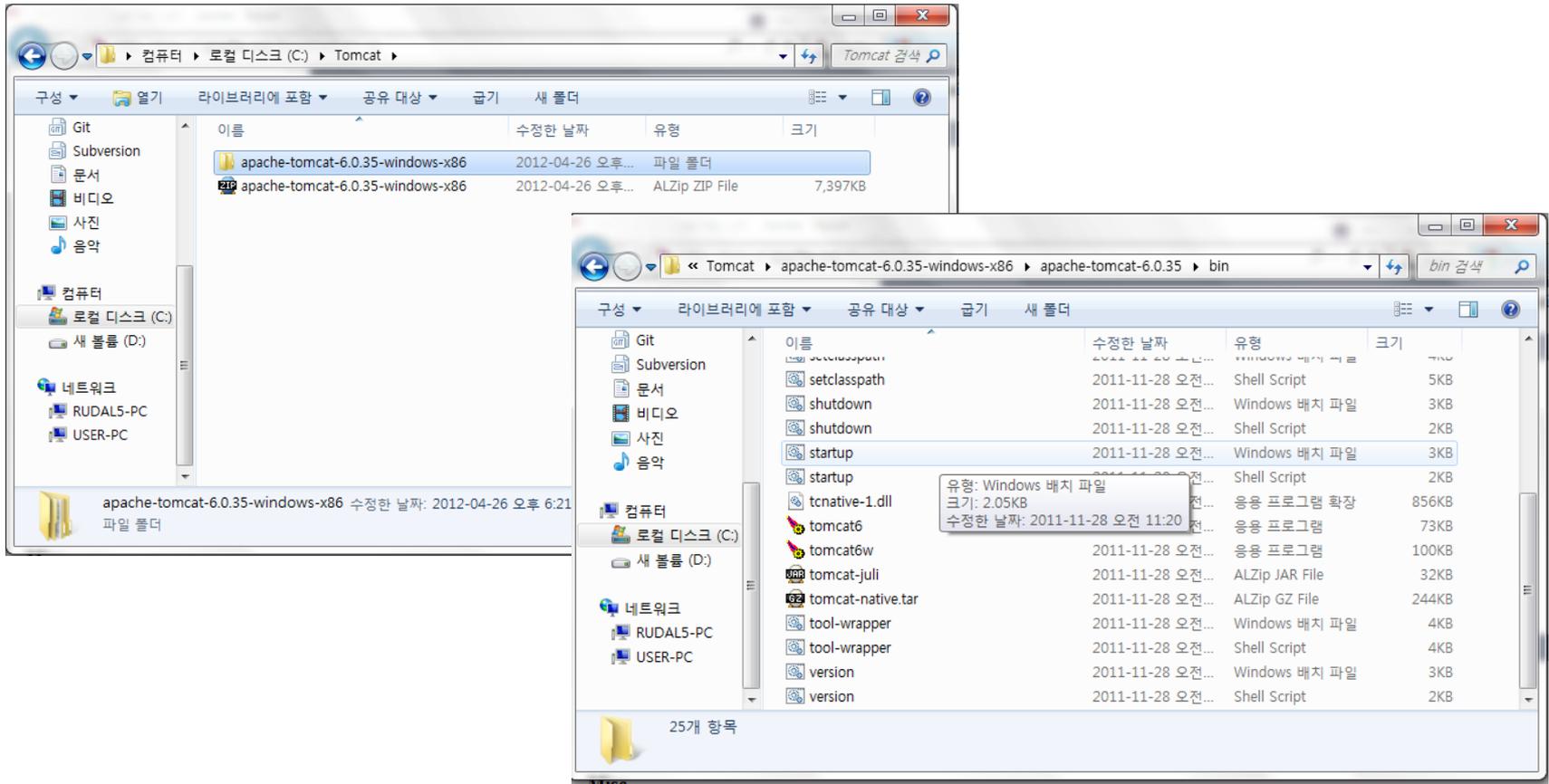
Tomcat 설치

- <http://tomcat.apache.org/> 접속 하여 Tomcat 6.0 download



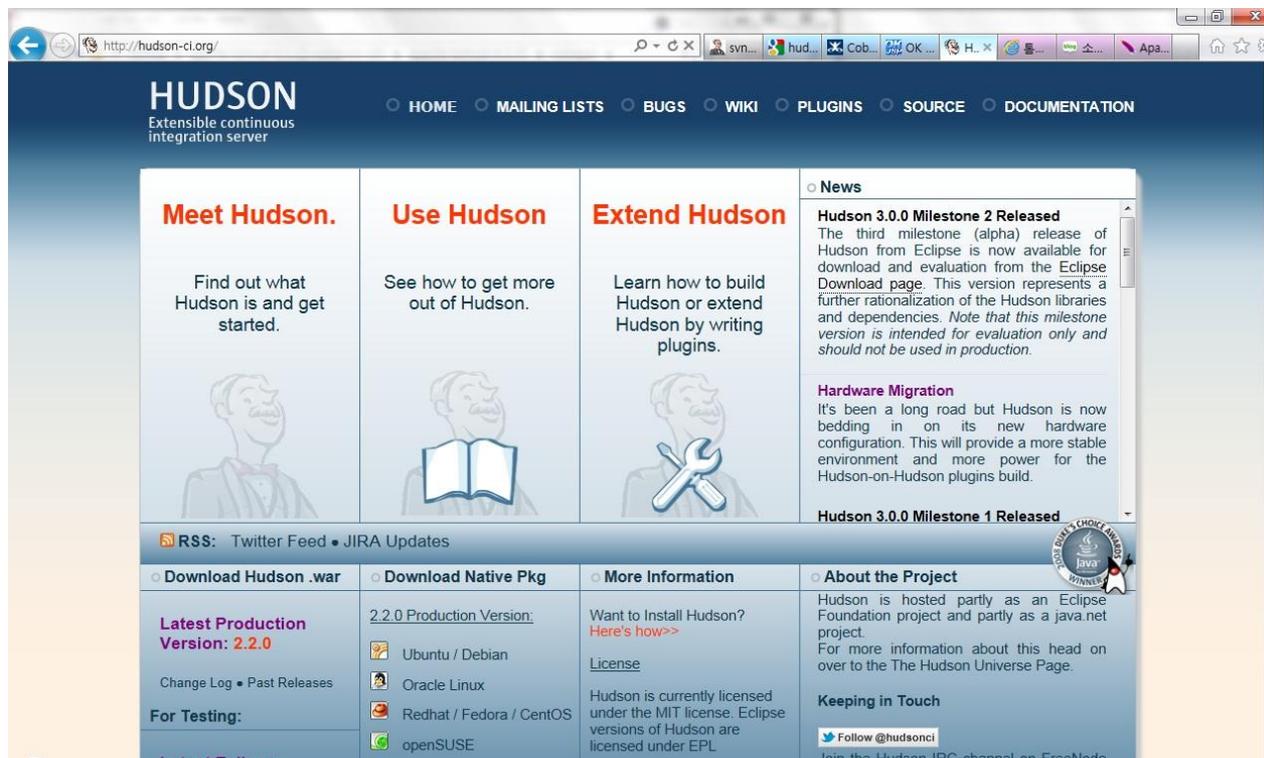
Tomcat 설치

- Tomcat 압축 파일 해제 후 startup 실행



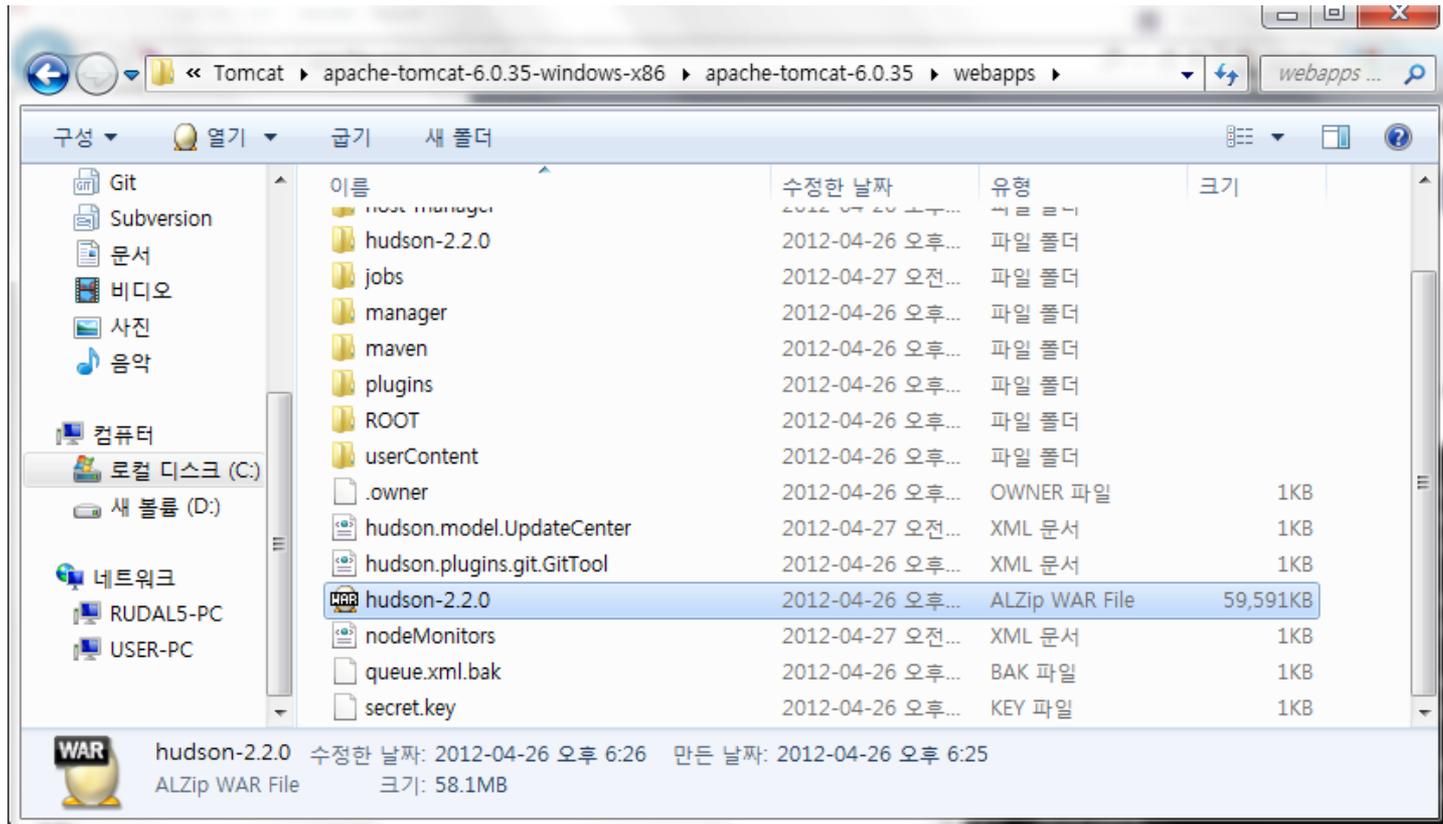
Hudson 설치

- <http://hudson-ci.org/> 접속 하여 Hudson Download



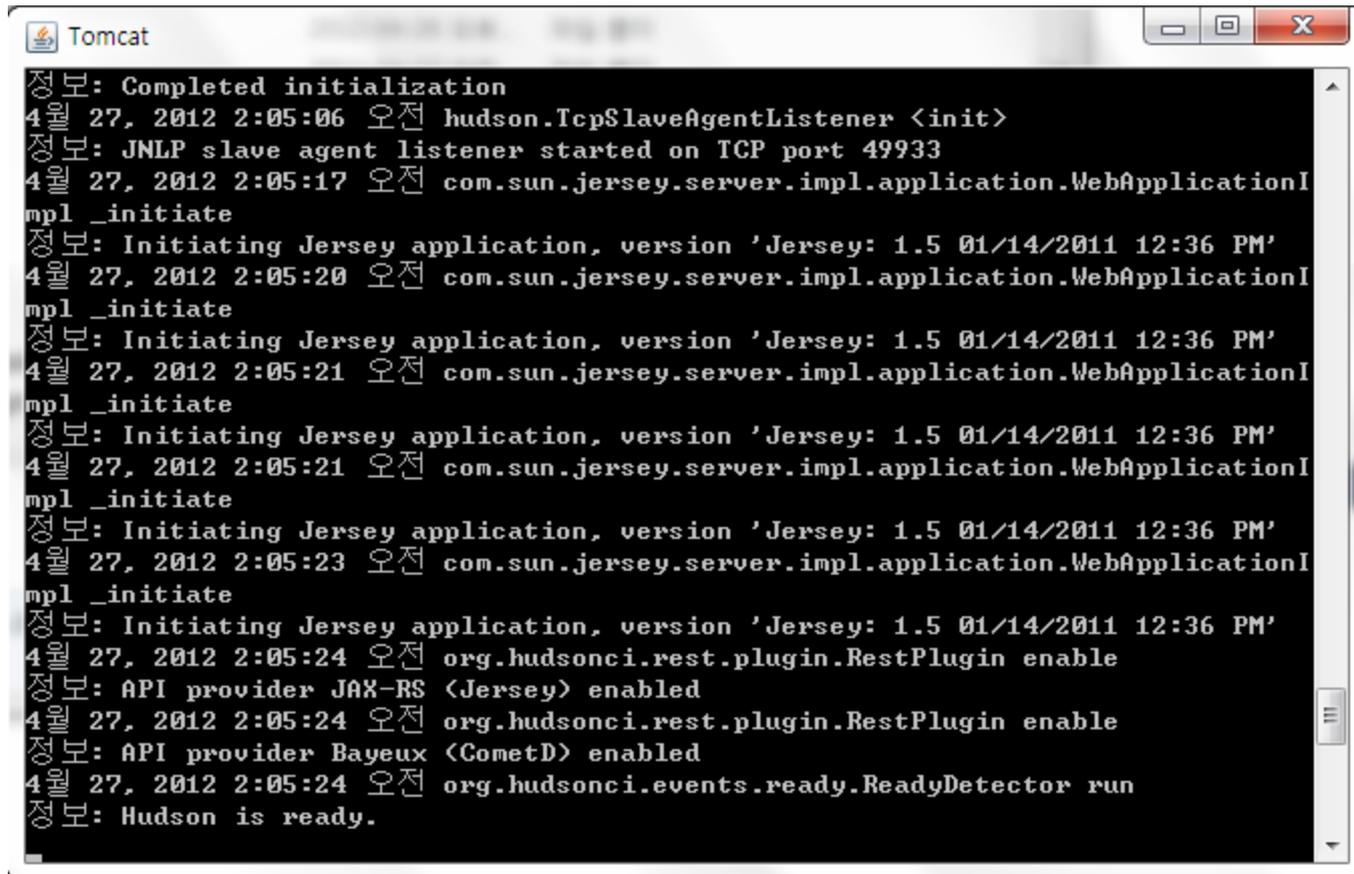
Hudson 설치

- hudson.war 파일을 tomcat 의 webapps 폴더에 저장



Hudson 설치

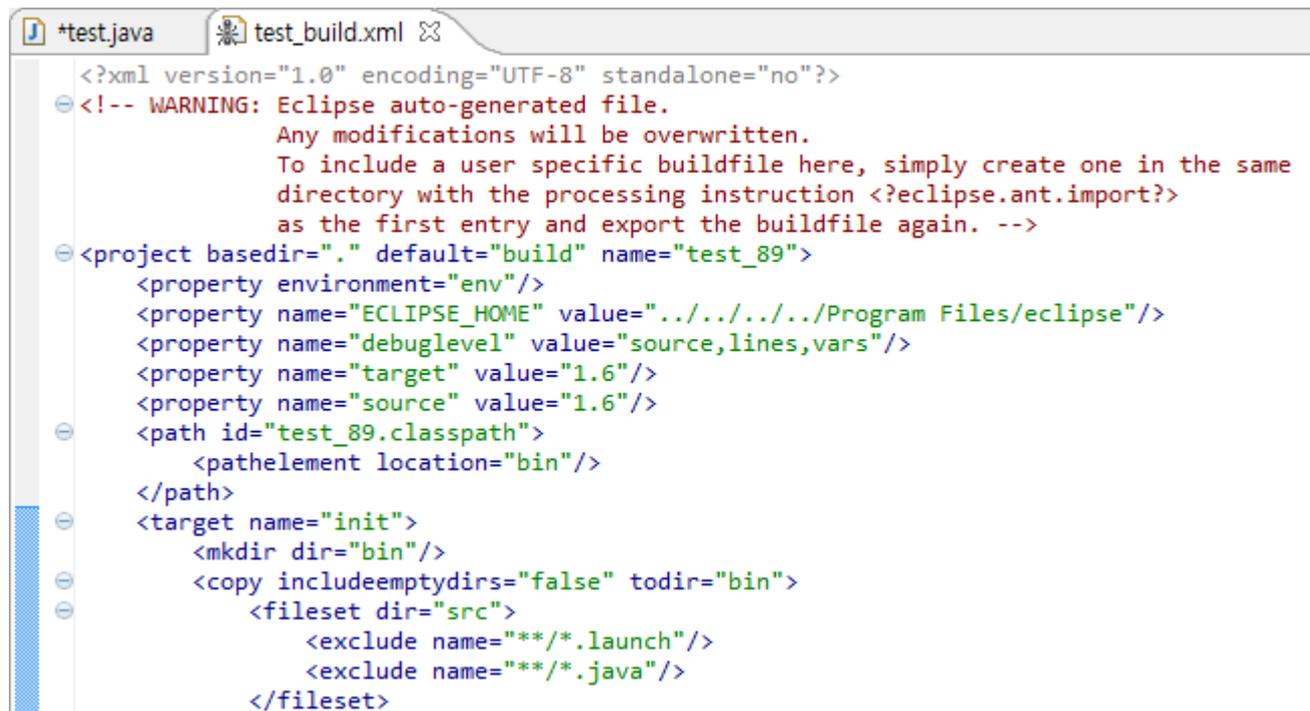
- Tomcat을 통한 Hudson 실행



```
Tomcat
정보: Completed initialization
4월 27, 2012 2:05:06 오전 hudson.TcpSlaveAgentListener <init>
정보: JNLP slave agent listener started on TCP port 49933
4월 27, 2012 2:05:17 오전 com.sun.jersey.server.impl.application.WebApplicationI
mpl_initiate
정보: Initiating Jersey application, version 'Jersey: 1.5 01/14/2011 12:36 PM'
4월 27, 2012 2:05:20 오전 com.sun.jersey.server.impl.application.WebApplicationI
mpl_initiate
정보: Initiating Jersey application, version 'Jersey: 1.5 01/14/2011 12:36 PM'
4월 27, 2012 2:05:21 오전 com.sun.jersey.server.impl.application.WebApplicationI
mpl_initiate
정보: Initiating Jersey application, version 'Jersey: 1.5 01/14/2011 12:36 PM'
4월 27, 2012 2:05:21 오전 com.sun.jersey.server.impl.application.WebApplicationI
mpl_initiate
정보: Initiating Jersey application, version 'Jersey: 1.5 01/14/2011 12:36 PM'
4월 27, 2012 2:05:23 오전 com.sun.jersey.server.impl.application.WebApplicationI
mpl_initiate
정보: Initiating Jersey application, version 'Jersey: 1.5 01/14/2011 12:36 PM'
4월 27, 2012 2:05:24 오전 org.hudsonci.rest.plugin.RestPlugin enable
정보: API provider JAX-RS <Jersey> enabled
4월 27, 2012 2:05:24 오전 org.hudsonci.rest.plugin.RestPlugin enable
정보: API provider Bayeux <CometD> enabled
4월 27, 2012 2:05:24 오전 org.hudsonci.events.ready.ReadyDetector run
정보: Hudson is ready.
```

Ant

- 자동 생성한 build.xml



```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!-- WARNING: Eclipse auto-generated file.
Any modifications will be overwritten.
To include a user specific buildfile here, simply create one in the same
directory with the processing instruction <?eclipse.ant.import?>
as the first entry and export the buildfile again. -->
<project basedir="." default="build" name="test_89">
  <property environment="env"/>
  <property name="ECLIPSE_HOME" value="../../../../Program Files/eclipse"/>
  <property name="debuglevel" value="source,lines,vars"/>
  <property name="target" value="1.6"/>
  <property name="source" value="1.6"/>
  <path id="test_89.classpath">
    <pathelement location="bin"/>
  </path>
  <target name="init">
    <mkdir dir="bin"/>
    <copy includeemptydirs="false" todir="bin">
      <fileset dir="src">
        <exclude name="**/*.launch"/>
        <exclude name="**/*.java"/>
      </fileset>
    </copy>
  </target>
</project>
```

Ant

- 자동 생성한 build.xml

A screenshot of an IDE window showing the content of a file named 'test_build.xml'. The window has two tabs: '*test.java' and 'test_build.xml'. The code is XML-based and defines several Ant targets. The 'clean' target deletes the 'bin' directory. The 'cleanall' target depends on 'clean'. The 'build' target depends on 'build-subprojects' and 'build-project'. The 'build-subprojects' target depends on 'init'. The 'build-project' target echoes a message, runs 'javac' with debug options, and sets the classpath. The 'build-refprojects' target has a description. The 'init-eclipse-compiler' target copies Eclipse compiler jars to the ant lib directory and unzips them, including a patternset for 'jdtCompilerAdapter.jar'.

```
</copy>
</target>
<target name="clean">
  <delete dir="bin"/>
</target>
<target depends="clean" name="cleanall"/>
<target depends="build-subprojects,build-project" name="build"/>
<target name="build-subprojects"/>
<target depends="init" name="build-project">
  <echo message="${ant.project.name}: ${ant.file}"/>
  <javac debug="true" debuglevel="${debuglevel}" destdir="bin" source="${source}" target="${target}">
    <src path="src"/>
    <classpath refid="test_89.classpath"/>
  </javac>
</target>
<target description="Build all projects which reference this project. Useful to propagate changes." name="build-refprojects"/>
<target description="copy Eclipse compiler jars to ant lib directory" name="init-eclipse-compiler">
  <copy todir="${ant.library.dir}">
    <fileset dir="${ECLIPSE_HOME}/plugins" includes="org.eclipse.jdt.core_*.jar"/>
  </copy>
  <unzip dest="${ant.library.dir}">
    <patternset includes="jdtCompilerAdapter.jar"/>
    <fileset dir="${ECLIPSE_HOME}/plugins" includes="org.eclipse.jdt.core_*.jar"/>
  </unzip>
</target>
```

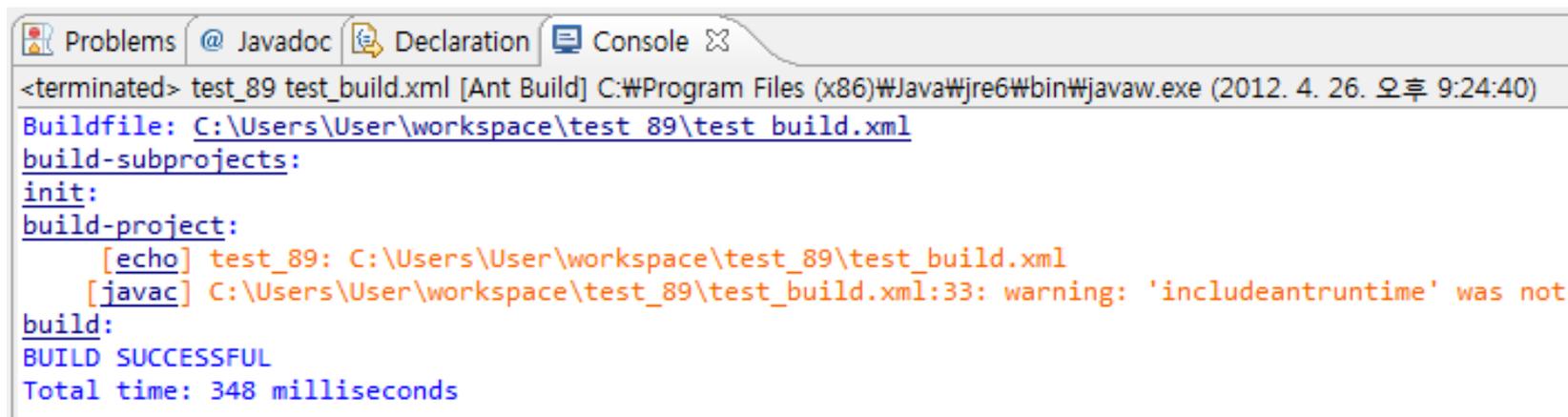
Ant

- 자동 생성한 build.xml

```
<target description="compile project with Eclipse compiler" name="build-eclipse-compiler">
  <property name="build.compiler" value="org.eclipse.jdt.core.JDTCompilerAdapter"/>
  <antcall target="build"/>
</target>
<target name="test (1)">
  <java classname="test" failonerror="true" fork="yes">
    <classpath refid="test_89.classpath"/>
  </java>
</target>
</project>
```

Ant

- build.xml 파일의 Build 결과



```
<terminated> test_89 test_build.xml [Ant Build] C:#Program Files (x86)#Java#jre6#bin#javaw.exe (2012. 4. 26. 오후 9:24:40)
Buildfile: C:\Users\User\workspace\test_89\test_build.xml
build-subprojects:
init:
build-project:
    [echo] test_89: C:\Users\User\workspace\test_89\test_build.xml
    [javac] C:\Users\User\workspace\test_89\test_build.xml:33: warning: 'includeantruntime' was not
build:
BUILD SUCCESSFUL
Total time: 348 milliseconds
```

Ant

- New works

작업명

- Build a free-style software project**
이것은 Hudson의 주요 기능입니다. Hudson은 어느 빌드 시스템과 어떤 SCM(형상관리)으로 묶인 당신의 프로젝트를 빌드할 것이고, 소프트웨어 빌드보다 다른 어떤 것에 자주 사용될 수 있습니다.
- Build a Maven 2/3 project (Legacy)**
Maven 2/3 (Legacy) 프로젝트를 빌드합니다. Hudson은 POM 파일의 이점을 가지고 있고 급격히 설정을 줄입니다.
- Monitor an external job**
이 유형의 작업은 원격 장비처럼 Hudson 외부에서 동작하는 프로세스의 실행을 기록하는 것을 허용합니다. 그렇게 설계되어서, 기존의 자동 시스템의 대시보드로서 Hudson을 사용할 수 있습니다. 자세한 설명은 [여기\(영문\)](#)을 보세요.
- Build multi-configuration project**
다양한 환경에서의 테스트, 플랫폼 특성 빌드, 기타 등등처럼 다수의 서로다른 환경설정이 필요한 프로젝트에 적합함.
- 기존 작업 복사**
Copy from

Ant

- Invoke Ant

Build

Invoke Ant

Targets

Build File

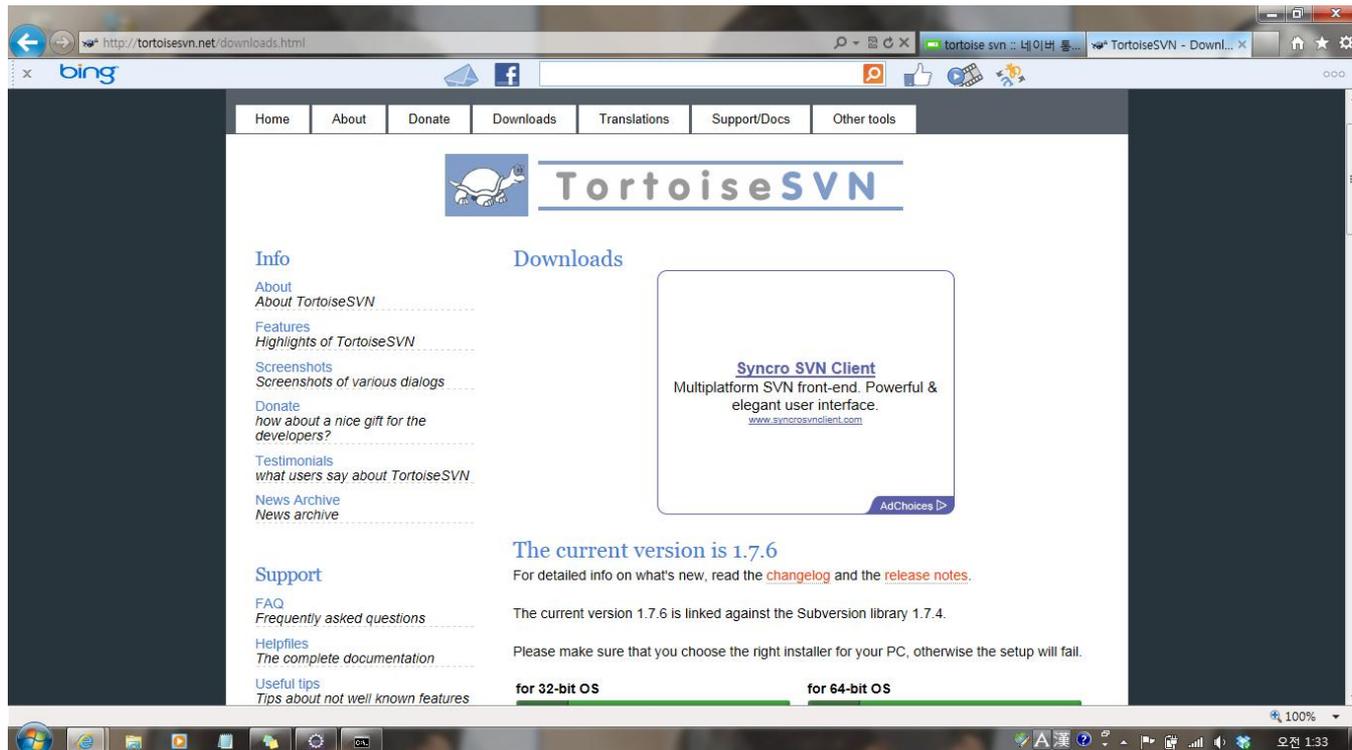
Properties

Java Options

Delete

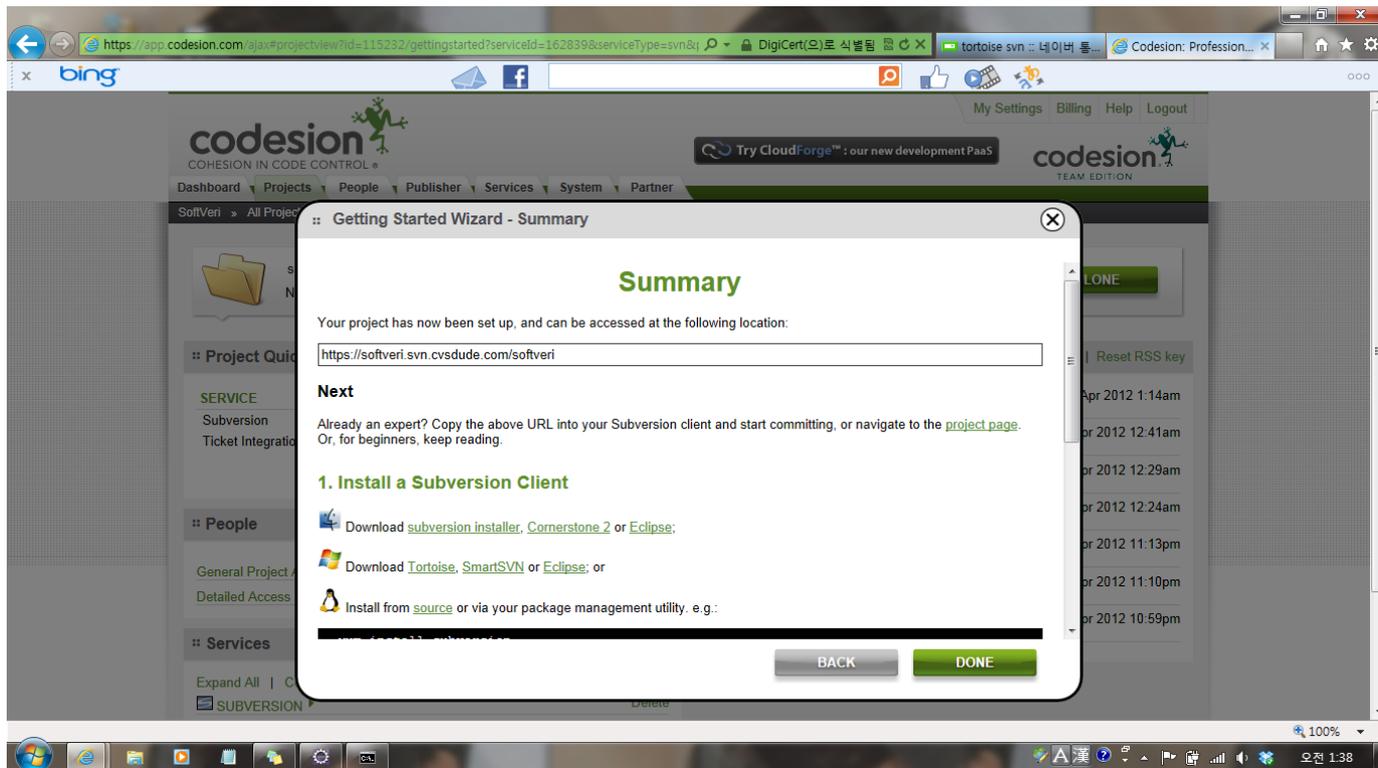
Tortoise SVN

- Tortoise SVN 홈페이지로 접속 후 최신버전 Download



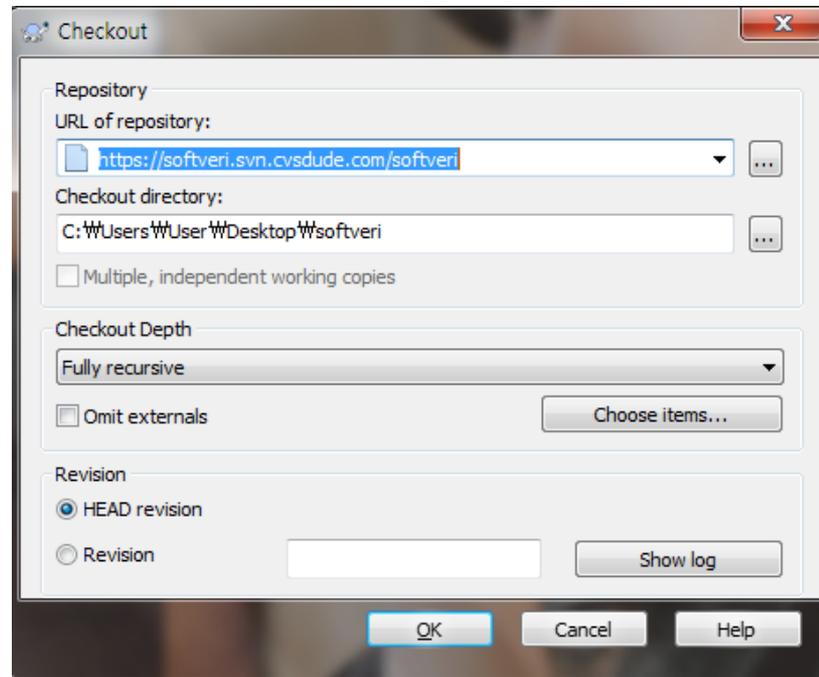
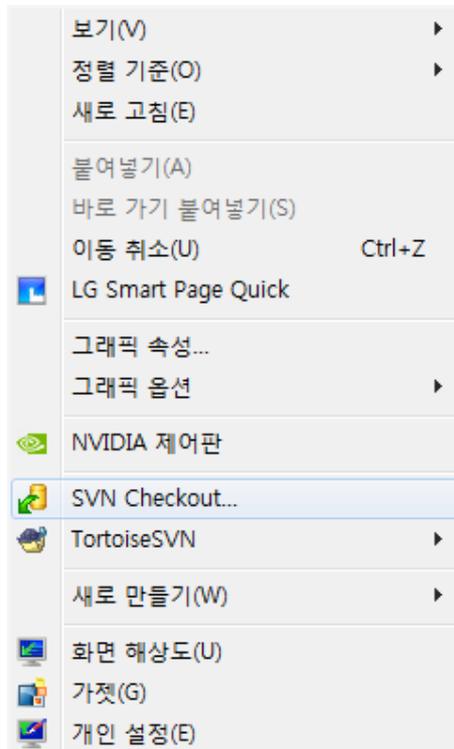
Tortoise SVN

- Codesion.com 사이트를 통해 SVN 저장소 생성



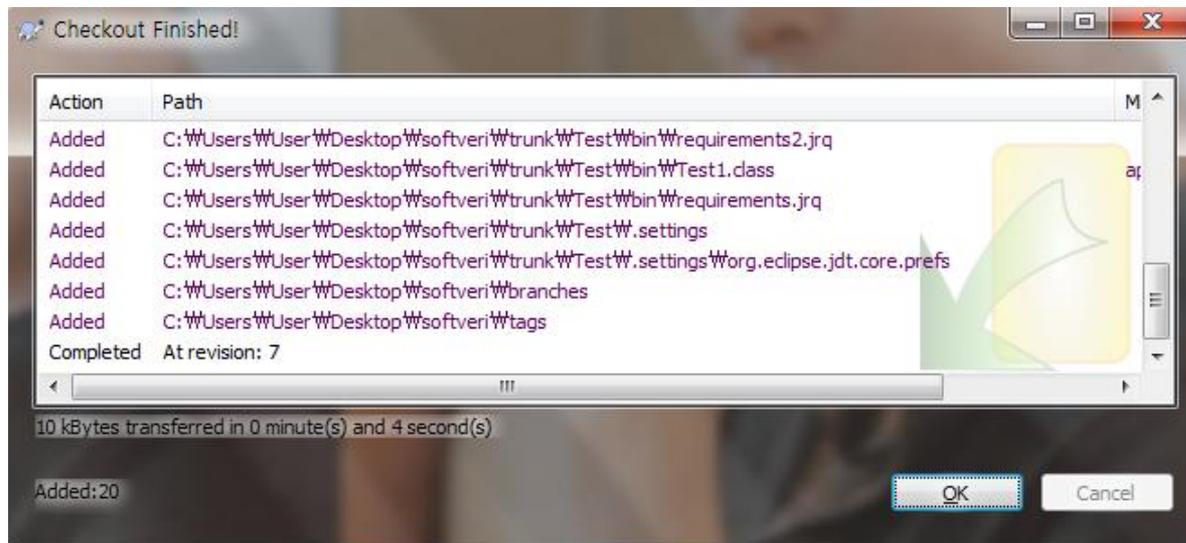
Tortoise SVN

- SVN Checkout을 통해 내컴퓨터에 SVN폴더 생성(URL에는 SVN 저장소의 주소 입력)



Tortoise SVN

- Checkout이 완료 된 후, 저장되었던 파일 update



Tortoise SVN

- SVN URL에 SVN 저장소의 주소를 등록

Source Code Management

- None
 CVS
 Git
 Subversion

Modules

Repository URL	<input type="text" value="https://softveri.svn.cvsdude.com/softveri"/> Update credentials	
Local module directory (optional)	<input type="text" value="."/>	
Repository depth option	<input type="text" value="infinity"/>	
Ignore externals option	<input type="checkbox"/>	

Add more locations...

Check-out Strategy

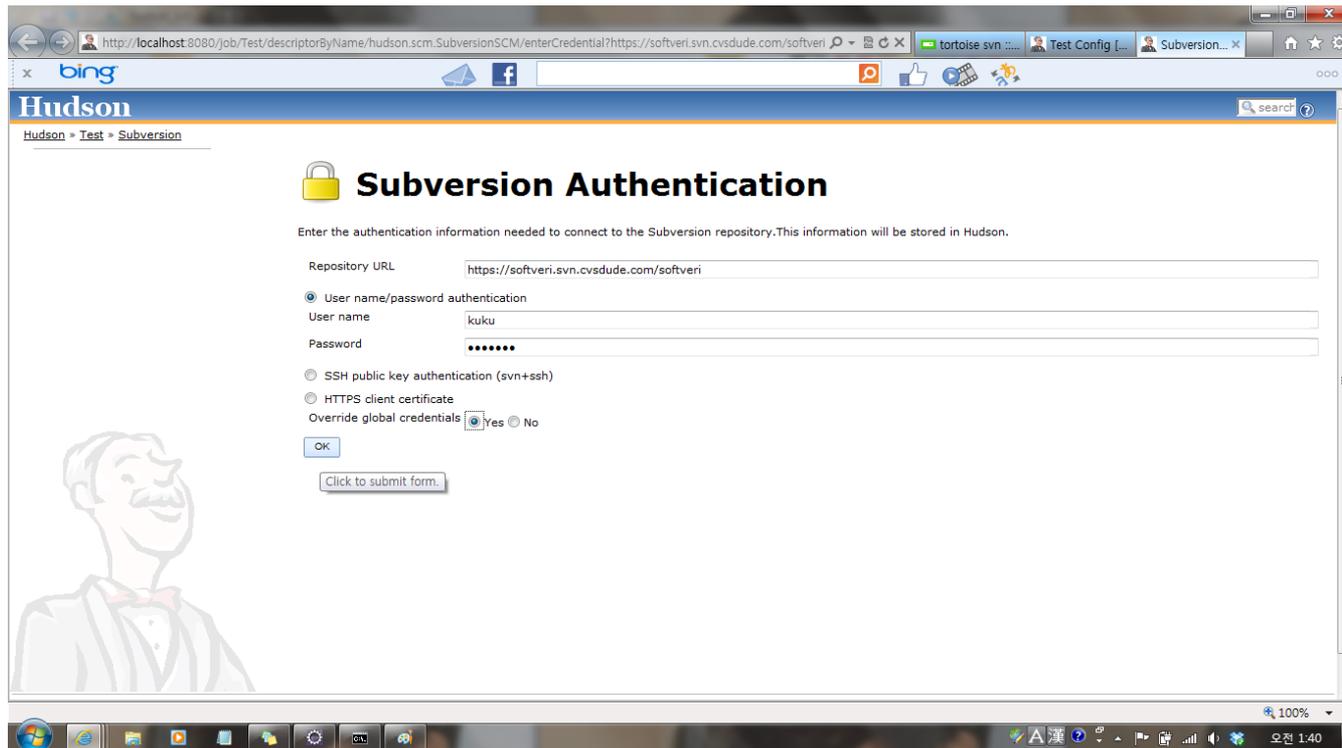
Use 'svn update' whenever possible, making the build faster. But this causes the artifacts from the previous build to remain when a new build starts.

Repository browser

Advanced...

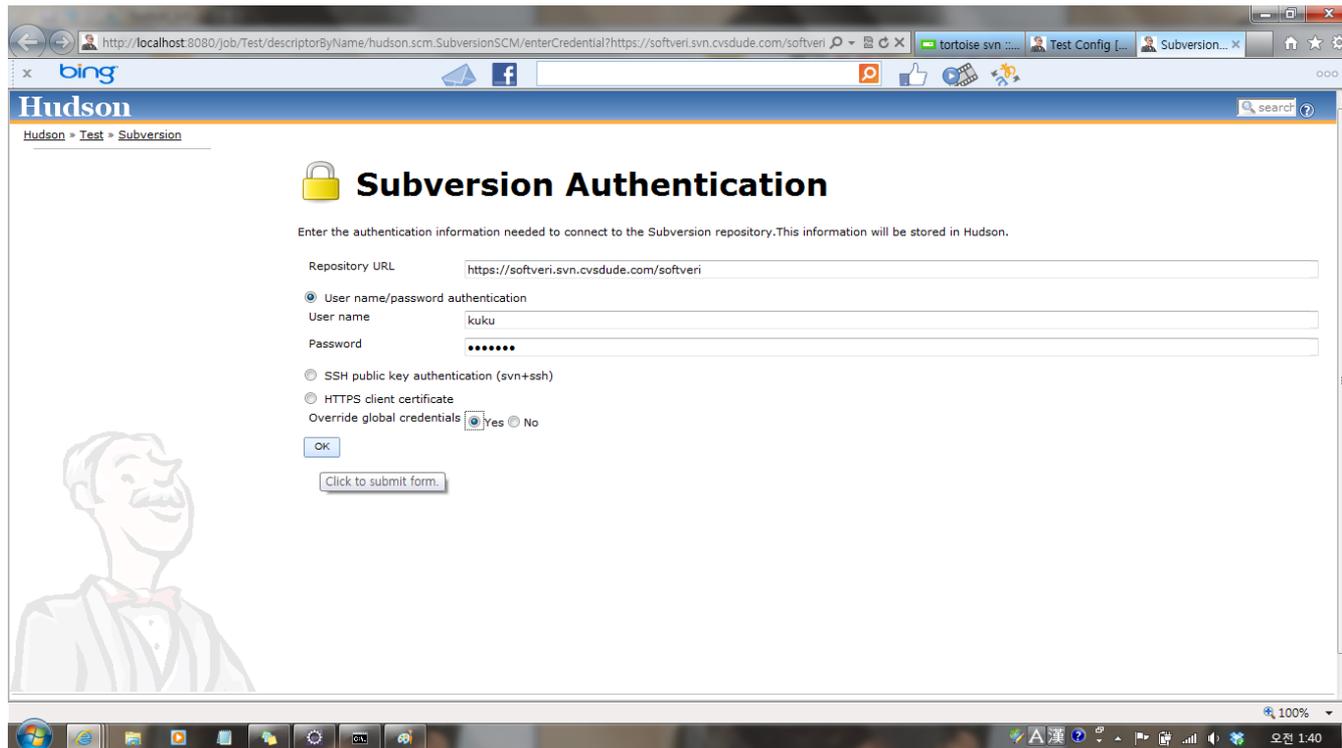
Tortoise SVN

- SVN 저장소에 로그인 필요할 경우 인증을 해줘야 함



Tortoise SVN

- SVN 저장소에 로그인 필요할 경우 인증을 해줘야 함



목차

1. CI & CTIP

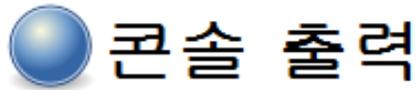
2. CTIP 환경설정

3. Hudson Result

2.1 Ant

2.2 Tortoise SVN

Result - Ant



```
Started by user anonymous
[workspace] $ cmd.exe /C "ant.bat -file test_build.xml build && exit %ERRORLEVEL%"
Buildfile: C:\apache-tomcat-6.0.35\webapps\jobs\test_ant\workspace\test_build.xml

build-subprojects:

init:

build-project:
  [echo] test_89: C:\apache-tomcat-6.0.35\webapps\jobs\test_ant\workspace\test_build.xml
  [javac] C:\apache-tomcat-6.0.35\webapps\jobs\test_ant\workspace\test_build.xml:33: warning:

build:

BUILD SUCCESSFUL
Total time: 0 seconds
[DEBUG] Skipping watched dependency update; build not configured with trigger: test_ant #9
Finished: SUCCESS
```



Result - Ant

[소개 내용 입력](#)

S	W	작업 ↓	최근 성공	최근 실패	최근 소요 시간	Console
		test_ant	3 min 27 sec (#9)	6 min 51 sec (#8)	0.66 sec	 

아이콘: [S](#) [M](#) [L](#)

[범례](#) [모든 것에 대해](#) [실패에 대해](#) [마지막 빌드에 대해](#)

“ 20% 만 성공 ”

-> Origin Source Code 에서 의도적으로 오류가 발생하도록 작성하였기 때문

Result - Tortoise SVN

- SVN에 저장된 코드를 hudson을 통해 빌드 성공한 화면

[소개 내용 입력](#)

All +	S	W	작업 ↓	최근 성공	최근 실패	최근 소요 시간	Console	
			test_ant	14 min (#10)	58 min (#8)	0.63 sec		
			Test	18 min (#18)	3 min 31 sec (#12)	6.8 sec		

아이콘: [S](#) [M](#) [L](#)

[범례](#) [모든 것에 대해](#) [실패에 대해](#) [마지막 빌드에 대해](#)

감사합니다
