

1. 도구 개요

Ant

소 개	자바 프로그래밍 언어에서 사용하는 자동화된 소프트웨어 빌드 도구 유닉스나 리눅스에서 사용되는 make와 비슷하나 자바언어로 구현되어 있어 자바 실행환경이 필요하며 자바 프로젝트들을 빌드하는데 표준으로 사용		
주요기능	패키지 빌드 자동화		
카테고리	Implementation	세부카테고리	빌드
커버리지	Package Build Automation	도구난이도	하
라이선스형태 / 비용	BSD License (Berkeley Software Distribution) / 무료	사전설치도구	JDK 1.5이상 Eclipse(본 매뉴얼에서 사용한 도구)
운영체제	Windows, Linux, Mac OS, UNIX	최신버전	1.8.4 (2012. 10)
특징	<ul style="list-style-type: none"> • Build 자동화(컴파일, Javadoc생성, 실행, FTP SCP, SFTP연결, CVS연동, 다른 공학도구와의 연동) • 배포 • 유닛 테스트 (JUnit 활용, HTML등 테스트 결과 보고서 작성) 		
적용회사 / 프로젝트	JAVA를 기반으로 진행되는 프로젝트, 대부분의 자바 IDE에 기본 적용		
관련 도구	Eclipse, NetBeans, Hudson, PMD등		
제작사	Apache Software Foundation		
공식 홈페이지	http://ant.apache.org/		

2. 기능 요약

Ant

자바 프로그래밍 언어에서 사용하는 자동화된 소프트웨어 빌드 도구

주요기능	지원여부
IDE 통합 및 도구지원	대부분의 도구 지원 (Eclipse, NetBeans 등)
원격 코드 Build자동화	지원(FTP, SCP, SFTP, SVN)
자동화 Build 중 Unit Test 수행	지원(JUnit)
테스트 수행 결과 보고서	지원(HTML)
JAVA Doc생성	지원
공학도구 연계	지원(Hudson, PMD, Maven 등)
Build후 배포	지원

3. 도구 실행 환경

Ant

JAVA를 지원하는 IDE상에서 설치 및 구현이 가능

- 다양한 OS를 지원
 - Windows : Windows XP / Windows 7 (32, 64-bit 모두 지원)
 - Linux : 32, 64-bit 지원
 - Mac OS X : 32, 64-bit 지원
 - UNIX : 32, 64-bit 지원
- JDK, 자바 기반 도구(IDE 등)이 필요
 - 코드 및 플러그인 형태, 도구에 포함되어 있는 형태로 제공

Ant**Java IDE (Eclipse, NetBeans 등)****JDK (Java development kit)****Windows / Linux / Mac OS / UNIX**

4. 도구 설치 방법

세부 목차

Ant

4.1 Ant 다운받기

4.2 Ant 설정하기

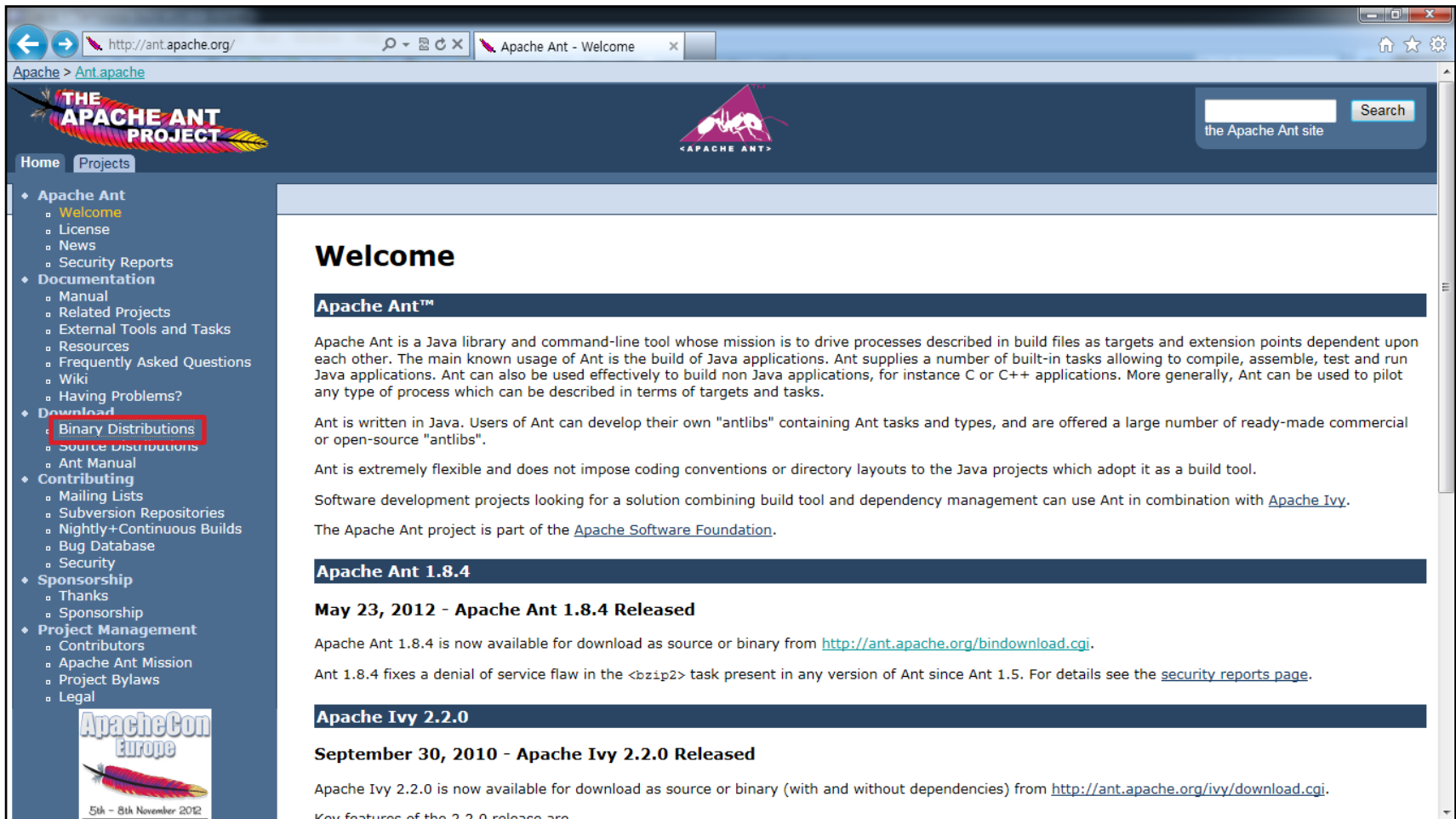
4.3 Eclipse에서 Ant 설치여부 확인하기

4. 도구 설치 방법

4.1 Ant 다운받기 (1/2)

Ant

- 대부분의 JAVA IDE는 Ant를 포함하고 있으나, Eclipse외부나 업데이트 시 설치 <http://ant.apache.org/> 에 접속 → Binary Distributions



4. 도구 설치 방법

4.1 Ant 다운받기 (2/2)

Ant

- 각 운영체제에 맞는 패키지 다운로드
 - Eclipse Windows용은 zip파일을 다운

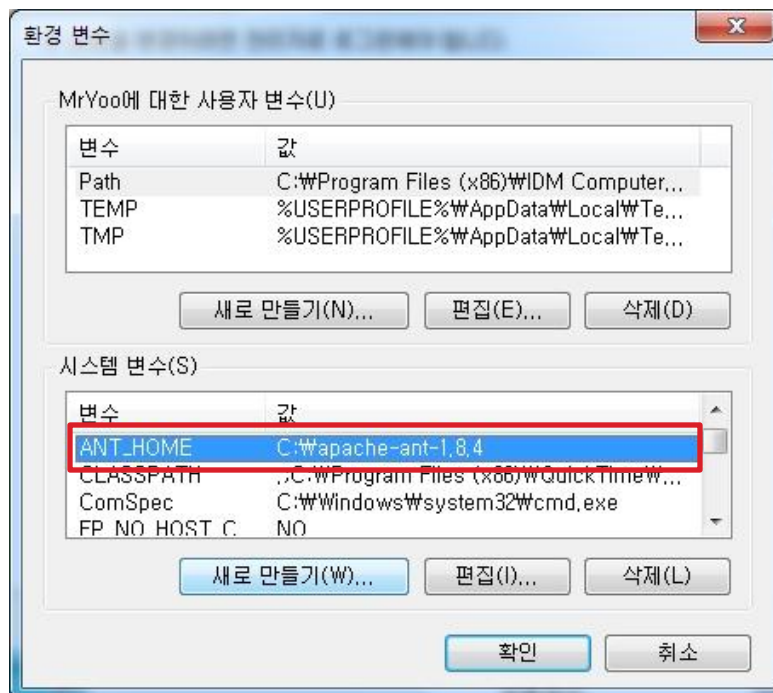
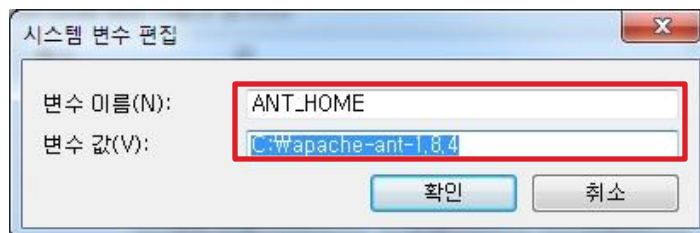
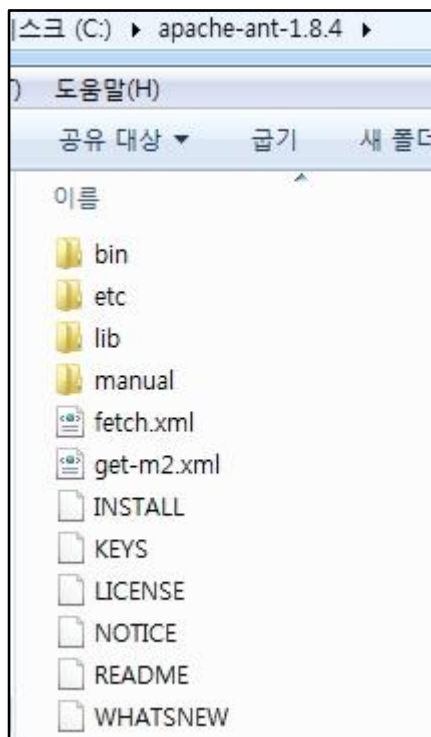
The screenshot shows the Apache Ant download page. The left sidebar contains a navigation menu with links like Manual, Download, Contributing, and Sponsorship. The main content area is titled 'Apache Ant™' and describes the tool. It includes sections for 'Downloading Apache Ant', 'Mirror', 'Current Release of Ant', and 'Old Ant Releases'. A red box highlights the 'Note' section, which states that Ant 1.8.4 was released on 23-May-2012 and may not be available on all mirrors. Another red box highlights the 'Tar files may require gnu tar to extract' section. At the bottom, a download bar shows the file 'apache-ant-1.8.4-bin.zip' (7.67MB) being downloaded from 'mirror.apache-kr.org'. A context menu is open over the download bar, showing options like '저장(S)', '다른 이름으로 저장(A)', and '저장 후 열기(O)'.

4. 도구 설치 방법

4.2 Ant 설정하기 (1/2)

Ant

- 압축 해제 및 환경변수 설정
 - 압축 해제 → ANT_HOME 환경변수 추가

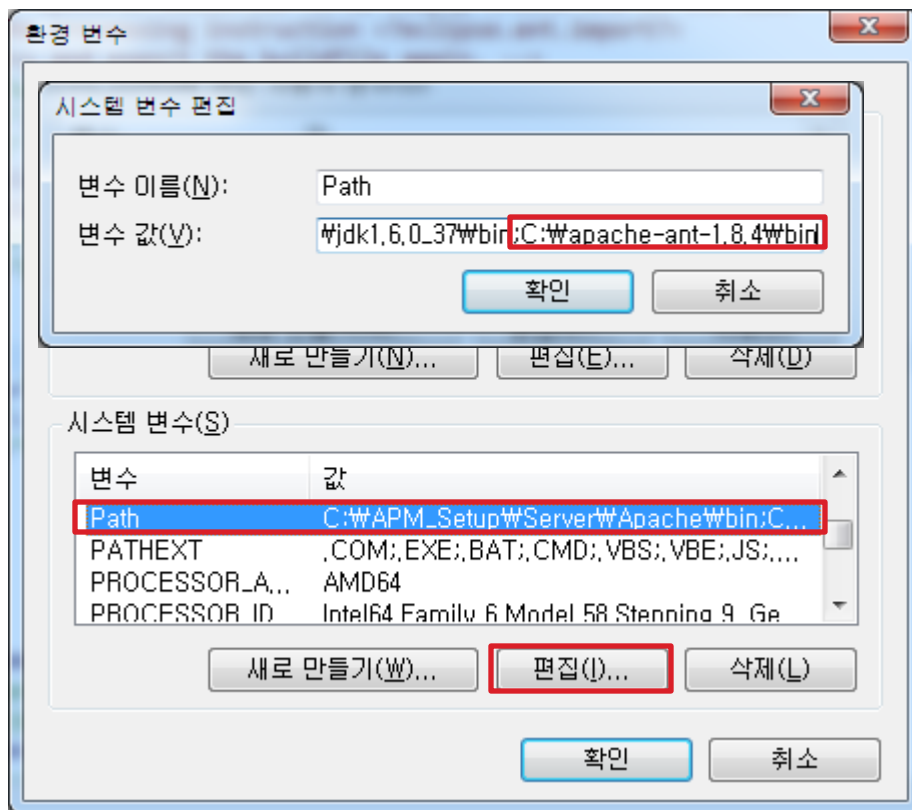


4. 도구 설치 방법

4.2 Ant 설정하기 (2/2)

Ant

- 압축 해제 및 환경변수 설정
 - Path에 Ant경로 추가

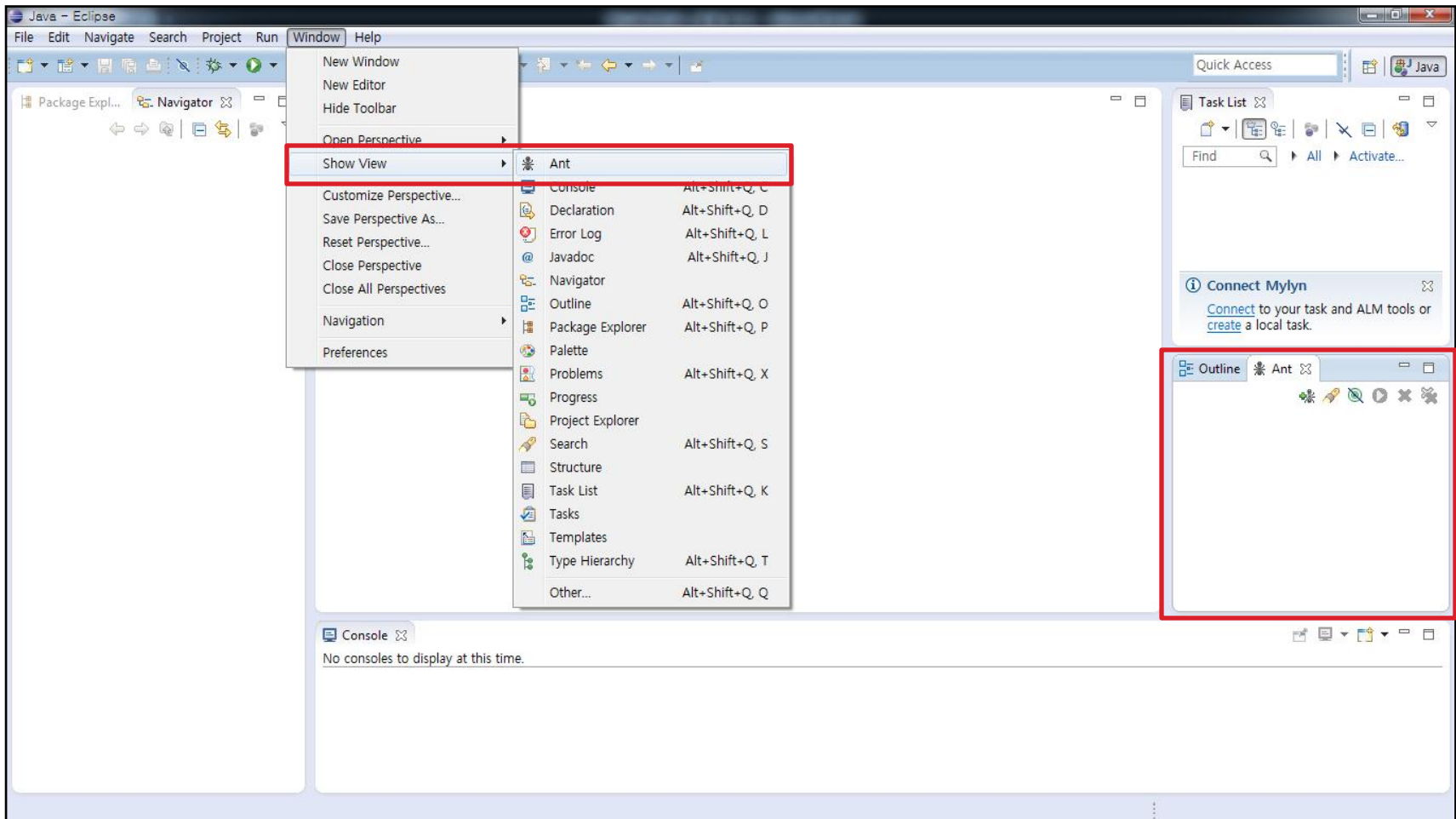


4. 도구 설치 방법

4.3 Eclipse에서 Ant 설치여부 확인하기

Ant

- 메뉴의 Windows → Show View → Ant를 클릭
 - Ant윈도우가 생성됨을 확인



5. 도구 기능 소개

세부목차

Ant

- 5.1 Ant의 개요
- 5.2 프로젝트에서 Ant Build파일 생성
- 5.3 Build.xml의 구조
- 5.4 Eclipse Ant Build 하기
- 5.5 Hudson과의 연계

5. 도구 기능 소개

5.1 Ant의 개요

Ant

Apache Ant는 빌드 도구 소프트웨어

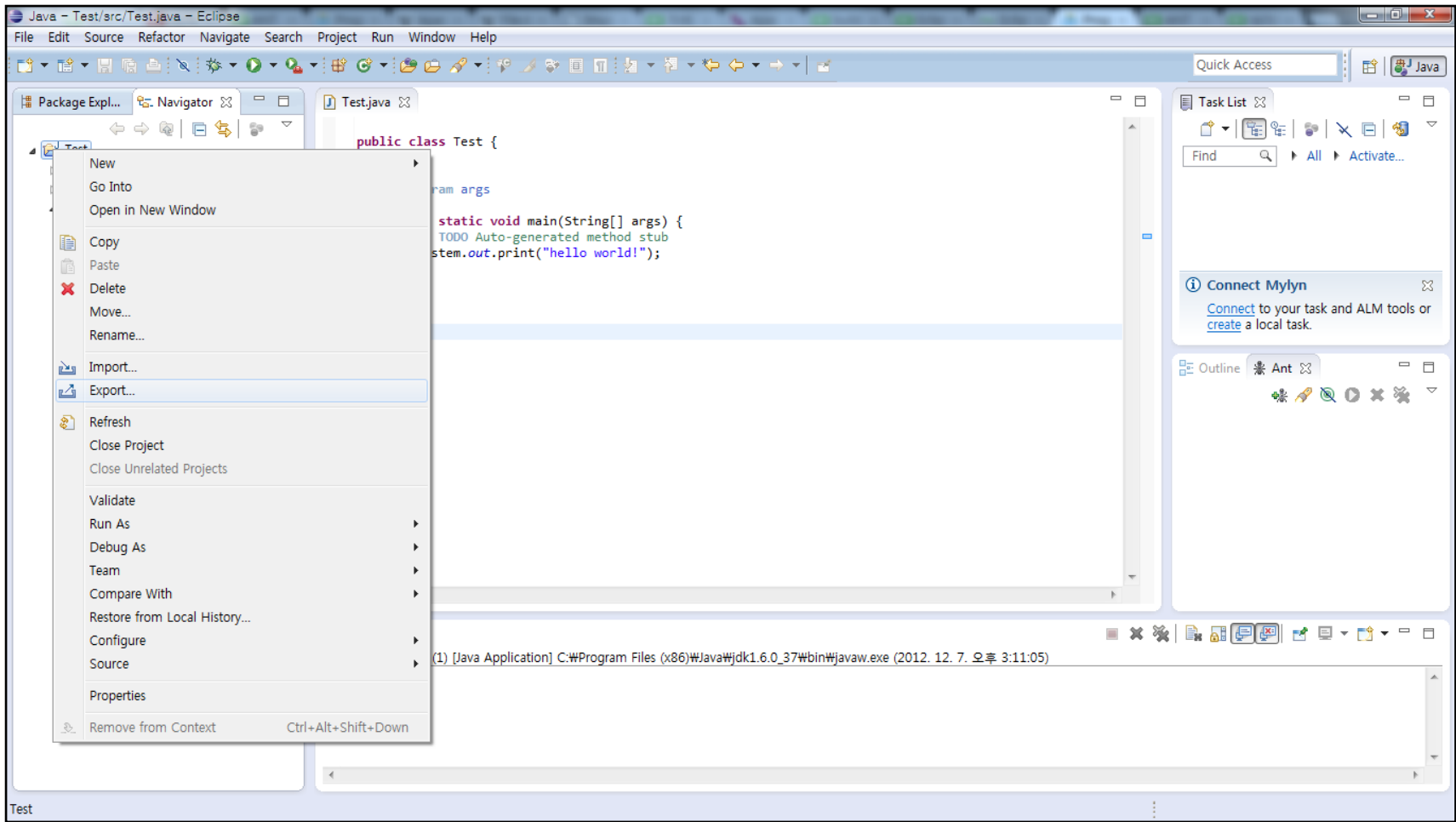
- GNU make 의 Java 버전
 - 운영 체제 (OS) 등 특정 환경에 의존하지 않는 빌드 도구
 - XML 문서 빌드(소프트웨어 구축)의 규칙을 작성하는 것이 특징
 - 통합 개발 환경 Eclipse 는 Ant 플러그인 이 기본으로 내장
 - 원래 Apache Tomcat 을 빌드하기 위해 개발된 것
- 태스크 기반의 XML 요소를 빌드 파일에 작성하여 빌드 규칙을 생성
 - 기본적으로 build.xml형태로 존재
 - Ant 플러그인으로 제공되고 있는 것을 외부에서 채용하는 것으로, 추가 가능
 - Ant의 응용 프로그래밍 인터페이스 (API)에 따라 Java로 작성하여 직접 작성 및 수정 가능
 - 최근 Java뿐만 아니라 IKVM.NET 프로젝트 및 Mono (소프트웨어) 프로젝트에서 Ant task for IKVMC 로 .NET Framework 에서의 이용 촉진 중

5. 도구 기능 소개

5.2 프로젝트에서 Ant Build파일 생성 (1/4)

Ant

- 프로젝트 우 클릭 → Export클릭

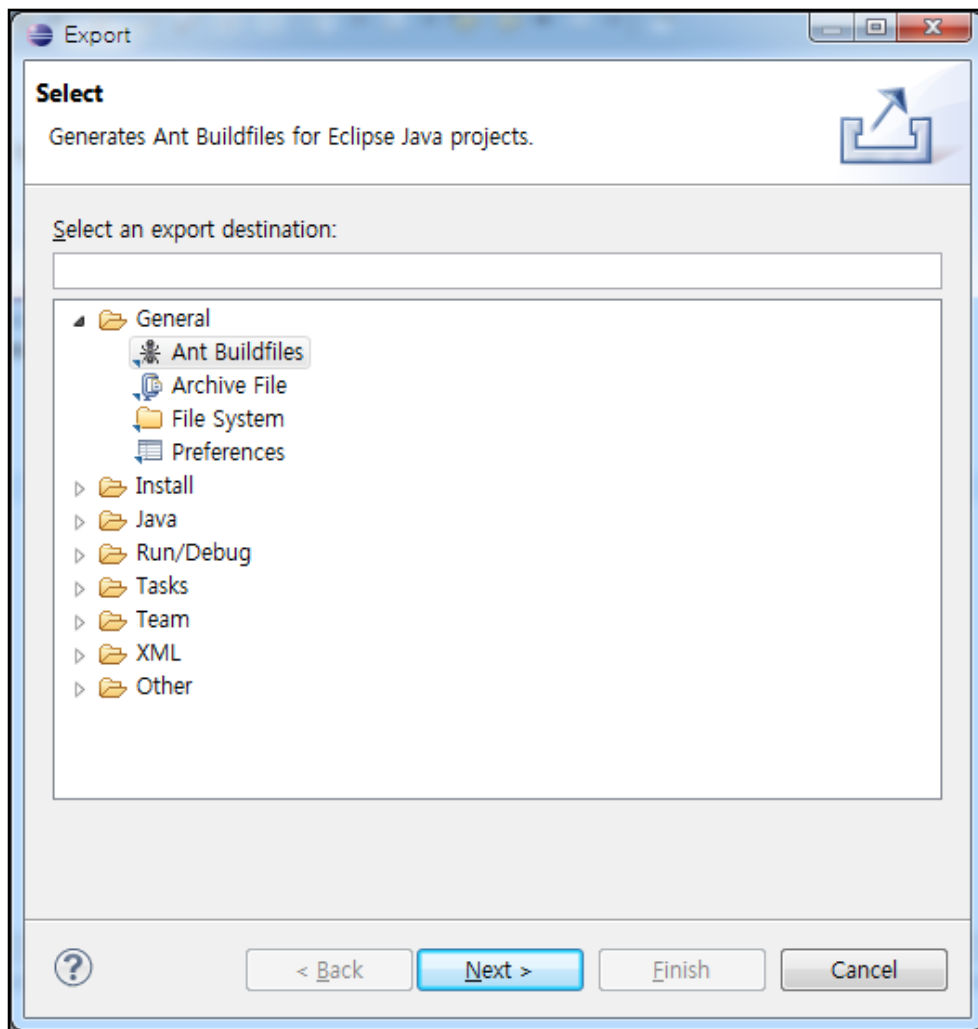


5. 도구 기능 소개

5.2 프로젝트에서 Ant Build파일 생성 (2/4)

Ant

- JAVA → “Ant Buildfiles”선택 → Next

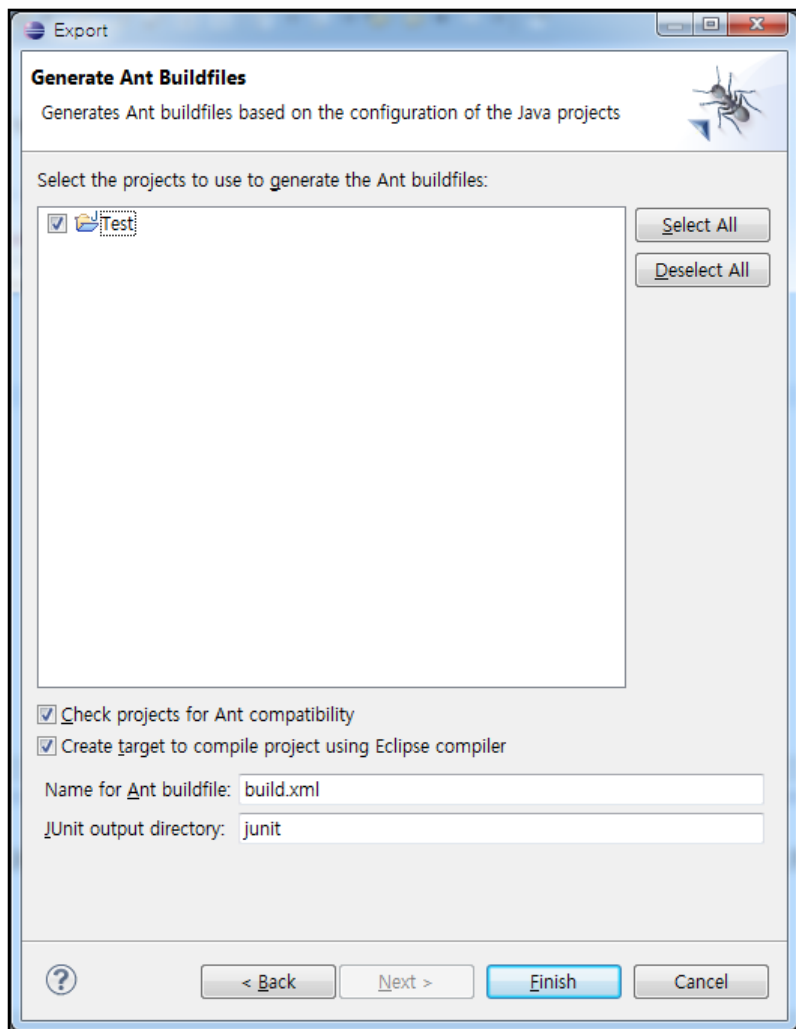


5. 도구 기능 소개

5.2 프로젝트에서 Ant Build파일 생성 (3/4)

Ant

- 프로젝트를 선택 → Finish

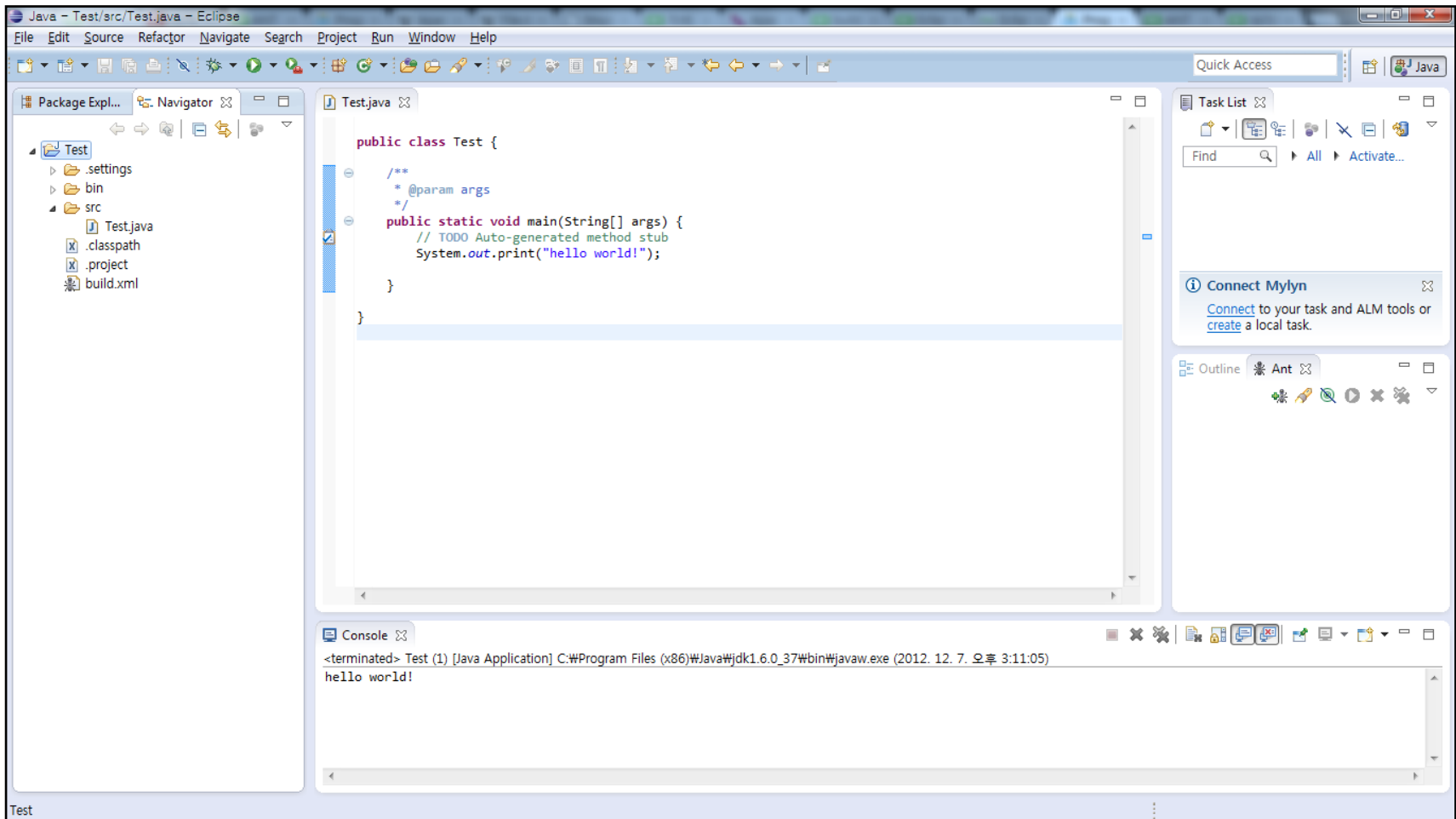


5. 도구 기능 소개

5.2 프로젝트에서 Ant Build파일 생성 (4/4)

Ant

- Build파일 생성 결과
 - Build.xml파일이 생성된 것을 확인할 수 있음



5. 도구 기능 소개

5.3 Build.xml의 구조 (1/7)

Ant

- Ant의 Build.xml파일 : Project요소, Target요소, Task요소를 포함
 - Eclipse에서 생성한 기본 build.xml파일

```
<?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">
  <property environment="env"/>
  <property name="debuglevel" value="source,lines,vars"/>
  <property name="target" value="1.6"/>
  <property name="source" value="1.6"/>
  <path id="Test.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"/>
  <target name="build-subprojects"/>
  <target depends="init" name="build-project">
    <echo message="${ant.project.name}: ${ant.file}"/>
    <javac debug="true" debuglevel="${debuglevel}" destdir="bin" includeantruntime="false" source="${source}" target="${target}">
      <src path="src"/>
      <classpath refid="Test.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}">

```


5. 도구 기능 소개

5.3 Build.xml의 구조 (2/7)

Ant

- 타깃 간의 의존 관계는 target 요소의 depends 속성으로 기술

- 그림예시 : dist 타깃은 compile 타깃에 의존
> compile → dist → clean 순으로 동작

- 일반적인 Ant의 명령어

- property : 속성을 지정
- mkdir : 새로운 디렉토리 생성
- copy : 파일, 디렉토리 복사
- javac : 컴파일
- jar : jar 파일 생성
- javadoc : javadoc 생성
- delete : 파일, 디렉토리 삭제
- Java : Java 프로그램을 실행
- Junit : 테스트 프레임 워크 JUnit 을 사용하여 Java 프로그램을 테스트
- Jnitreport : junit 작업 출력 결과 파일을 사용하여 HTML 형식 등에 대응하는 보고서를 생성
- ftp : FTP 연결을 시작하고 파일 업로드 , 다운로드 등을 가능하게 함
- zip : 지정된 디렉토리나 파일을 ZIP 형식으로 압축 · 보관
- echo : 콘솔 (명령 라인 환경)에 문자열 출력
- splash : 런타임에 지정된 시간 동안 시작 표시
- buildnumber : 빌드 번호를 업데이트

```

1 <project name="project_name" default="all" basedir=".">
2   <target name="all" depends="compile, dist, clean" >
3
4   </target>
5
6   <target name="compile">
7
8   </target>
9
10  <target name="dist" depends="compile">
11
12  </target>
13
14  <target name="clean">
15
16  </target>
17 </project>

```

5. 도구 기능 소개

5.3 Build.xml의 구조 (3/7)

Ant

- property : 속성을 지정
 - 속성에서는 대소문자를 구별
 - > foo.src라는 이름의 속성으로 "src"를 지정 시

```
<property name="foo.src" value="src" />
```

- > 빌드 스크립트의 다른 부분에서 이 속성을 참조하려면 "\${foo.src}"와 같이 기록

```
<javac srcdir="${foo.src}" destdir="${build}" />
```

- > 파일을 읽어 속성을 설정

```
<javac file="foo.properties" />
```

- mkdir : 새로운 디렉토리 생성
 - 디렉토리 생성 예시

```
<mkdir dir="${dist}" />  
<mkdir dir="${dist}/lib" />
```

5. 도구 기능 소개

5.3 Build.xml의 구조 (4/7)

Ant

- copy : 파일, 디렉토리 복사

- 파일 하나 복사 예시

```
<copy file="myfile.txt" tofile="mycopy.txt" / >
```

- 디렉토리에서 다른 디렉토리 복사 예시

```
<copy todir="../new/dir ">  
  <fileset dir="src_dir ">  
</copy>
```

- 특정 디렉토리의 원하는 파일만 지정하여 복사 : *.java 파일을 제외한 나머지 파일 복사 예시

> exclude는 특정 내용을 제외하는 키워드이고 **의 경우 src_dir 디렉토리 아래 모든 디렉토리를 재귀적으로 탐색

```
<copy todir="../dest/dir ">  
  <fileset dir="src_dir ">  
    <exclude name="**/*.java" />  
  </fileset>  
</copy>
```

5. 도구 기능 소개

5.3 Build.xml의 구조 (5/7)

Ant

- javac : JAVA 소스파일의 컴파일
 - 재귀적으로 탐색하여, class파일이 없거나 class 파일이 java 파일보다 오래된 경우에만 컴파일
 - \${src}와 그 하위 디렉토리에 있는 모든 java 파일을 컴파일 예시
 - > 결과를 \${build} 디렉토리에 저장, 클래스패스에는 xyz.jar가 포함되고, 디버깅 옵션을 켜고 컴파일

```
<javac srcdir="${src}" destdir="${build}" classpath="xyz.jar" debug="on" />
```

- \${src}와 \${src2} 및 그 하위 디렉토리에 있는 java파일을 컴파일 : 결과를 \${build} 디렉토리에 저장
 - > mypackage/p1과 mypackage/p2에 있는 파일만을 사용하고, 클래스패스에 xyz.jar가 포함
 - > 여기서 include가 특정 내용을 포함하는 키워드 입니다

```
<javac srcdir="${src}:${src2}" destdir="${build}"
include="mypackage/p1/**,mypackage/p2/**"
exclude="mypackage/p1/testpackage/**" classpath="xyz.jar" debug="on" />
```

5. 도구 기능 소개

5.3 Build.xml의 구조 (6/7)

Ant

- jar : 지정된 파일들을 jar로 묶습니다.

- \${build}/classes 밑에 있는 파일을 app.jar로 묶기

```
<jar destfile="${dist}/lib/app.jar" basedir="${build}/classes" />
```

- mypackage/text 밑에 있는 파일만을 묶고 Test.class는 제외

```
<jar destfile="${dist}/lib/app.jar" basedir="{build}/classes"
include="mypackage/test/**" exclude="**/Test.class" />
```

- \${build}/classes와 \${src}/resources 밑에 있는 파일을 app.jar로 묶되, Test.class는 제외

```
<jar destfile="${dist}/lib/app.jar" ">
  <fileset dir="${build}/classes" exclude="**/Test.class" />
  <fileset dir="${src}/resources" />
</jar>
```

- javadoc : javadoc 문서를 생성,

- src 디렉토리 밑에 있는 소스파일을 읽어 javadoc 문서를 생성하여 \${doc} 디렉토리에 저장

```
<javadoc destdir="${doc}" ">
  <fileset dir="${src}" ">
  </fileset>
</javadoc>
```

5. 도구 기능 소개

5.3 Build.xml의 구조 (7/7)

Ant

- delete : 하나의 파일 또는 디렉토리와 그 하위 디렉토리, fileset에 지정된 파일을 삭제
 - /lib/ant.jar 파일을 삭제

```
<delete file="/lib/ant.jar" />
```

- lib 디렉토리와 그 하위 디렉토리를 삭제

```
<delete dir="/lib" />
```

- 현재 디렉토리와 그 하위 디렉토리에서 확장자가 bak인 모든 파일을 삭제

```
<delete>  
  <fileset dir="." include="**/*.bak" />  
</delete>
```

- build 디렉토리와 그 하위 디렉토리를 삭제합니다. includeEmptyDirs를 "true"로 설정하면 fileset을 사용할 때 빈 디렉토리도 포함

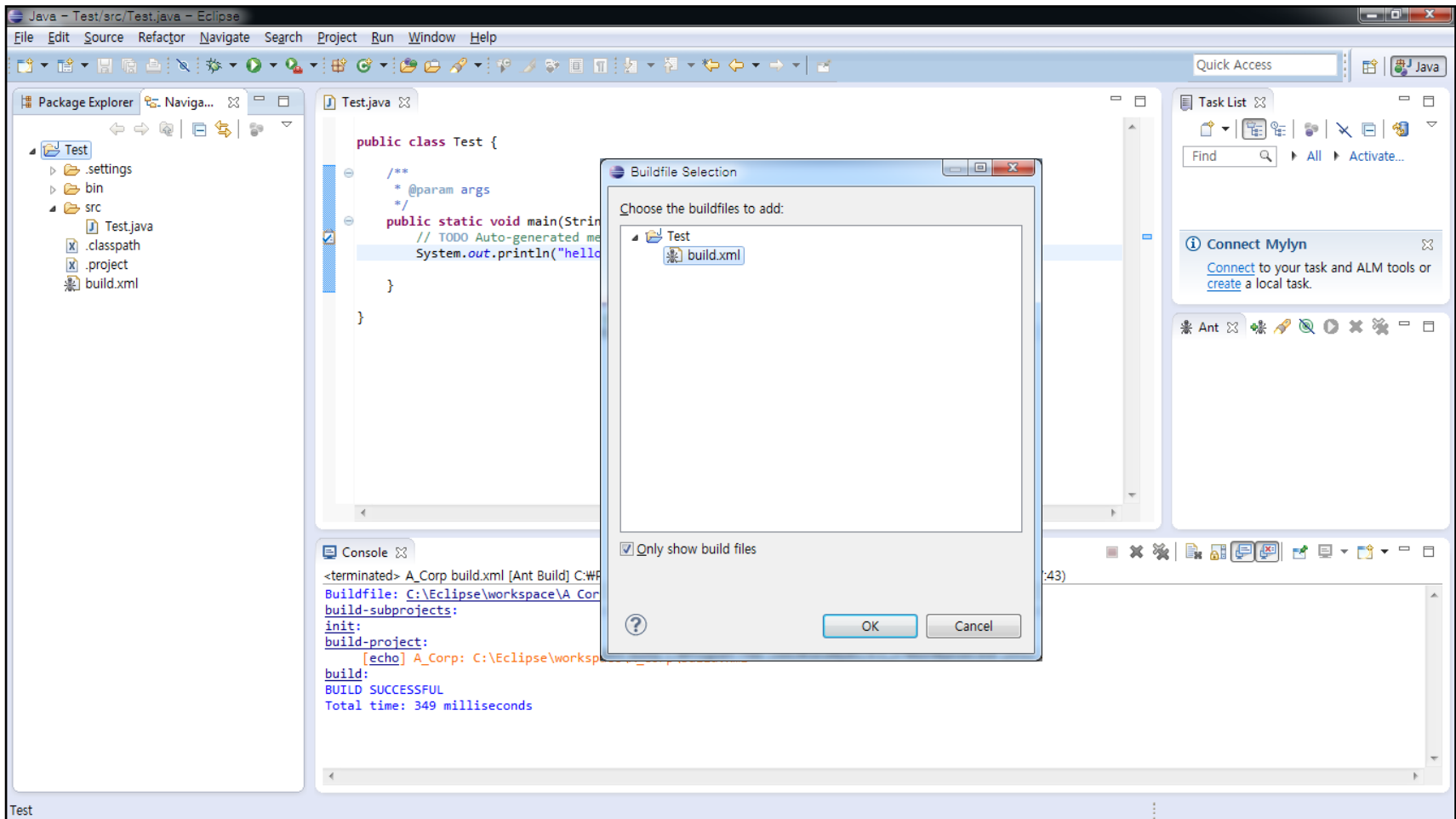
```
<delete includeEmptyDirs="true">  
  <fileset dir="build" />  
</delete>
```

5. 도구 기능 소개

5.4 Eclipse Ant Build 하기 (1/2)

Ant

- Add Buildfile → Finish클릭
 - Eclipse내부에서 생성된 Ant파일 읽기

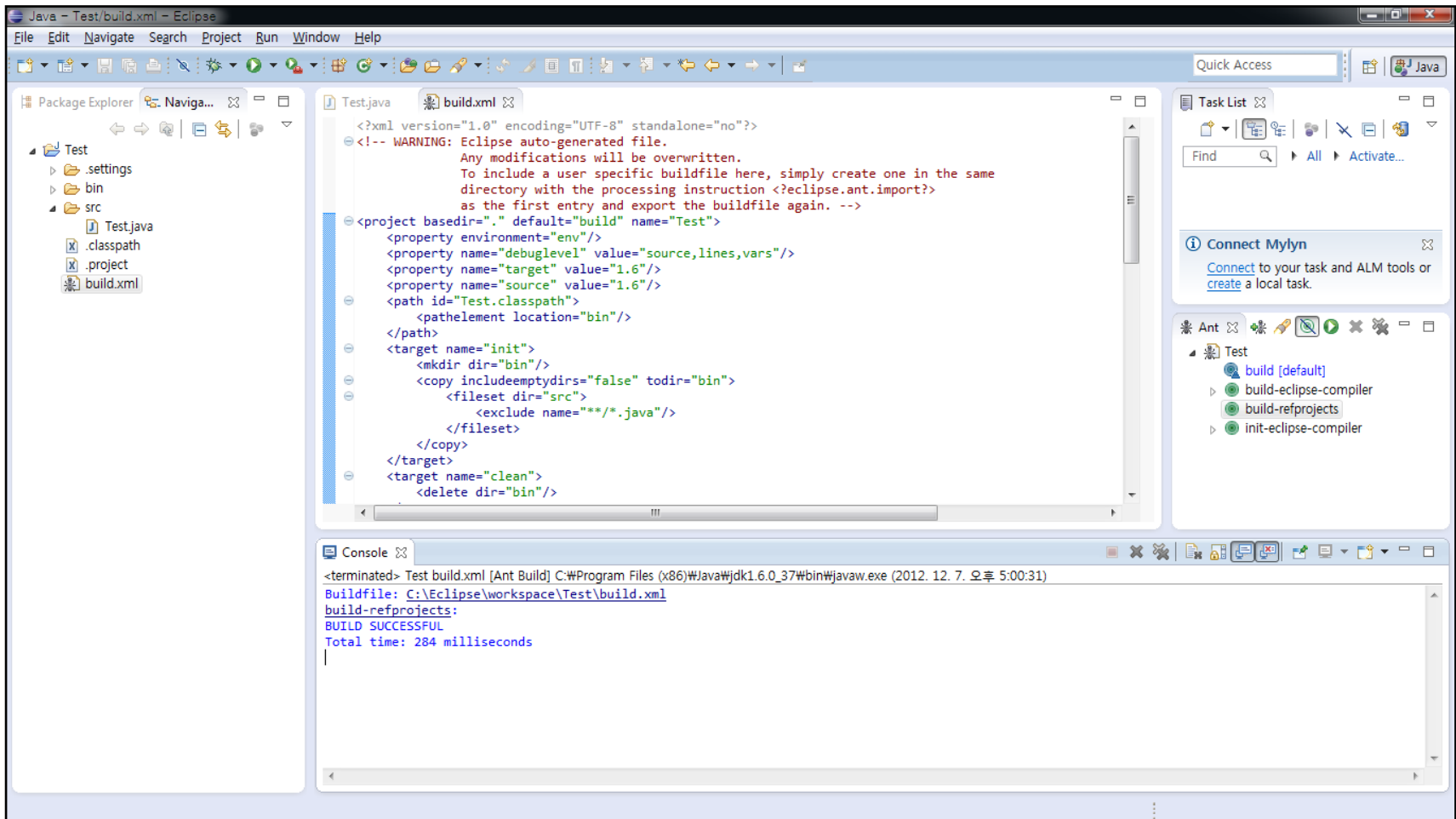


5. 도구 기능 소개

5.4 Eclipse Ant Build 하기 (2/2)

Ant

- Eclipse Ant의 Build 결과
 - Ant 프로젝트 트리 탐색기의 Build할 부분을 선택 → Run the selected target버튼 / 더블클릭



5. 도구 기능 소개

5.5 Hudson과의 연계 (1/3)

Ant

- Hudson 프로젝트 생성
 - 새 작업 → 작업 명 입력 → OK

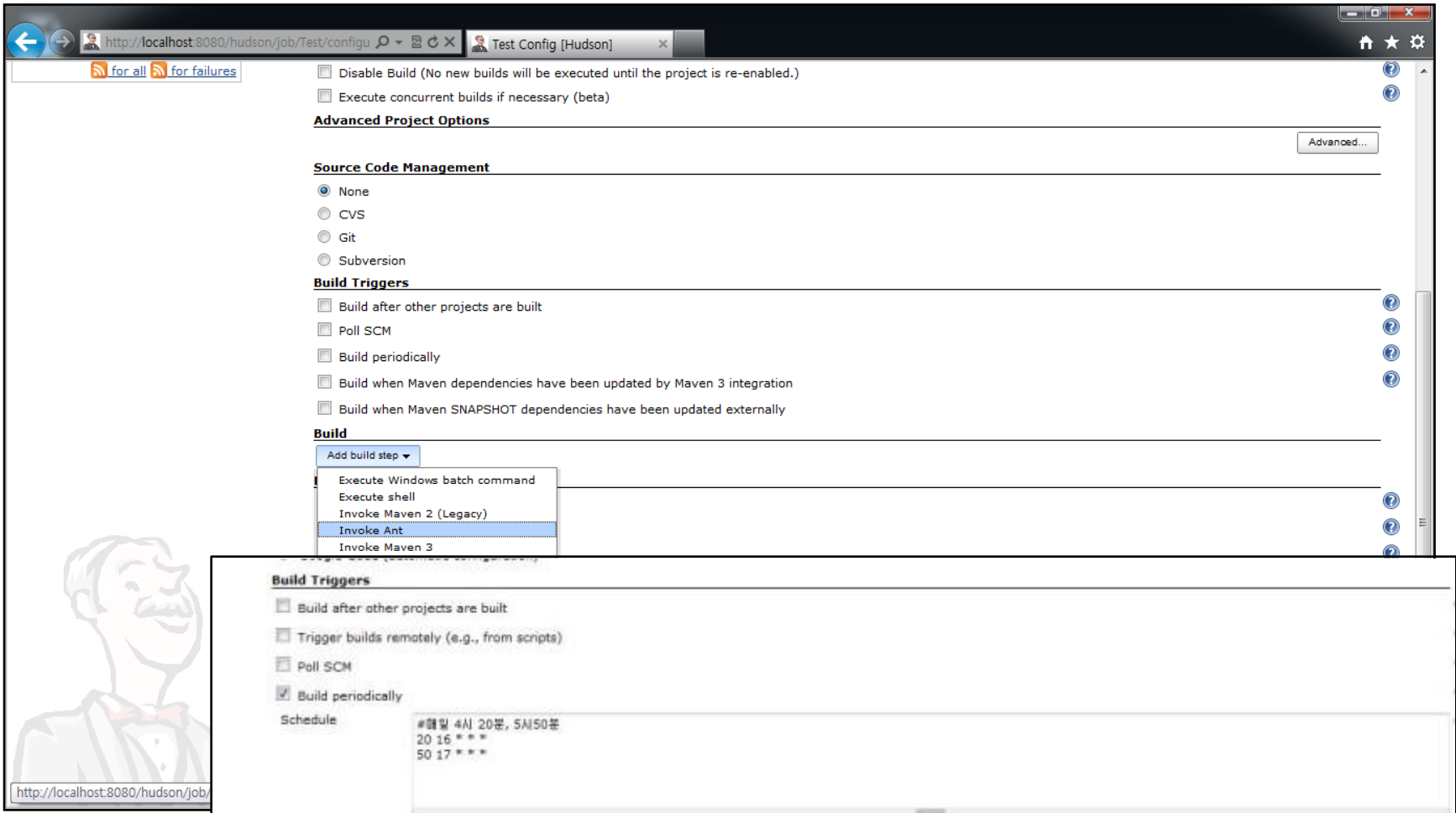


5. 도구 기능 소개

5.5 Hudson과의 연계 (2/3)

Ant

- Ant설정 : Build → Add Build step → Invoke Ant
 - Build Triggers에서 빌드 시점 지정 가능



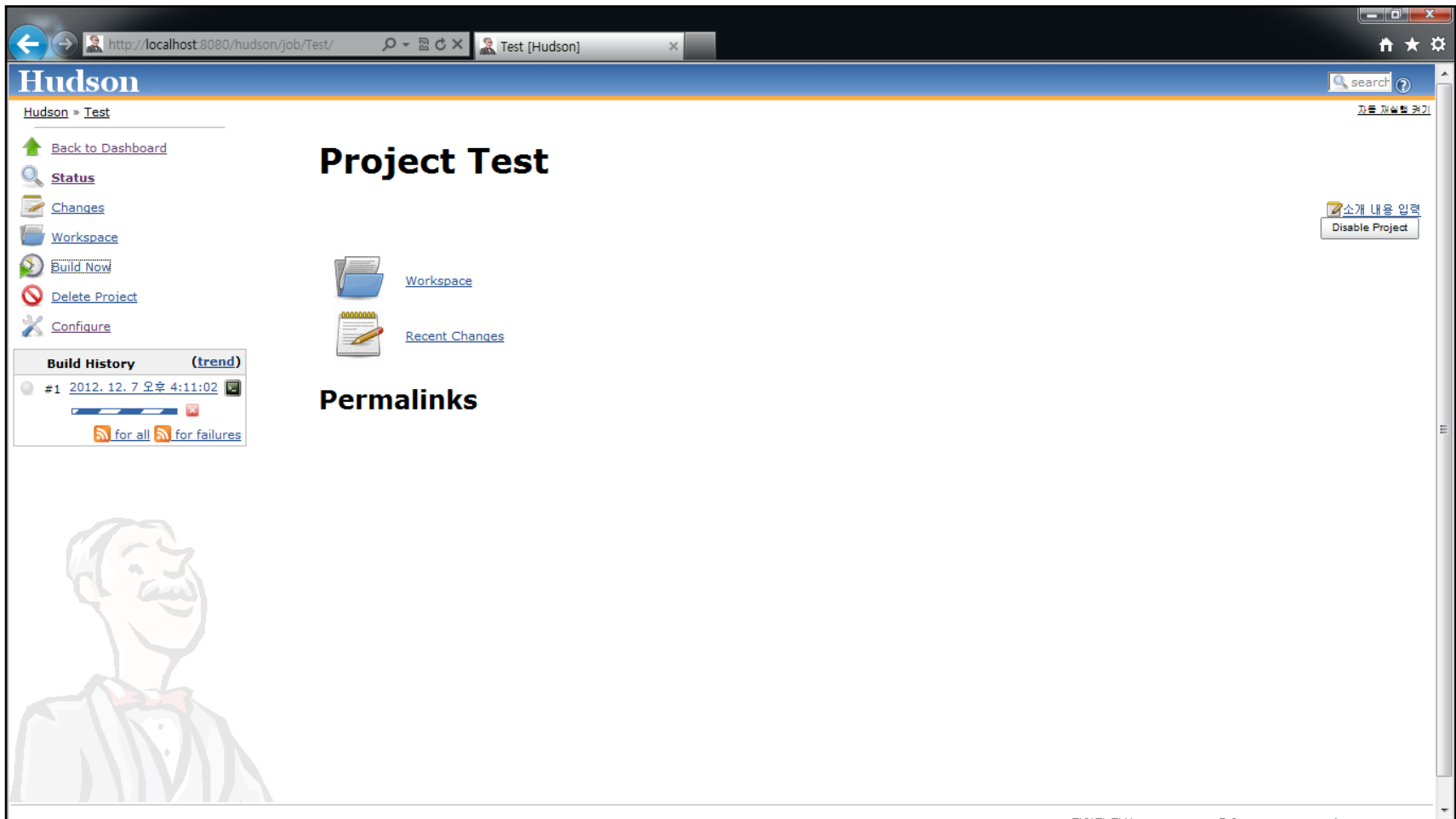
The screenshot displays the Hudson web interface for configuring a job named 'Test Config'. The 'Build' section is active, showing a dropdown menu for 'Add build step' with 'Invoke Ant' selected. Other options include 'Execute Windows batch command', 'Execute shell', 'Invoke Maven 2 (Legacy)', and 'Invoke Maven 3'. An inset window shows the 'Build Triggers' section, where 'Build periodically' is checked, and a cron schedule is entered: '# 매일 4시 20분, 5시 50분' and '20 16 * * *' and '50 17 * * *'.

5. 도구 기능 소개

5.5 Hudson과의 연계 (3/3)

Ant

- Ant Build수행
 - Build now를 눌러 실행하거나, 설정된 Build plan에 따라 Build 수행



6. 도구 활용 예제

세부 목차

Ant

- 6.1 예제 소개
- 6.2 build.xml 생성
- 6.3 Eclipse 내부에서 Ant 사용하기
- 6.4 Hudson 설정 및 연계

6. 도구 활용 예제

6.1 예제소개 (1/2)

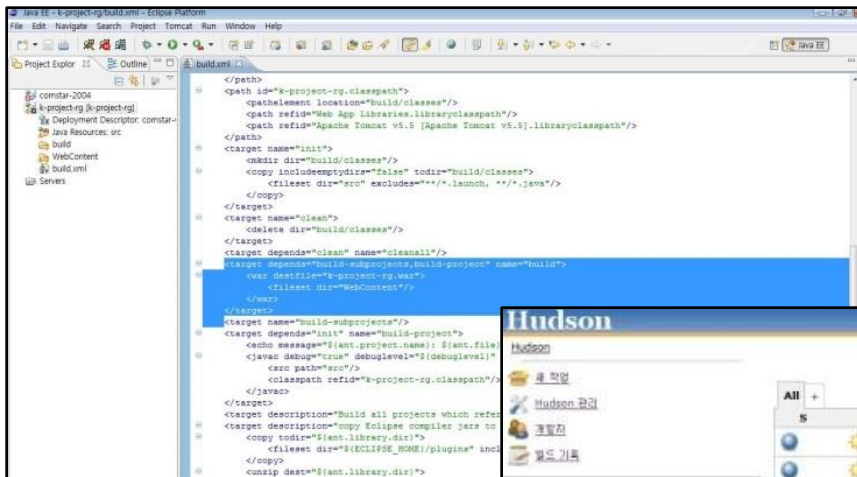
Ant

- 예제 시스템 : A업체의 Ant 적용
 - A업체의 상황 및 개선예정사항

A업체는 자바 기반의 객체지향 프로그래밍을 위해 Eclipse를 사용

A업체는 빈번한 코드 변경이 일어나지만, Build와 Compile 및 Test가 자동화 되지 않아
찾은 배포 문제를 겪음

A업체는 Ant와 Hudson을 사용하여 자동 생성 체계를 갖추



All	S	W	작업	최근 성공	최근 실패	최근 소요 시간
			AntTest	40 min (#30)	11 days (#25)	1.9 sec
			Exam01	40 sec (#26)	13 days (#17)	1.6 sec
			WEB-Project	1 hr 10 min (#6)	1 hr 35 min (#6)	2.7 sec

이미콘: S M L

Legend: for all for failures for 3

6. 도구 활용 예제

6.1 예제소개 (2/2)

Ant

A업체의 몇 가지 이슈사항

소스코드의 변경이 자주 일어남

잦은 변경으로 인한 테스트와 배포과정의 자동화가 필요

➡ 공개 측정도구인 Ant를 도입, 프로그램을 효율적으로 측정, 관리하기로 결정

Ant 도입효과

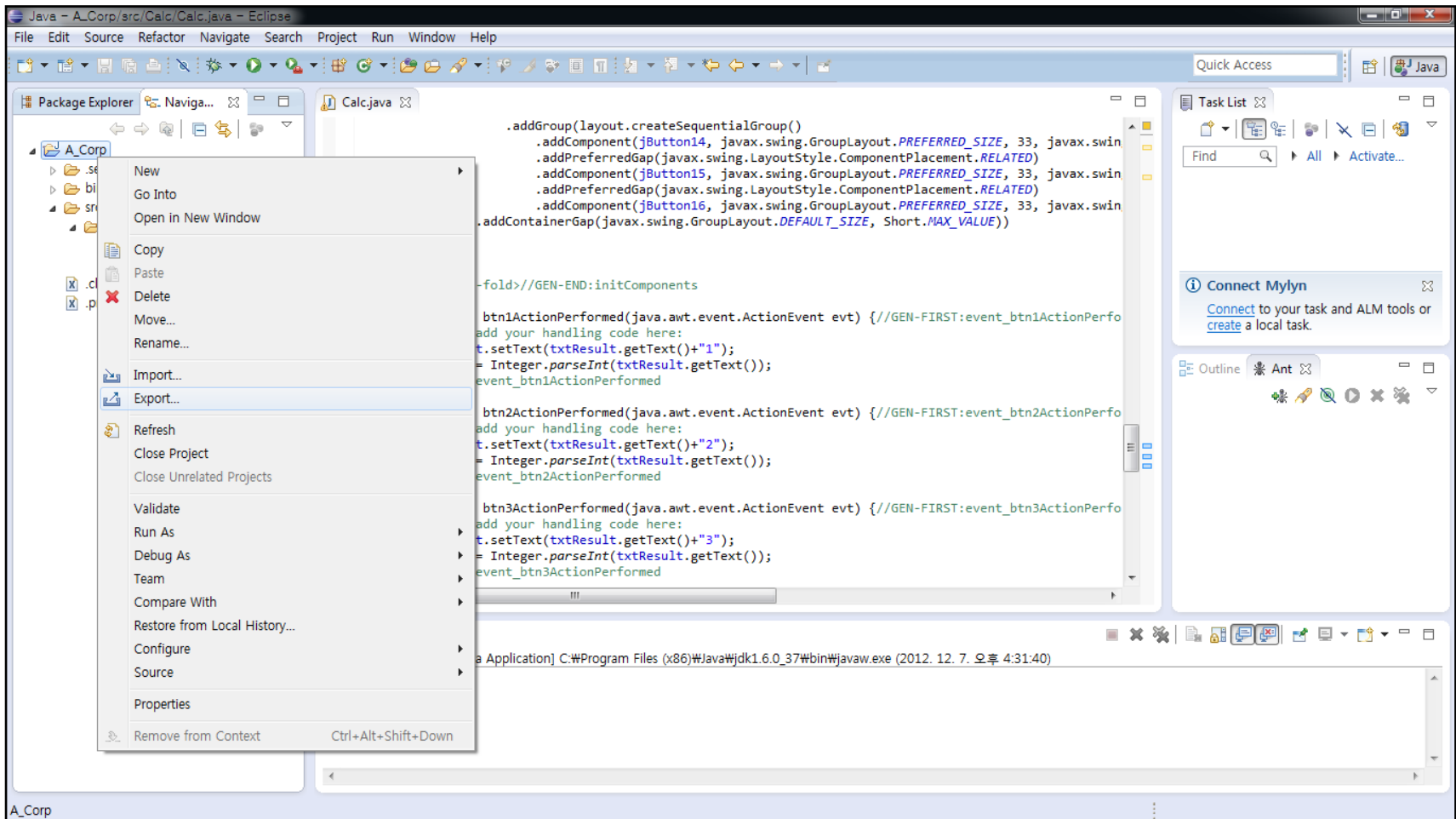
- Build부터 Test, 배포까지 자동화
- 자동화된 체계로 인한 노력 절감
- 정해진 Build로 인한 최신화 유지

6. 도구 활용 예제

6.2 build.xml생성 (1/4)

Ant

- 프로젝트 우 클릭 → Export클릭
 - Build.xml파일을 추출하는 과정

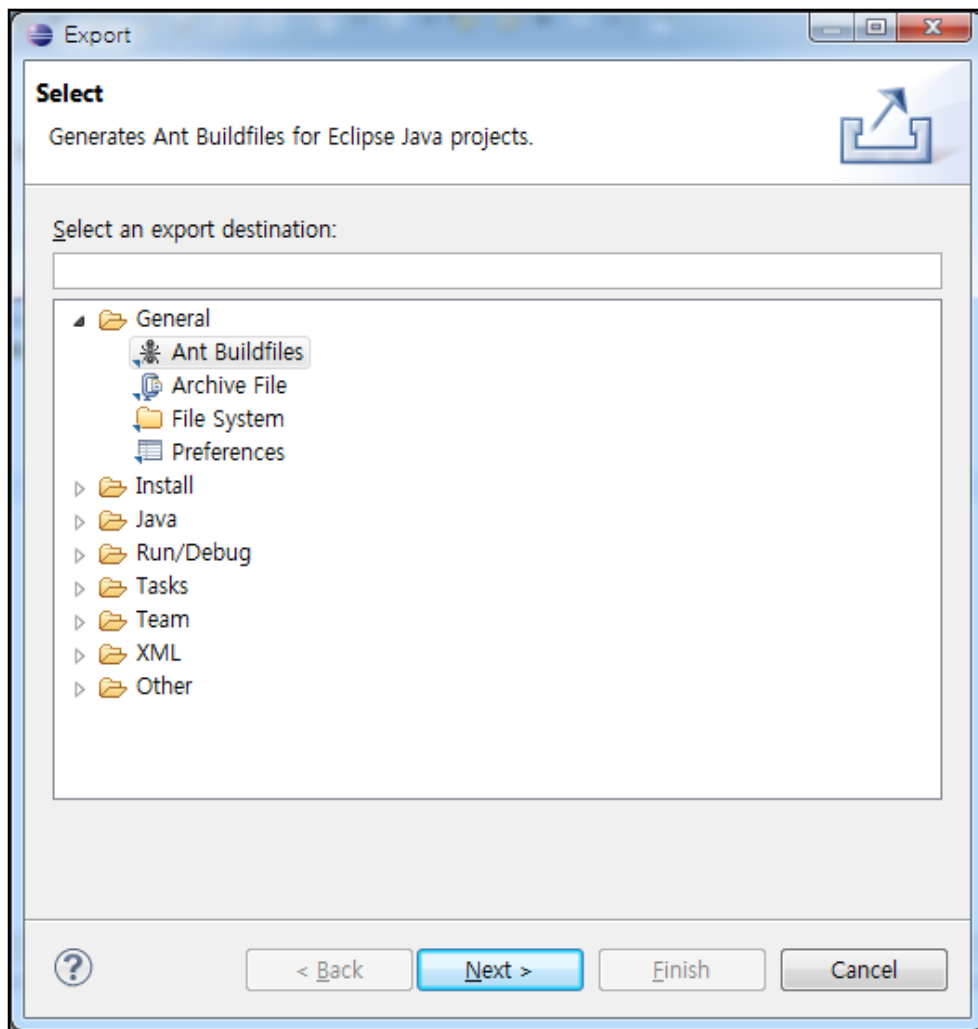


6. 도구 활용 예제

6.2 build.xml 생성 (2/4)

Ant

- JAVA → “Ant Buildfiles” 선택 → Next

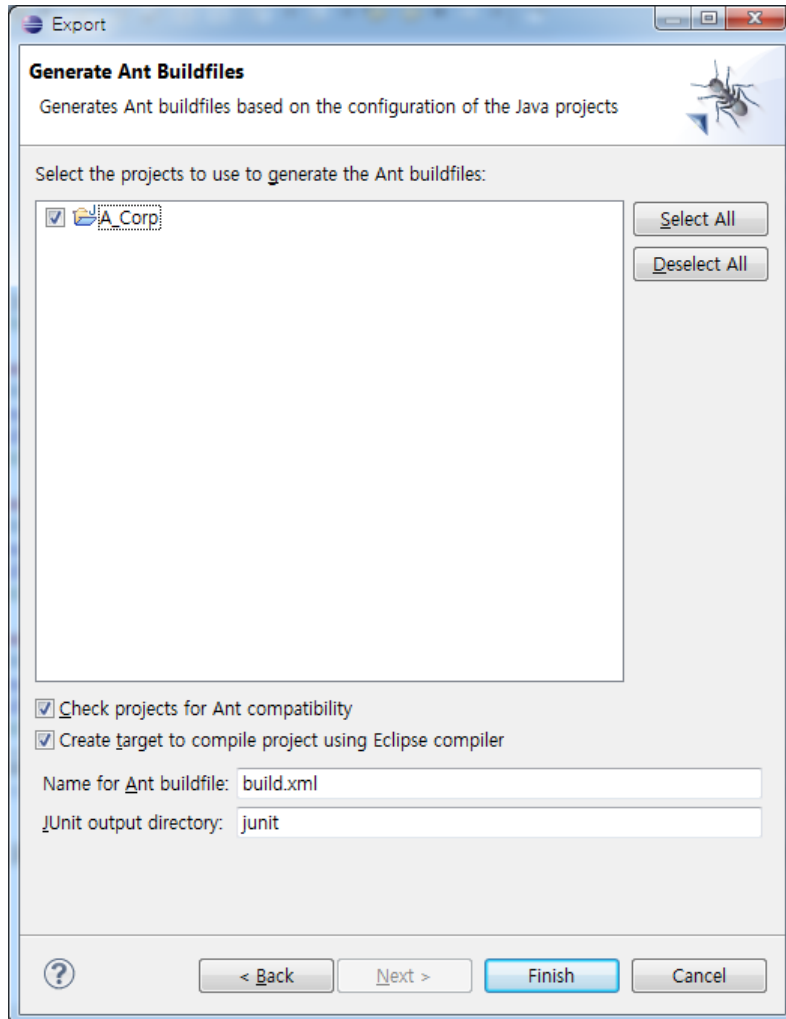


6. 도구 활용 예제

6.2 build.xml 생성 (3/4)

Ant

- 프로젝트를 선택 → Finish

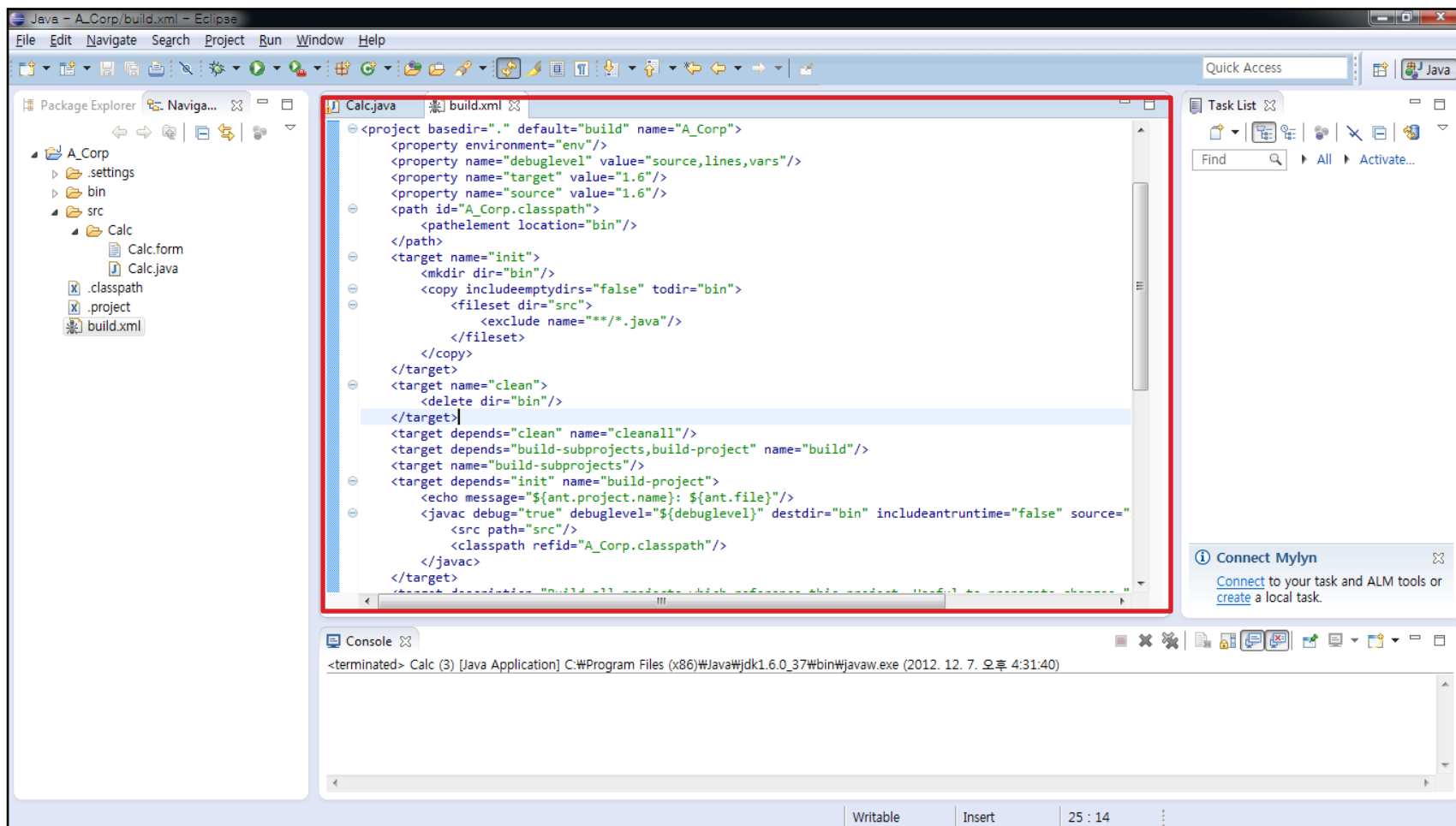


6. 도구 활용 예제

6.2 build.xml생성 (4/4)

Ant

- 생성된 Build 파일을 수정
 - 생성된 Build파일을 자동화 Build목표에 맞게 수정

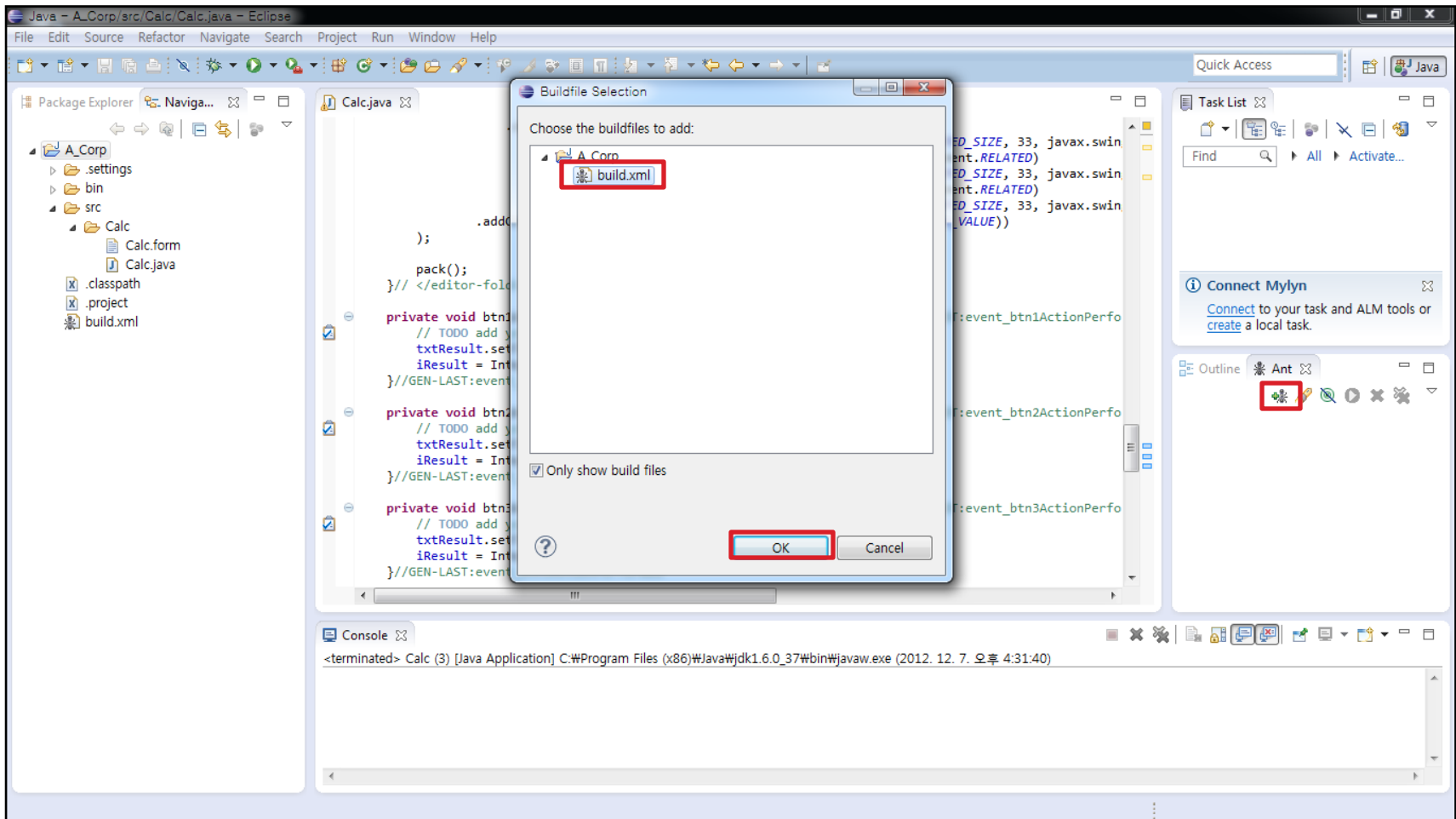


6. 도구 활용 예제

6.3 Eclipse내부에서 Ant사용하기 (1/2)

Ant

- Add Buildfile → Finish클릭
 - Eclipse내부에서 생성된 Ant파일 읽기



6. 도구 활용 예제

6.3 Eclipse내부에서 Ant사용하기 (2/2)

Ant

- Eclipse Ant의 Build 결과
 - Ant 프로젝트 트리 탐색기의 Build할 부분을 선택 → Run the selected target버튼 / 더블클릭

The screenshot shows the Eclipse IDE interface during an Ant build. The main editor displays the `build.xml` file with the following content:

```
<project basedir="." default="build" name="A_Corp">
  <property environment="env"/>
  <property name="debuglevel" value="source,lines,vars"/>
  <property name="target" value="1.6"/>
  <property name="source" value="1.6"/>
  <path id="A_Corp.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"/>
  <target name="build-subprojects"/>
  <target depends="init" name="build-project">
    <echo message="${ant.project.name}: ${ant.file}"/>
  </target>
</project>
```

The right sidebar shows the Ant task list with the following structure:

- A_Corp
 - build [default] (selected)
 - build-eclipse-compiler
 - build-project
 - build-refprojects
 - build-subprojects
 - Calc (1)
 - Calc (3)
 - clean

The bottom console window shows the build output:

```
<terminated> A_Corp build.xml [Ant Build] C:\Program Files (x86)\Java\jdk1.6.0_37\bin\javaw.exe (2012. 12. 7. 오후 4:46:05)
Buildfile: C:\Eclipse\workspace\A_Corp\build.xml
build-subprojects:
init:
build-project:
[echo] A_Corp: C:\Eclipse\workspace\A_Corp\build.xml
build:
BUILD SUCCESSFUL
Total time: 391 milliseconds
```

6. 도구 활용 예제

6.4 Hudson설정 및 연계

Ant

- Hudson에서 Build설정 및 Build결과

Build Triggers

☐ Build after other projects are built

☐ Trigger builds remotely (e.g., from scripts)

☐ Poll SCM

☒ Build periodically

Schedule

```
#매일 4시 20분, 5시50분
20 16 * * *
50 17 * * *
```

Build

Invoke Ant

Ant Version

Targets

Build File

Properties

Hudson

새 작업
Hudson 관리
계정관리
설정기록

빌드 대기 목록
빌드 대기 항목 없습니다.

빌드 실행 상태

No.	상태
1	대기 중
2	대기 중

All +

S	W	작업	최근 성공	최근 실패	최근 소요 시간
		AntTest	40 min (#30)	11 days (#25)	1.9 sec
		Exam01	40 sec (#26)	13 days (#17)	1.6 sec
		WEB-Project	1 hr 10 min (#6)	1 hr 35 min (#4)	2.7 sec

이미콘: S M L

Legend for all for failures for success

7. FAQ

Ant

질문1) Ant의 약자는 무엇인가요? 단순히 재미에서 따온 것인가요?

➡ 답변1 : Ant는 Another Neat Tool의 약어입니다.

질문2) Ant가 개발된 동기나 배경은 어떠합니까?

➡ 답변1 : JSP/Servlet 표준구현 엔진(Tomcat으로 발전되기 전)을 오픈소스화 하던 중 make를 이용하여 작업하였으나, 다른 환경에서는 이를 수행할 수 있는 오픈소스 소프트웨어가 없었기 때문에 개발하게 되었습니다. 그리하여 Tomcat을 빌드하기 위한 툴에서 현재의 모습으로 발전하게 되었습니다.

8. 도구 평가

Ant

- 활용성
 - 빌드 자동화 도구 중에 최고의 활용성
- 범용성
 - 대부분의 자바 IDE도구에 기본 채택
- 호환성
 - JAVA환경에서 완벽하게 작동, (버전 별 최소 요구 자바버전이 존재, 1.5이상에서 모두 가능)
- 성능
 - 빌드 수행 능력에 있어서 일반적인 Build성능과 별다른 차이가 없다, 오히려 자동화로 효율적
- 기타
 - 다른 관리 도구와 연계 지원

도구평가 의견

- Ant는 Eclipse에서 기본 탑재되어 있으며, 자동화 빌드를 지원
- 자체 JUnit 테스트 기능을 지원
- XML을 수정하여 원하는 형태의 Build를 자동화 할 수 있으며, 명령어를 통해 플러그인 생성 및 테스트까지 완벽하게 지원

9. 용어 정리

Ant

본 매뉴얼에서 사용하고 있는 용어의 정리

JUnit

Java기반 테스트를 위한 프레임워크로, 단위모듈(ex: Method)이 정확히 구현되었는지를 확인할 수 있는 도구

단위테스트

단위테스트는 단위코드에서 문제발생소지가 있는 모든 부분을 테스트하는 작업

빌드 자동화

개발자가 개발한 소프트웨어의 개발과정중 Build과정을 자동화 한 것. Build시점을 정할 수 있으며, 시점에는 트리거 발생 시, 업데이트 시, 일단위(Daily Build)등이 존재