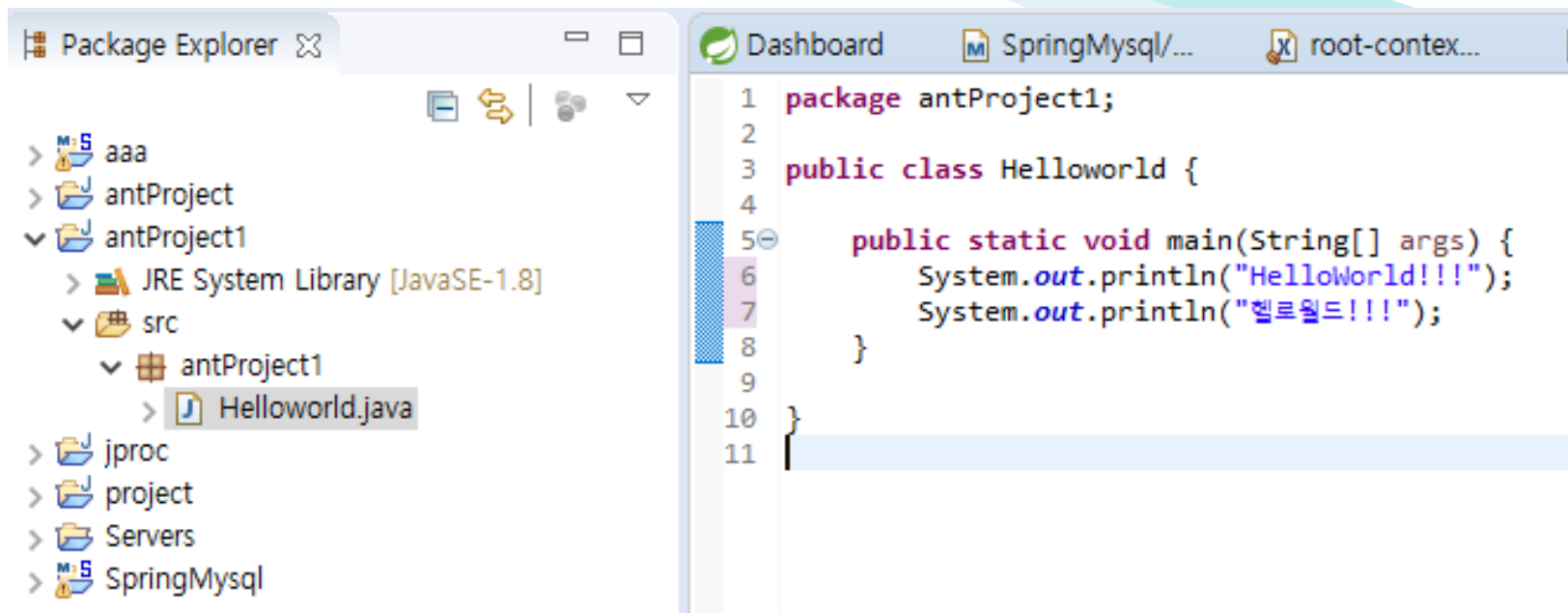


애플리케이션 빌드

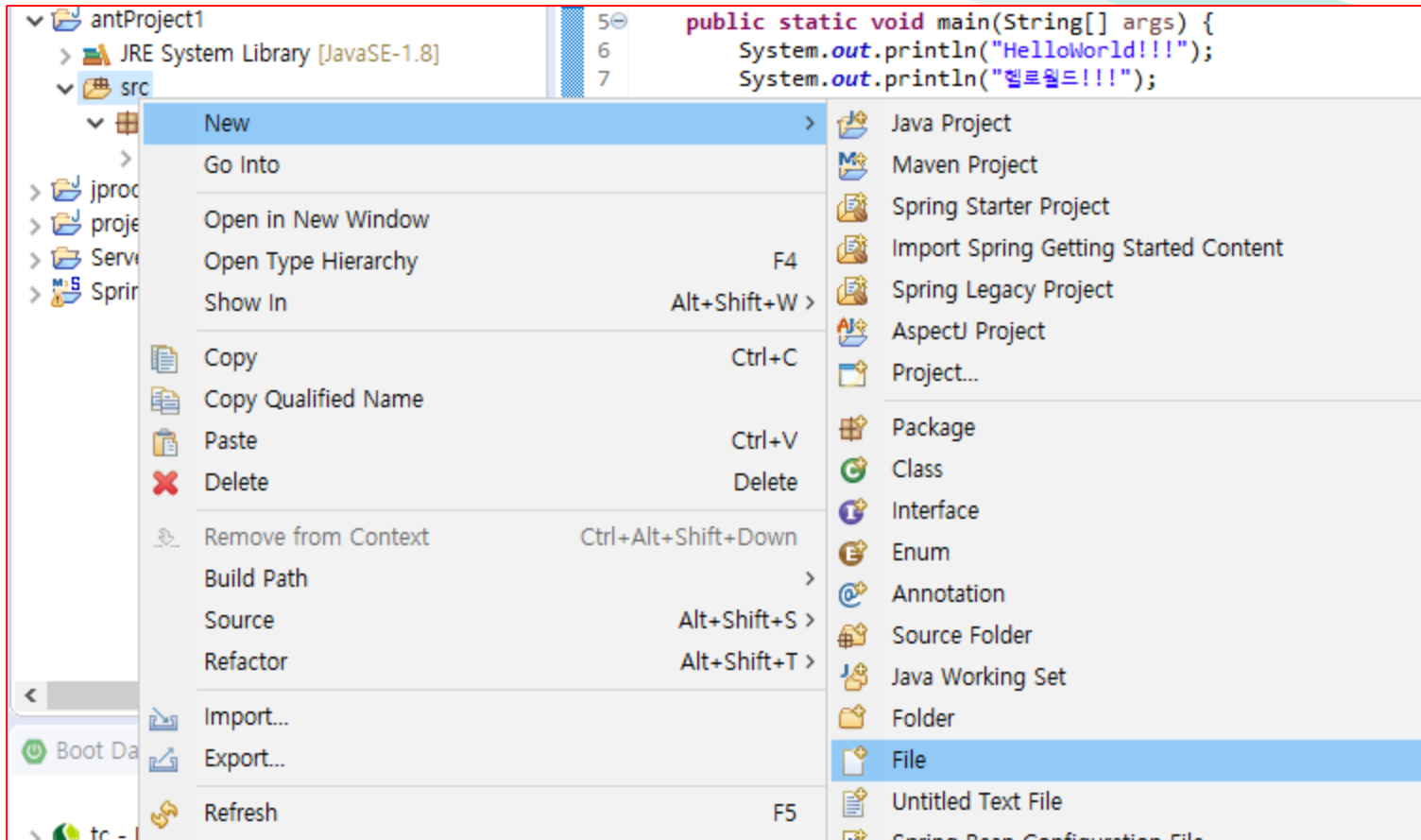
빌드

1. Ant 프로젝트 생성



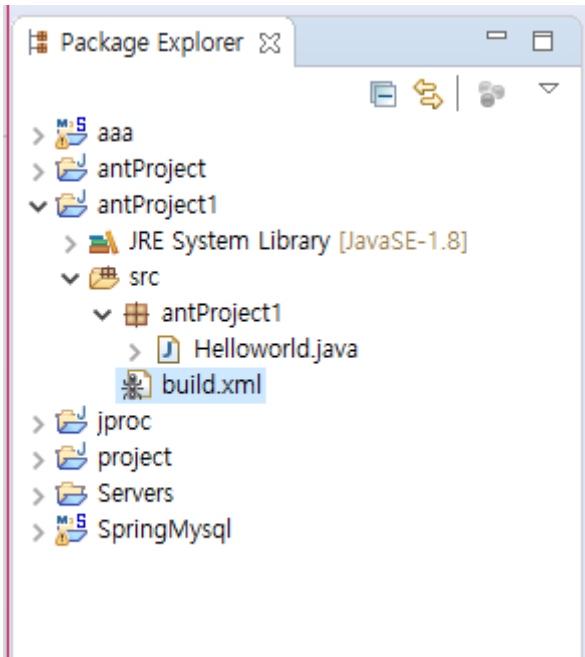
빌드

2. 프로젝트 오른쪽 마우스 → new → File에서 build.xml 생성



빌드

2. 프로젝트 오른쪽 마우스→new→File에서 build.xml 생성



빌드

2. 프로젝트 오른쪽 마우스→new→File에서 build.xml 생성

```
1  <?xml version="1.0"?>
2
3  <project name="antProject1" default="main" basedir=".">
4      <target name="main">
5          <javac srcdir="./src" destdir="./bin" includeantruntime="false">
6              </javac>
7          <java classpath="bin" classname="antProject1.Helloworld">
8              </java>
9      </target>
10 </project>
11
```

- ✓ project라는 대 단위가 있고 속성으로 **name**과 **default**, **basedir**있다.
- ✓ name은 프로젝트의 이름이다. 사실 크게 중요하진 않음
- ✓ default는 중요한데 이 빌드에 특정한 명령을 주지 않는다면 저 **default**까지 만드는걸 목표로 삼는다는 것이다.
- ✓ basedir은 빌드를 할 경로의 기준을 의미한다. **..**을 쓴다면 현재 **build.xml**폴더의 위치를 기준으로 삼는다는 것이다.
- ✓ 이제 target을 만들어 준다. project는 여러개의 target으로 이루어져있으며 빌드의 목표는 결국 target이다.
- ✓ 우리는 target의 목표를 main으로 잡았다. 결국 실행하면 main target이 실행되게 된다.

빌드

2. build.xml

```
1 <?xml version="1.0"?>
2
3 <project name="antProject1" default="main" basedir=".">
4   <target name="main">
5     <javac srcdir="./src" destdir="./bin" includeantruntime="false">
6     </javac>
7     <java classpath="bin" classname="antProject1.Helloworld">
8     </java>
9   </target>
10 </project>
11
```

- ✓ 해당 ant설정파일을 보자.
- ✓ default를 main으로 했으므로 target을 main을 찾아서 한다.
- ✓ target은 바로 하나의 작업이라고 생각하면 된다.
- ✓ target안에는 총 2개의 작업이 있다. 하나는 javac이고 하나는 java이다.

빌드

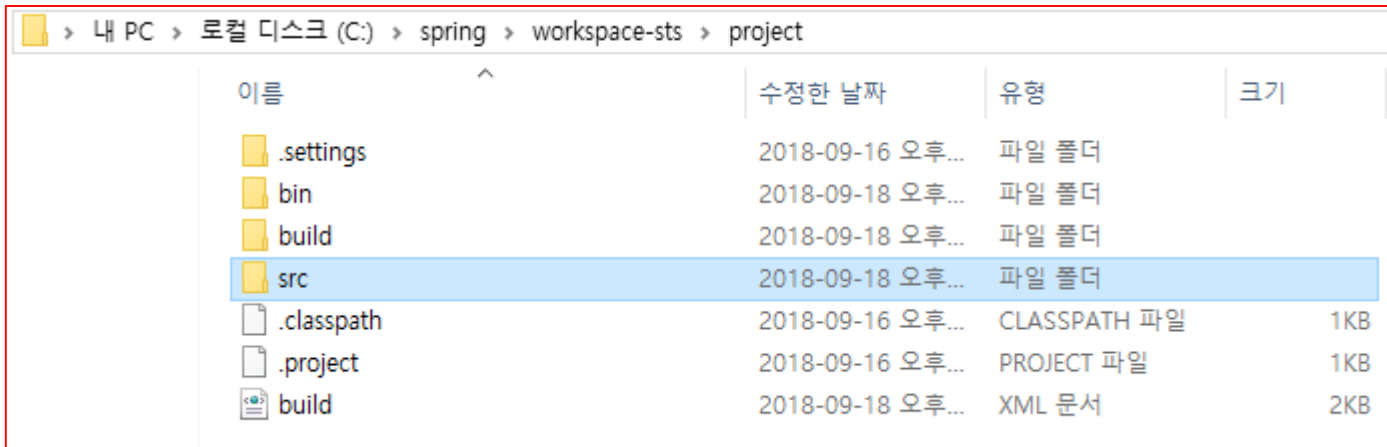
2. build.xml

- ✓ javac을 보면 srcdir과 destdir 두가지가 있다. 뒤의 includeantruntime은 그냥 false로 지정하고 사용하면 됨
- ✓ srcdir의 경우에는 빌드할 소스파일의 위치이다. 즉 java파일들의 위치
- ✓ destdir의 경우에는 빌드한 파일을 위치할 경로이다. 즉 class파일들이 저장될 위치
- ✓ javac으로 인해서 bin파일에는 class파일들이 위치하게 될것이다.
- ✓ 그중에서 main파일을 실행하려고 한다.
- ✓ main파일의 위치는 antProject1에 Helloworld클래스 내부에 존재하므로 풀 경로를 **classname**에 적어주고, 그 패키지가 들은 경로의 이름인 ./bin을 **classpath**에 적어준다.

빌드

2. build.xml

- ✓ javac을 보면 srcdir과 destdir 두가지가 있다. 뒤의 includeantruntime은 그냥 false로 지정하고 사용하면 됨
- ✓ srcdir의 경우에는 빌드할 소스파일의 위치이다. 즉 java파일들의 위치
- ✓ destdir의 경우에는 빌드한 파일을 위치할 경로이다. 즉 class파일들이 저장될 위치
- ✓ javac으로 인해서 bin파일에는 class파일들이 위치하게 될것이다.
- ✓ 그중에서 main파일을 실행하려고 한다.
- ✓ main파일의 위치는 antProject1에 Helloworld클래스 내부에 존재하므로 풀 경로를 **classname**에 적어주고, 그 패키지가 들은 경로의 이름인 ./bin을 **classpath**에 적어준다.

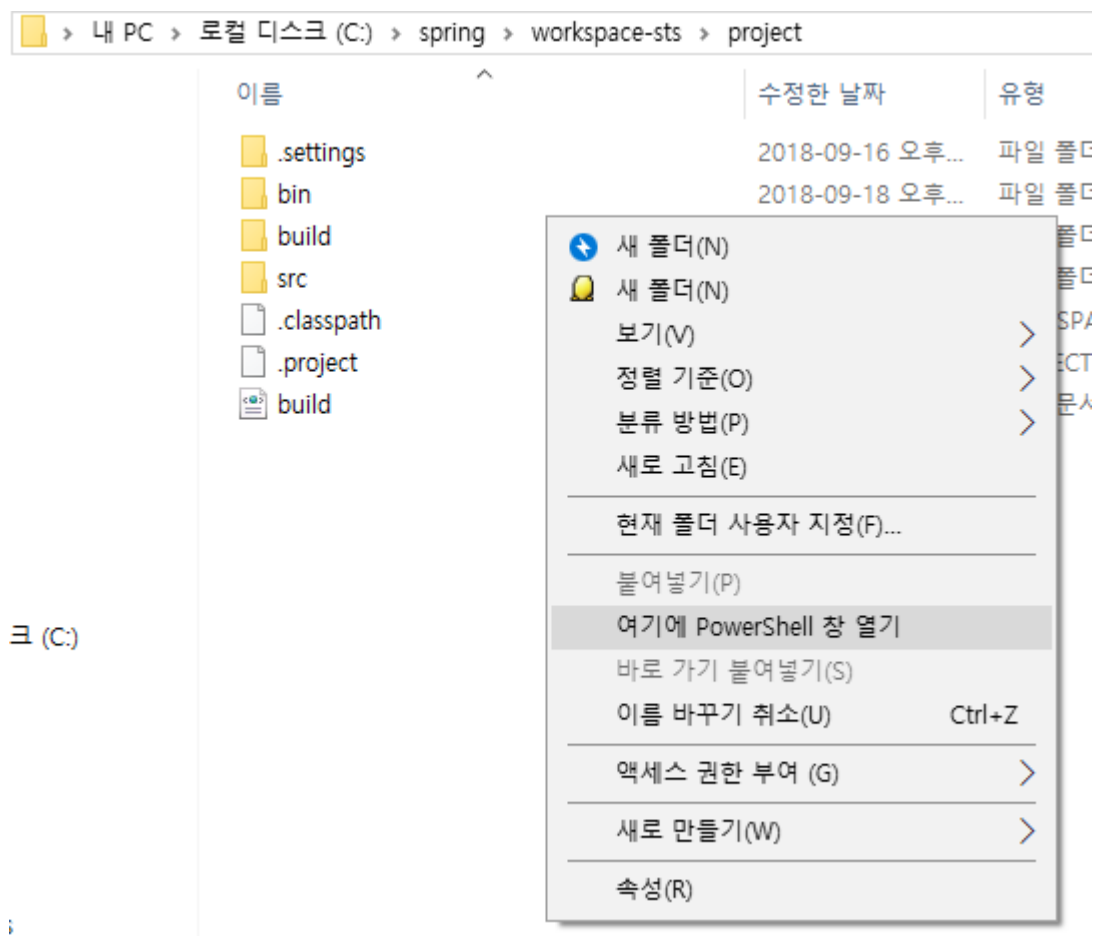


이름	수정한 날짜	유형	크기
.settings	2018-09-16 오후...	파일 폴더	
bin	2018-09-18 오후...	파일 폴더	
build	2018-09-18 오후...	파일 폴더	
src	2018-09-18 오후...	파일 폴더	
.classpath	2018-09-16 오후...	CLASSPATH 파일	1KB
.project	2018-09-16 오후...	PROJECT 파일	1KB
build	2018-09-18 오후...	XML 문서	2KB

빌드

2. build.xml

- ✓ shift+우클릭을 하면 해당경로에서 PowerShell을 열어보자



빌드

2. build.xml

✓ Ant 실행

```
Windows PowerShell
PS C:\spring\workspace-sts\project> ant
Buildfile: C:\spring\workspace-sts\project\build.xml

clean:
    [delete] Deleting directory C:\spring\workspace-sts\project\build

init:
    [mkdir] Created dir: C:\spring\workspace-sts\project\build
    [mkdir] Created dir: C:\spring\workspace-sts\project\build\classes

compile:
    [javac] Compiling 1 source file to C:\spring\workspace-sts\project\build\classes

jar:
    [jar] Building jar: C:\spring\workspace-sts\project\build\jar\anttest-1.0.0.jar

main:

BUILD SUCCESSFUL
Total time: 2 seconds
PS C:\spring\workspace-sts\project>
```

빌드

2. build.xml

- ✓ Ant 실행
- ✓ 원래는 ant main같은 형식으로 사용해야 한다.
- ✓ 원래는 **ant target명** 같은 형식으로 사용
- ✓ 그러나 ant main => ant같은 형식으로 축약할 수 있었던 것은 default로 main을 설정해놨기 때문
- ✓ 이는 보통 다른 ant빌드를 썼던 프로젝트도 마찬가지이다.
- ✓ 가장 중요한 target의 경우를 default로 지정해주는 것이 관례이다. 따라서 프로젝트를 열어봤는데 ant 시스템을 사용중이라면
- ✓ 보통은 ant라는 명령어 하나만으로 빌드를 할 수 있다

```
PS C:\spring\workspace-sts\project> ant main
Buildfile: C:\spring\workspace-sts\project\build.xml

clean:
[delete] Deleting directory C:\spring\workspace-sts\project\build

init:
[mkdir] Created dir: C:\spring\workspace-sts\project\build
[mkdir] Created dir: C:\spring\workspace-sts\project\build\classes

compile:
[javac] Compiling 1 source file to C:\spring\workspace-sts\project\build\classes

jar:
[jar] Building jar: C:\spring\workspace-sts\project\build\jar\anttest-1.0.0.jar

main:

BUILD SUCCESSFUL
Total time: 2 seconds
PS C:\spring\workspace-sts\project>
```

빌드

2. build.xml

- ✓ 설정파일을 3개로 분할
- ✓ 각각 init, compile, main으로 분할

```
1 <?xml version="1.0"?>
2
3 <project name="antProject1" default="main" basedir=".">
4   <target name="init">
5     <mkdir dir="./bin" />
6   </target>
7
8   <target name="compile" depends="init">
9     <javac srcdir="./src" destdir="./bin" includeantruntime="false">
10
11     </javac>
12   </target>
13
14   <target name="main" depends="compile">
15     <java classpath="./bin" classname="antProject1.Helloworld">
16
17     </java>
18   </target>
19 </project>
```

빌드

2. build.xml

- ✓ 먼저 init의 경우에는 compile하기 전에 선행해서 하는 작업들
- ✓ 예를 들자면 파일을 만드는 작업들이다.
- ✓ mkdir을 사용해서 bin파일을 만들었다

```
Windows PowerShell
PS C:\spring\workspace-sts\project> ant
Buildfile: C:\spring\workspace-sts\project\build.xml

clean:
[delete] Deleting directory C:\spring\workspace-sts\project\build

init:
[mkdir] Created dir: C:\spring\workspace-sts\project\build
[mkdir] Created dir: C:\spring\workspace-sts\project\build\classes

compile:
[javac] Compiling 1 source file to C:\spring\workspace-sts\project\build\classes

jar:
[jar] Building jar: C:\spring\workspace-sts\project\build\jar\anttest-1.0.0.jar

main:

BUILD SUCCESSFUL
Total time: 2 seconds
PS C:\spring\workspace-sts\project> █
```

빌드

2. build.xml

- ✓ compile의 경우 java파일을 javac을 사용해서 class파일로 바꾸는 작업이다.
- ✓ 해당 파일들을 통째로 묶어서 compile이라는 target으로 만든다.
- ✓ compile target속성에 depends라는 속성이 있다.
- ✓ 이 depends 역할은 이 target 작업을 하기 전에 앞의 작업이 선행되어야 한다는 것을 뜻한다.
- ✓ 위의 경우 compile을 실행하기 전에 init이 실행되어야 한다는 것
- ✓ 만약 내가 compile을 호출할 경우 자동으로 init이 먼저 실행되고 그 다음에 compile이 실행된다는 이야기이다.
- ✓ 이런식으로 target을 쪼개도 상관없는 이유는 depends 속성을 이용해서 꼬리에 꼬리로 타고넘어가서 같이 빌드가 된다.
- ✓
- ✓ 마지막으로 main에서 java코드를 실행하게 된다.
- ✓ 이 main역시 compile target을 필요로 하므로 main을 실행하려면 먼저 compile이 선행되어야 하고
- ✓ compile을 실행하려면 먼저 init이 선행되어야 한다

빌드

2. build.xml

- ✓ 빌드를 하면 main을 실행하고 main을 위해서 compile이 실행되고 compile을 위해서 init이 실행된다.
- ✓ 결국에 타고 타고 올라가게 된다.
- ✓ 그러면 compile을 명시적으로 빌드하면 어떻게 될까?

```
PS C:\spring\workspace-sts\project> ant compile
Buildfile: C:\spring\workspace-sts\project\build.xml

init:

compile:

BUILD SUCCESSFUL
Total time: 0 seconds
PS C:\spring\workspace-sts\project> .
```

빌드

2. build.xml

- ✓ clean target을 만들어 보자.
- ✓ uninstall로직의 target이름을 clean이라고 정하는 경우가 많다.
- ✓ 이런 clean target은 내가 빌드 시스템을 이용해서 빌드한 파일을 모조리 삭제할 수 있다.

```
1 <?xml version="1.0"?>
2
3 <project name="antProject1" default="main" basedir=".">
4   <target name="init">
5     <mkdir dir="./bin" />
6   </target>
7
8   <target name="compile" depends="init">
9     <javac srcdir="./src" destdir="./bin" includeantruntime="false">
10    </javac>
11  </target>
12
13  <target name="main" depends="compile">
14    <java classpath="./bin" classname="antProject1.Helloworld">
15    </java>
16  </target>
17  <target name="clean" >
18    <delete dir="./bin"/>
19  </target>
20 </project>
```


빌드

2. build.xml

- ✓ 이 설정파일은 무슨 문제가 있을까?
 - ✓ 바로 모든 값들이 변수가 아니라 절대 값으로 되어있다.
 - ✓ 이러한 설정파일은 유연하게 대처할 수 없다.
 - ✓ 예를들어서 java파일이 들어있는 src파일의 경로가 이동했다고 치자.
 - ✓ main/src로 이동했거나 이름이 source로 바뀌었거나. 그럴 경우 모든 변수를 일일이 찾아서 다 바꿔줘야 한다.
 - ✓ 이러한 경로를 변수로 사용할 수 있다면 당연하지만 매우 좋을것이다.
-
- ✓ 그걸 위해 존재하는 것이 property이다. 물론 property만 변수처럼 사용할 수 있는 것은 아니다.
 - ✓ 그러나 property는 변수로 사용할수 있는 가장 기본적인 녀석이다.

2. build.xml

```
1 <project name="antProject1" default="main" basedir=". ">
2   <property name="name" value="Test" />
3   <property name="version" value="1.0.0" />
4   <property name="src.dir" value="${basedir}/src" />
5   <property name="build.dir" value="${basedir}/bin" />
6
7   <target name="init">
8     <mkdir dir="${build.dir}" />
9   </target>
10
11  <target name="compile" depends="init">
12    <javac srcdir="${src.dir}" destdir="${build.dir}" includeantruntime="false">
13    </javac>
14  </target>
15
16  <target name="main" depends="compile">
17    <java classpath="${build.dir}" classname="antProject1.Helloworld">
18    </java>
19  </target>
20
21  <target name="clean">
22    <delete dir="${build.dir}" />
23  </target>
24 </project>
```

빌드

2. build.xml

- ✓ 해당 프로젝트는 property로 변수를 지정해서 사용한 예제이다.
- ✓ property를 총 4개를 지정했다.
- ✓ 보통의 경우 버전과 앱의 이름을 property로 지정한다.
- ✓ 달러사인(\$)을 볼수 있을 것이다.
- ✓ 이 \${변수}의 경우 변수의 값을 호출한다는 것

```
PS C:\spring\workspace-sts\project> ant
Buildfile: C:\spring\workspace-sts\project\build.xml

clean:

init:
    [mkdir] Created dir: C:\spring\workspace-sts\project\build
    [mkdir] Created dir: C:\spring\workspace-sts\project\build\classes

compile:
    [javac] Compiling 1 source file to C:\spring\workspace-sts\project\build\classes

jar:
    [jar] Building jar: C:\spring\workspace-sts\project\build\jar\anttest-1.0.0.jar

main:

BUILD SUCCESSFUL
Total time: 2 seconds
PS C:\spring\workspace-sts\project>
```

빌드

3. Ant프로젝트의 기본

1) property

(1)Name : 프로젝트의 이름을 기술한다. 외부적으로 쓰이는 일은 드물지만 이 프로젝트의 이름을 의미한다. 가장 자주 쓰이는 이름은 ant.project.name이다.

(2)name : 외부 파일을 만들때 사용할 프로젝트의 이름을 기술한다. jar파일을 만들 때 이 이름을 참조한다. 요새는 maven에서는 artifactId라고 기술되는 존재이다.

(3)groupid : 이 프로젝트가 전체적으로 가질 루트 패키지 명을 의미한다. 패키지명을 참조할 때 이 이름을 참조한다.

(4)project.version : 프로젝트의 버전을 의미한다. 간략하게 version이라고 쓸 때도 많다. ant에서 가장 자주 사용하는 방식은 ant.version이다.

(5)src.dir : 소스파일의 경로를 의미한다. 특별한 일이 있지 않는 한(있는 경우 못봄) src로 적는다.

(6)build.dir : 빌드 된 파일의 경로를 의미한다. 대부분의 케이스에서는 build라는 이름을 사용한다.

(7)classes.dir : 빌드 된 클래스파일의 경로를 의미한다. 대부분 build의 하위 폴더로서 classes 혹은 bin을 사용하나 대부분에서는 classes를 사용한다.

빌드

3. Ant프로젝트의 기본

2) target:

- (1)clean : ant로 생성된 파일을 다시 깔끔히 지운다.
- (2)init : 컴파일을 하기전에 해야할 선행작업들을 해둔다.
- (3)compile : javac를 사용해서 모든 소스파일을 컴파일한다.
- (4)main : 내가 메인으로 할 작업을 한다. 보통의 경우에는 jar파일 생성, war파일 생성, main실행 등의 역할을 하게된다.

3. build.xml

```
1 <?xml version="1.0"?>
2 <project name="antProject2" default="main" basedir=".">
3   <property name="Name" value="Ant Test"></property>
4   <property name="name" value="anttest"></property>
5   <property name="groupid" value="antProject2"></property>
6   <property name="project.version" value="1.0.0"></property>
7
8   <property name="src.dir" value="src"></property>
9   <property name="build.dir" value="build"></property>
10  <property name="classes.dir" value="${build.dir}/classes"></property>
11  <property name="jar.dir" value="${build.dir}/jar"></property>
12
13  <target name="clean">
14    <delete dir="${build.dir}"></delete>
15  </target>
16  <target name="init">
17    <mkdir dir="${build.dir}"></mkdir>
18    <mkdir dir="${classes.dir}"></mkdir>
19  </target>
20  <target name="compile" depends="init">
21    <javac srcdir="${src.dir}" destdir="${classes.dir}" includeantruntime="false">
22    </javac>
23  </target>
24  <target name="jar" depends="compile">
25    <jar destfile="${jar.dir}/${name}-${project.version}.jar" basedir="${classes.dir}">
26      <manifest>
27        <attribute name="Main-Class" value="antProject2.Helloworld"></attribute>
28      </manifest>
29    </jar>
30  </target>
31  <target name="run" depends="compile">
32    <java jar="${jar.dir}/${name}-${project.version}.jar" fork="true">
33    </java>
34  </target>
35  <target name="main" depends="clean,jar">
36  </target>
37 </project>
```

3. build.xml

External Tools Configurations
Create, manage, and run configurations
Run an Ant build file.

Name: antProject1 build.xml

Check targets to execute:

Name	Description
<input checked="" type="checkbox"/> clean	
<input checked="" type="checkbox"/> init	
<input checked="" type="checkbox"/> compile	
<input checked="" type="checkbox"/> jar	
<input checked="" type="checkbox"/> run	
<input checked="" type="checkbox"/> main [...]	

6 out of 6 selected
☐ Sort targets
☐ Hide internal targets not selected for execution

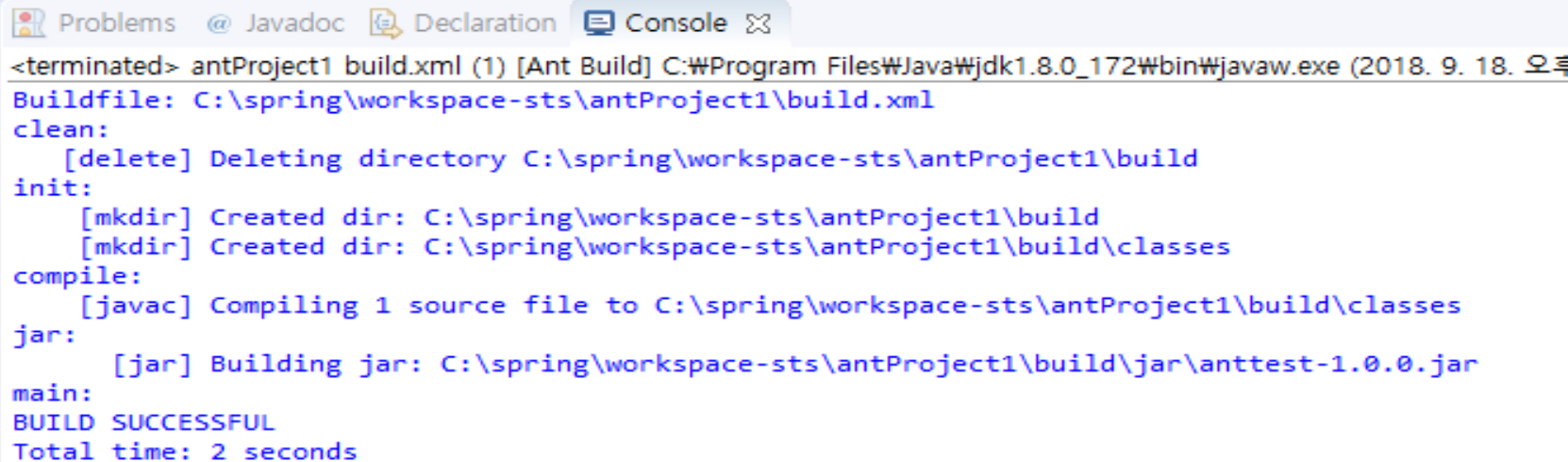
Target execution order:
main, clean, init, compile, jar, run

Filter matched 11 of 11 items

Buttons: Run, Close, Revert, Apply, Order...

빌드

4. 실행



The screenshot shows an IDE console window with the following output:

```
<terminated> antProject1 build.xml (1) [Ant Build] C:\Program Files\Java\jdk1.8.0_172\bin\javaw.exe (2018. 9. 18. 오후 10:00:00)
Buildfile: C:\spring\workspace-sts\antProject1\build.xml
clean:
  [delete] Deleting directory C:\spring\workspace-sts\antProject1\build
init:
  [mkdir] Created dir: C:\spring\workspace-sts\antProject1\build
  [mkdir] Created dir: C:\spring\workspace-sts\antProject1\build\classes
compile:
  [javac] Compiling 1 source file to C:\spring\workspace-sts\antProject1\build\classes
jar:
  [jar] Building jar: C:\spring\workspace-sts\antProject1\build\jar\anttest-1.0.0.jar
main:
BUILD SUCCESSFUL
Total time: 2 seconds
```

