



May 2019, IPT Course
Java Web Debelopment

Build Tools Basics

Trayan Iliev

tiliev@ipproduct.org

<http://ipproduct.org>

Copyright © 2003-2019 IPT - Intellectual
Products & Technologies

About me



Trayan Iliev

- CEO of IPT – Intellectual Products & Technologies
- Oracle® certified programmer 15+ Y
- end-to-end reactive fullstack apps with Java, ES6/7, TypeScript, Angular, React and Vue.js
- 12+ years IT trainer
- Voxxed Days, jPrime, jProfessionals, BGOUG, BGJUG, DEV.BG speaker
- Organizer RoboLearn hackathons and IoT enthusiast (<http://robolearn.org>)

Where to Find the Code?

Java Web Development projects and examples are available @ GitHub:

<https://github.com/iproduct/course-java-web-development>

Agenda for This Session

- ❖ Ant
- ❖ Maven
- ❖ Gradle
- ❖ Practical examples

Apache Ant

- ❖ Apache Ant is a software tool for automating software build processes, which originated from the Apache Tomcat project in early 2000.
- ❖ Replacement for the Make build tool of Unix, created due to a number of problems with Unix's make.
- ❖ Similar to Make but is implemented using the Java language, requires the Java platform, and is best suited to building Java projects.
- ❖ With Make, actions required to create a target are specified as shell commands which are specific to the platform on which runs.
- ❖ Ant solves this problem by providing a large amount of built-in functionality designed to behave similarly on all platforms.

Apache Ant - I

```
<?xml version="1.0"?>

<project name="Invoicing with Views" default="compile">

<presetdef name="javac"><javac includeantruntime="false" /></presetdef>

<property name="build.dir" location="${basedir}/bin"/>

    <target name="clean" description="remove intermediate files">

        <delete dir="${build.dir}"/>
    </target>

    <target name="clobber" depends="clean" description="remove all artifact files">

        <delete file="invoicing.jar"/>
    </target>

    <target name="compile" description="compile the Java source code to class files">

        <mkdir dir="${build.dir}"/>

        <javac srcdir="." destdir="${build.dir}"/>
    </target>
```

Apache Ant - II

...

```
<target name="jar" depends="compile" description="create a Jar file for the app">
  <jar destfile="invoicing.jar">
    <fileset dir="${build.dir}" includes="**/*.class"/>
    <manifest>
      <attribute name="Main-Class" value="invoicing.view.MainMenu"/>
    </manifest>
  </jar>
</target>
<target name="run" depends="compile" description="run the application">
  <java classname="invoicing.view.MainMenu"
    classpath="${java.class.path};${build.dir}" dir="." fork="true" />
</target>
<target name="runJar" depends="jar" description="run the app from the jar file">
  <java jar="invoicing.jar" dir="." failonerror="true" fork="true" />
</target>
</project>
```

Apache Ant Sample Tasks

```
<java classname="test.Main">
  <arg value="-h"/>
  <classpath>
    <pathelement location="dist/test.jar"/>
    <pathelement path="${java.class.path}"/>
  </classpath>
</java>

<java dir="${exec.dir}"
  jar="${exec.dir}/dist/test.jar"
  fork="true"
  failonerror="true"
  maxmemory="128m">
  <arg value="-h"/>
  <classpath>
    <pathelement location="dist/test.jar"/>
    <pathelement path="${java.class.path}"/>
  </classpath>
</java>

<junit printsummary="yes" haltonfailure="yes">
  <classpath>
    <pathelement location="${build.tests}"/>
    <pathelement path="${java.class.path}"/>
  </classpath>
  <formatter type="plain"/>
  <test name="my.test.TestCase"
    haltonfailure="no" outfile="result">
    <formatter type="xml"/>
  </test>
  <batchtest fork="yes" todir="${reports.tests}">
    <fileset dir="${src.tests}">
      <include name="**/*Test*.java"/>
      <exclude name="**/AllTests.java"/>
    </fileset>
  </batchtest>
</junit>
```


Maven Dependency Management

- ❖ Apache Maven – <https://spring.io/guides/gs/maven/>
- ❖ Common arguments: **mvn compile**, **mvn package**, **mvn install**, **mvn clean** **deploy** **site-deploy**
- ❖ Example configuration:

```
<?xml version="1.0" encoding="UTF-8" ?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>org.iproduct.spring</groupId>
  <artifactId>01-introduction-maven</artifactId>
  <version>1.0-SNAPSHOT</version>
```

Maven Configuration (continued)

```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>5.0.5.RELEASE</version>
  </dependency>
</dependencies>

<repositories>
  <repository>
    <id>io.spring.repo.maven.release</id>
    <url>http://repo.spring.io/release/</url>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
  </repository>
</repositories>
```

Maven Configuration (continued)

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <configuration>
        <source>9</source>
        <target>9</target>
      </configuration>
    </plugin>
  </plugins>
</build>

</project>
```

Maven Configuration (enhanced)

```
<dependencyManagement>
  <dependencies>
    <dependency>
      <groupId>org.springframework</groupId>
      <artifactId>spring-framework-bom</artifactId>
      <version>5.0.5.RELEASE</version>
      <type>pom</type>
      <scope>import</scope>
    </dependency>
  </dependencies>
</dependencyManagement>

<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
  </dependency>
</dependencies>
```

Gradle Dependency Management

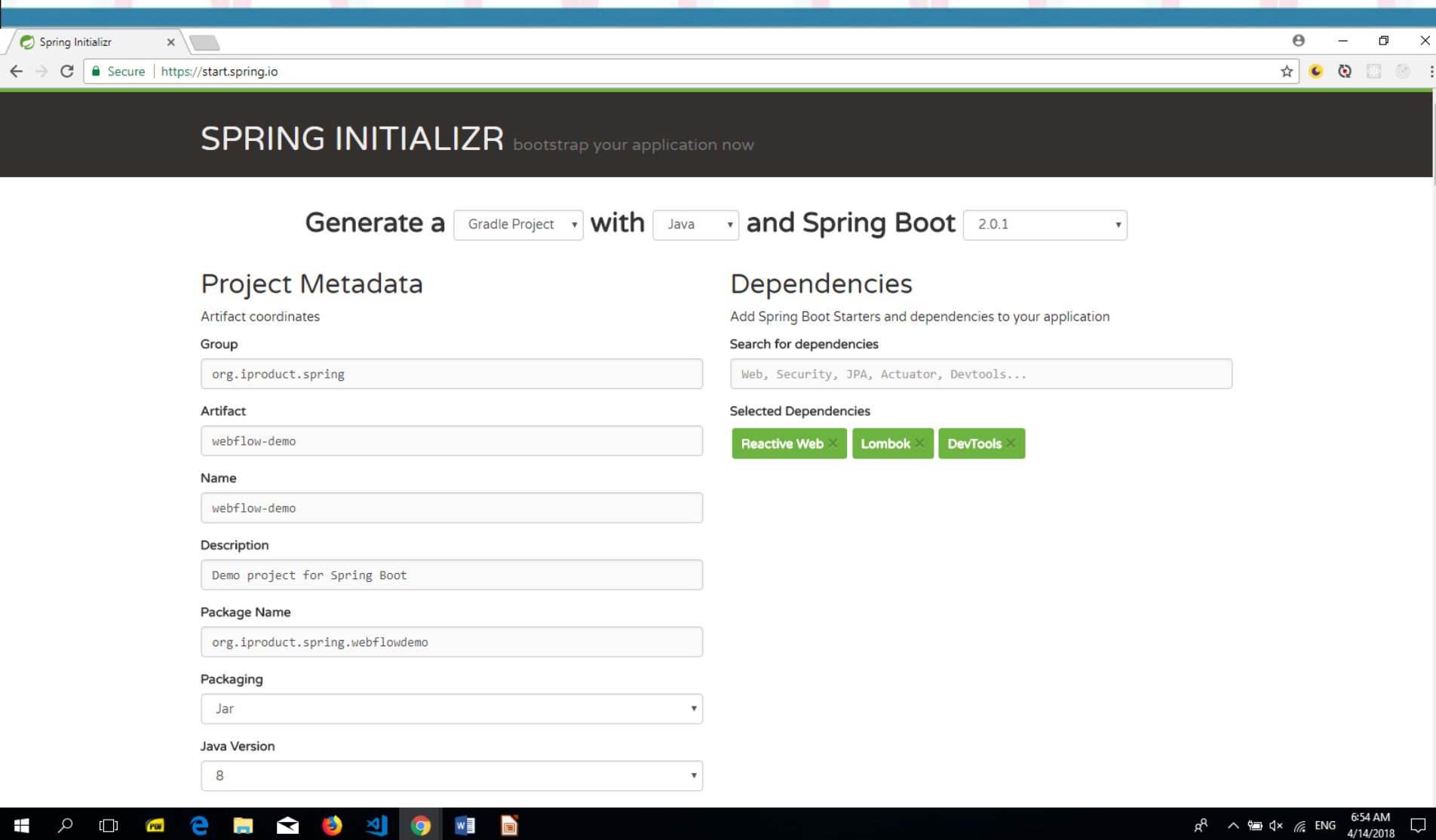
- ❖ Gradle – <https://spring.io/guides/gs/gradle/>
- ❖ Init new project/ convert existing from Maven: **gradle init**
- ❖ Build project: **gradle build**
- ❖ Build project: **gradle run**
- ❖ Example configuration:

```
group 'org.iproduct.spring'  
version '1.0-SNAPSHOT'  
apply plugin: 'java'  
apply plugin: 'application'  
mainClassName =  
    'org.iproduct.spring.demo.xmlconfig.HelloWorldSpringDI'  
  
sourceCompatibility = 1.8
```


Gradle Configuration (continued)

```
task runApp(type: JavaExec) {  
    classpath = sourceSets.main.runtimeClasspath  
    main =  
'org.iproduct.spring.demo.xmlconfig.HelloWorldSpringDI'  
}  
  
repositories {  
    mavenLocal()  
    mavenCentral()  
    maven { url "https://repo.spring.io/snapshot" }  
    maven { url "https://repo.spring.io/milestone" }  
}  
  
dependencies {  
    compile group: 'org.springframework',  
           name: 'spring-context', version: '5.0.5.RELEASE'  
    testCompile group: 'junit', name: 'junit', version: '4.12'  
}
```

Making Projects Easy: Spring Boot 2



The screenshot shows the Spring Initializr web application in a browser window. The browser's address bar shows the URL <https://start.spring.io>. The page has a dark header with the text "SPRING INITIALIZR" and the tagline "bootstrap your application now". Below the header, there is a form to generate a project. The form is divided into two main sections: "Project Metadata" and "Dependencies".

Generate a Gradle Project **with** Java **and Spring Boot** 2.0.1

Project Metadata
Artifact coordinates

Group

Artifact

Name

Description

Package Name

Packaging
Jar

Java Version
8

Dependencies
Add Spring Boot Starters and dependencies to your application

Search for dependencies

Selected Dependencies
Reactive Web Lombok DevTools

Thank's for Your Attention!



Trayan Iliev

**CEO of IPT – Intellectual Products
& Technologies**

<http://iproduct.org/>

<http://robolearn.org/>

<https://github.com/iproduct>

<https://twitter.com/trayaniliev>

<https://www.facebook.com/IPT.EACAD>

<https://plus.google.com/+IproductOrg>