



บทที่ 9

Spring Boot



ธีระยุทธ ทองเครือ

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยขอนแก่น

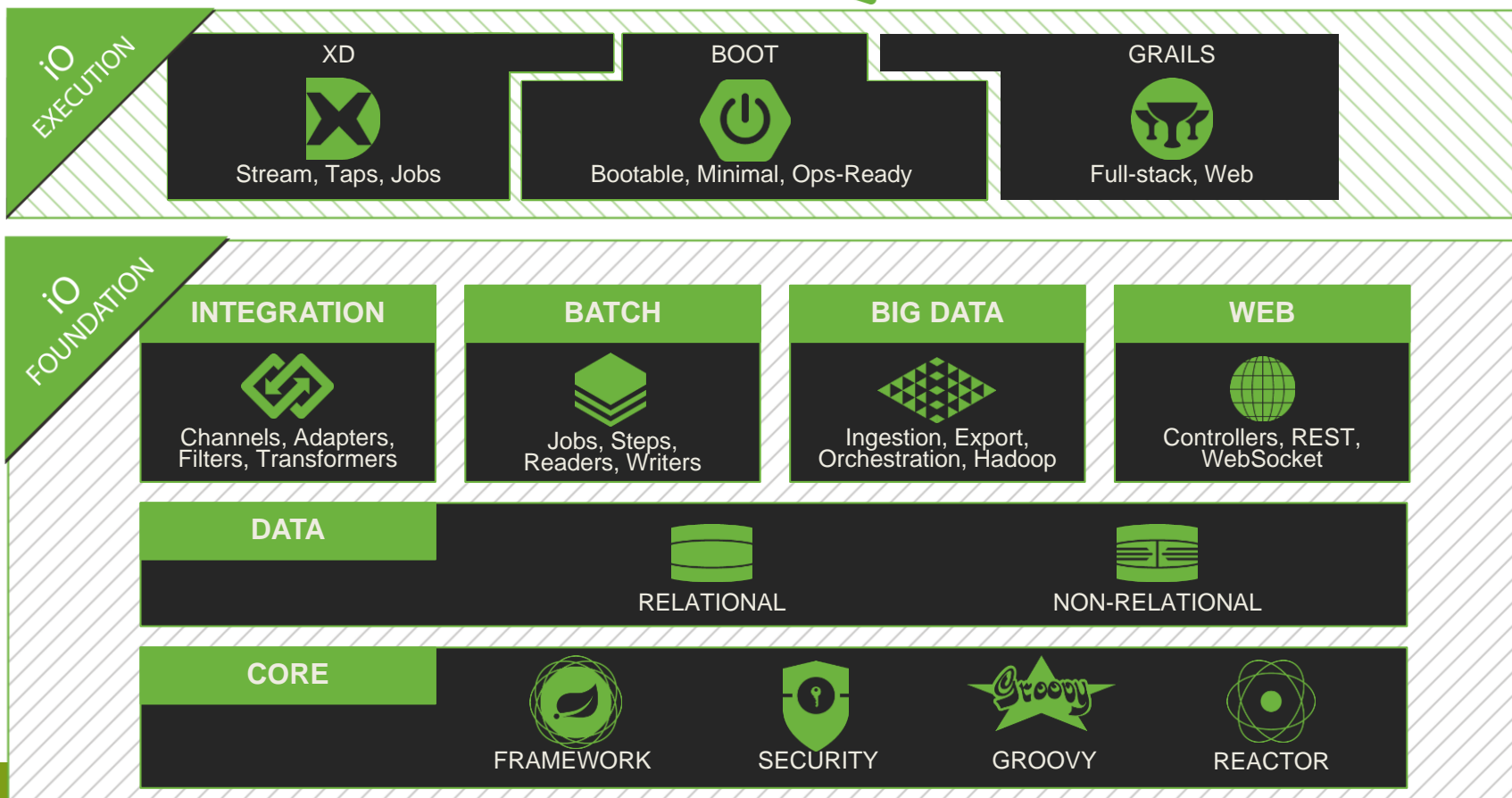




Spring Framework



- ❖ Spring Framework คือ กรอบงานที่ใช้ในการพัฒนาจาวาแอปพลิเคชัน และช่วยในการวางโครงสร้างพื้นฐาน (infrastructure) ของแอปพลิเคชัน





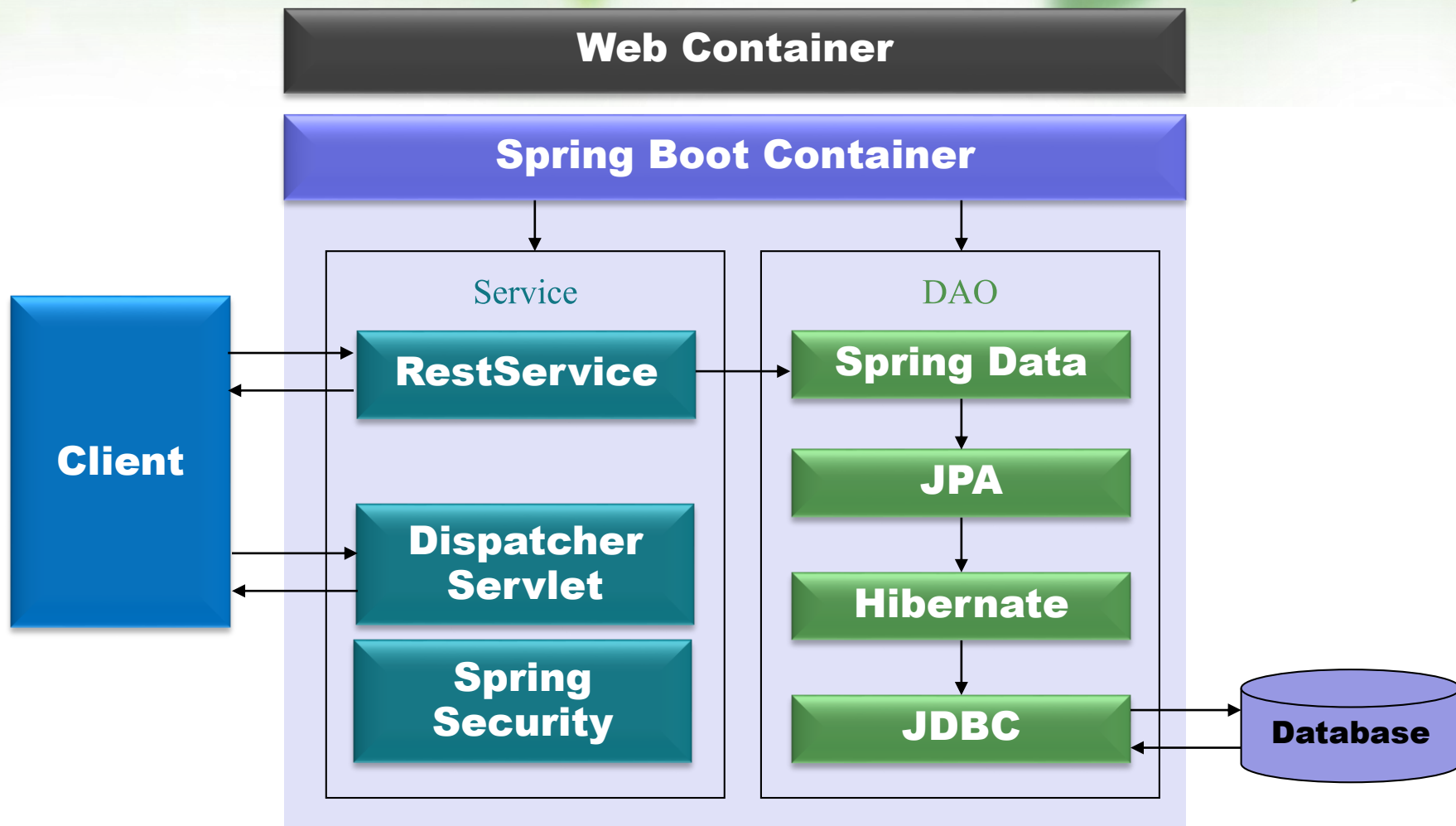
Spring Boot

- ❖ Spring Boot คือ เครื่องมือสำหรับเริ่มต้นพัฒนาแอปพลิเคชันด้วย Spring Framework อย่างรวดเร็ว มีชุด Starter Library ให้เรียกใช้งานได้สะดวก ลดขั้นตอนการเพิ่ม library อื่นๆ ที่เกี่ยวข้อง
- ❖ มี Auto Configuration ซึ่งช่วยลดการกำหนดค่าที่ยุ่งยากใน Spring Framework
- ❖ มี Built-in web server เช่น Tomcat, Jetty, Undertow ทำให้สามารถ start ตัวเอง ขึ้นมาได้ง่าย โดยไม่ขึ้นกับ Server ใดๆ ช่วยให้การสร้าง Web Application, Web API, Micro Service ง่ายขึ้น





Spring Boot Architecture

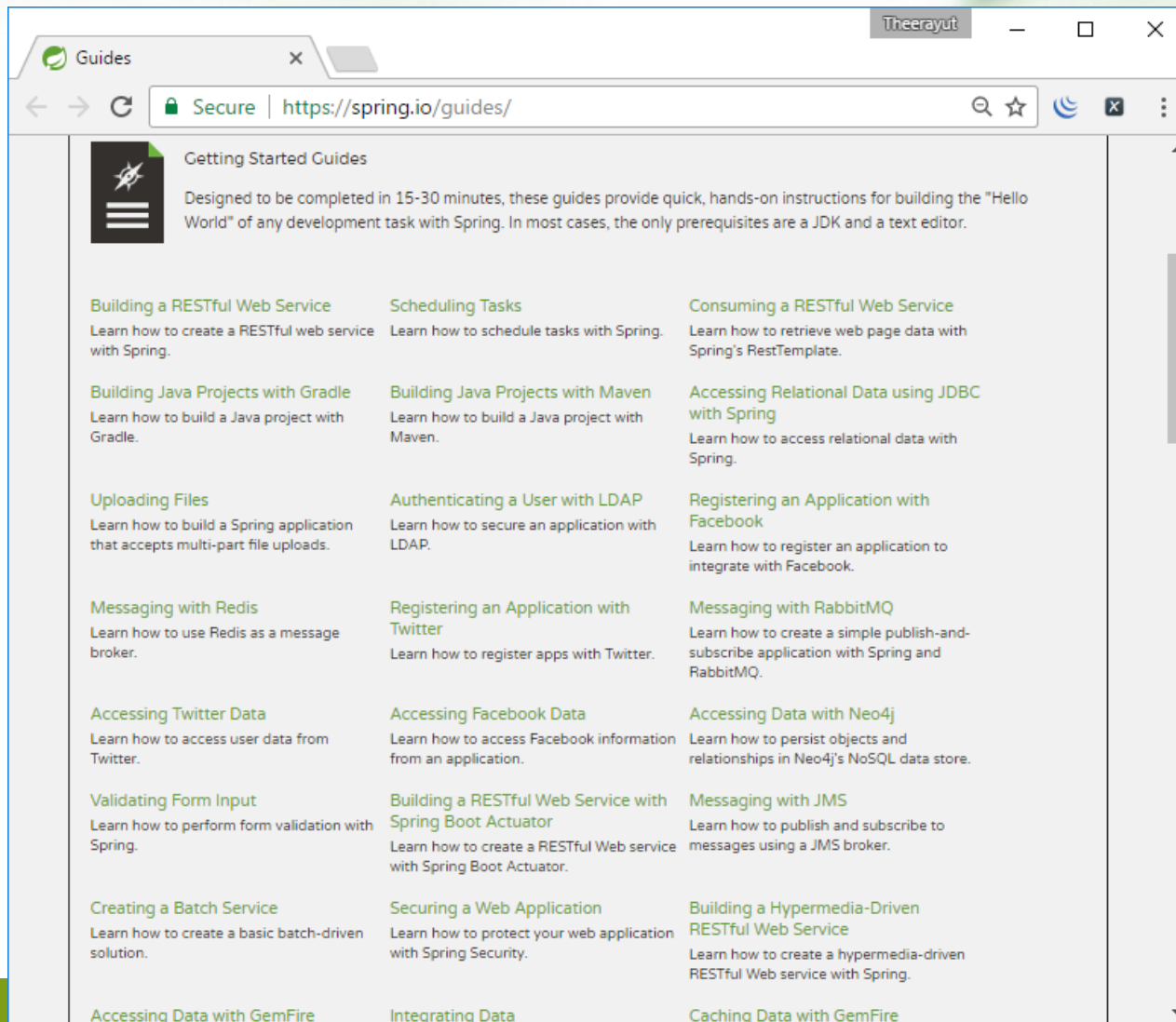




Spring Boot ทำอะไรได้บ้าง



ตัวอย่างโค้ดและคำอธิบาย Spring Boot ในเคสต่างๆ -> <https://spring.io/guides>



The screenshot shows a web browser window with the URL <https://spring.io/guides/>. The page is titled "Getting Started Guides" and features a grid of 18 guides. Each guide includes a title, a brief description, and a small icon. The guides are organized into three columns and six rows.

Getting Started Guides		
Designed to be completed in 15-30 minutes, these guides provide quick, hands-on instructions for building the "Hello World" of any development task with Spring. In most cases, the only prerequisites are a JDK and a text editor.		
Building a RESTful Web Service Learn how to create a RESTful web service with Spring.	Scheduling Tasks Learn how to schedule tasks with Spring.	Consuming a RESTful Web Service Learn how to retrieve web page data with Spring's RestTemplate.
Building Java Projects with Gradle Learn how to build a Java project with Gradle.	Building Java Projects with Maven Learn how to build a Java project with Maven.	Accessing Relational Data using JDBC with Spring Learn how to access relational data with Spring.
Uploading Files Learn how to build a Spring application that accepts multi-part file uploads.	Authenticating a User with LDAP Learn how to secure an application with LDAP.	Registering an Application with Facebook Learn how to register an application to integrate with Facebook.
Messaging with Redis Learn how to use Redis as a message broker.	Registering an Application with Twitter Learn how to register apps with Twitter.	Messaging with RabbitMQ Learn how to create a simple publish-and-subscribe application with Spring and RabbitMQ.
Accessing Twitter Data Learn how to access user data from Twitter.	Accessing Facebook Data Learn how to access Facebook information from an application.	Accessing Data with Neo4j Learn how to persist objects and relationships in Neo4j's NoSQL data store.
Validating Form Input Learn how to perform form validation with Spring.	Building a RESTful Web Service with Spring Boot Actuator Learn how to create a RESTful Web service with Spring Boot Actuator.	Messaging with JMS Learn how to publish and subscribe to messages using a JMS broker.
Creating a Batch Service Learn how to create a basic batch-driven solution.	Securing a Web Application Learn how to protect your web application with Spring Security.	Building a Hypermedia-Driven RESTful Web Service Learn how to create a hypermedia-driven RESTful Web service with Spring.
Accessing Data with GemFire	Integrating Data	Caching Data with GemFire



Spring Boot application starters



Spring Boot มีชุดแม่แบบ หรือ Starter พร้อมใช้ แบ่งตามประเภทของแอปพลิเคชัน มีชื่อ artifactId ขึ้นต้นด้วย spring-boot-starter-* สามารถนำชื่อ Dependency ไประบุในไฟล์ POM ของโปรเจก Maven ซึ่งจะได้ชุด Library .jar ต่างๆ ที่เกี่ยวข้องมาเก็บไว้ในโปรเจกแบบอัตโนมัติ

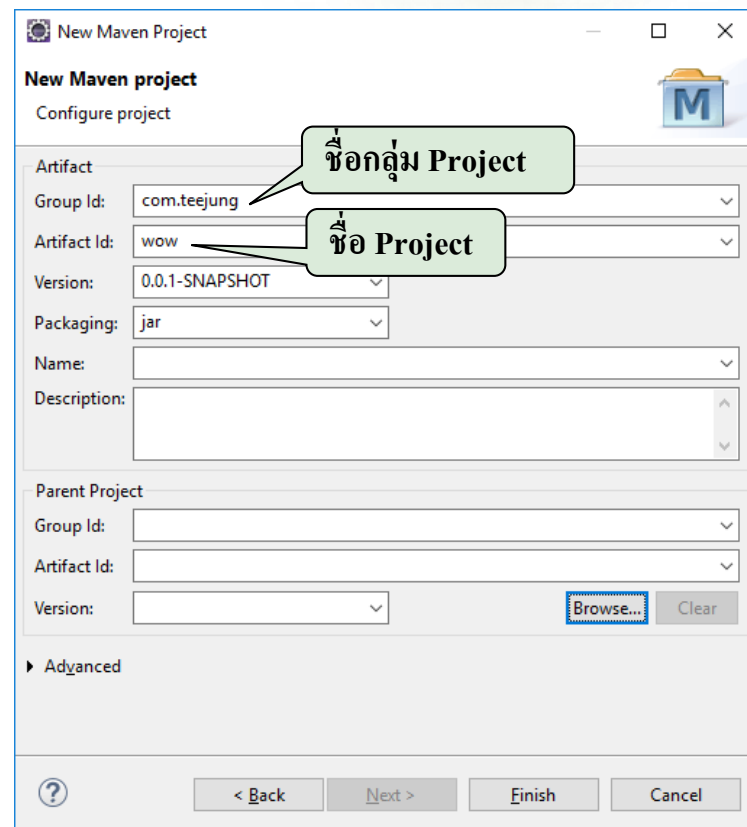
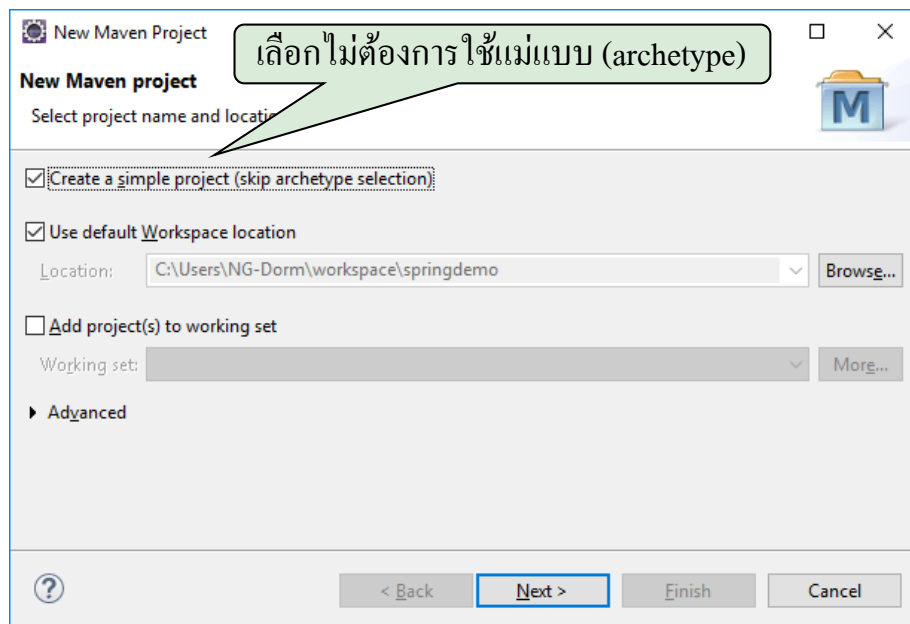
ชื่อ Dependency	คำอธิบาย
spring-boot-starter	ใช้กับแอปพลิเคชันที่ใช้เฉพาะส่วนหลักของ Spring Boot ในส่วน auto-configuration ต่างๆ
spring-boot-starter-web	ใช้ในการพัฒนาเว็บแบบ full-stack ประกอบด้วย Tomcat และ spring-webmvc
spring-boot-starter-jdbc	ใช้กับการติดต่อฐานข้อมูลผ่าน JDBC
spring-boot-starter-data-jpa	สนับสนุน JPA (Java Persistence API) ประกอบด้วย spring-data-jpa, spring-orm และ Hibernate
spring-boot-starter-data-mongodb	สนับสนุน MongoDB NoSQL Database
spring-boot-starter-data-rest	ใช้พัฒนา Web API (RESTful API) จาก spring-data-rest-webmvc
spring-boot-starter-freemarker	ใช้ Render หน้าแสดงผลโดยไม่ใช้โค้ดภาษาจาวาด้วย FreeMarker templating engine
spring-boot-starter-thymeleaf	ใช้ Render หน้าแสดงผลโดยไม่ใช้โค้ดภาษาจาวาด้วย Thymeleaf templating engine
spring-boot-starter-ws	ใช้พัฒนา Web Services
spring-boot-starter-cloud-connectors	ใช้ติดต่อกับ “Spring Cloud Connectors” เช่น Cloud Foundry และ Heroku
spring-boot-starter-security	ใช้จัดการความปลอดภัยในการเข้าถึงระบบ
spring-boot-starter-social-facebook	ใช้ติดต่อกับ Facebook API
spring-boot-starter-social-twitter	ใช้ติดต่อกับ Twitter API



การสร้าง Spring Boot Application



❖ ที่โปรแกรม Eclipse เลือกเมนู File > New > Maven Project





การกำหนดค่าใน pom.xml



- ❖ ใช้แท็ก <parent> กำหนดค่าเริ่มต้นของ Spring Boot
- ❖ แต่ละ Dependency ให้ใส่ groupId เป็น **"org.springframework.boot"**

กำหนด dependency เริ่มต้นของ
Spring Boot Application

Dependency สำหรับพัฒนาเว็บ
แอปพลิเคชันแบบ MVC

ใช้ระบุให้ Maven สามารถสร้าง executable jar file
(อาจเรียกว่า fat jars)

```
<project xmlns="...">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.teejung</groupId>
  <artifactId>wow</artifactId>
  <version>0.0.1-SNAPSHOT</version>

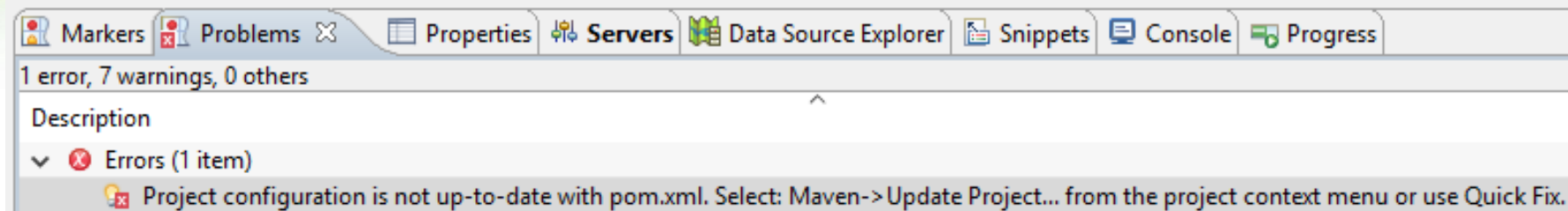
  <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.5.2.RELEASE</version>
  </parent>

  <dependencies>
    <dependency>
      <groupId>org.springframework.boot</groupId>
      <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
  </dependencies>

  <build>
    <plugins>
      <plugin>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-maven-plugin</artifactId>
      </plugin>
    </plugins>
  </build>
</project>
```




กรณีเกิด not up-to-date

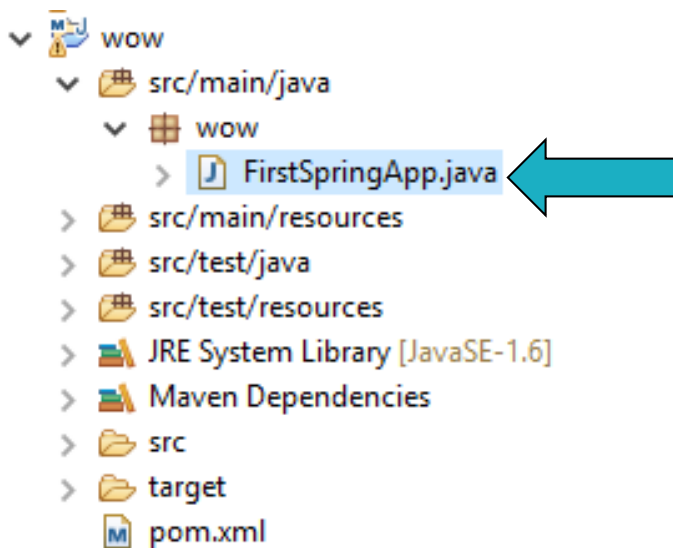


- ❖ หาก Save ไฟล์ pom.xml แล้ว Project มี Error ดังภาพ ให้คลิกขวาที่ Project เลือกเมนู Maven > Update Project...
- ❖ แล้วเลือกที่ “Force Update of Snapshots/Release” และกดปุ่ม OK



การสร้างคลาสหลัก

- ❖ สร้าง package ใหม่ภายใต้ src/main/java
 - ข้อแนะนำ ควรเก็บคลาสต่างๆ ไว้ใน package เสมอ โดยอาจตั้งชื่อ package จาก domain ของแอปพลิเคชัน จากหลังไปหน้า เช่น project.example.com จะตั้งชื่อเป็น com.example.project
- ❖ สร้างคลาสหลัก เช่น FirstSpringApp โดยให้มีเมธอด main บรรทัดอยู่





โครงสร้างการเก็บไฟล์



- firstspring
 - src/main/java
 - firstspring
 - MainApplication.java
 - src/main/resources
 - static
 - templates
 - application.properties
 - src/test/java
 - src/test/resources
 - JRE System Library [JavaSE-1.6]
 - Maven Dependencies
 - src
 - target
 - pom.xml
- โฟลเดอร์สำหรับเก็บคลาสจาวา ซึ่งอาจสร้าง package ย่อยแยกส่วน model และ controller ออกจากกันได้
- ไฟล์หลักสำหรับใช้เริ่มทำงาน ซึ่งมี method main() อยู่
- โฟลเดอร์สำหรับเก็บไฟล์ static ที่ไม่มีการเปลี่ยนแปลง เช่น html, js, css, รูปภาพ
- โฟลเดอร์สำหรับเก็บไฟล์ส่วน View ของ MVC
- ไฟล์เก็บข้อมูลเกี่ยวกับ Spring Boot Application เช่น URL ของฐานข้อมูล



คลาสหลัก

```
package wow;
```

```
import org.springframework.boot.SpringApplication;  
import org.springframework.boot.autoconfigure.EnableAutoConfiguration;  
import org.springframework.context.annotation.ComponentScan;
```

`@EnableAutoConfiguration` — ระบุ annotation เพื่อให้มีการ config แอปพลิเคชันแบบอัตโนมัติ

`@ComponentScan` — ระบุ annotation เพื่อให้ค้นหาคลาสที่เกี่ยวข้องอัตโนมัติ

```
public class FirstSpringApp {
```

```
    public static void main(String[] args) {
```

กำหนดจุดเริ่มต้นของแอปพลิเคชันด้วยเมธอด main

```
        SpringApplication.run(FirstSpringApp.class, args);
```

```
    }
```

สั่งให้ Spring Boot Application เริ่มทำงาน โดยระบุชื่อคลาสเริ่มต้น

ใช้รับค่าทาง command line

```
}
```




@EnableAutoConfiguration

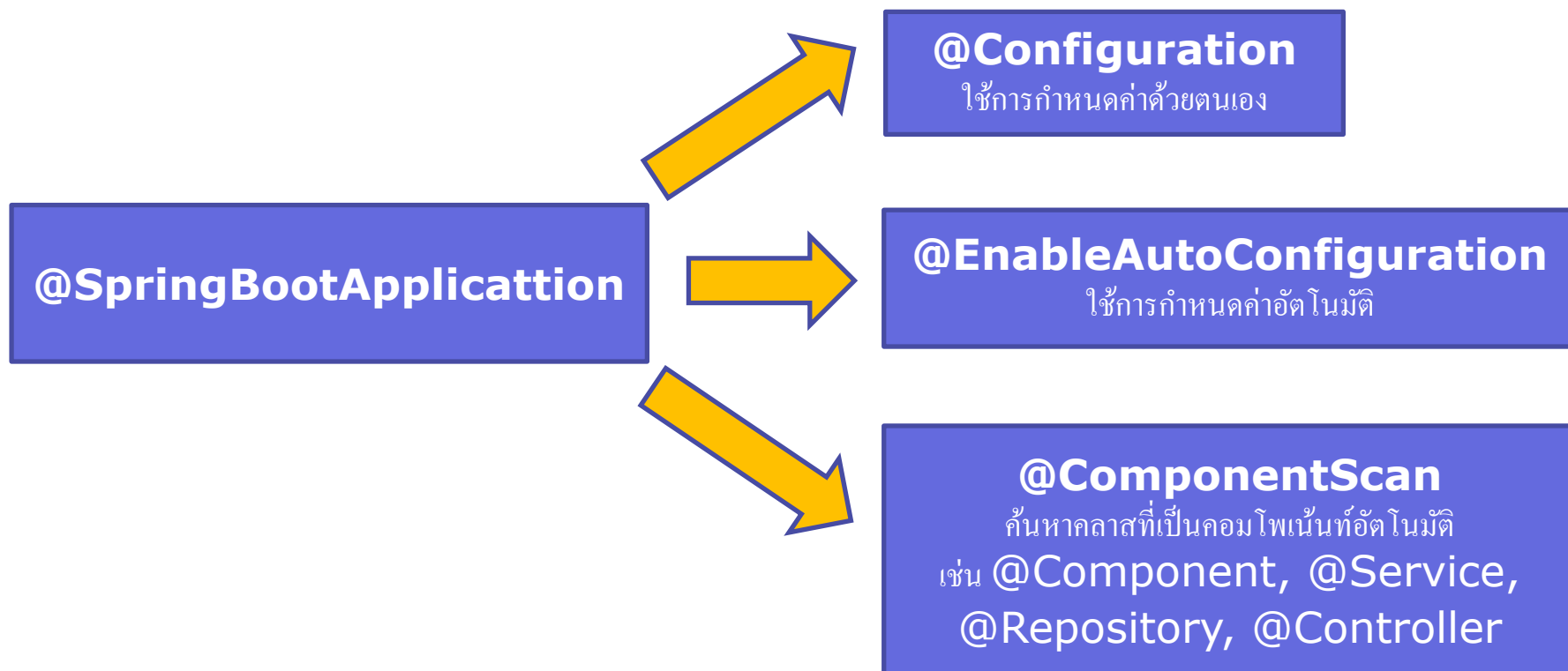


- ❖ เมื่อมีการระบุ annotation นี้ Spring Boot จะคาดเดาการ configuration โดยพิจารณาจาก dependencies (.jar) ที่กำหนดไว้ในโปรเจกแบบอัตโนมัติ
- ❖ เช่น หากใช้ spring-boot-starter-web จะถูกเพิ่ม Tomcat และ Spring MVC ซึ่งจะถูก setup ค่าให้เป็นเว็บแอปพลิเคชัน และ start Tomcat Web Server แบบอัตโนมัติ
- ❖ Auto-configuration จะทำงานได้ดีกับ Dependency ที่เป็น Starter ของ Spring นักพัฒนาอาจไม่ใช่ Starter ก็ได้ แต่ Spring Boot ก็ยังคงทำงานโดยการพยายามกำหนด configuration ที่ดีที่สุดให้กับแอปพลิเคชัน



@SpringBootApplication

- ❖ สามารถใช้ Annotation **@SpringBootApplication** ประกาศหนึ่คลาส ซึ่งเทียบเท่ากับการกำหนด 3 annotation





การสร้าง Controller



```
package wow;
```

```
import org.springframework.web.bind.annotation.GetMapping;
```

```
import org.springframework.web.bind.annotation.RestController;
```

```
@RestController ●————— ระบุว่าให้คลาสนี้เป็น Controller ของ Web API
```

```
public class FristController {
```

```
    @GetMapping("/hello") ●————— กำหนด Path ของ HTTP request ที่ใช้เข้าถึงเมธอดนี้แบบ GET
```

```
    public String home() {
```

```
        return "Hello World"; ●————— ส่ง HTTP response กลับไปด้วยข้อความธรรมดา (text plain)
```

```
    }
```

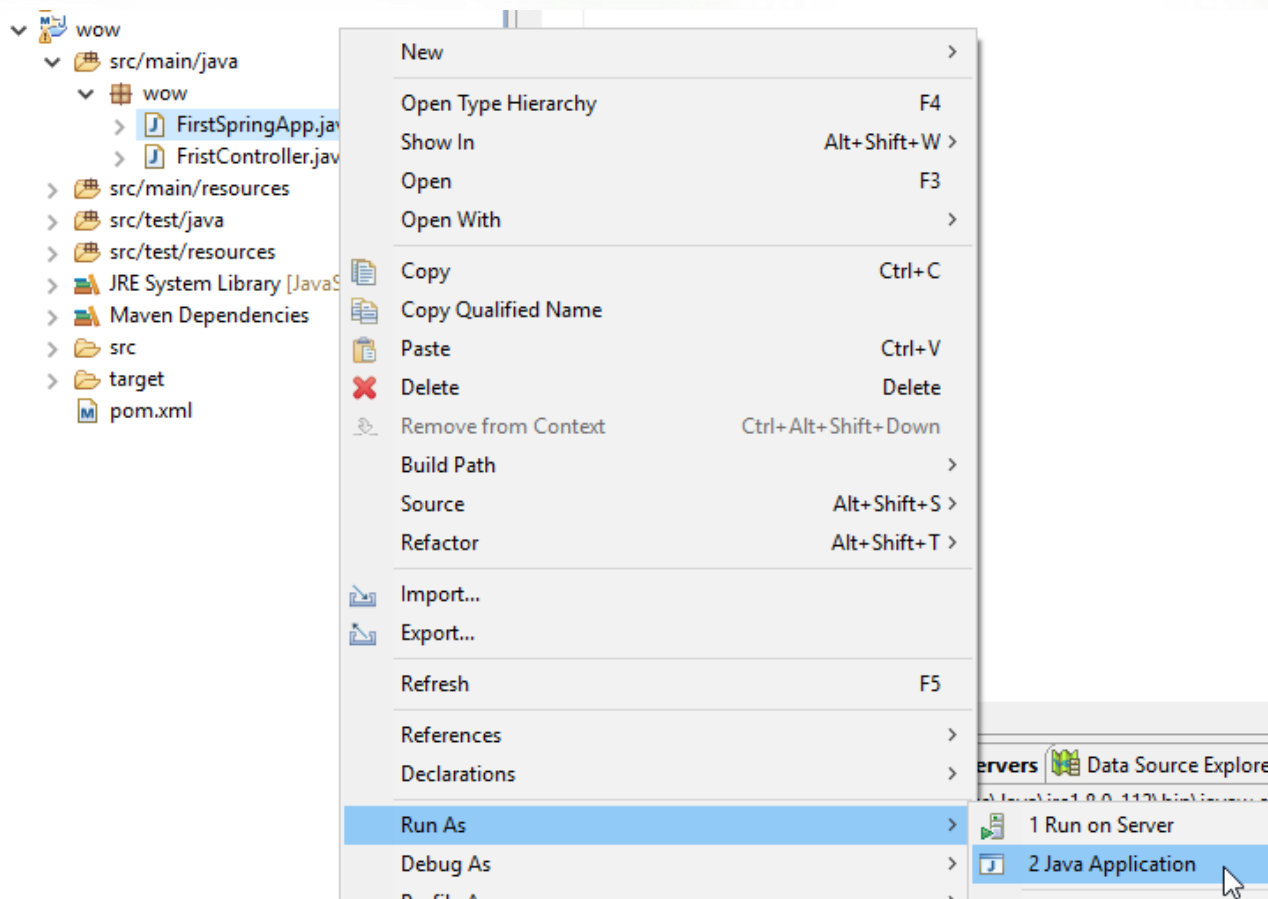
```
}
```



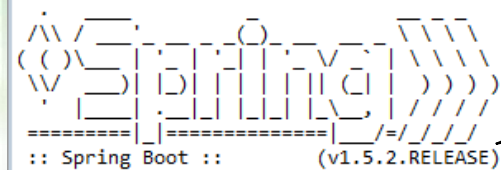
การรัน Spring Boot Application



❖ คลิกขวาที่คลาสหลัก เลือก Run As > Java Application



การรัน Spring Boot Application



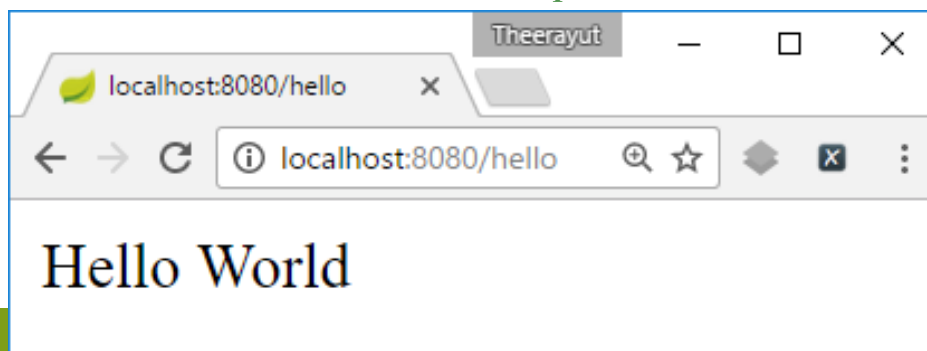
Version ของ Spring Boot ที่ใช้

```
2560-04-04 00:10:36.403 INFO 4608 --- [main] wow.WowApplication : Starting WowApplication on NG-Dorm with PID 4608 (C:\Users\WG-Dc
2560-04-04 00:10:36.403 INFO 4608 --- [main] wow.WowApplication : No active profile set, falling back to default profiles: default
2560-04-04 00:10:36.560 INFO 4608 --- [main] ationConfigEmbeddedWebApplicationContext : Refreshing org.springframework.boot.context.embedded.AnnotationC
2560-04-04 00:10:39.259 INFO 4608 --- [main] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat initialized with port(s): 8080 (http)
2560-04-04 00:10:39.307 INFO 4608 --- [main] o.apache.catalina.core.StandardService : Starting service Tomcat
2560-04-04 00:10:39.307 INFO 4608 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet Engine: Apache Tomcat/8.5.11
2560-04-04 00:10:39.526 INFO 4608 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext
2560-04-04 00:10:39.526 INFO 4608 --- [ost-startStop-1] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 2966 ms
2560-04-04 00:10:39.807 INFO 4608 --- [ost-startStop-1] o.s.b.w.servlet.ServletRegistrationBean : Mapping servlet: 'dispatcherServlet' to [/]
2560-04-04 00:10:39.822 INFO 4608 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'characterEncodingFilter' to: [/]
2560-04-04 00:10:39.822 INFO 4608 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'hiddenHttpMethodFilter' to: [/]
2560-04-04 00:10:39.822 INFO 4608 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'httpPutFormContentFilter' to: [/]
2560-04-04 00:10:39.822 INFO 4608 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'requestContextFilter' to: [/]
2560-04-04 00:10:40.349 INFO 4608 --- [ma : Looking for @ControllerAdvice: org.springframework.boot.context.
2560-04-04 00:10:40.541 INFO 4608 --- [ma : Mapped "{[/hello],methods=[GET]}" onto java.lang.String wow.Firs
2560-04-04 00:10:40.557 INFO 4608 --- [ma : Mapped "{[/error]}" onto public org.springframework.http.Respons
2560-04-04 00:10:40.557 INFO 4608 --- [ma : Mapped "{[/error],produces=[text/html]}" onto public org.springf
2560-04-04 00:10:40.729 INFO 4608 --- [ma : Mapped URL path [/webjars/**] onto handler of type [class org.sp
2560-04-04 00:10:40.744 INFO 4608 --- [ma : Mapped URL path [/**] onto handler of type [class org.springfran
2560-04-04 00:10:40.838 INFO 4608 --- [ma : Mapped URL path [/**/favicon.ico] onto handler of type [class or
2560-04-04 00:10:41.119 INFO 4608 --- [ma : Registering beans for JMX exposure on startup
2560-04-04 00:10:41.244 INFO 4608 --- [ma : Tomcat started on port(s): 8080 (http)
2560-04-04 00:10:41.260 INFO 4608 --- [ma : Started WowApplication in 5.539 seconds (JVM running for 6.397)
```

มี path /hello เรียกได้ในแบบ GET

ใช้ port 8080

ผลลัพธ์จากการส่ง HTTP Request ผ่าน Browser

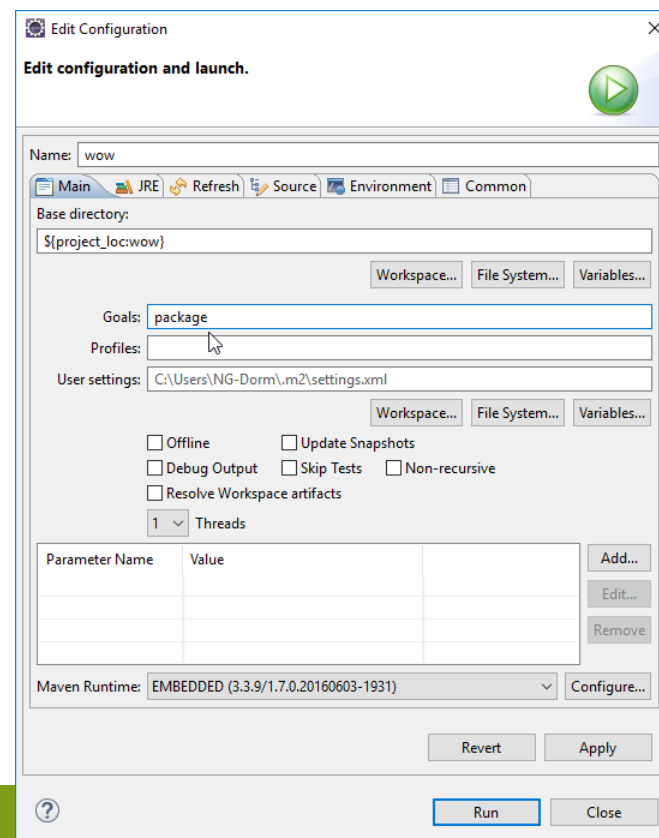
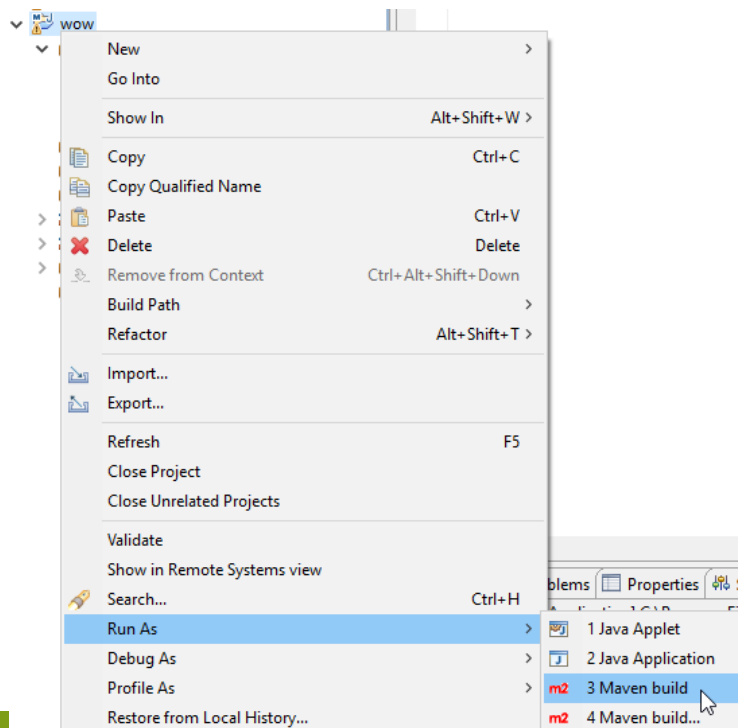




การสร้าง Executable jar file (fat jars)



- ❖ Spring Boot Application ที่สร้างเสร็จแล้ว สามารถนำไปรันที่เครื่องใดก็ได้ที่มี JRE แต่ต้องสร้าง Executable jar file ขึ้นมาก่อน
- ❖ คลิกขวาที่ชื่อ Project เลือก Run As > Maven build
- ❖ ที่ช่อง Goals ใส่คำว่า "package" และกดปุ่ม Run

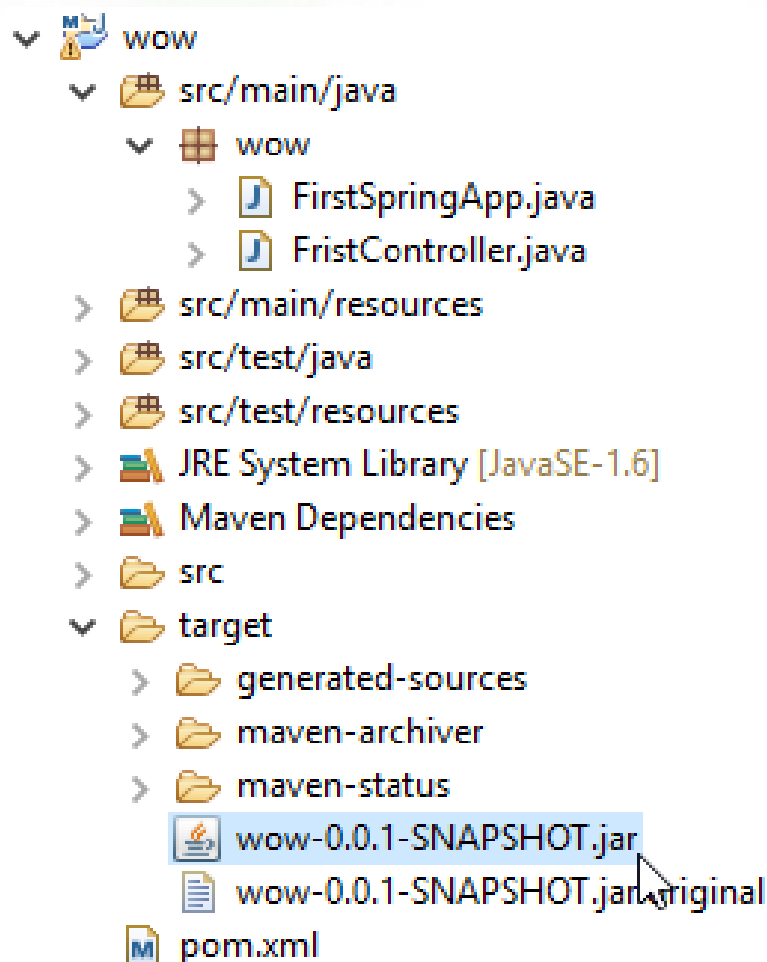




การสร้าง Executable jar file (fat jars)



❖ คลิกขวาที่ชื่อ Project > Refresh จะเห็นไฟล์ Executable jar file (.jar) ภายในโฟลเดอร์ target

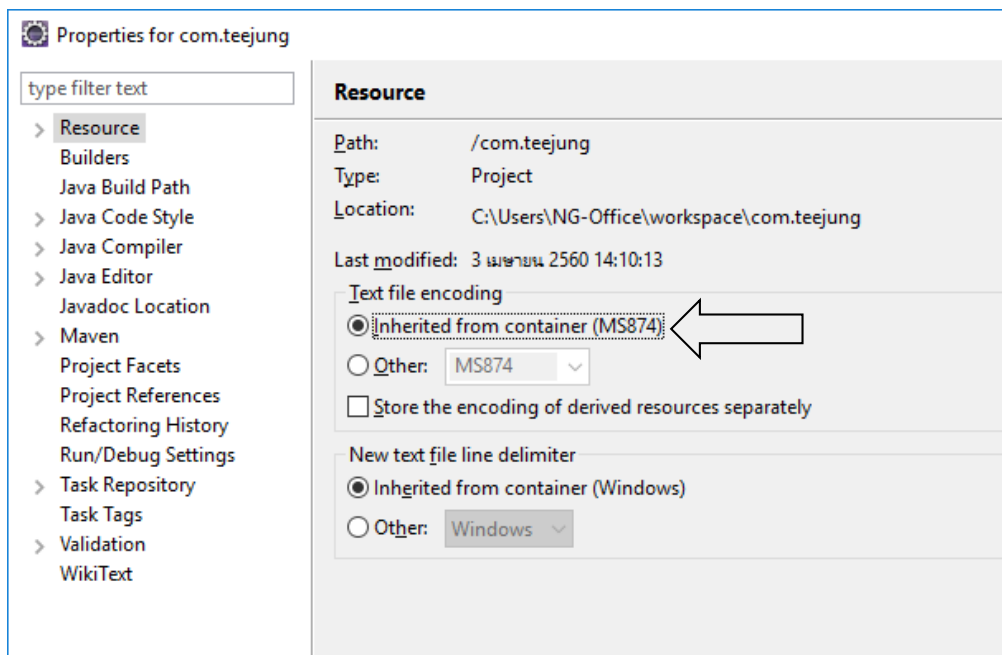




กรณีไม่สามารถ package ได้



ให้คลิกขวาที่ชื่อ Project แล้วเลือก Property > Resource > เลือก Inherit from container

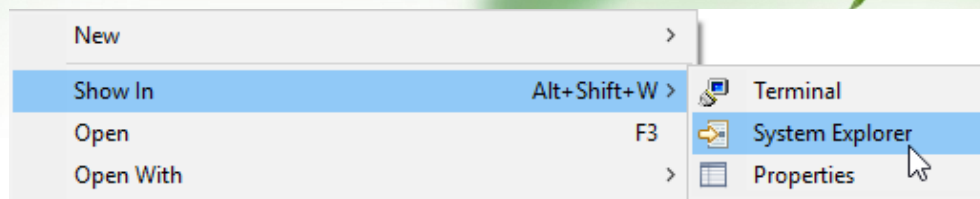




การรัน Executable jar file

❖ คลิกขวาที่โฟลเดอร์ target

เลือก Show In > System Explorer



❖ กดปุ่ม Shift ค้างไว้แล้วคลิกขวาที่โฟลเดอร์ target

เลือก Open command window here

❖ พิมพ์คำสั่งเพื่อเปิดการทำงาน

ไฟล์ .jar ที่สร้างขึ้น รูปแบบดังนี้

`java -jar ชื่อไฟล์.jar`

```
C:\WINDOWS\system32\cmd.exe - java -jar wow-0.0.1-SNAPSHOT.jar

C:\Users\NG-Dorm\workspace\wow\target>java -jar wow-0.0.1-SNAPSHOT.jar

Spring
:: Spring Boot :: (v1.5.2.RELEASE)

2560-04-08 23:35:38.616 INFO 9556 --- [          main] wow.FirstSpringApp
.1-SNAPSHOT on NG-Dorm with PID 9556 (C:\Users\NG-Dorm\workspace\wow\target\wow-0.0.1-SNAPSHOT.jar)
s\NG-Dorm\workspace\wow\target)
2560-04-08 23:35:38.632 INFO 9556 --- [          main] wow.FirstSpringApp
ng back to default profiles: default
2560-04-08 23:35:38.788 INFO 9556 --- [          main] ationConfigEmbeddedWebApplicationContext
```



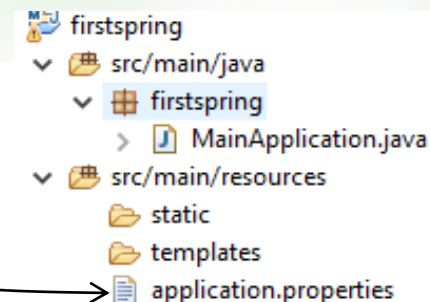
ไฟล์ application.properties



❖ การ Config ค่าเกี่ยวกับ Spring Boot Application จะถูกกำหนดในไฟล์ application.properties

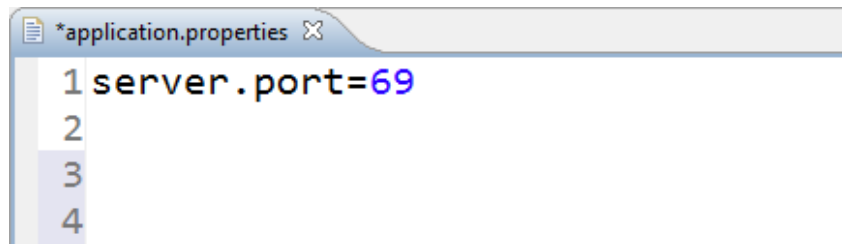
❖ นักพัฒนาจะต้องสร้างไฟล์นี้ขึ้นมาเอง ซึ่งอยู่ใน src/main/resources

❖ การเก็บข้อมูลจะอยู่ในรูปแบบ



property-name = value

❖ แม้แบบ spring-boot-starter-web มี Web Container Tomcat ฝังมาในตัวแล้ว และกำหนด port เป็น 8080 ดังนั้นควร Stop Server ที่เคยเปิดอยู่ เพื่อป้องกัน port ชนกัน หรือเปลี่ยน port ของ Spring Boot Application โดยเพิ่ม property ดังนี้





ตัวอย่าง property



```
# EMBEDDED SERVER CONFIGURATION (ServerProperties)
server.port=8080
server.address= # bind to a specific NIC
server.session-timeout= # session timeout in seconds
```

```
# HTTP encoding (HttpEncodingProperties)
```

```
# encoding of HTTP requests/responses
spring.http.encoding.charset=UTF-8
```

```
# enable http encoding support
spring.http.encoding.enabled=true
```

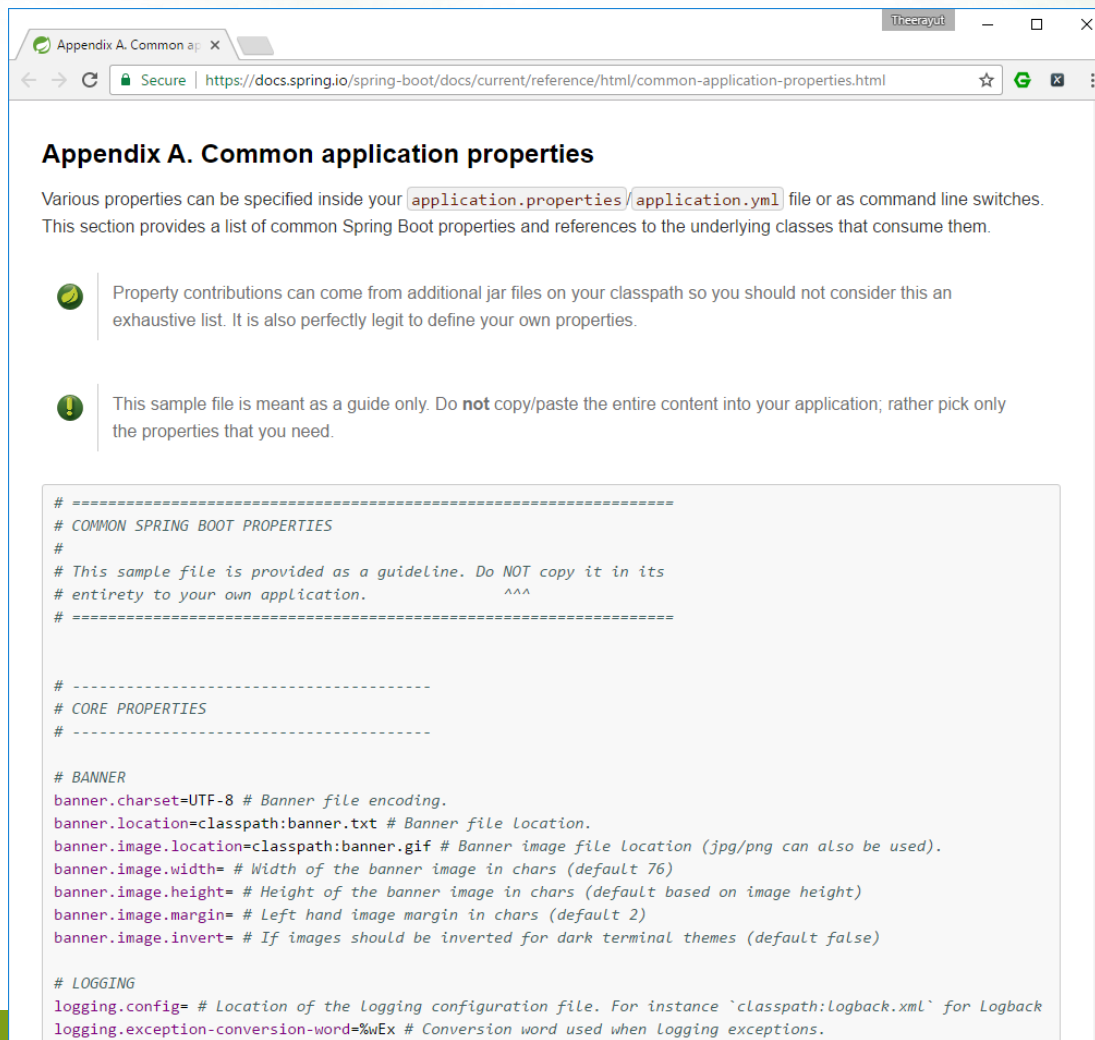
```
# force the configured encoding
spring.http.encoding.force=true
```



Common application properties



<https://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html>



The screenshot shows a web browser window with the address bar displaying the URL `https://docs.spring.io/spring-boot/docs/current/reference/html/common-application-properties.html`. The page title is "Appendix A. Common application properties". The main content area contains the following text:

Various properties can be specified inside your `application.properties`/`application.yml` file or as command line switches. This section provides a list of common Spring Boot properties and references to the underlying classes that consume them.

Property contributions can come from additional jar files on your classpath so you should not consider this an exhaustive list. It is also perfectly legit to define your own properties.

This sample file is meant as a guide only. Do **not** copy/paste the entire content into your application; rather pick only the properties that you need.

```
# =====
# COMMON SPRING BOOT PROPERTIES
#
# This sample file is provided as a guideline. Do NOT copy it in its
# entirety to your own application.      ^^^
# =====

# -----
# CORE PROPERTIES
# -----

# BANNER
banner.charset=UTF-8 # Banner file encoding.
banner.location=classpath:banner.txt # Banner file location.
banner.image.location=classpath:banner.gif # Banner image file location (jpg/png can also be used).
banner.image.width= # Width of the banner image in chars (default 76)
banner.image.height= # Height of the banner image in chars (default based on image height)
banner.image.margin= # Left hand image margin in chars (default 2)
banner.image.invert= # If images should be inverted for dark terminal themes (default false)

# LOGGING
logging.config= # Location of the logging configuration file. For instance `classpath:logback.xml` for Logback
logging.exception-conversion-word=%wEx # Conversion word used when logging exceptions.
```




Developer tools

- ❖ Spring Boot มีเครื่องมือสำหรับนักพัฒนาที่ใช้ในช่วงวงจรการพัฒนา โดยมี dependency ดังนี้

```
<dependency>  
  <groupId>org.springframework.boot</groupId>  
  <artifactId>spring-boot-devtools</artifactId>  
  <optional>true</optional>  
</dependency>
```

- ❖ การกำหนดแท็ก optional เป็น true ช่วยให้ง่ายต่อการสร้าง Production Application โดย devtools จะถูก disable อัตโนมัติ เมื่อรันด้วย Executable jar file
- ❖ เมื่อ class ใดมีการแก้ไขโค้ดและ compile ใหม่ จะ restart application อัตโนมัติ
- ❖ สามารถกำหนด path ที่ไม่ต้องการ restart ในไฟล์ application.properties ได้ เช่น

spring.devtools.restart.exclude=static/,public/****