



# บทที่ 8

## Maven Project

ธีระยุทธ ทองเครือ

ภาควิชาวิทยาการคอมพิวเตอร์ คณะวิทยาศาสตร์

มหาวิทยาลัยขอนแก่น



# Maven

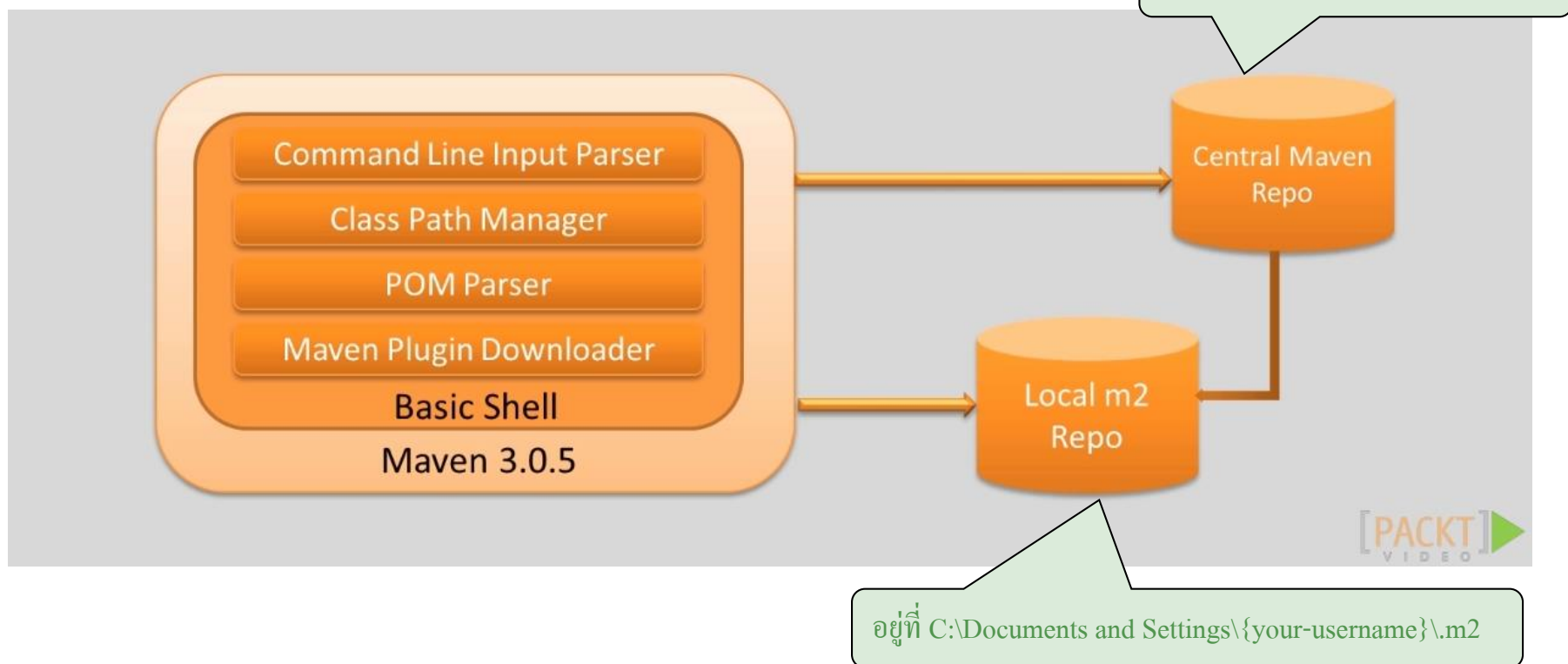
- ❖ Maven คือ เครื่องมือสำหรับใช้ในการบริหารจัดการ Project โดยมีเป้าหมายคือ ช่วยในการ build project แบบอัตโนมัติ
- ❖ Maven ใช้ Project Object Model (POM) ซึ่งเป็นไฟล์ข้อมูลเกี่ยวกับการสร้าง Project ในรูปแบบ XML นักพัฒนาสามารถกำหนดข้อมูล เช่น
  - รายการไลบรารีที่อ้างอิง (Dependency list) ช่วยให้ไม่ต้องไปโหลดไฟล์ Library เอง
  - ส่วนการทำ Unit Test
  - การกำหนด environment ต่างๆ
- ❖ Maven มีแม่แบบ (archetype) ในการช่วยสร้างแอปพลิเคชันชนิดต่างๆ
- ❖ ใน Eclipse มี Maven ติดมาแล้ว สามารถสร้าง Project ในรูปแบบ Maven ได้เลย



# Maven Architecture



- ❖ Central Maven Repository คือ Server ที่ใช้เก็บ Library ทั้งหมด
- ❖ Library ที่ถูกอ้างอิงมาใช้แล้ว จะอยู่บน Local m2 Repository เพื่อใช้กับ Project อื่นๆ ไม่ต้องโหลดใหม่จาก Central Maven Repository ทุกครั้ง





# การสร้าง Maven Web Project

❖ เลือกเมนู File ► New ► Maven Project

New Maven Project

New Maven project

Select project name and location

☐ Create a simple project (skip archetype selection)

☒ Use default Workspace location

Location: C:\Users\NG-Dorm\workspace\ Browse...

☐ Add project(s) to working set

Working set: More...

► Advanced

? < Back Next > Finish Cancel

เลือกที่นี่ หากไม่ต้องการใช้แม่แบบ (archetype) ใช้ในกรณีการสร้าง Application ทั่วไป





# การเลือกแม่แบบ Project



New Maven Project

**New Maven project**  
Select an Archetype

Catalog: All Catalogs Configure...

Filter:

Group Id	Artifact Id	Version
org.apache.maven.archetypes	maven-archetype-profiles	1.0-alpha-4
org.apache.maven.archetypes	maven-archetype-quickstart	1.1
org.apache.maven.archetypes	maven-archetype-site	
org.apache.maven.archetypes	maven-archetype-site-simple	
org.apache.maven.archetypes	maven-archetype-webapp	

An archetype which contains a sample Maven Webapp project.

☒ Show the last version of Archetype only ☐ Include snapshot archetypes Add Archetype...

► Advanced

? < Back Next > Finish Cancel

เลือกใช้แม่แบบเว็บแอปพลิเคชัน  
(Web application archetype)



# กำหนดชื่อกุ่ม และชื่อ Project



New Maven Project

New Maven project

Specify Archetype parameters

Group Id:

Artifact Id:

Version:

Package:

Properties available from archetype:

Name	Value

► Advanced

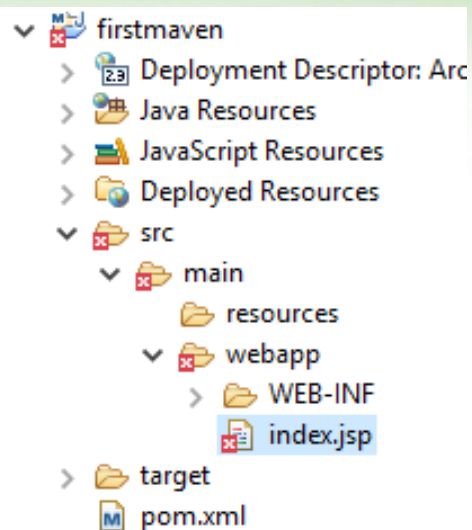
Buttons: ? < Back Next > Finish Cancel

Annotations:

- ชื่อกุ่ม Project - อาจใช้ชื่อ domain ขององค์กร  
เจ้าของ project นำมาเขียนจากหลังไปหน้า
- ชื่อ Project



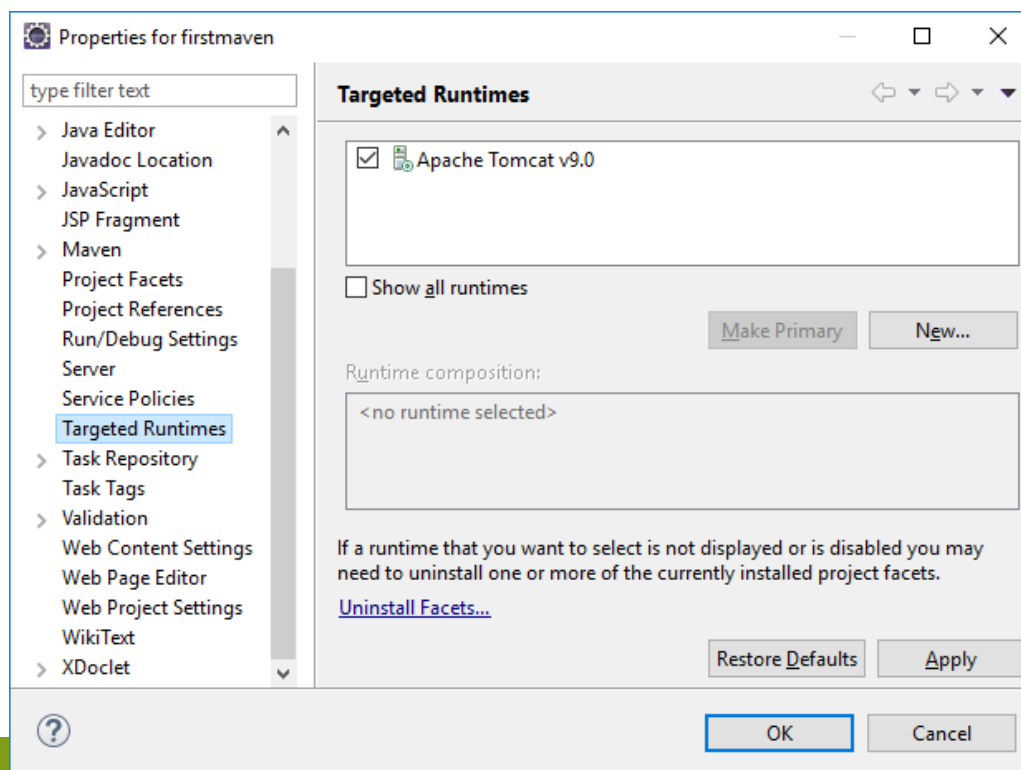
# กำหนด Server สำหรับ Project



❖ Maven Project ที่สร้างขึ้นจะยังไม่รู้จักคลาส HttpServlet  
แก้ไขโดยคลิกขวาที่ชื่อ Project เลือก *Properties*

❖ เลือก *Targeted Runtimes*

❖ เลือก Web Server ที่เพิ่มไว้แล้ว เช่น Tomcat

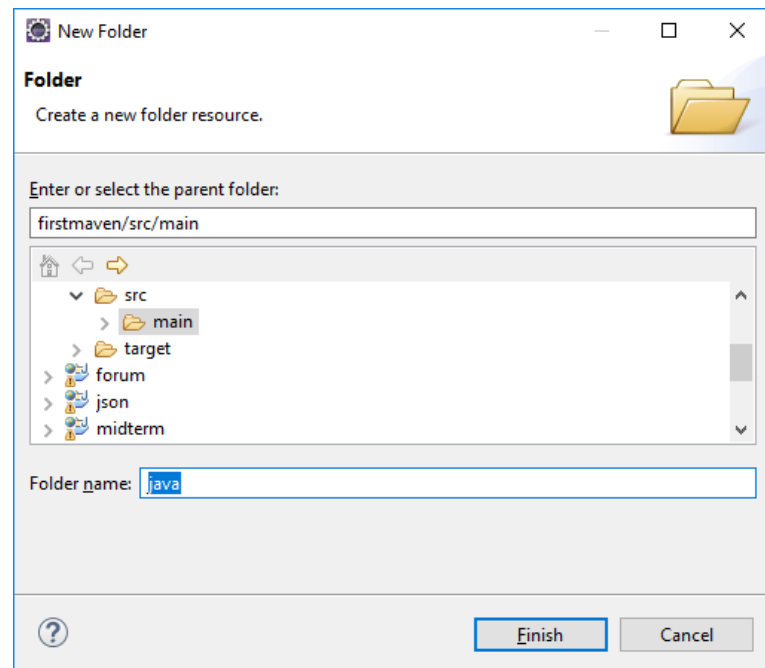
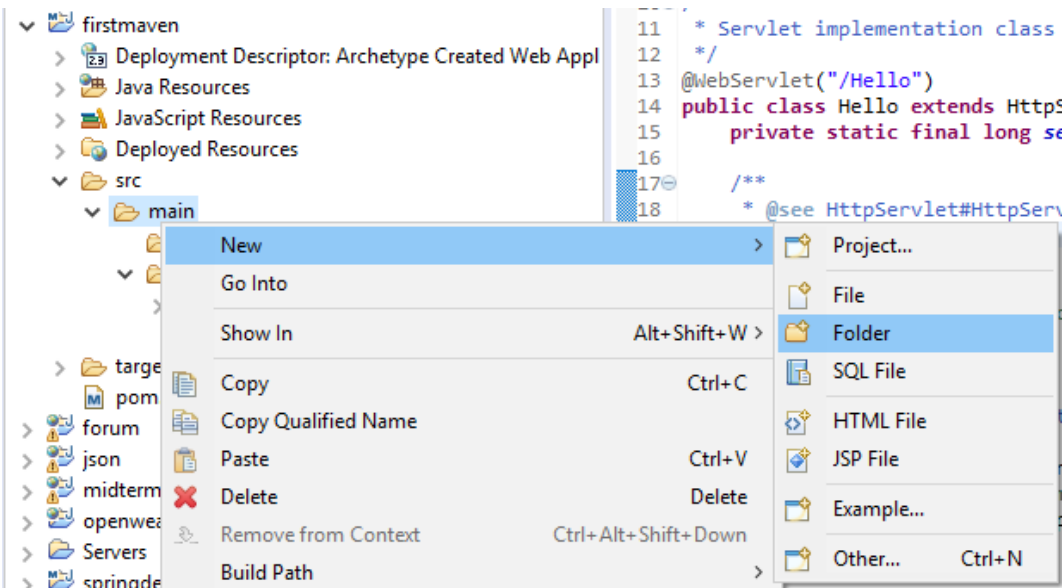




# สร้างโฟลเดอร์เก็บโค้ด .java



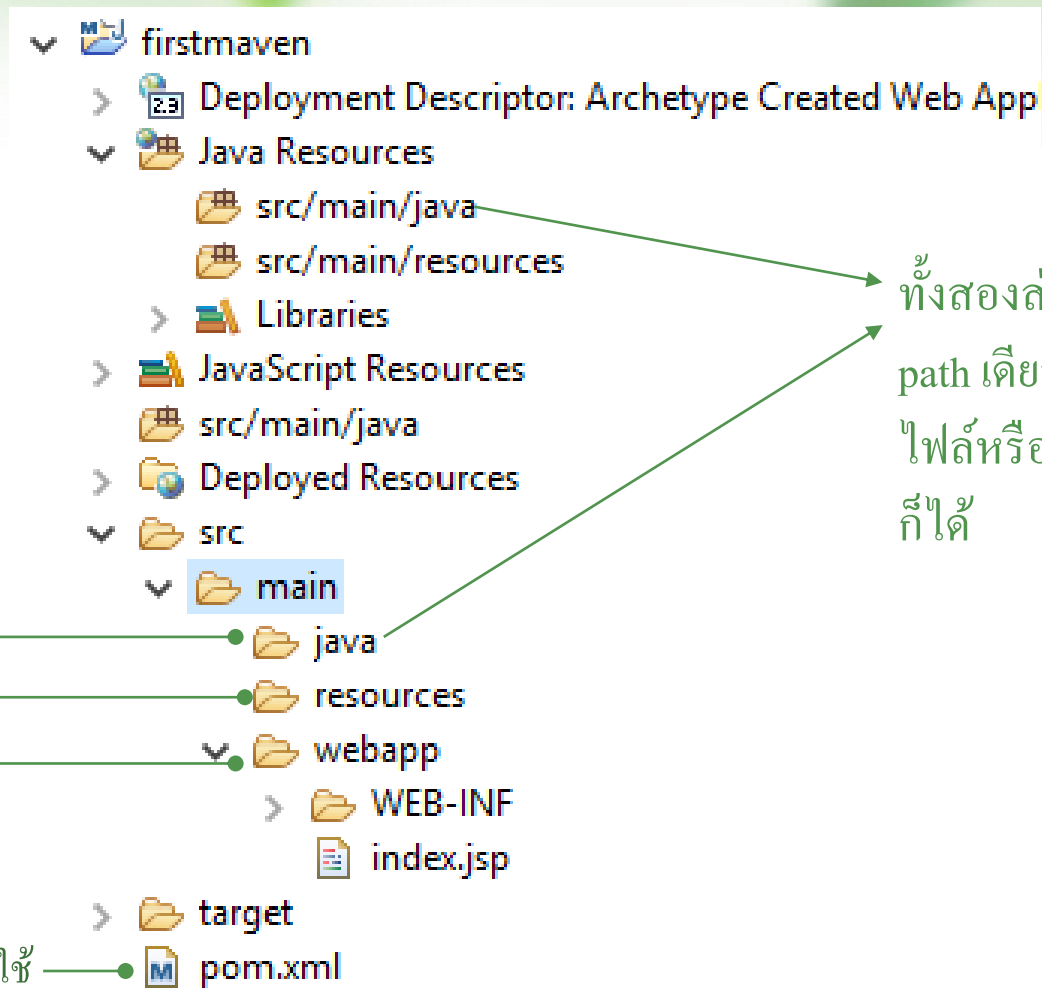
- ❖ แม่แบบเว็บแอปพลิเคชันจะไม่สร้างโฟลเดอร์สำหรับเก็บไฟล์ .java มาให้ ให้สร้างโฟลเดอร์ใหม่เองชื่อ java ภายใต้โฟลเดอร์ main
- ❖ คลิกขวาที่โฟลเดอร์ main เลือก New > Folder
- ❖ ใส่ชื่อโฟลเดอร์ "java"







# โครงสร้างโฟลเดอร์ Maven Project



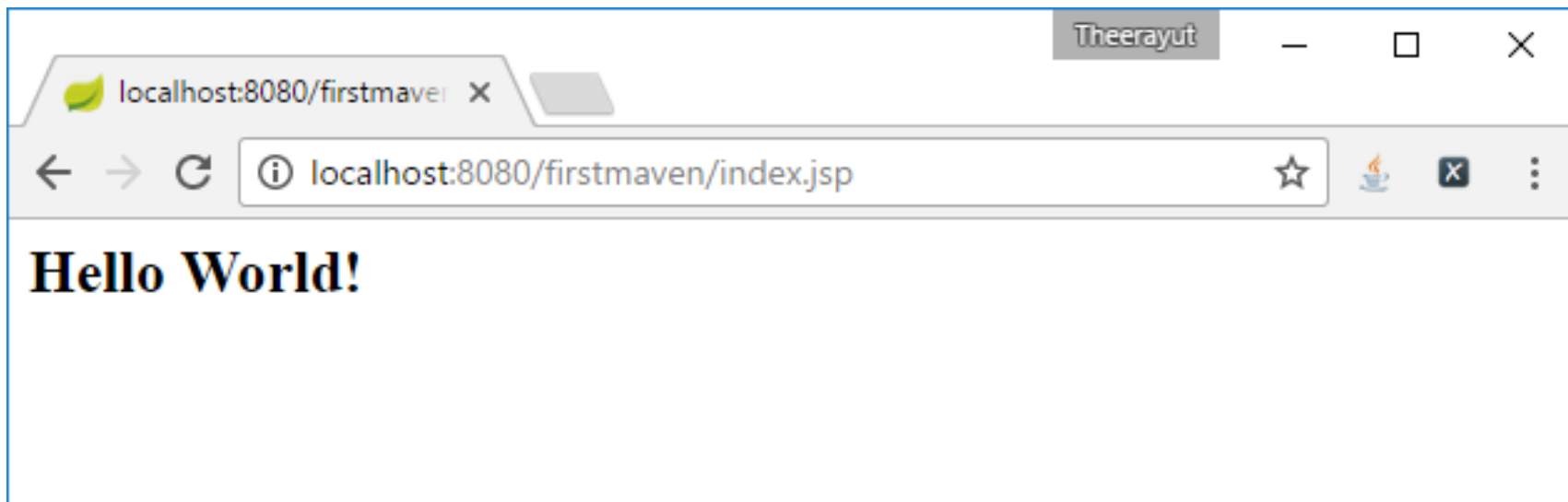
ทั้งสองส่วนหมายถึง path เดียวกัน จะสร้างไฟล์หรือเข้าถึงที่ส่วนใดก็ได้



# ทดสอบการทำงาน JSP



- ❖ เพิ่ม Maven Project ที่สร้างขึ้นเข้าไปใน Server
- ❖ Start Server และทดสอบรันไฟล์ index.jsp ที่แม่แบบสร้างไว้ให้แล้ว





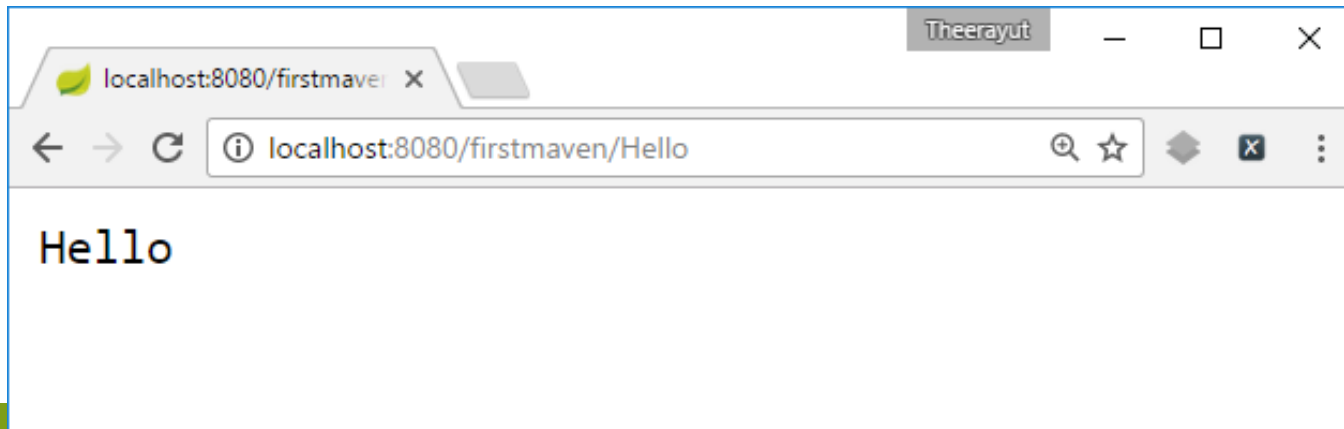
# ทดสอบการทำงาน Servlet

❖ สร้าง Servlet ในโฟลเดอร์ main/java

❖ restart server และทดสอบ

!!!! ใน Maven Project จะต้อง  
ใส่ path ของ Servlet เอง

```
@WebServlet("/Hello")  
public class Hello extends HttpServlet {  
  
    protected void doGet(HttpServletRequest request,  
        HttpServletResponse response) throws ServletException, IOException {  
  
        response.getWriter().println("Hello");  
    }  
}
```





# การเพิ่มไลบรารีสำหรับ Maven Project



- ❖ นักพัฒนาสามารถนำไลบรารีมาใช้ โดยหาได้จาก 2 แหล่ง
  - เว็บไซต์ผู้สร้าง Library โดยตรง
  - Central Maven Repository
- ❖ การเพิ่ม Library คือ การนำชื่อ และรุ่นของ Library มาเพิ่มข้อมูลลงในไฟล์ pom.xml โดยวางภายใต้แท็ก <dependencies></dependencies> หลังจากบันทึกไฟล์ pom.xml แล้ว Maven จะติดต่อไปยัง Maven Repository เพื่อโหลดไฟล์ที่เกี่ยวข้องมาเก็บไว้ใน Project อัดโนมัติ ดังนั้นหากไม่เคยใช้ Library นั้นในเครื่องที่พัฒนาใช้ จะต้องต่อ internet ด้วย
- ❖ การโหลด Library จาก Maven Repository จะทำเฉพาะครั้งแรกและครั้งเดียวเท่านั้น หากเป็น Library ที่เคยเรียกใช้แล้วจะนำไฟล์จาก cache มาเพิ่มให้โดยไม่มีการดาวน์โหลดใหม่



# การค้นหาลibrary จาก Maven Repository

❖ นักพัฒนาสามารถค้นหาลibrary เพื่อนำมากำหนดใช้ใน Project ได้จาก

<http://search.maven.org>

Maven Repository: jdbc r X

← → C mvnrepository.com/search?q=jdbc+mysql

Indexed Artifacts (5.96M)

5956k  
2978k  
0  
2004 2017

Popular Categories

- Aspect Oriented
- Actor Frameworks
- Application Metrics
- Build Tools
- Bytecode Libraries
- Command Line Parsers
- Cache Implementations
- Cloud Computing
- Code Analyzers

Found 1535 results

- MySQL Connector/J** 1,882 usages  
mysql » mysql-connector-java  
MySQL JDBC Type 4 driver  
GPL
- Mariadb Java Client** 101 usages  
org.mariadb.jdbc » mariadb-java-client  
JDBC driver for MariaDB and MySQL  
LGPL
- MySQL Connector/MXJ** 67 usages  
mysql » mysql-connector-mxj  
MySQL Connector/MXJ is a Java Utility package for deploying and managing a MySQL database.  
GPL
- MySQL DataStore** 4 usages  
org.geotools.jdbc » jdbc-mysql

ระบุ keyword ของ  
Library ที่ต้องการค้นหา

Version	Repository	Usages	Date
6.0.5	Central	33	(Oct, 2016)
6.0.4	Central	15	(Aug, 2016)
6.0.3	Central	21	(Jun, 2016)
6.0.2	Central	10	(Mar, 2016)
5.1.41	Central	2	(Feb, 2017)
5.1.40	Central	69	(Sep, 2016)
5.1.39	Central	157	(May, 2016)
5.1.38	Central	196	(Dec, 2015)
5.1.37	Central	65	(Oct, 2015)
5.1.36	Central	98	(Jun, 2015)
5.1.35	Central	124	(May, 2015)
5.1.34	Central	128	(May, 2015)
5.1.33	Central	34	(May, 2015)
5.1.32	Central	35	(May, 2015)

ชื่อของ Library ที่จะ copy  
ไปวางไว้ในไฟล์ pom.xml

Home » mysql » mysql-connector-java » 6.0.5

**MySQL Connector/J » 6.0.5**  
MySQL JDBC Type 4 driver

License: GPL 2.0

Categories: MySQL Drivers

Organization: Oracle Corporation

HomePage: <http://dev.mysql.com/doc/connector-j/en/>

Date: (Oct 10, 2016)

Files: [Download \(JAR\)](#) (1.8 MB)

Repositories: Central

Used By: 1,882 artifacts

Maven Gradle SBT Ivy Grape Leiningen Buildr

```
<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-  
java -->  
<dependency>  
  <groupId>mysql</groupId>  
  <artifactId>mysql-connector-java</artifactId>  
  <version>6.0.5</version>  
</dependency>
```





# ตัวอย่างไฟล์ pom.xml



```
<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>com.teejung</groupId>
  <artifactId>firstmaven</artifactId>
  <packaging>war</packaging>
  <version>0.0.1-SNAPSHOT</version>
```

} ข้อมูล Project

แท็กใช้ครอบชื่อ Library  
ทั้งหมดของ Project

```
<dependencies>
```

แท็ก dependency ใช้  
ครอบชื่อ 1 Library

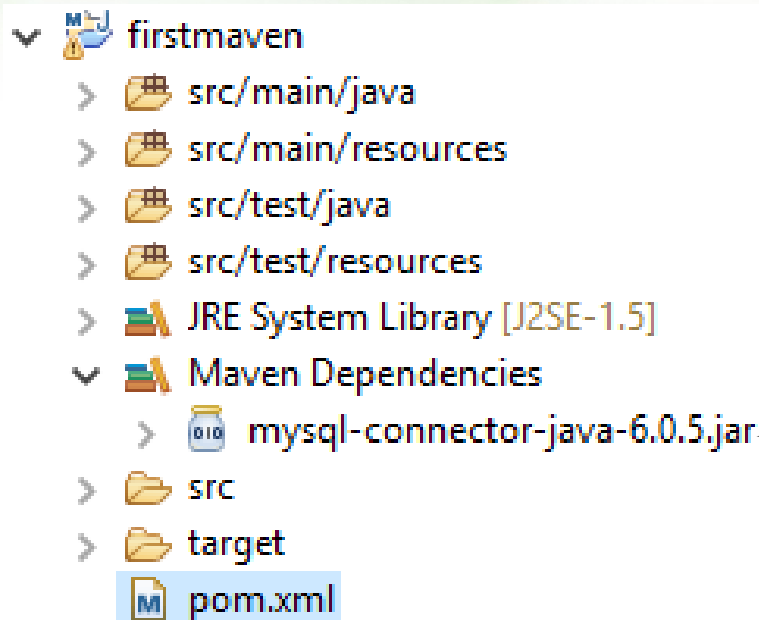
```
  <dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>6.0.5</version>
  </dependency>
```

```
</dependencies>
```

```
</project>
```



# ตัวอย่างไลบรารีที่ถูกโหลดแบบอัตโนมัติ



ไฟล์ Library ที่ Maven  
โหลดมาให้อัตโนมัติ

หากมีการลบแท็ก **<dependency>** ของไลบรารีใดออก

ไฟล์ .jar ก็จะถูกดึงออกจาก Project แบบอัตโนมัติ

แต่ไฟล์ .jar นั้นยังเก็บพัก (cache) ไว้ในโปรแกรม Eclipse อยู่

จะถูกนำกลับมาใช้ได้เมื่อมีการเพิ่มไลบรารีอีกครั้งใน Project ใดๆก็ได้



# การใช้ Library ที่โหลดเสร็จแล้ว



- ❖ เมื่อได้ Library ที่ต้องการแล้ว ไฟล์ต่างๆจะอยู่ใน Class Path อัตโนมัติ
- ❖ นักพัฒนาสามารถ import ในโค้ดโปรแกรม อ้างอิงคลาสจาก Library ที่ Maven โหลดมาใช้ได้ทันที

# ทดสอบใช้ Library ติดต่อฐานข้อมูลกับ Servlet

```
@WebServlet("/Hello")
public class Hello extends HttpServlet {
    protected void doGet(HttpServletRequest request, ...) {
        response.setContentType("text/html; charset=utf-8");
        PrintWriter out = response.getWriter();
        try {
            Class.forName("com.mysql.jdbc.Driver");
            String dbURL = "jdbc:mysql://localhost/blueshop?characterEncoding=utf-8";
            Connection con = DriverManager.getConnection(dbURL, "root", "");
            Statement statement = con.createStatement();
            ResultSet resultSet = statement.executeQuery("select * from product");
            while (resultSet.next()) {
                int pid = resultSet.getInt("pid");
                String pname = resultSet.getString("pname");
                String pdetail = resultSet.getString("pdetail");
                int price = resultSet.getInt("price");
                out.println(pid + "," + pname + "," + pdetail + "," + price + "<br>");
            }
        } catch (ClassNotFoundException e) {
            System.err.println("Error loading driver: " + e);
        } catch (SQLException e) {
            System.err.println("Error database connection: " + e);
        }
    }
}
```

