HANDS ON 1

# **CREATE A SPRING WEB PROJECT USING MAVEN**

Follow steps below to create a project: 

1. Go to <https://start.spring.io/>
2. Change Group as “com.cognizant”
3. Change Artifact Id as “spring-learn”
4. Select Spring Boot DevTools and Spring Web
5. Create and download the project as zip
6. Extract the zip in root folder to Eclipse Workspace
7. Build the project using ‘mvn clean package -Dhttp.proxyHost=proxy.cognizant.com -Dhttp.proxyPort=6050 -Dhttps.proxyHost=proxy.cognizant.com -Dhttps.proxyPort=6050 -Dhttp.proxyUser=123456’ command in command line
8. Import the project in Eclipse "File > Import > Maven > Existing Maven Projects > Click Browse and select extracted folder > Finish"
9. Include logs to verify if main() method of SpringLearnApplication.
10. Run the SpringLearnApplication class.

# **CODE:**

## **HOMECONTROLLER.JAVA**

package com.example.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.ui.Model;

@Controller

public class HomeController {

    @RequestMapping("/")

    public String home(Model model) {

        model.addAttribute("message", "Welcome to Spring MVC!");

        return "home";

    }

}

## **WEB.XML:**

package com.example.controller;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.ui.Model;

@Controller

public class HomeController {

    @RequestMapping("/")

    public String home(Model model) {

        model.addAttribute("message", "Welcome to Spring MVC!");

        return "home";

    }

}

dispactcher-servlet.xml  
  
<beans xmlns="http://www.springframework.org/schema/beans"

       xmlns:mvc="http://www.springframework.org/schema/mvc"

       xmlns:context="http://www.springframework.org/schema/context"

       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

       xsi:schemaLocation="

          http://www.springframework.org/schema/beans

          http://www.springframework.org/schema/beans/spring-beans.xsd

          http://www.springframework.org/schema/context

          http://www.springframework.org/schema/context/spring-context.xsd

          http://www.springframework.org/schema/mvc

          http://www.springframework.org/schema/mvc/spring-mvc.xsd">

    <context:component-scan base-package="com.example.controller" />

    <mvc:annotation-driven />

    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">

        <property name="prefix" value="/WEB-INF/views/" />

        <property name="suffix" value=".jsp" />

    </bean>

</beans>

## **HOME.JSP:**

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>

<html>

  <head><title>Home Page</title></head>

  <body>

    <h2>${message}</h2>

  </body>

</html>

## **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.example</groupId>

    <artifactId>spring-web-demo</artifactId>

    <version>1.0-SNAPSHOT</version>

    <packaging>war</packaging>

    <name>Spring Web Demo</name>

    <dependencies>

*<!-- Spring Web MVC -->*

        <dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-webmvc</artifactId>

            <version>5.3.32</version>

        </dependency>

*<!-- JSTL -->*

        <dependency>

            <groupId>javax.servlet</groupId>

            <artifactId>jstl</artifactId>

            <version>1.2</version>

        </dependency>

*<!-- Servlet API -->*

        <dependency>

            <groupId>javax.servlet</groupId>

            <artifactId>javax.servlet-api</artifactId>

            <version>4.0.1</version>

            <scope>provided</scope>

        </dependency>

    </dependencies>

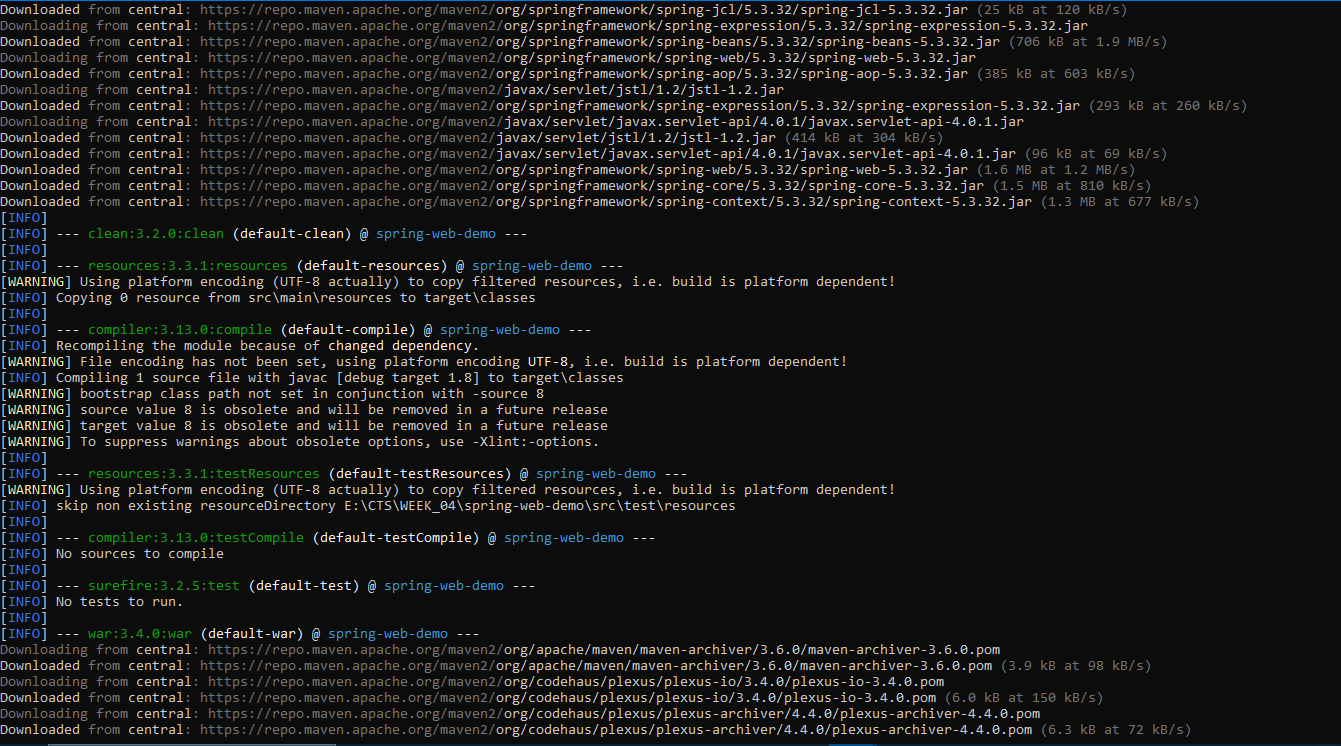
    <build>

        <finalName>spring-web-demo</finalName>

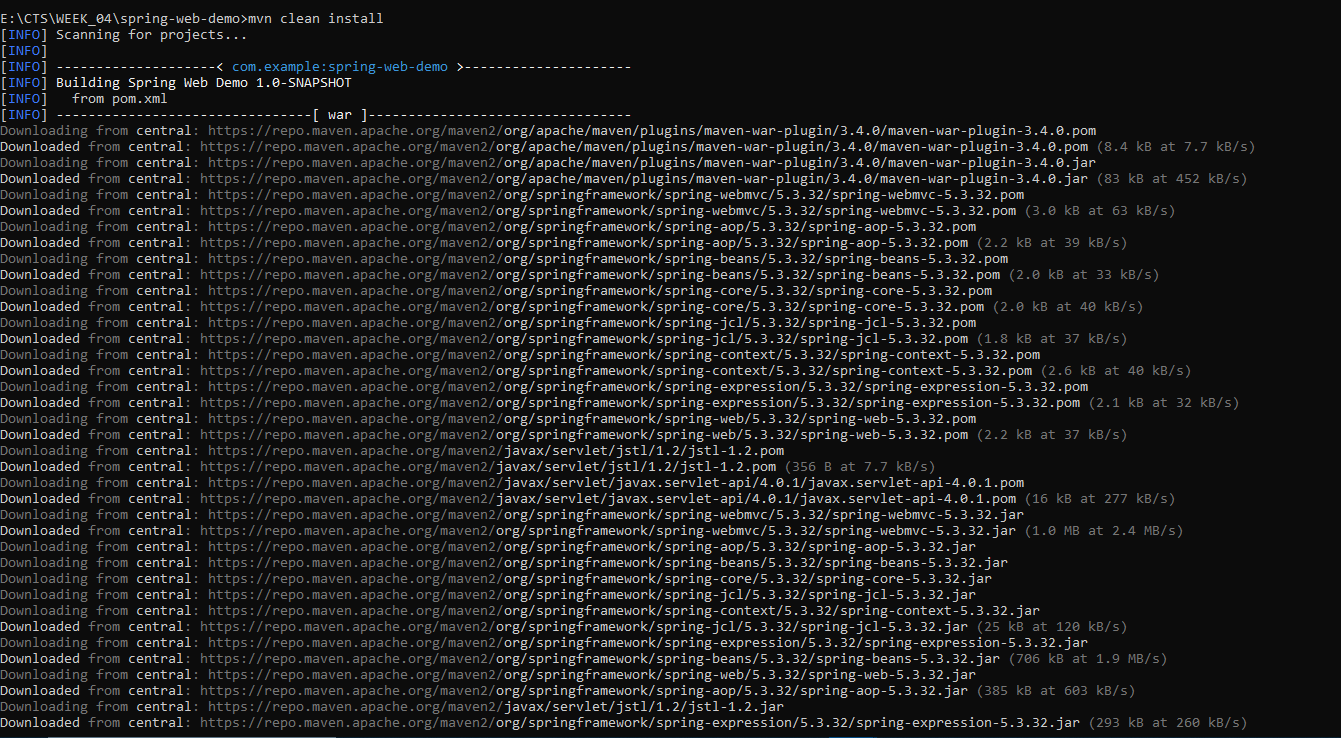
    </build>

</project>

# **OUTPUT**



A black screen with white lines

AI-generated content may be incorrect.

HANDS ON 2

# **SPRING CORE – LOAD SIMPLEDATEFORMAT FROM SPRING CONFIGURATION XML**

SimpleDateFormat with the pattern ‘dd/MM/yyyy’ is created in multiple places of an application. To avoid creation of SimpleDateFormat in multiple places, define a bean in Spring XML Configuration file and retrieve the date.  
  
Follow steps below to implement:

Create spring configuration file date-format.xml in src/main/resources folder of 'spring-learn' project

Open https://docs.spring.io/spring-framework/docs/current/spring-framework-reference/core.html#beans-factory-metadata

Copy the XML defined in the section of previous step URL and paste it into date-format.xml

Define bean tag in the XML with for date format. Refer code below.

# **CODE:**

## 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.example</groupId>

    <artifactId>spring-country-xml</artifactId>

    <version>1.0-SNAPSHOT</version>

    <dependencies>

*<!-- Spring Core -->*

        <dependency>

            <groupId>org.springframework</groupId>

            <artifactId>spring-context</artifactId>

            <version>5.3.32</version>

        </dependency>

    </dependencies>

</project>

## COUNTRY.JAVA:

package com.example;

public class Country {

    private String name;

    private String capital;

*// Getters and Setters*

    public String getName() {

        return name;

    }

    public void setName(String name) {

        this.name = name;

    }

    public String getCapital() {

        return capital;

    }

    public void setCapital(String capital) {

        this.capital = capital;

    }

    @Override

    public String toString() {

        return "Country{name='" + name + "', capital='" + capital + "'}";

    }

}

## COUNTRY.XML:

<?xml *version*="1.0" *encoding*="UTF-8"?>

<beans *xmlns*="http://www.springframework.org/schema/beans"

*xmlns:xsi*="http://www.w3.org/2001/XMLSchema-instance"

*xsi:schemaLocation*="

         http://www.springframework.org/schema/beans

         http://www.springframework.org/schema/beans/spring-beans.xsd">

    <bean *id*="country" *class*="com.example.Country">

        <property *name*="name" *value*="India"/>

        <property *name*="capital" *value*="New Delhi"/>

    </bean>

</beans>

## APP.JAVA:

package com.example;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

public class App {

    public static void main(String[] args) {

        ApplicationContext context = new ClassPathXmlApplicationContext("country.xml");

        Country country = context.getBean("country", Country.class);

        System.out.println("Loaded Country: " + country);

    }

}

# **OUTPUT:**

A screenshot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer

AI-generated content may be incorrect.