Aug 31st Lecture and Project spring-mvc-demo

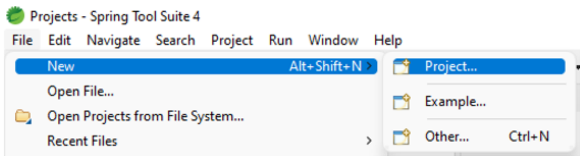
Create a Maven MVC project and deploy on an IDE Tomcat server. This document describes creating a MVC project in the SpringTool IDE. Other documents describes Tomcat server deployments within the IDE, on local machine, and on AWS.

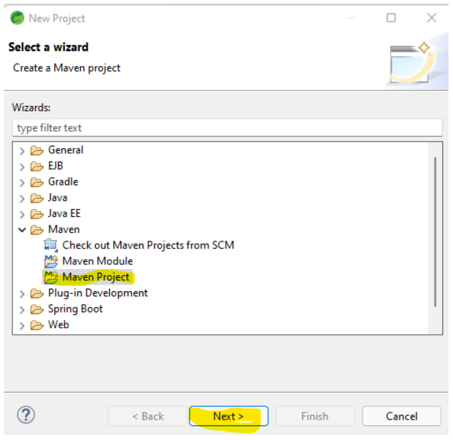
The Spring Framework is an application framework and inversion of control container for the Java platform. The framework's core features can be used by any Java application, but there are extensions for building web applications on top of the Java EE.

Table of Contents

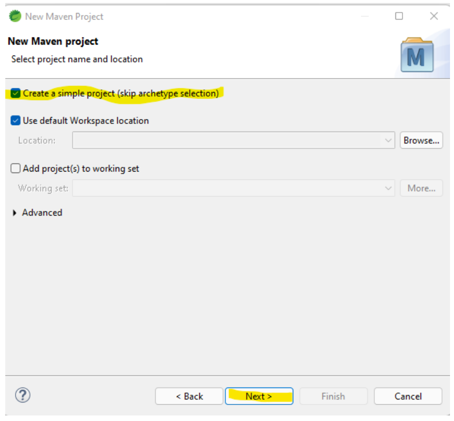
Create a Maven MVC project and deploy on an IDE Tomcat server

Create New Maven Project


Projects - Spring Tool Suite 4
File Edit Navigate Search Project Run Window Help
New Alt+Shift+N
Open File...
Open Projects from File System...
Recent Files
Project...
Example...
Other... Ctrl+N



Select "Create a simple project …"
Click "Next >"


New Maven Project

New Maven project
Select project name and location

- [x] Create a simple project (skip archetype selection)
- [x] Use default Workspace location

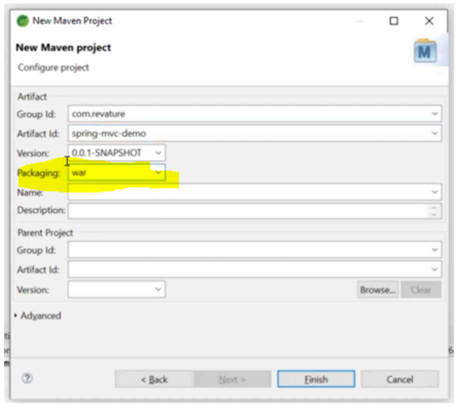
Location:

- [ ] Add project(s) to working set

Working set:

Advanced

Enter the Group Id: com.revature
Enter the Artifact Id: spring-mvc-demo
Select Packaging: **war**
Note: packaging must be war to deploy on a Tomcat server.
Click Finish


New Maven Project

New Maven project

Configure project

Artifact

Group Id: com.revature
Artifact Id: spring-mvc-demo
Version: 0.0.1-SNAPSHOT
Packaging: war
Name:
Description:

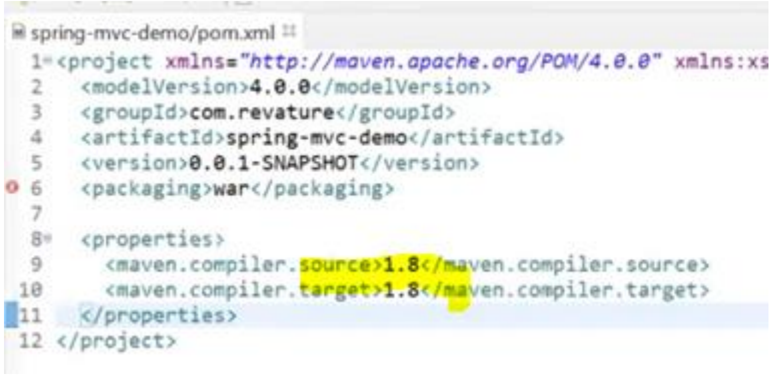
Parent Project

Group Id:
Artifact Id:
Version:

Update the pom.xml

Update Java Version

BEFORE PROCEEDING MAKE SURE THE PROJECT IS ON Java 1.8 in the pom.xml and has been rebuilt.



spring-mvc-demo/pom.xml

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xs
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.revature</groupId>
  <artifactId>spring-mvc-demo</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>

  <properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
  </properties>
</project>
```

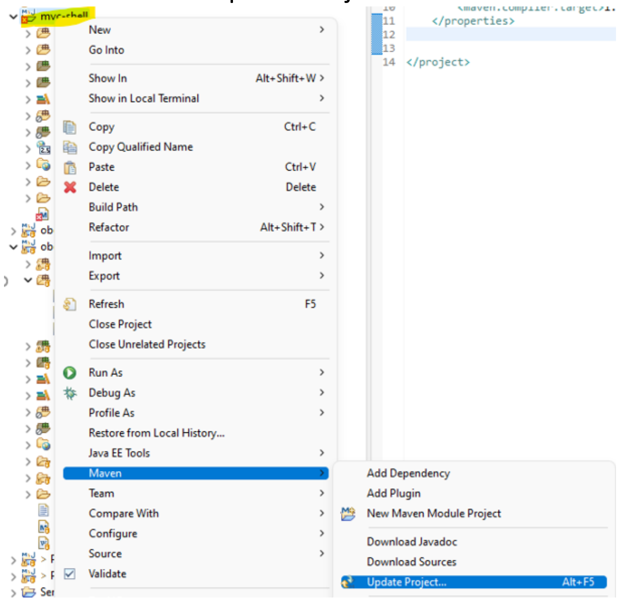
Add the Java 1.8 in the pom.xml

```
	<properties>
		<maven.compiler.source>1.8</maven.compiler.source>
		<maven.compiler.target>1.8</maven.compiler.target>
	</properties>
```

Rebuild the Maven project

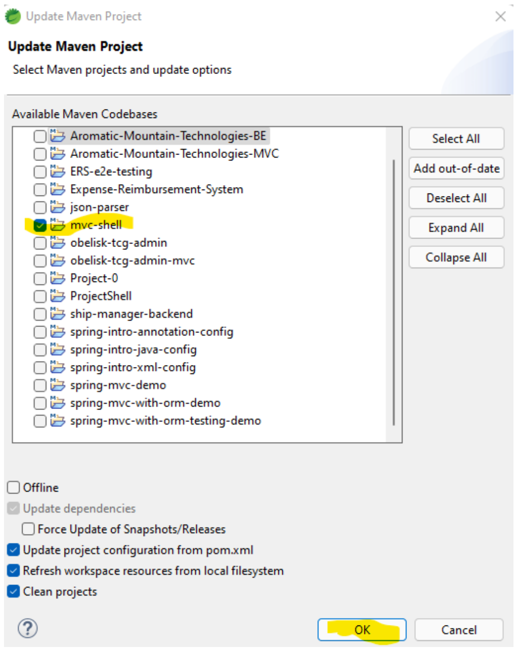
Right click on the Project

Select Maven ➔ Update Project…



```
  </properties>

</project>
```

The project should already be selected.
Click “OK”


Update Maven Project

Update Maven Project

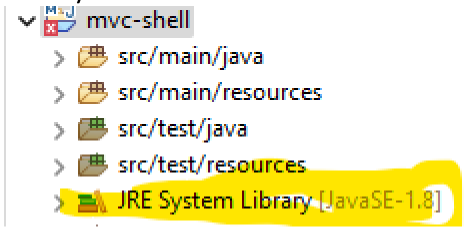
Select Maven projects and update options

Available Maven Codebases

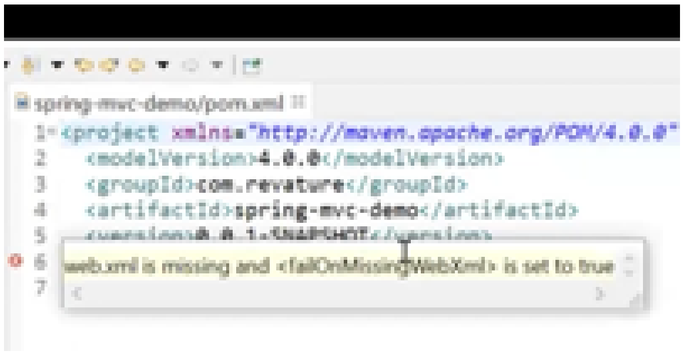
- [ ] Aromatic-Mountain-Technologies-BE
- [ ] Aromatic-Mountain-Technologies-MVC
- [ ] ERS-e2e-testing
- [ ] Expense-Reimbursement-System
- [ ] json-parser
- [x] mvc-shell
- [ ] obelisk-tcg-admin
- [ ] obelisk-tcg-admin-mvc
- [ ] Project-0
- [ ] ProjectShell
- [ ] ship-manager-backend
- [ ] spring-intro-annotation-config
- [ ] spring-intro-java-config
- [ ] spring-intro-xml-config
- [ ] spring-mvc-demo
- [ ] spring-mvc-with-orm-demo
- [ ] spring-mvc-with-orm-testing-demo

- [ ] Offline
- [x] Update dependencies
- [ ] Force Update of Snapshots/Releases
- [x] Update project configuration from pom.xml
- [x] Refresh workspace resources from local filesystem
- [x] Clean projects

Library should now be at the new Java version



Correct Error: Web.xml is missing



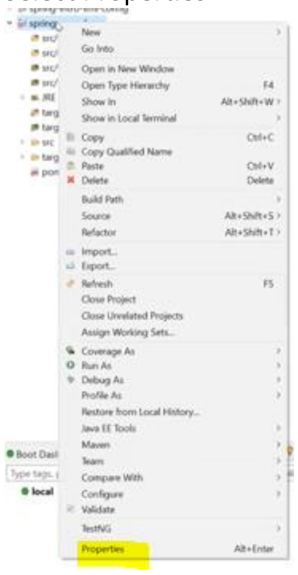
```
<project xmlns="http://maven.apache.org/POM/4.0.0"
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.revature</groupId>
  <artifactId>spring-mvc-demo</artifactId>
  <version>0.0.1-SNAPSHOT</version>
```

```
web.xml is missing and <failOnMissingWebXml> is set to true
```

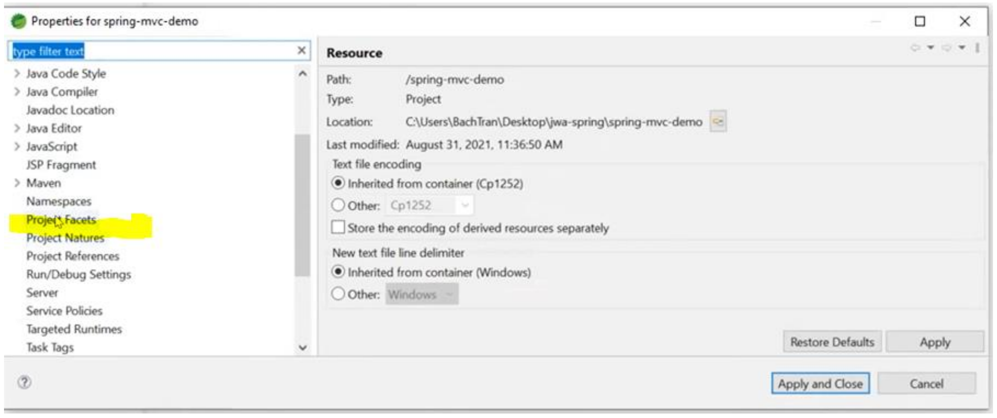
Need to generate Web.xml

Right click on project

Select Properties



Select Project Facets

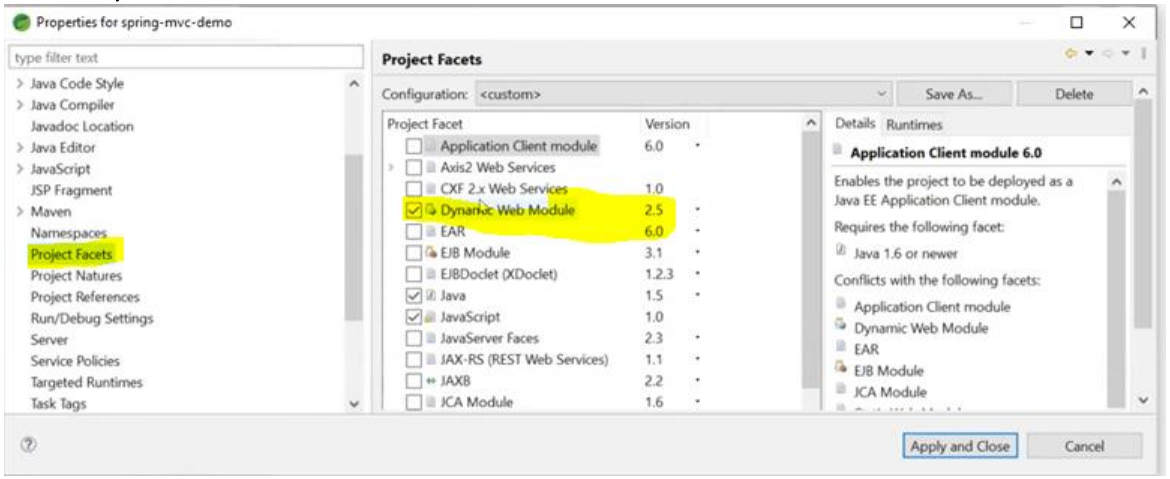


Path: /spring-mvc-demo
Type: Project
Location: C:\Users\BachTran\Desktop\jwa-spring\spring-mvc-demo
Last modified: August 31, 2021, 11:36:50 AM

Text file encoding
[✓ Inherited from container (Cp1252)]
Other: Cp1252
Store the encoding of derived resources separately

New text file line delimiter
[✓ Inherited from container (Windows)]
Other: Windows

Check “Dynamic Web Module”


Properties for spring-mvc-demo

type filter text

- Java Code Style
- Java Compiler
- Javadoc Location
- Java Editor
- JavaScript
- JSP Fragment
- Maven
- Namespaces
- Project Facets
- Project Natures
- Project References
- Run/Debug Settings
- Server
- Service Policies
- Targeted Runtimes
- Task Tags

Project Facets

Configuration: <custom>   Save As...   Delete

Project Facet	Version
☐ Application Client module	6.0
☐ Axis2 Web Services	
☐ CXF 2.x Web Services	1.0
☑ Dynamic Web Module	2.5
☐ EAR	6.0
☐ EJB Module	3.1
☐ EJBDoclet (XDoclet)	1.2.3
☑ Java	1.5
☑ JavaScript	1.0
☐ JavaServer Faces	2.3
☐ JAX-RS (REST Web Services)	1.1
☐ JAXB	2.2
☐ JCA Module	1.6

Details   Runtimes

Application Client module 6.0

Enables the project to be deployed as a Java EE Application Client module.

Requires the following facet:

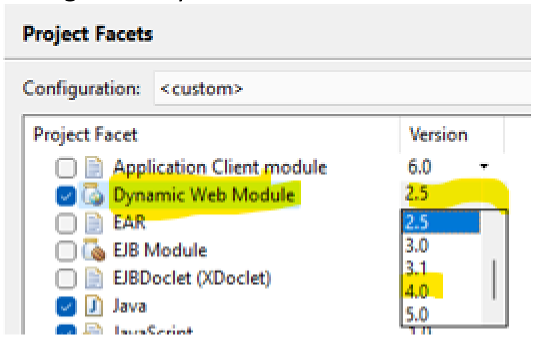
Java 1.6 or newer

Conflicts with the following facets:

Application Client module
Dynamic Web Module
EAR
EJB Module
JCA Module

Apply and Close   Cancel

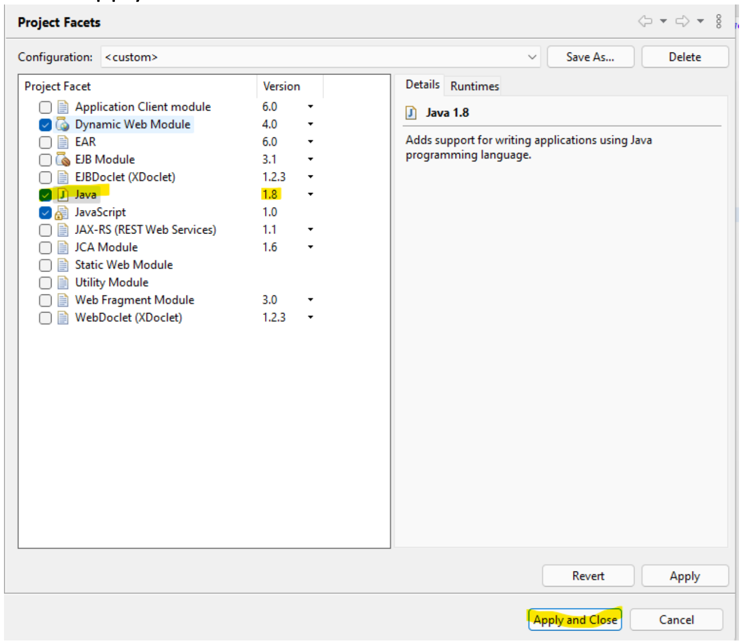
Change the “Dynamic Web Module” to 4.0


Project Facets

Configuration: <custom>

Project Facet	Version
☐ Application Client module	6.0
☑ Dynamic Web Module	2.5
☐ EAR	
☐ EJB Module	
☐ EJBDoclet (XDoclet)	
☑ Java	

Change to new version Java 1.8 or newer version (if not already set)
Click “Apply and Close”


Project Facets

Configuration: <custom>   Save As...   Delete

Project Facet	Version
☐ Application Client module	6.0
☑ Dynamic Web Module	4.0
☐ EAR	6.0
☐ EJB Module	3.1
☐ EJBDoclet (XDoclet)	1.2.3
☑ Java	1.8
☑ JavaScript	1.0
☐ JAX-RS (REST Web Services)	1.1
☐ JCA Module	1.6
☐ Static Web Module	
☐ Utility Module	
☐ Web Fragment Module	3.0
☐ WebDoclet (XDoclet)	1.2.3

Details   Runtimes

Java 1.8

Adds support for writing applications using Java programming language.

Revert   Apply

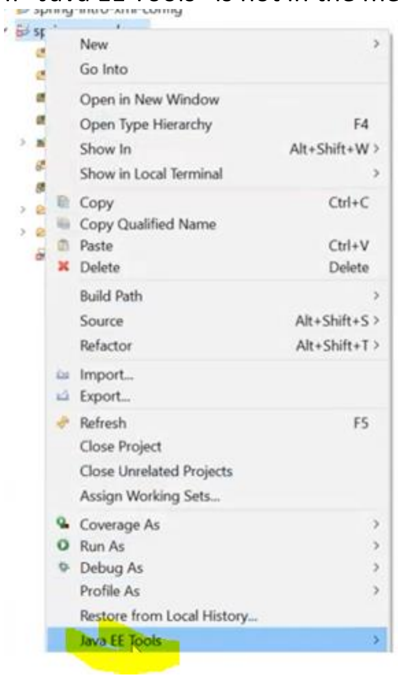
Apply and Close   Cancel

Install Java EE Tool if needed

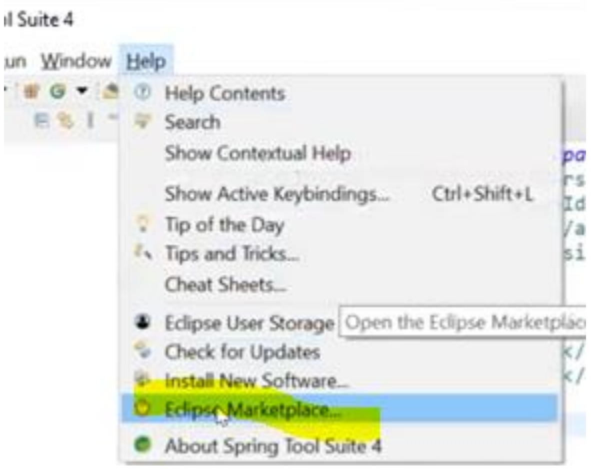
Right click on project

If “Java EE Tools” is in the menu list skip to “**Generate Deployment Descriptor Stub**”

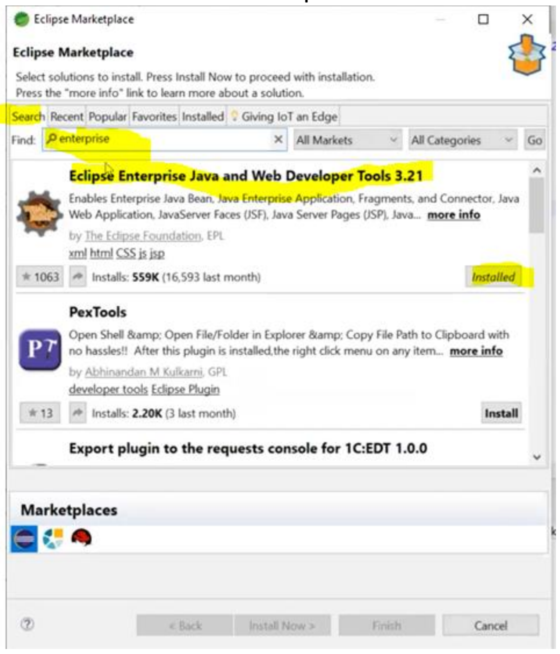
If “Java EE Tools” is not in the menu list follow the next two steps



<u>Step 1: Install via Eclipse Market Place under the Help menu</u>



Step 2: Search tab: enter enterprise
If not already installed, install the component
Once the installation completes restart the IDE for the changes to take affect



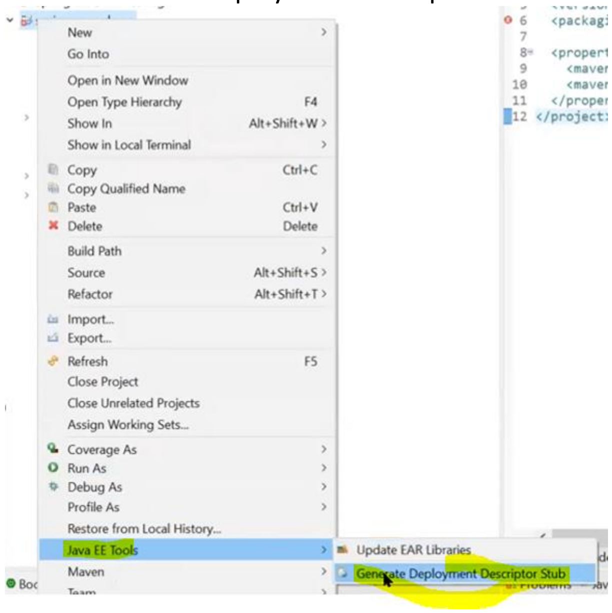
Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Eclipse Enterprise Java and Web Developer Tools 3.21
Enables Enterprise Java Bean, Java Enterprise Application, Fragments, and Connector, Java Web Application, JavaServer Faces (JSF), Java Server Pages (JSP), Java... more info
by The Eclipse Foundation, EPL
xml html CSS js jsp
★ 1063 Installs: 559K (16,593 last month)

PexTools
Open Shell &amp; Open File/Folder in Explorer &amp; Copy File Path to Clipboard with no hassles!! After this plugin is installed,the right click menu on any item... more info
by Abhinandan M Kulkarni, GPL
developer tools Eclipse Plugin
★ 13 Installs: 2.20K (3 last month)

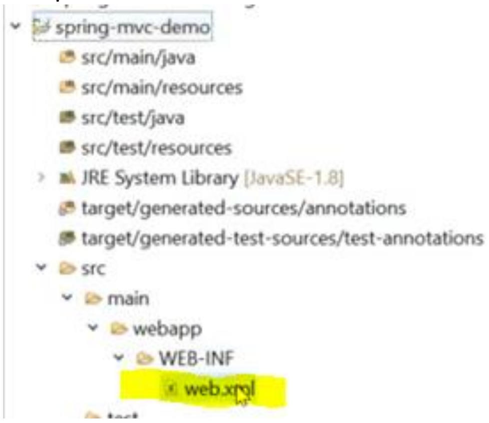
Export plugin to the requests console for 1C:EDT 1.0.0

Generate Deployment Descriptor Stub
Right click on the project
Select Java EE Tools
Select Generate Deployment Descriptor Stub

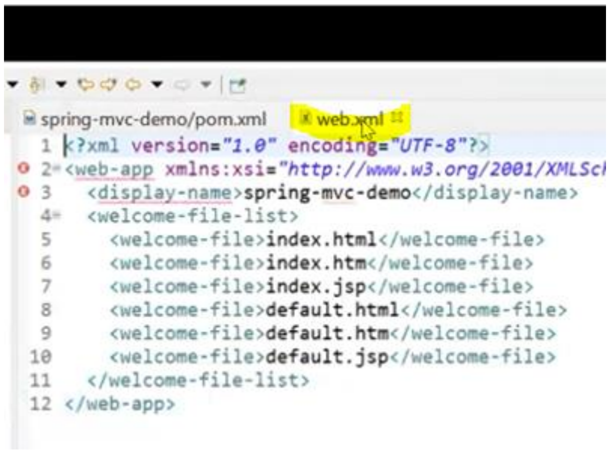


```
<packag
<propert
  <maver
  <maver
</proper
</project
```

The web.xml is generated in following location.
The pom.xml error should now be resolved.



```
spring-mvc-demo
  src/main/java
  src/main/resources
  src/test/java
  src/test/resources
  JRE System Library [JavaSE-1.8]
  target/generated-sources/annotations
  target/generated-test-sources/test-annotations
  src
    main
      webapp
        WEB-INF
          web.xml
```

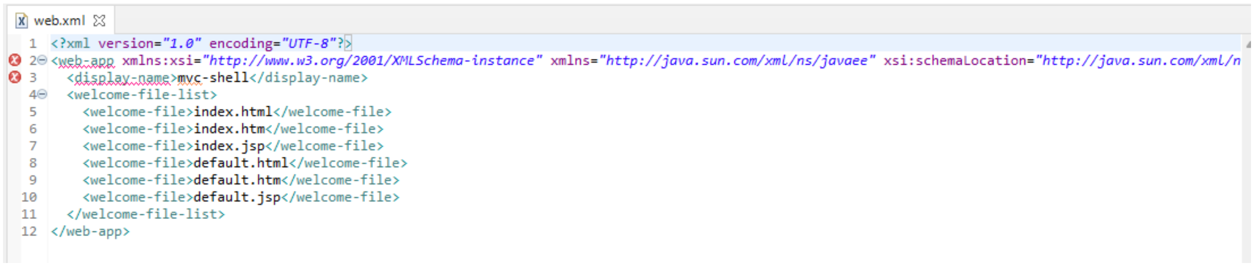


spring-mvc-demo/pom.xml   web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSc
  <display-name>spring-mvc-demo</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

Correct web.xml error

Open Web.xml File



web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee" xsi:schemaLocation="http://java.sun.com/xml/n
  <display-name>mvc-shell</display-name>
  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

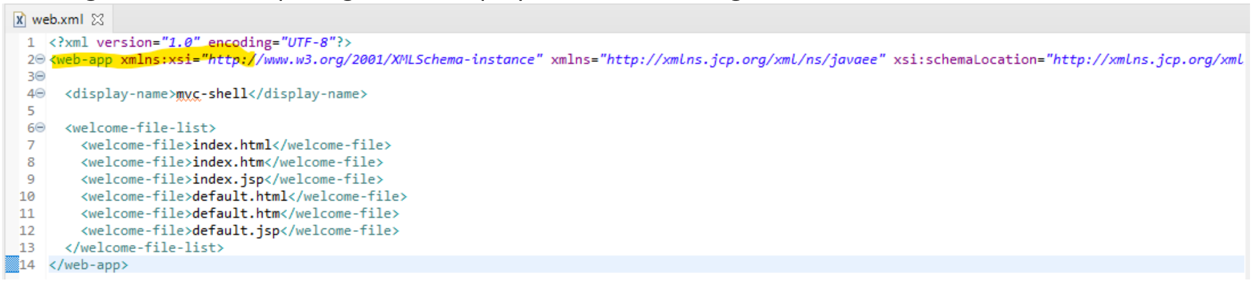
Update outdated link:
From:

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
version="2.5">
```

To:

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" version="4.0">
```

You might need to do spacing on the display name as well to get rid of the errors.


web.xml

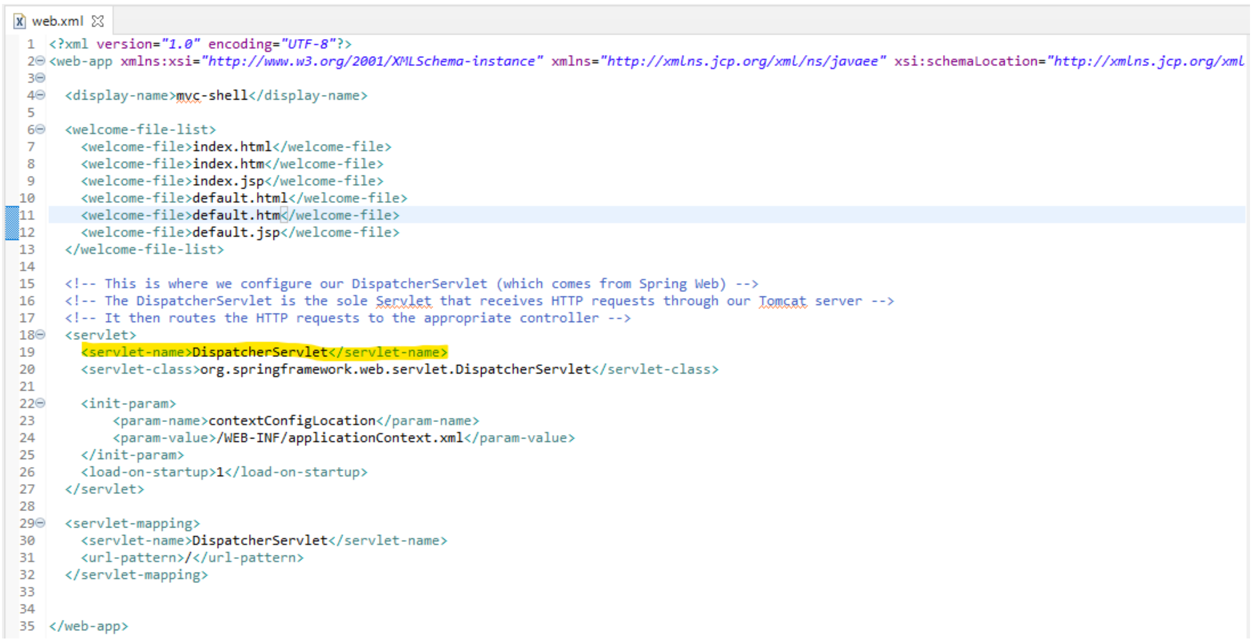
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml

  <display-name>mvc-shell</display-name>

  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>
</web-app>
```

Update the web.xml file

The web.xml file is used for the Tomcat server configuration.

Add DispatcherServlet to web.xml

```
  <!-- This is where we configure our DispatcherServlet (which comes from Spring Web) -->
  <!-- The DispatcherServlet is the sole Servlet that receives HTTP requests through our Tomcat
server -->
  <!-- It then routes the HTTP requests to the appropriate controller -->
  <servlet>
  	<servlet-name>DispatcherServlet</servlet-name>
  	<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
  	
  	<init-param>
  		<param-name>contextConfigLocation</param-name>
  		<param-value>/WEB-INF/applicationContext.xml</param-value>
  	</init-param>
  	<load-on-startup>1</load-on-startup>
  </servlet>
  
  <servlet-mapping>
  	<servlet-name>DispatcherServlet</servlet-name>
  	<url-pattern>/</url-pattern>
  </servlet-mapping>
```



web.xml

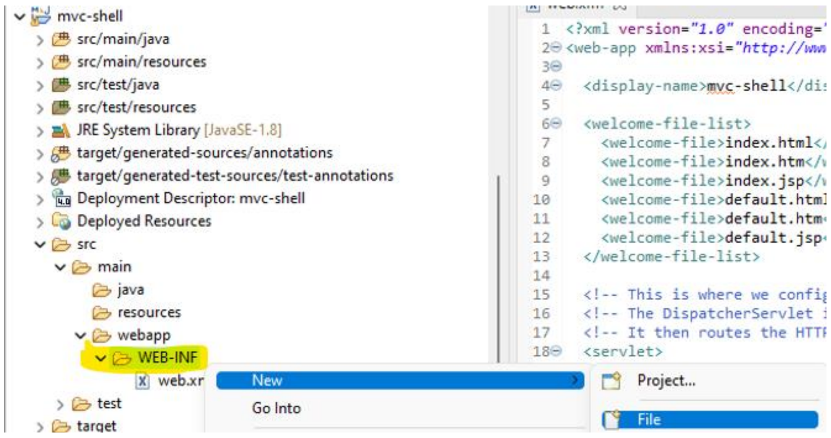
```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml

  <display-name>mvc-shell</display-name>

  <welcome-file-list>
    <welcome-file>index.html</welcome-file>
    <welcome-file>index.htm</welcome-file>
    <welcome-file>index.jsp</welcome-file>
    <welcome-file>default.html</welcome-file>
    <welcome-file>default.htm</welcome-file>
    <welcome-file>default.jsp</welcome-file>
  </welcome-file-list>

  <!-- This is where we configure our DispatcherServlet (which comes from Spring Web) -->
  <!-- The DispatcherServlet is the sole Servlet that receives HTTP requests through our Tomcat server -->
  <!-- It then routes the HTTP requests to the appropriate controller -->
  <servlet>
    <servlet-name>DispatcherServlet</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>

    <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/applicationContext.xml</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>

  <servlet-mapping>
    <servlet-name>DispatcherServlet</servlet-name>
    <url-pattern>/</url-pattern>
  </servlet-mapping>


</web-app>
```

All http request goes to the DispatcherServlet which in turn sends to the controller(s) endpoints.
“Front Controller Design Pattern”

Create an applicationContext.xml File

Create the applicationContext.xml file in the directory:
spring-mvc-demo\src\main\webapp\WEB-INF\

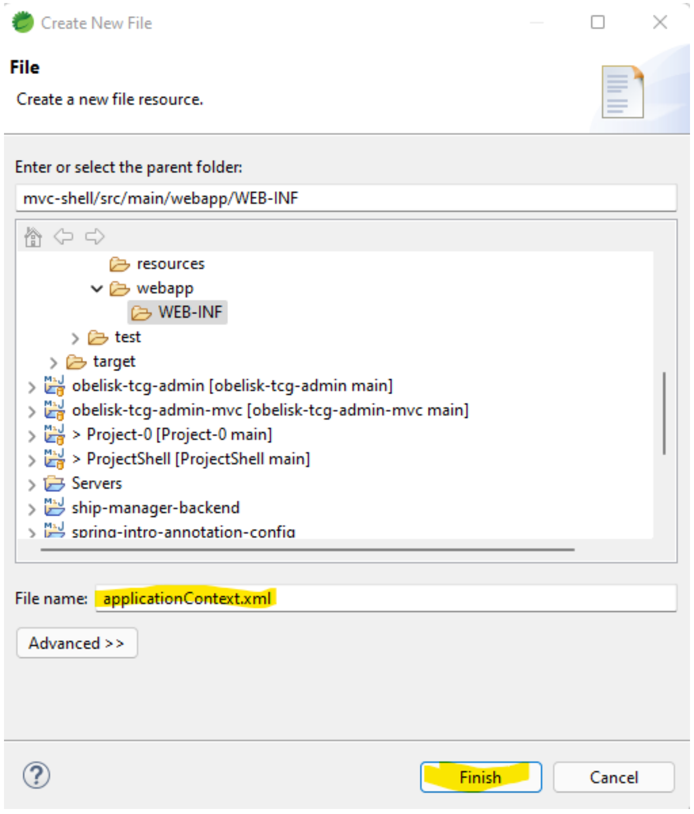
Right click on WEB-INF folder
Select New ➔ File



```
mvc-shell
  src/main/java
  src/main/resources
  src/test/java
  src/test/resources
  JRE System Library [JavaSE-1.8]
  target/generated-sources/annotations
  target/generated-test-sources/test-annotations
  Deployment Descriptor: mvc-shell
  Deployed Resources
  src
    main
      java
      resources
      webapp
        WEB-INF
          web.xr
    test
  target
```

```
<?xml version="1.0" encoding='
<web-app xmlns:xsi="http://ww

  <display-name>mvc-shell</di

  <welcome-file-list>
    <welcome-file>index.html</
    <welcome-file>index.htm</
    <welcome-file>index.jsp</
    <welcome-file>default.html
    <welcome-file>default.htm
    <welcome-file>default.jsp
  </welcome-file-list>

  <!-- This is where we confi
  <!-- The DispatcherServlet i
  <!-- It then routes the HTT
  <servlet>
```

Enter the “File name:” applicationContext.xml
Click “Finish”



Create New File

File

Create a new file resource.

Enter or select the parent folder:

mvc-shell/src/main/webapp/WEB-INF

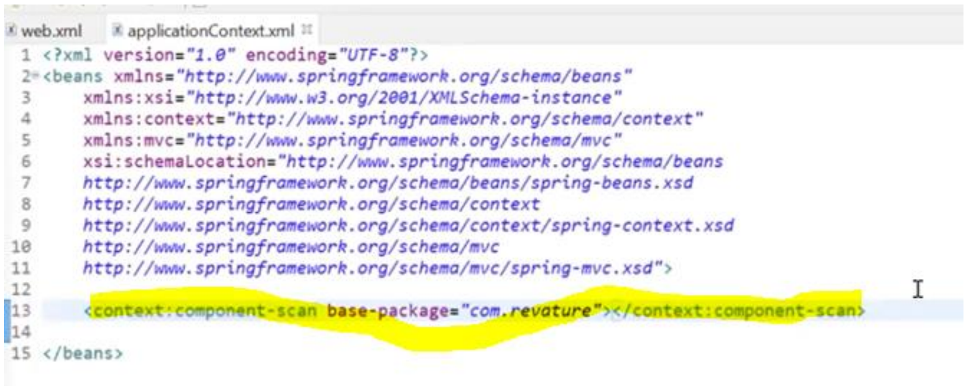
```
      resources
      webapp
        WEB-INF
    test
  target
obelisk-tcg-admin [obelisk-tcg-admin main]
obelisk-tcg-admin-mvc [obelisk-tcg-admin-mvc main]
> Project-0 [Project-0 main]
> ProjectShell [ProjectShell main]
Servers
ship-manager-backend
spring-intro-annotation-config
```

File name: applicationContext.xml

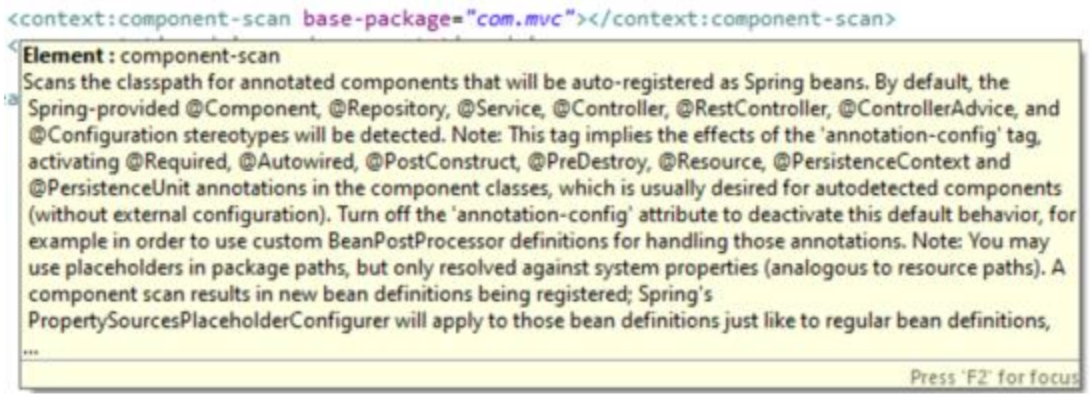
Update the applicationContext.xml File

Add the following to the xml file:

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        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.revature"></context:component-scan>
        <mvc:annotation-driven></mvc:annotation-driven>

</beans>
```



```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    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.revature"></context:component-scan>

</beans>
```



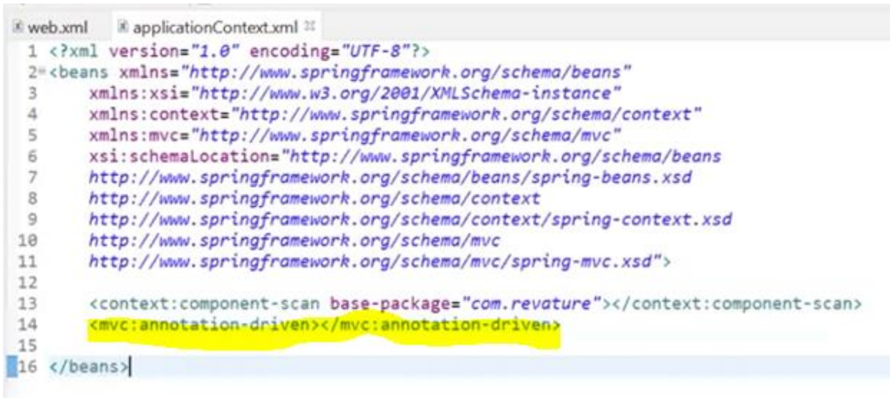
```
<context:component-scan base-package="com.mvc"></context:component-scan>
```

Element : component-scan
Scans the classpath for annotated components that will be auto-registered as Spring beans. By default, the Spring-provided @Component, @Repository, @Service, @Controller, @RestController, @ControllerAdvice, and @Configuration stereotypes will be detected. Note: This tag implies the effects of the 'annotation-config' tag, activating @Required, @Autowired, @PostConstruct, @PreDestroy, @Resource, @PersistenceContext and @PersistenceUnit annotations in the component classes, which is usually desired for autodetected components (without external configuration). Turn off the 'annotation-config' attribute to deactivate this default behavior, for example in order to use custom BeanPostProcessor definitions for handling those annotations. Note: You may use placeholders in package paths, but only resolved against system properties (analogous to resource paths). A component scan results in new bean definitions being registered; Spring's PropertySourcesPlaceholderConfigurer will apply to those bean definitions just like to regular bean definitions, ...

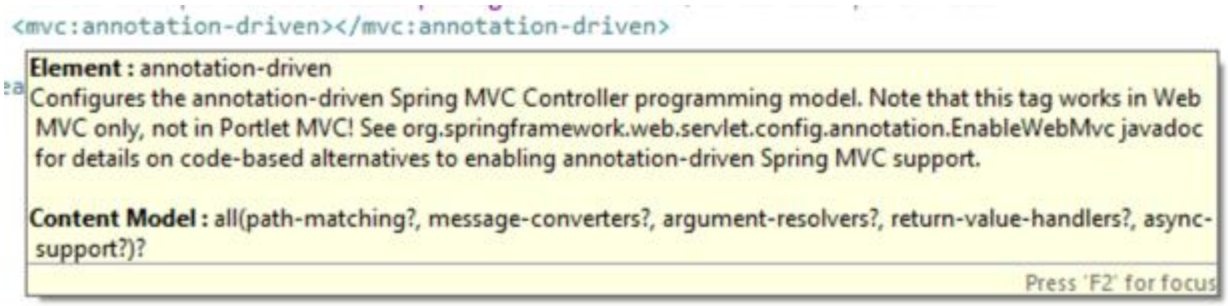
Make sure to update the base package: com.revature to the project's definition.

Make sure to add the <mcv:annotation> tags

mcv:annotation allows for the @Component, @Service and @RestController tags.



```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:context="http://www.springframework.org/schema/context"
    xmlns:mvc="http://www.springframework.org/schema/mvc"
    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.revature"></context:component-scan>
    <mvc:annotation-driven></mvc:annotation-driven>

</beans>
```



```
<mvc:annotation-driven></mvc:annotation-driven>
```

Element : annotation-driven
Configures the annotation-driven Spring MVC Controller programming model. Note that this tag works in Web MVC only, not in Portlet MVC! See org.springframework.web.servlet.config.annotation.EnableWebMvc javadoc for details on code-based alternatives to enabling annotation-driven Spring MVC support.

Content Model : all(path-matching?, message-converters?, argument-resolvers?, return-value-handlers?, async-support?)?

Add other project dependencies to pom.xml

Add dependencies tags

Between the dependencies tags

- Add dependencies for Spring Framework
- Add dependencies for Tomcat
- Add dependencies for other common project items (will vary depending on project):
    - hibernate
    - mariadb
    - junit.jupiter
    - h2database
    - mockito
    - logback

```
<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
https://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mvc</groupId>
  <artifactId>mvc-shell</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>

        <properties>
                <maven.compiler.source>1.8</maven.compiler.source>
                <maven.compiler.target>1.8</maven.compiler.target>
        </properties>

        <dependencies>
                <!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>javax.servlet-api</artifactId>
                        <version>4.0.1</version>
                        <scope>provided</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-
databind -->
                <dependency>
                        <groupId>com.fasterxml.jackson.core</groupId>
                        <artifactId>jackson-databind</artifactId>
                        <version>2.12.5</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-webmvc</artifactId>
                        <version>5.3.9</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
                <dependency>
                        <groupId>org.projectlombok</groupId>
                        <artifactId>lombok</artifactId>
                        <version>1.18.20</version>
                        <scope>provided</scope>
                </dependency>

                <!-- Database Related dependencies -->
                <!-- https://mvnrepository.com/artifact/org.springframework/spring-orm -->
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-orm</artifactId>
                        <version>5.3.9</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->
                <dependency>
                        <groupId>org.hibernate</groupId>
                        <artifactId>hibernate-core</artifactId>
                        <version>5.5.7.Final</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.apache.tomcat/tomcat-dbcp -->
                <dependency>
                        <groupId>org.apache.tomcat</groupId>
                        <artifactId>tomcat-dbcp</artifactId>
                        <version>10.0.10</version>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.mariadb.jdbc/mariadb-java-client -->
                <dependency>
                        <groupId>org.mariadb.jdbc</groupId>
                        <artifactId>mariadb-java-client</artifactId>
                        <version>2.7.4</version>
                </dependency>

                <!-- Testing related dependencies -->
                <!-- https://mvnrepository.com/artifact/org.springframework/spring-test -->
                <dependency>
                        <groupId>org.springframework</groupId>
                        <artifactId>spring-test</artifactId>
                        <version>5.3.9</version>
                        <scope>test</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.junit.jupiter/junit-jupiter-api -->
                <dependency>
                        <groupId>org.junit.jupiter</groupId>
                        <artifactId>junit-jupiter-api</artifactId>
                        <version>5.8.0-M1</version>
                        <scope>test</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/com.h2database/h2 -->
                <dependency>
                        <groupId>com.h2database</groupId>
                        <artifactId>h2</artifactId>
                        <version>1.4.200</version>
                        <scope>test</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.mockito/mockito-inline -->
                <dependency>
                        <groupId>org.mockito</groupId>
                        <artifactId>mockito-inline</artifactId>
                        <version>3.12.4</version>
                        <scope>test</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.mockito/mockito-junit-jupiter -->
                <dependency>
                        <groupId>org.mockito</groupId>
                        <artifactId>mockito-junit-jupiter</artifactId>
                        <version>3.12.4</version>
                        <scope>test</scope>
                </dependency>
                <!-- https://mvnrepository.com/artifact/org.skyscreamer/jsonassert -->
                <dependency>
                        <groupId>org.skyscreamer</groupId>
                        <artifactId>jsonassert</artifactId>
                        <version>1.5.0</version>
                        <scope>test</scope>
                </dependency>


                <!-- dependency definitions from logging-demo -->
                <dependency>
                        <groupId>ch.qos.logback</groupId>
                        <artifactId>logback-classic</artifactId>
                        <version>1.2.4</version>
                </dependency>
                <dependency>
                        <groupId>org.slf4j</groupId>
                        <artifactId>slf4j-api</artifactId>
                        <version>1.7.32</version>
                </dependency>

        </dependencies>

</project>
```

Complete the MVC demo project

The above is the basic configuration for any Spring MVC project.

To complete the spring-mvc-demo project add the following packages and file.

\spring-mvc-demo\src\main\java\

Package: com\revature\controller

TestController.java

TestController

Package: com\revature\dto

LoginDTO.java

LoginDTO

Spring MVC Project Next Steps for Tomcat Server(s)

After completing the code in this document, follow the steps in the "Tomcat 02 IDE Setup Aug 31st" Setup document which walks through the steps to create a Tomcat server in the Spring Tool IDE and run this MVC based applications.

Another Tomcat document "Tomcat 03 localhost Deploy war Sep 8th" detail Tomcat setup on a local computer, deploy and run an MVC application war file and connect to the application through localhost.

Another Tomcat document "Tomcat 04 AWS EC2 Sep 8th" detail Tomcat installation and setup on an AWS EC2 instance, deploy and run an MVC application war file and connect to the application through an AWS URL.

Appendix: Recording Transcript for This Document

Bach Tran: Alright, so let's see let's go ahead and try to configure spring mvc so um.
Bach Tran: yeah first step to do that would be.
Bach Tran: To create a new project, so I think everybody should follow, along with this, just so that you can get like the basic skeleton core.
Bach Tran: Creating your project to back end.
Bach Tran: Because there are some configuration steps are definitely involved here so new go to New let's go to other and to maven project.
Bach Tran: Right so.
Bach Tran: COM dot ravaged.
Bach Tran: And we'll name this project spring mvc DEMO.
Bach Tran: The difference here is for the packaging, instead of it being a jar it's going to be a war file.
Bach Tran: So you do want to make sure to change from jar to war.
Bach Tran: Because a jar that's a Java archive file, while a war file is a web archive file, this is what's required to actually deploy your application onto tomcat the server.
Bach Tran: So yeah you do need to have this water file that will be deployed tomcat essentially.
Bach Tran: So whenever you build your project it's going to generate the war file and then boom that's when you actually deploy it to the tomcat server.
Bach Tran: So yeah let's click finish.
Bach Tran: And once I do that and go to the poem dot xml it's actually going to complain that web dot xml is missing and fail on missing web xml is set to true that's actually.
Bach Tran: Good and that's because we need to generate the Web dot xml file for this project, so the first step here, I will do is right click on the project.
Bach Tran: Go to property.
Bach Tran: And once you are inside of the properties, you want to go to project facets.
Bach Tran: Right so you'll click on project facets and you'll see a section called dynamic web module that should already be checked.
Bach Tran: But what we want to do is change the version because 2.5 is pretty old will want to change over to version, for in our case.
Bach Tran: Which is support for Java service API requiring Java one page or do it right so 4.0 is what we want to change this version two.
Bach Tran: and
Bach Tran: yeah let's see if it.
Bach Tran: Oh yeah.
Bach Tran: it's gonna give me an error here, it requires Java one plant or new and it won't let me apply we're missing a step here, actually, we need to go back you.
Bach Tran: configure our properties to.
Bach Tran: 1.8.
Bach Tran: So we do this step for.
Bach Tran: So you'll want to have these properties.
Bach Tran: And you will want to right click.
Bach Tran: maven update project.
Bach Tran: And now it's on 1.8 now it's still going to complain about the Web dot xml missing, but we just want to right click on the project again go to properties and go back to the dynamic web module and set the version of 4.0 and now it's good to go, so we can actually apply.
Bach Tran: apply and close.
Bach Tran: All right, once that's done, you will want to right click on the project again go to Java EE tool.
Bach Tran: And just to make sure like it might actually be missing from some of your ID is it missing from anybody.
Nicholas Hailey: mine.
Nicholas Hailey: yeah were you right click.
Bach Tran: Does anybody else have it or is everybody's missing.
Bach Tran: Okay yeah.
Bach Tran: Maybe maybe it's missing from most people.
So.
Bach Tran: The thing that you have to do here is go to help and eclipse marketplace.
Bach Tran: And we want to install a plug in here for.
Bach Tran: Having the Java EE tools pump up.
David Huynh: i'm actually getting the air with my properties, this is property built air non parcel palm.
Bach Tran: That is.
Bach Tran: that's a different issue let's see.
Bach Tran: yeah you want to share your screen.
Bach Tran: I think you're just missing the.
Bach Tran: That should go inside of the project tak.
David Huynh: Oh.
Bach Tran: yeah.
David Huynh: that's my bed, thank you.

Bach Tran: yeah that's it alright cool.
Bach Tran: Alright yeah so um basically the plugin that you're going to need to install would be the eclipse enterprise Java web developer tool.
Bach Tran: Let me make sure, let me look at some of the other ones.
Bach Tran: yeah so this one right here so go over to search and.
Bach Tran: let's do that search for enterprise so just in the search bar.
Bach Tran: And that should have this pop up right here so eclipse enterprise Java web developer tools you're going to want to just have this installed basically.
Nicholas Hailey: luck, I see that your says 3.1.
Nicholas Hailey: Much of a difference.
Bach Tran: Oh yeah it shouldn't matter too much just you'll just need to have this installed basically.
McCabe Sommers: Do we needed change and installation settings or do we just confirm.
Bach Tran: Ah, I think you have what looks actually see what it looks like.
McCabe Sommers: Okay.
McCabe Sommers: Find how right you're here okay it's multiple screens to get me.
McCabe Sommers: boom boom.
McCabe Sommers: Okay, it was looking like to me I guess.
yeah.
Bach Tran: yeah I think yeah you can just click confirm on now, you need the on one.
awesome.
Yves Bouele: By the way, see that much a passage gst I don't have gst my in my list.
Bach Tran: And yeah just just have the ones are required check and I think you should be here to.
Bach Tran: The only thing we need this for is the Java you tools to actually generate the Web xml and that's pretty much all we need.
Bach Tran: None of the other stuff is particularly important for us.
Bach Tran: But once that's installed then.
Bach Tran: You should have Java EE tools in the drop down whenever you right click on the project.
Bach Tran: And then the one that you want to click is generate deployment descriptor Stub.
Bach Tran: And what that will do is just create that web dot xml file and so now my palm is no longer complaining.
Bach Tran: Because the Web dot xml is now generated so it's inside of source, so the source folder down here it's inside the main and then it's inside of a folder called web APP that was awesome created it's a web APP and then web in web dot xml is right there.
Bach Tran: So if we open up.
Bach Tran: The web dot xml.
Bach Tran: it's actually going to complain here.
Bach Tran: That is because.
Bach Tran: Well, let me, let me check some settings real quick, just to be sure, for point, though.
Bach Tran: yeah alright that's good um let's see so.
Bach Tran: yeah it's going to say right here in valley element name that's because i'm on an older version for my schema location here, so I actually want to change this from 2.5 to 4.0.
Bach Tran: manually.
Bach Tran: So we'll just do that.
Bach Tran: and
Bach Tran: Now let's see if that works.
Bach Tran: cannot find declaration of elements well.
Bach Tran: Let me see.
Bach Tran: This is probably not the right.
Bach Tran: yeah.
Bach Tran: These wings are old now doesn't work anymore, let me see you see this is still about link.
Bach Tran: open up this in the browser.
Bach Tran: job.
Bach Tran: Okay, this is the new link right here.
Bach Tran: So let me.
Bach Tran: copy and paste it right here.
Bach Tran: The namespace will be.
Bach Tran: This location, right here.
Bach Tran: and
Bach Tran: that's why right here.
Bach Tran: Right.
Bach Tran: The target namespace schema dot.
Bach Tran: space.
Nicholas Hailey: One Eric Jamie.
Bach Tran: I remember, I tried this before, but it was actually generating the xml not the older.
Bach Tran: huh.
Bach Tran: Let me try deleting this file real quick.
Bach Tran: generating the outdated version.
Bach Tran: Even though I.
Bach Tran: owed it to us 4.0 right here.
interesting.

Bach Tran: So it's trying to generate the 2.5 version.
Bach Tran: refreshed.
Bach Tran: Java EE tools generate a descriptor.
Bach Tran: All right, let's let that load.
Bach Tran: see if it gives the right version.
Bach Tran: And this time I actually did that's pretty interesting.
Bach Tran: One air in Dallas he seven.
Bach Tran: Let me just say this file it's going to go away.
Bach Tran: yeah alright cool all the errors are.
Bach Tran: Gone that is should be correct, so if you.
Bach Tran: have something like this, let me put it in the chat you can actually just copy and paste this in directly.
Bach Tran: And it should be exactly the structure here.
Bach Tran: So what is this file, for it is a file that tells Tom cat.
Bach Tran: How to actually work with this project, how to start up the application and get things running so instead of having a main method.
Bach Tran: The way that this works is that this web dot xml is like the entry point to our application instead within the tomcat.
Bach Tran: environment so inside of here, we need to specify a service to actually you.
Bach Tran: To basically start up our application context and all that is required.
Bach Tran: So.
Bach Tran: yeah let's actually go over to define some services so.
Bach Tran: We need to create a service that tag here.
Bach Tran: And there is a service that.
Bach Tran: Are the basic idea of a circle, it is it's some kind of object.
Bach Tran: Within Java EE that will take in a http request and send a response.
Bach Tran: it's very similar to controllers.
Bach Tran: In that sense, but in the case of spring mvc we just have a single server.
Bach Tran: That route, the requests so it's still receives the requests just this single serve what but it routes, the requests to the appropriate controller objects that are annotate with at controller So here we need to map the dispatcher certainly, so let me put in the comments here.
Bach Tran: We need to this is where we configure our dispatcher surplus.
Bach Tran: which comes from spring Web.
Bach Tran: In order.
Bach Tran: yeah yeah, so this is where we can figure out this backer service which comes from sort of what.
Bach Tran: The dispatcher serve, what is the sole survivor it.
Bach Tran: That receives http requests.
Bach Tran: Through our tomcat server.
Bach Tran: Then routes, the http requests to the appropriate.
Bach Tran: Control.
Bach Tran: So here we will give our service.
Bach Tran: And name will call it.
Bach Tran: The dispatcher server.
Bach Tran: And we need to specify a class or this certainly org dot spring framework dot web dot servile it dot dispatcher circle right notice no this package name here, this is coming from spring framework, so we need to actually grab the dependency.
Bach Tran: Right now, we don't have it so well we'll fill out our palm not xml after we do this configuration here.
Nicholas Hailey: To get a question.
Bach Tran: yeah let's take a look here.
Bach Tran: What did we have to do to remove the red line.
Bach Tran: Under the.
Bach Tran: web APP part in the Web yeah it's just a bug like all I did was put in an extra line and say disappear.
Bach Tran: So it's not really an issue it's just like.
Bach Tran: The ID freaking out here are.
Bach Tran: reminders just taking a minute, for it to go yeah yeah so it's just like this inconsistent air that this pops up sometimes.
Bach Tran: it's not really a problem.
Yves Bouele: For me, are going back up everything stopped by wake up or don't have anything after Labor and I have a.
Yves Bouele: Delay in my POM xml.
Yves Bouele: here.
Yves Bouele: And I actually have to.
Yves Bouele: Do it to.
Bach Tran: save your POM xml is it's not saved right.
Bach Tran: yeah.
Bach Tran: All right now go and generate your web xml so right click and.
Nicholas Hailey: don't forget to update first got an.
Bach Tran: update and make sure to update cute so update maven and right click maven update project.
Nicholas Hailey: Right, I may even.
Though.
Yves Bouele: Okay.
Bach Tran: All right, and then from there, you want to actually generate your web nights and also you need to right click.
Bach Tran: I actually did you do the project facets for so I like it on the project go to properties, make sure that that.

Did.
Bach Tran: to 2.5 yeah change to 4.0.
Bach Tran: and
Then apply.
Bach Tran: yeah plainclothes.
Bach Tran: All right now right click.
Yves Bouele: click what.
Yves Bouele: crowd right way.
Bach Tran: Java EE tools and.
Bach Tran: descriptor.
Bach Tran: You might have press the first one, you need to press the second one.
Yves Bouele: yeah perfect, as a first one.
Yves Bouele: Oh, this one.
yeah yeah.
Yves Bouele: Placing the first one.
Okay.
Yves Bouele: generates the xml.
Yves Bouele: I see.
Bach Tran: yeah and then they use so let's take a look at it and see if it's on the right version.
Bach Tran: yeah yeah that looks fine so you'll see an x here, but if you like, just change like put in like an extra line or something and save that it's gonna disappear.
Bach Tran: And yeah.
Yves Bouele: to copy what you had on the day on the set.
Bach Tran: yeah.
Yves Bouele: You should do it too okay thanks.
Nicholas Hailey: I use snippet a lot in case I get behind so those little snippet to in a graduate things move so much faster sometimes to as well.
Bach Tran: All right, alright so let's see.
Bach Tran: um.
Bach Tran: Let me think here yeah i'm going to i'm going to try this i'm going to do a bit of different configuration see if it works if it doesn't know.
Bach Tran: will change it back to my original way, but I think this is easier to understand, so what we need is an Internet parameter tagged and then for the parameter name, we need to configure the.
Bach Tran: context configuration right, so what is the location of this.
Bach Tran: Application context file, essentially, and then we need to say all right, it is inside of the Web INF folder.
Bach Tran: Application context dot xml.
Bach Tran: So what that will do is whenever this circle it gets started up right gets instantiated.
Bach Tran: This property here get set to slash web dash slash application context xml.
Bach Tran: And this is familiar territory now right so like this is where we could define our beans do configuration and just load up all that's necessary in our application inside of this application context about xml.
Bach Tran: We also have this.
Bach Tran: mapping tag that we need to.
Bach Tran: also fill out here.
Bach Tran: So certainly name, we basically are mapping a URL.
Bach Tran: Like all requests that go to a particular URL.
Bach Tran: will be sent to a particular server so here our service, it is the dispatcher serve.
Bach Tran: So we need to provide the name or dispatcher sort of live.
Bach Tran: And we want all requests to be sent over to the dispatcher, certainly in the way that we do that is by putting a slash.
Bach Tran: All requests all requests go to this dispatcher service now dispatchers circle it then sends it over to the appropriate control and.
Bach Tran: Though this is what's known as.
Bach Tran: The front controller.
Bach Tran: Design pattern essentially.
Bach Tran: So the way you can think of it is all requests go through like this front Controller and then dispatches those requests, hence the name dispatcher serve it to the actual controllers with our various URL mappings.
Bach Tran: So that's what is being done right here.
Bach Tran: So that means that we do need an application context dot xml.
Bach Tran: So.
Nicholas Hailey: Bob, can I ask a question, where is your serve lit closing tag located.
Nicholas Hailey: So so so that it goes inside of that.
Nicholas Hailey: Okay, well, I couldn't I just.
Nicholas Hailey: Thank you.
Bach Tran: let's.
See.
Bach Tran: All right, so yeah let's create that application context xml oh actually it's time for lunch so we'll come back we'll finish up with this example and then.
Bach Tran: From there that's when i'll move into the one on ones, other than that will basically put you into the various project to breakout rooms, where you can be working on the project.

Bach Tran: Alright, so this is where we left off right, so we have the Web dot xml.
Bach Tran: And, basically, the purpose of this file is to configure and let tomcat the server that we're going to be using know what to actually start up.
Bach Tran: Whenever the application loads right and so it's going to look through this file, this is what's known as a deployment descriptor right so whenever we.
Bach Tran: generated the deployment descriptor, this is what it actually is and it's going to look through here it's going to find the service that tag it's going to look for this dispatcher service class.
Bach Tran: And then it's going to instantiate this service and basically manage it, and when this dispatcher service, it is initialized it's going to have this configuration right here that looks for an application context dot xml file.
Bach Tran: And so, this xml file is what we actually use to configure all of the different beings, we could enable.
Bach Tran: annotation scanning right components scanning inside of this file because defined as being and so we're going to do that right inside of this web INF folder so we just create a new file in here we'll just call it application context taught xml.
Bach Tran: And let me go ahead and just copy and paste into this file.
Bach Tran: The beans tag with the definition.
Bach Tran: And so, let me go ahead and give this to you all, as well, so.
Bach Tran: I just put it in the chat you can copy and paste it fix up the formatting with control shift right.
Bach Tran: And then from there you're going to basically be on the same page.
Bach Tran: So inside of this file when I want to do is enable components scanning, so I will do contexts components scan.
Bach Tran: base package equals combat ready.
Bach Tran: All right, and if you remember that was all that was needed to actually enable the annotation configuration.
Bach Tran: For spring.
Bach Tran: And so that's all we need to put anything that is a inside of the Cabinet revenue package or a sub package will those annotations will be picked up.
Bach Tran: let's see.
Bach Tran: All right, let's put another one in NBC default survival it learner let me see we actually need this one.
Bach Tran: Fears a handler for serving static resources by forwarding to the server containers defaults are use of this handler allows a mapping the dispatch service, while still allowing sort of container service static reason well.
Bach Tran: No, we don't really need this one right now, because we're not doing static resources so.
Bach Tran: yeah that's not required actually one other one I will put, though, is mvc.
Bach Tran: annotation driven.
Bach Tran: So.
Bach Tran: This one configured the annotation driven spring mvc controller programming model.
Bach Tran: And yeah this This basically enables the APP controller annotation so that, when we put that on any of our to controller classes those basically become our control.
Bach Tran: that's all that's me so we just need these two lines in here for our configuration.
Bach Tran: Alright, so yeah let's go over to source main Java, we will create the package i'm not Robert Turner dot controller.
Nicholas Hailey: hold on one second i'm gonna get that snippet of this way don't slow everybody down.
Bach Tran: yeah let me paste this as well.
Nicholas Hailey: Or, I could just do that that's just fine you know.
Bach Tran: Alright, so we're going to right click COM dot Controller and then create a new class.
Bach Tran: let's call this one test controller.
Bach Tran: Let me check a message real quick.
Bach Tran: All right, yeah so test controller just a symbol class want to click finish and at the top here we're going to put typically right, we would use.
Bach Tran: Or at service or one of those annotations here we're going to use controller, which is a special annotation in the for indicating that we want for this to be a controller one day i'd forgotten right we can't actually import anything because we don't have the dependency so.
Bach Tran: Inside of our palm not xml we do need to go ahead and list out the various dependencies so.
Bach Tran: let's go ahead and create the dependencies tag.
Bach Tran: and go over to the end repository.
Bach Tran: And let's search for service that.
Bach Tran: We need the server API itself so we'll go.
Bach Tran: Over and grab that 4.0 point one.
Bach Tran: Let me put this in the chat.
Bach Tran: And all right we're gonna paste.
Bach Tran: snow what I wanted see.
Bach Tran: Alright, and we are.
Bach Tran: And we also need spring weather in DC so.
Bach Tran: We will search for spring Web.
Bach Tran: So it's just going to be this spring dash web, which is the spring web module.
Bach Tran: I point 3.9.
Bach Tran: So you get that leap there as well.
Bach Tran: Right.
Bach Tran: And spring web, so this module it depends on spring beans in spring core.
Bach Tran: Right so.
Bach Tran: spring beans dependency injection and spring core for the core utilities right, I actually even tells you right there just pretty cool.
Bach Tran: um what else we need we need Jackson data by right which will convert our objects into json and from json into the object.

Bach Tran: So same dependency as required for Java right.
Bach Tran: So we'll just grab Jackson data by.
Bach Tran: And place on on it.
Bach Tran: Alright, so right there.
Bach Tran: So yeah just search for Jackson the divine will put that in the chat as well.
Bach Tran: yeah that should be the three dependencies that are required.
Bach Tran: All right, so yeah now we can actually import the.
Bach Tran: They get rid of the controller annotation interesting wait, let me see.
Bach Tran: Oh, I did get rid of the controller manage hmm.
Bach Tran: Well, we can always use rest controller.
Bach Tran: that's interesting, let me actually.
Bach Tran: see what annotations there are.
Bach Tran: So spring spring Web.
Bach Tran: right here.
Bach Tran: spring framework dot Web.
Bach Tran: Top bind dot annotation.
Bach Tran: yeah let me get to my old Code as well, like today, actually get rid of that.
Bach Tran: stereotype talk controller.
Bach Tran: or dot spring framework dot.
Bach Tran: stereotype trawler okay let's see so.
Bach Tran: I should be.
Bach Tran: yeah they did.
Bach Tran: They did update recently and they did make a lot of changes I did hear about that.
Bach Tran: Let me see, let me see.
Bach Tran: spring web see.
Bach Tran: yeah i'm just curious i'm gonna go back to a previous version 5.3 Point six and just see if there's a different.
Bach Tran: let's see is there the controller annotation in the older version.
Bach Tran: yeah there was there definitely was the controller annotation.
Bach Tran: So that's pretty strange that it's not available anymore.
Bach Tran: let's see spring core.
Bach Tran: um.
Bach Tran: Let me just try doing the import directly here.
Bach Tran: Really.
Bach Tran: Let me see.
Bach Tran: Could that be a problem if we are trying to do at a component.
Bach Tran: Now, apparently, the maven dependency structure need had a bit of a change so.
Bach Tran: We are.
Bach Tran: going to include some other dependencies as well.
Bach Tran: Let me actually see how we might do that.
Bach Tran: One second.
Bach Tran: it's a bit curious how it doesn't have these.
Bach Tran: annotations like we do.
Bach Tran: over here.
Bach Tran: See spring context.
Bach Tran: Up beans core and expression for.
Bach Tran: NBC.
Bach Tran: We have.
Bach Tran: beans and core.
Bach Tran: Okay, so let's try putting the context dependency in there as well, so.
Bach Tran: going to go to the Paul not xml and we will.
Bach Tran: Put inside of here the spring contacts dependency as well.
Bach Tran: And let's see okay plays together nicely or not.
Bach Tran: So.
Bach Tran: springtime context plug it straight in there.
Bach Tran: let's all just have these four dependencies.
Bach Tran: A.
Bach Tran: Put a couple of lines here there we are that's the four.
Bach Tran: And yeah let's see if we have the component annotation yeah cool so there it is, we can also put in the controller annotation so let's.
Bach Tran: go ahead and do that.
Bach Tran: And for our very first controller here, we will.
Bach Tran: make a really simple and point so let's let's do public string.
Bach Tran: it's called it's Hello and return Hello.
Bach Tran: world.
Bach Tran: Right something like that.
Bach Tran: Alright, so one thing we also need to do is.
Bach Tran: At response body, so that this string goes into the response body itself because normally if we don't put this here it's going to go

into.
Bach Tran: it's going to try to contact of view resolved and sort of a static file and that's not really something that we're dealing with we're just.
Bach Tran: Using spring web mvc to actually create a back end up teaching right with that just sends and receives http requests not static fine, so we need this response body here to.
Bach Tran: Basically.
Bach Tran: turn this string and included inside of the body of the responses, so now up here we can specify a get mapping.
Bach Tran: there's different parameters that you can put in so like path right, so we could do path equal Hello.
Bach Tran: And there's also some other options as well, so, for example, produces.
Bach Tran: let's see what would this.
Bach Tran: will be here doesn't give me the Intel a sense.
Bach Tran: Media type, so this I want this to be json, for example, I would indicate that it is um.
Bach Tran: let's see a search for a json type.
Bach Tran: That it is application json, for example.
Bach Tran: So right.
Bach Tran: Oh yeah it needs to be inside of.
Bach Tran: curly braces that.
Bach Tran: The value for attribute mapping produces must be a constant satin on a constant expression.
Bach Tran: let's see is this going to work.
Bach Tran: cannot convert from media type to.
Bach Tran: strain on.
Bach Tran: I think it might just be application.
Bach Tran: json.
Bach Tran: yeah so that's what you do right there right, so this is going to be a json response, and we want to put inside of the header of the response that this is some json that we are sending that if it were like an image, then you would do.
Bach Tran: Something like image slash.
Bach Tran: jpg something like that, but, in our case it's going to be some json so application flushes.
Bach Tran: All right, so how do we actually get this to run, there is no main method so it's not quite straightforward yet, but.