

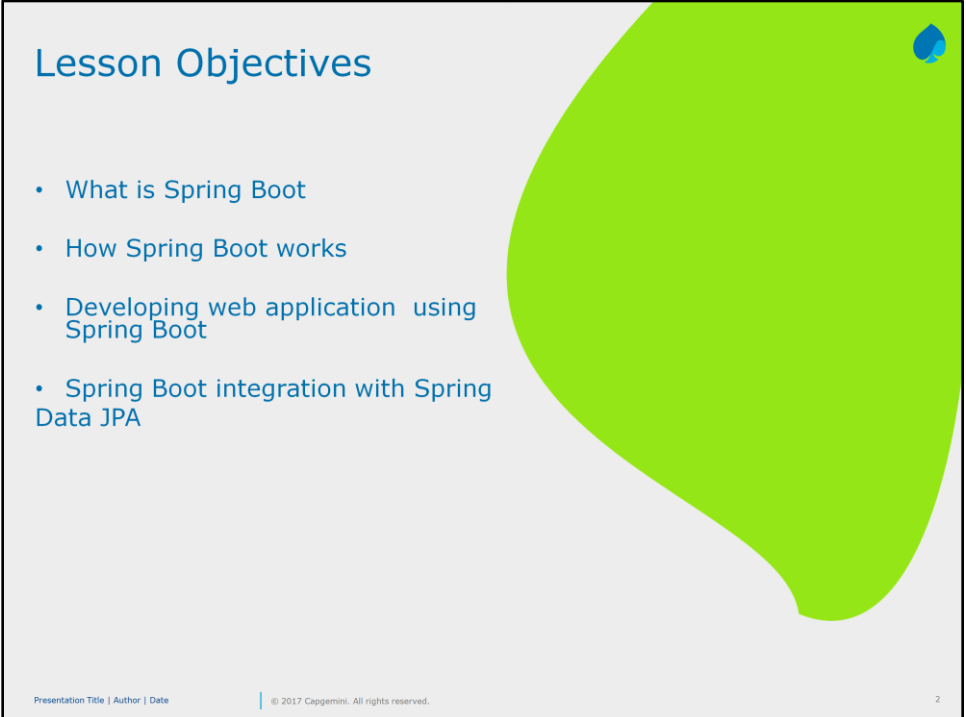
Instructor Notes:

Add instructor notes
here.



Instructor Notes:

Add instructor notes here.



Lesson Objectives

- What is Spring Boot
- How Spring Boot works
- Developing web application using Spring Boot
- Spring Boot integration with Spring Data JPA


Presentation Title | Author | Date | © 2017 Capgemini. All rights reserved. 2

Following contents would be covered:

- What is Spring Boot
- How Spring Boot works
- Developing web application using Spring Boot
- Spring Boot integration with Spring Data JPA

Instructor Notes:

Spring Boot



Prerequisites to start working with Spring Boot

- Knowledge of basic spring concepts
- jdk 1.8 or higher
- IDE i.e Spring STS (has maven built into it)

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

3

Spring STS can be downloaded from spring.io/tools

Instructor Notes:

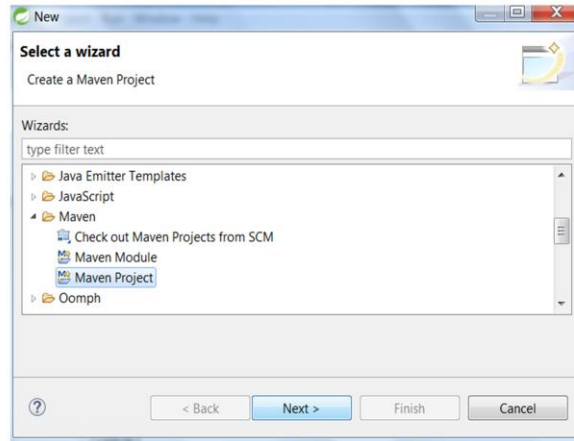
Ways to create Spring Boot project

1. Using the Spring Tool Suite IDE (STS)
2. Spring Initializer
3. Spring command line interface

Instructor Notes:

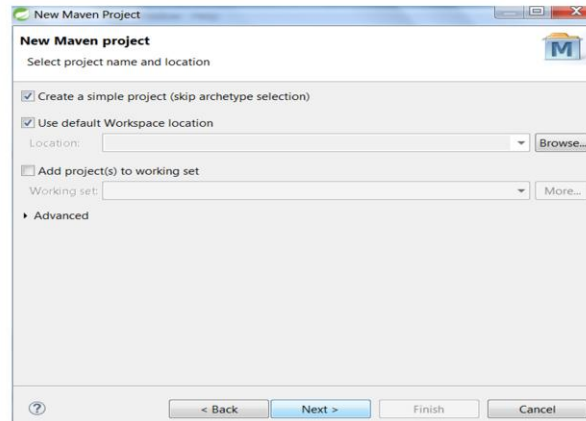
Creating a spring boot application using STS IDE

Click on menu , File →New –Other - Maven -Maven Project- Click on Next



Instructor Notes:

Creating a spring boot application using STS IDE
Select the checkbox, "Create a simple project" and Click On Next-



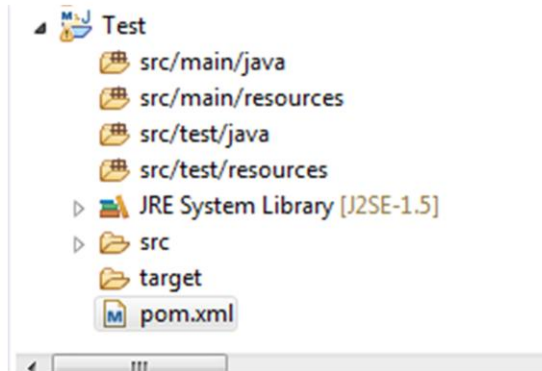
Instructor Notes:

Specify the group id, artifact id, name and description
Click On Finish. Observe the folder structure of the newly created project



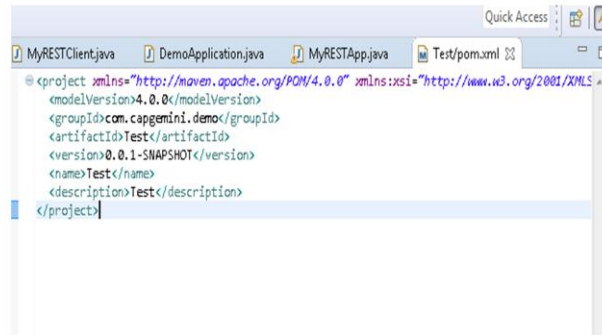
Instructor Notes:

Double click on the generated pom.xml file



Instructor Notes:

The default pom.xml is shown below

A screenshot of an IDE window showing a file named 'Test/pom.xml'. The file contains the following XML code:

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.cappgemini.demo</groupId>
  <artifactId>Test</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>Test</name>
  <description>Test</description>
</project>
```

Instructor Notes:

Add the following code in the pom.xml under the <description> tag 

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.0.1.RELEASE</version>
    <relativePath /> <!-- lookup parent from repository -->
</parent>
<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
</dependencies>
```

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

10

```
<parent>
```

```
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>2.0.1.RELEASE</version>
    <relativePath /> <!-- lookup parent from repository -->
</parent>
```

Above entry will bring in all the dependency management features of Spring boot .

There is no need to declare all the dependencies one by one in pom.xml

```
<dependencies>
```

```
    <dependency>
```

```
        <groupId>org.springframework.boot</groupId>
```

```
        <artifactId>spring-boot-starter-web</artifactId>
```

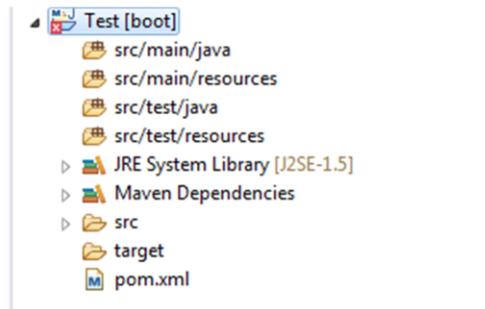
```
    </dependency>
```

```
</dependencies>
```

Above will integrate Spring MVC and autoconfigure the project for us. When u add the Spring boot starter web dependency in pom.xml, this brings in the Spring MVC sub framework dependency into the application.

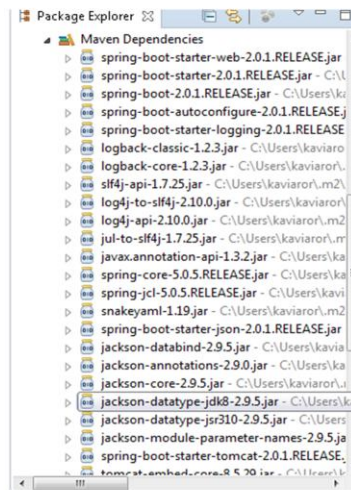
Instructor Notes:

Observe that "Maven Dependencies" has been included into the project



Instructor Notes:

Without Spring Boot, these jar files are among those that you would have had to copy physically into the project



Instructor Notes:

Create a new java class having the following code

```
@SpringBootApplication
public class Client {

    public static void main(String[] args) {
        SpringApplication.run(Client.class,args);
    }

}
```

Run the above program as a regular java application
There is no need to deploy this application on any external server

Note: this class must be kept in the topmost package.

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

13

SpringApplication.run(): Starts Spring, creates Spring context , applies annotations and sets up embedded container

Instructor Notes:



Run the application as a java application and observe the console as shown below

```
.Client      : Starting Client on LINUX267 with PID 11808 (C:\spring_boot\test\target\classes started by kaviaror in C:\spr
.Client      : No active profile set, falling back to default profiles: default
~verApplicationContext : Refreshing org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationConte
omcat.TomcatWebServer : Tomcat initialized with port(s): 8081 (http)
core.StandardService : Starting service [tomcat]
s.core.StandardEngine : Starting Servlet Engine: Apache Tomcat/8.5.29
4prLifecycleListener : The APR based Apache Tomcat Native library which allows optimal performance in production environments was n
.[localhost].[/]      : Initializing Spring embedded WebApplicationContext
~textLoader         : Root WebApplicationContext: initialization completed in 2480 ms
~vletRegistrationBean : Servlet dispatcherServlet mapped to [/]
lterRegistrationBean  : Mapping filter: 'characterEncodingFilter' to: [/]
lterRegistrationBean  : Mapping filter: 'hiddenHttpMethodFilter' to: [/]
lterRegistrationBean  : Mapping filter: 'httpPutFormContentFilter' to: [/]
lterRegistrationBean  : Mapping filter: 'requestContextFilter' to: [/]
mpleUrlHandlerMapping : Mapped URL path [/**/favicon.ico] onto handler of type [class org.springframework.web.servlet.resource.Resou
thappingHandlerAdapter : Looking for @ControllerAdvice: org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServer
thappingHandlerMapping : Mapped "[(/error)]" onto public org.springframework.http.ResponseEntity<java.util.Map<java.lang.String, java
thappingHandlerMapping : Mapped "[(/error),produces=[text/html]]" onto public org.springframework.web.servlet.ModelAndView org.spring
mpleUrlHandlerMapping : Mapped URL path [/webjars/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceCe
mpleUrlHandlerMapping : Mapped URL path [/**] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpReque
omBeanExporter        : Registering beans for JMX exposure on startup
omcat.TomcatWebServer : Tomcat started on port(s): 8081 (http) with context path ''
.Client              : Started Client in 4.632 seconds (JVM running for 5.691)
```

Instructor Notes:**Create a class which acts as a controller**

```
@RestController
public class HelloController {
    @RequestMapping("/hello")
    public String sayHi() {
        return "Hi";
    }
}
```

As we have not mapped any URLs to methods in the controller class, this step becomes necessary

Instructor Notes:

Creating a spring boot application using STS IDE

After adding the controller class, navigate to browser and type <http://localhost:8081/hello>

And observe the "Hi" message displayed on the browser page

We have a fully running Java spring web application developed using Spring boot

Rapid application development is what Spring boot is about.

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

16

Use the following to change the default port 8080 on which Tomcat listens

application.properties file

src ->main ->resources .. keep file here

server.port=8081

Instructor Notes:

How Spring Boot works



1. The application is started from the Java main class
2. Spring boot initialises Spring context that comprises the Spring app and honours autoconfig initialisers, configuration and annotations which direct how to initialise and startup the spring context
3. Embedded server container is started and autoconfigured

This removes the need for web.xml

Spring has chosen "Tomcat" as the default container

Instructor Notes:

How Spring Boot works



@SpringBootApplication

A convenience annotation that wraps commonly used annotations.

Used in place of the following 3 different annotations

1. @configuration : Instructs that a Spring configuration class is being used instead of XML to define the components

2. @EnableAutoconfiguration : is a Spring boot specific annotation

Instructs that the application should auto configure the other frameworks included as dependency with Spring.

3. @ComponentScan : Scans project for Spring components annotated with @Service, @Repository, @Component

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

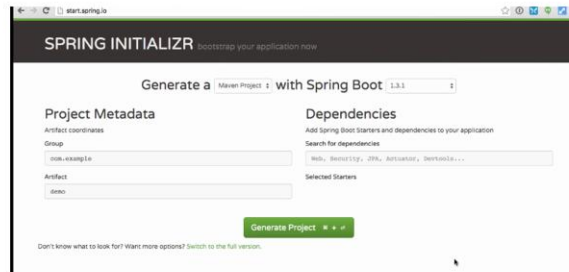
18

@EnableAutoconfiguration : This annotation told Spring boot to automatically set up so that we can use Spring controllers without doing any other integration work with MVC framework

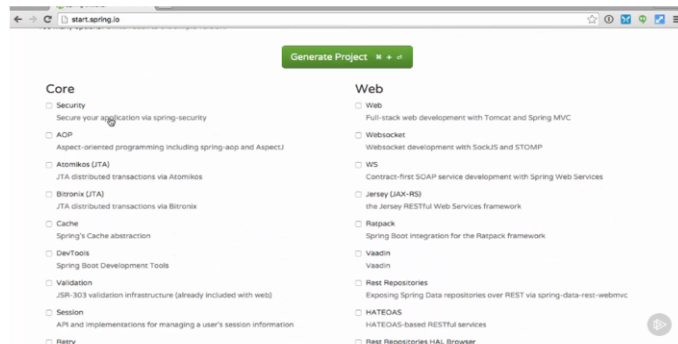
Instructor Notes:

Spring Initializr

Navigate to the following URL
start.spring.io

The screenshot shows the Spring Initializr web application. At the top, it says "SPRING INITIALIZR bootstrap your application now". Below this, there's a header "Generate a Maven Project with Spring Boot 1.3.1". The main content is divided into two sections: "Project Metadata" and "Dependencies". Under "Project Metadata", there are input fields for "Group" (containing "io.spring"), "Artifact" (containing "demo"), and "Version" (containing "1.0.0"). Under "Dependencies", there's a search bar and a list of "Selected Starters" which includes "web", "spring-web", "spring-mvc", "spring-data-jpa", "spring-security", and "spring-thymeleaf". A green "Generate Project" button is at the bottom right. A small link at the bottom left says "Don't know what to look for? Want more options? Switch to the full version."

Click on "switch to full version" link

Instructor Notes:

Select appropriate checkboxes which represent the different dependencies you want to include in the project and then click on "Generate Project"

Observe the zip file created for you.
This contains the folder structure of the project

Instructor Notes:

Spring boot command line interface

The Spring Boot CLI is a command line tool.

You don't necessarily need to use the CLI to work with Spring Boot

You can download the Spring CLI distribution from the Spring software repository
spring-boot-cli-xxx.BUILD-SNAPSHOT-bin.zip

Once downloaded, follow the instructions written in install.txt

Instructor Notes:

Thoughts to ponder



Why move to containerless deployment

Why run the application as a plain Java program

Instructor Notes:

Container deployments



Make a jar file of the application and deploy on the container

- Pre setup and configuration required

- Need to use files like web.xml to tell the container how to work

- Environmental configuration may be required. eg JNDI

Instructor Notes:

Application deployments

When container is bundled inside the application, it is a better choice as

- The applications runs anywhere that Java is setup

- No need to find hosting environment

- Container is embedded inside the application which tells the container how to set up the app so that it can be access via HTTP

- Environmental configuration is internal to the application

Instructor Notes:

Demo

1. Simple Java application using Spring Boot
2. Restful web application using Spring Boot
3. Spring boot application which integrates with Spring Data JPA

Instructor Notes:

Trainer can summarize
the points

Summary



What we have seen so far:

- What is Spring Boot and how it works
- Create a Java application up and running using Spring Boot
- Create a Restful web application using Spring Boot
- Create a Spring boot application which integrates with Spring Data JPA

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

26

Add the notes here

Instructor Notes:

Question 1: Option 2

Question 2: True

Question 3: SOAP
messages

Review Question

Question 1:

Question 2:

Question 3:

Presentation Title | Author | Date

© 2017 Capgemini. All rights reserved.

27

Add the notes here.