**Problems associated with old process**.(before build tools)

• Downloadingdependencies. (maven local/remote/centralrepo accumulates all dependencies and can be used later in any new projectimplementation)

• Compiling source code into binary code. (no need to compile manually : mvn compile will do the job)

• Packaging that binary code. (no need to packagemanually : mvn package will do the job)

• Running tests. (mvn test : automatically compiles and runs unit test cases on application code)

• Deploymentto production systems. (Jenkins CI/CD process uses maven to build the app)

• Reusing the user created project binary code in another project. (mvn install and mvn deploy : places packaged product at maven repo.)

• Flexibility of different actionsin different environments. (Plugins & Profiles)

**Build Tools**

Build tools are programs that automate the creation of executable applications from source code. Building incorporates compiling, linking and packaging the code into a usable or executable form.

**Two important files of Maven**

• pom.xml – located in the project root folder

• Project Object Model is the fundamental unit of work in Maven. It is an XML file that contains information about the project and configuration details used by Maven to build the project.

• settings.xml – located in maven installation directory $M2\_HOME/conf/settings.xml [globalsettings] User’s home directory ${user.home}/.m2/settings.xml [user settings] Both files are optional. If both files are present, the valuesin the user home settingsfile overrides the valuesfrom global settingsfile.

• Contains configuration that are not specific to project but are global in nature.

The Super POM is Maven's default POM. All POMs extend the Super POM unless explicitly set, meaning the configuration specified in the Super POM is inherited by the POMs you created for your projects.

Annotation of Help

• @Controller • @Service • @Repository • @RequestMapping • @Autowired

**PRACTICAL**

Project creation

File 🡪 other 🡪 maven 🡪 web-app (becoz creating a web app)

Then you need to check the whether src/main/java , src/test/java these folders are created or not during the project creation if not create it manually in project directory.

Pom.xml – heart of maven (all dependencies req are placed here)

Web.xml 🡪 dispatcher servlet and its mapping becoz the request comes here first(into web.xml) then dispatcher control the flow.

Previously all the servlets are mentioned here before using the dispatcher servlet but only using dispatcher servlet is enough now.

Name of the servlet in servlet-name and name used in servlet-mapping and the file dispacther.xml the name need to be unique else nothing works.

url-pattern / ()

web.xml

<servlet>

<servlet-name>dispatcher</servlet-name>

<servletclass>org.springframework.web.servlet.DispatcherServlet</servlet-class>

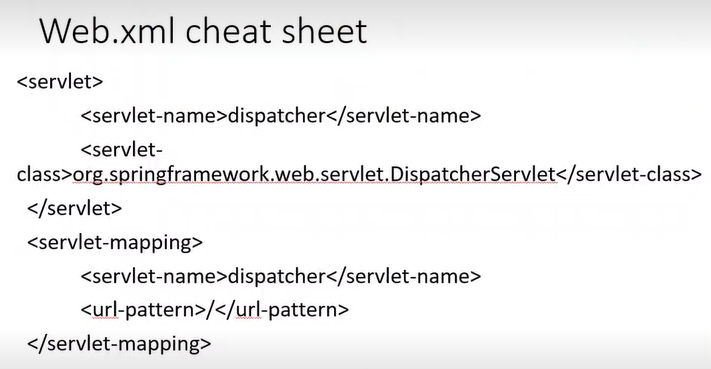
</servlet>

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<url-pattern>/</url-pattern>

</servlet-mapping>



Pom.xml in dependency tag we need to copy our needed dependencies

**pom.xml**

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>4.1.6.RELEASE</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>4.1.6.RELEASE</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-jdbc</artifactId>

<version>4.0.3.RELEASE</version>

</dependency>

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>servlet-api</artifactId>

<version>2.5</version>

</dependency>

--------------------------------

4 dependencies are explained

**Spring-context** if you deeper you will get to know

**Spring-webmvc** (helps in webbased componenct of our appln like controllers)

**Jstl** servlet library (java standard library) (dynamic html page )

**Jdbc** library for database interactions

For http request and response classes javax servlet

<url> </url> used tomention url from where to download dependencies

Dispatcher-servlet.xml(supportive file for dispatcher servlet mentioned in web.xml)(flow definition is written then dispatcher knows how to redirect flow)

**Dispatcher-servlet.xml** In web-inf folder

<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"

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

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

xsi:schemaLocation="

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

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

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

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

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

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

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

http://www.springframework.org/schema/util/spring-util-3.0.xsd

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

http://www.springframework.org/schema/task/spring-task-3.0.xsd

">

<context:annotation-config></context:annotation-config>

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

<bean

class="org.springframework.web.servlet.view.InternalResourceVi

ewResolver">

---------------------------------

Annotation-config we are using this it is mentioned in dispatcher-servlet.xml file

Text

Description automatically generated

When dispatcher gets the control it searches for user-defined controllers to provide locn where to scan we need to give the base-package name com.mohan.mycontroller

The above named package need to be present in src/main/java

In above view resolve is also defined as after dispatcher gets view and model it checks for view resolver whether this type of file is present or not

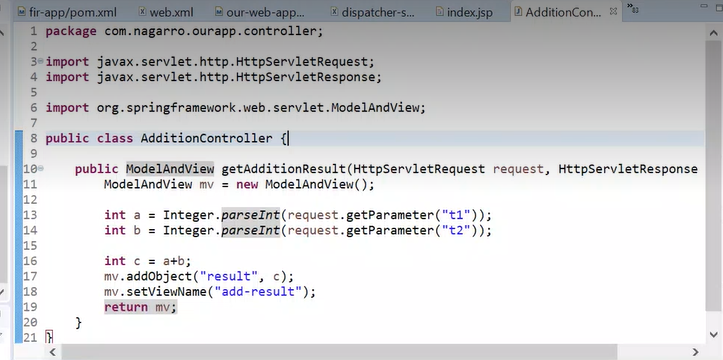
If it gets as mentioned above it further processes

Graphical user interface, text, application, email

Description automatically generated

In the base controller package need to create new NewController.java

Controller should be there after the name of the file.



Paasing the result using addObject method to

We used setViewName which need to be created

Graphical user interface, text

Description automatically generated

Here we gave the key which we used in controller class the give the result.

Making the normal calss as controller class

Graphical user interface, text, application, email

Description automatically generated

After reaching the controller class the request needs ot know where to go

So added the requestmappping anottation

/add which we used in we jsp page

In img libraries used need to be imported.

Do maven build

Here,

And in goal write clean install

And in jre section select jdk

Windows🡪showview🡪 others then write servers

Doubleclick on servername to change the port

Flow

Userrequest 🡪Index.jsp🡪web.xml and lands on dispatch rservlet🡪dispatcher support file🡪scan base package if it finds it thn checks for request mapping and processes it 🡪dsipatcher in web.xml 🡪dispathcer supor tfile and here looks for view resolver and checks for it add-result with suffix mentioned in directory structure🡪ds]ispatcher 🡪 support file 🡪add-result.jsppage and displays result

Dispatcher processes anything coming after rhe root url slash as we mentioned in url tag in pom.xml

Graphical user interface, application

Description automatically generated

Goals in Maven • There are more than 20 goals in maven, each having a specific task to perform.

• Each goal lying before the specified goal to run is also executed in hierarchical order.

1.Validate 2.Compile 3.Test Compile 4.Test 5.Package 6. Install 7.Deploy

Diagram

Description automatically generated

Eg:com.nagarro.ourapp.controller (package naming where business logic is written)

If bydefault jdk is not visible while maven build

Windows🡪preferences🡪standarvm🡪directory and choose the jdk location then apply and close.