**MVC**

Model view Controller

Client => request => Dispatcher Servlet(Finds which Controller to call)

install tomcat

add in eclipse servers

add in projects java bui;dpath add library server runtime

**Using XML**

index page =action=>web.xml It has servlet name, servlet class(Dispatcher Servlet), servlet mapping

web.xml forward request 🡺name-servlet.xml (Dispatcher Servlet)

name-servlet .xml will have the base backage to search the controller with action type

Controller class will have @Controller and @RequestMapping(“/pathname”)

For internal view resolver in xml

<**bean**

class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>

<**property** name=*"prefix"*>

<**value**>/WEB-INF/views/</**value**>

</**property**>

<**property** name=*"suffix"*>

<**value**>.jsp</**value**>

</**property**>

</**bean**>

**Using Java based Approach**

index page =action=>web.xml It has servlet name, servlet class(Dispatcher Servlet), servlet mapping

Design class that will inherit abstract dispatcher servlet with methods

Configure method to search for configclass in package

Config class will have @Configuration

Controller class will have @Controller and @RequestMapping(“/pathname”)

@Controller

Methodname(req,res)|| Methodname(@RequestParam(“t1”) int I) ||reqparam(value=t1) int i

{

I=Integer.parseInt(req.getParam(“inp1”)

j=Integer.parseInt(req.getParam(“inp2”)

k=i+j

modelandview mv= new modelandview

mv.setViewName(“display.jsp”)

mv.addObject(“result”,k)

return mv}

**using values of controller in index**

<%= request.getAttribute(“result”) =%>

**By using expression language**

${result}

Set isELIgnored=false in beans def

**Internal View Resolver**

To return idex.jsp without .jsp

Internal View Resolver v = new()

v.setPrefix(“/web-inf”)

v.setSuffux(“.jsp”)