**Create Spring MVC**

Structure in WEB-INF

Appconfig-mvn.xml

appConfig-root.xml

web.xml

1. **Web.xml**
2. Create a context parameter contextConfigLocation

This parameter will point appConfig-root.xml

1. Create servlet DispatchServlet
2. Create servlet-mapping for our dispatcher that url-pattern to evenry thing



**Init-param** 🡪 it is attribute saved in servlet scope

**Context-param** 🡪 attribute save in application scope

1. Create lister **ContextLoderListener**

To tie a life cycle of ApplicationContext to ServeltContext

Creates WebApplicationContext 🡪 to access ServletContextAware beans getServletContext

1. **ServletConfig**:

A unique object per servlet. We cannot set attributes in ServletConif

1. **ServletContext**

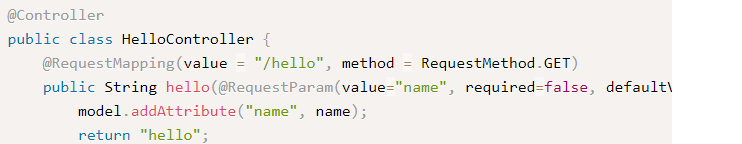
It is an object that contains meta information about your Web application

We can set attribute in ServletContext and will be accessible via appluation

1. appConfig-mvc.xml



**InternalResourceViewResolver**

Map the logical name of the view to physical

Servlets extens HttpServlet {

Void doGet()

doPost();

doDelete();

}

URLS were defined in web.xml

**MVC structure**

1. WebXML
   1. Create dispatch servlet
   2. Create context attribute 🡪 contextConifgLocation point to springConfig.xml
   3. ContextLoaderLister 🡪 tight spring applicationContext to SerletContext
2. springConfig-mvc.xml 🡪 add InternalResourceViewResolver
3. springConig.xml 🡪 add spring-config-mvc in there

**Spring model,ModelMap and ModelAndView**

Model: An interface that represent the object that is going to be rendered  
 has 2 methods addAttribute () and mergeAttributes()

ModelMap: an Interface that represents the MapObject contains objects being displayed  
 ModelAndView: contains all information we need to return to page

**BindingResult**

It is an interface that extends Errors 🡪 used for validatin

Typical Controller in MVC

@Controller  
 public class UserController  
 @RequestMapping(value="createUser", method = RequestMethod.POST)  
 public ModelAndView createUser(@ModelAttribute("user") @Valid User user,BindingResult

result,ModelMap model)