<https://stackjava.com/category/spring-mvc>

DispatcherServlet

 Java configuration that registers and initializes the DispatcherServlet. This class is auto-detected by the Servlet container

**public** **class** **MyWebApplicationInitializer** **implements** WebApplicationInitializer {

@Override

**public** **void** onStartup(ServletContext servletCxt) {

*// Load Spring web application configuration*

AnnotationConfigWebApplicationContext ac = **new** AnnotationConfigWebApplicationContext();

ac.register(AppConfig.class);

ac.refresh();

*// Create and register the DispatcherServlet*

DispatcherServlet servlet = **new** DispatcherServlet(ac);

ServletRegistration.Dynamic registration = servletCxt.addServlet("app", servlet);

registration.setLoadOnStartup(1);

registration.addMapping("/app/\*");

}

}

Hoặc cấu hình trong web.xml configuration to register and initialize the DispatcherServlet:

<web-app>

<listener>

<listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>

</listener>

<context-param>

<param-name>contextConfigLocation</param-name>

<param-value>/WEB-INF/app-context.xml</param-value>

</context-param>

<servlet>

<servlet-name>app</servlet-name>

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

<init-param>

<param-name>contextConfigLocation</param-name>

<param-value></param-value>

</init-param>

<load-on-startup>1</load-on-startup>

</servlet>

<servlet-mapping>

<servlet-name>app</servlet-name>

<url-pattern>/app/\*</url-pattern>

</servlet-mapping>

</web-app>

@EnableWebMvc is equivalent to <mvc:annotation-driven /> in XML. It enables support for @Controller-annotated classes that use @RequestMapping to map incoming requests to a certain method. You can read detailed information about what it configures by default and how to customise the configuration in the [reference documentation](http://docs.spring.io/spring/docs/3.2.x/spring-framework-reference/html/mvc.html#mvc-config).