**通过对SpringMVC启动过程的深入研究，期望掌握Java Web容器启动过程；掌握SpringMVC启动过程；了解SpringMVC的配置文件如何配置，为什么要这样配置；掌握SpringMVC是如何工作的；掌握Spring源码的设计和增强阅读源码的技巧。**

**目录**

**1.Web容器初始化过程**

**2.SpringMVC中web.xml配置**

**3.认识ServletContextListener**

**4.认识ContextLoaderListener**

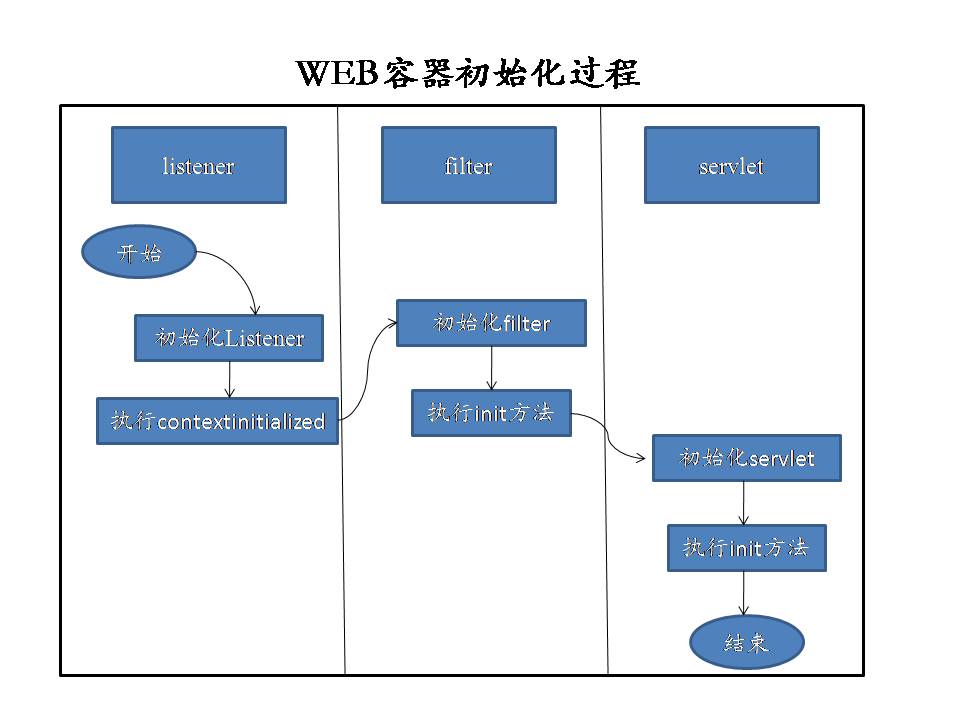
**5.DispatcherServlet初始化（HttpServletBean • FrameworkServlet • DispatcherServlet）**

**6.ContextLoaderListener与DispatcherServlet关系**

**7.DispatcherServlet的设计**

**8.DispatcherServlet工作原理**

**一、Web容器初始化过程**

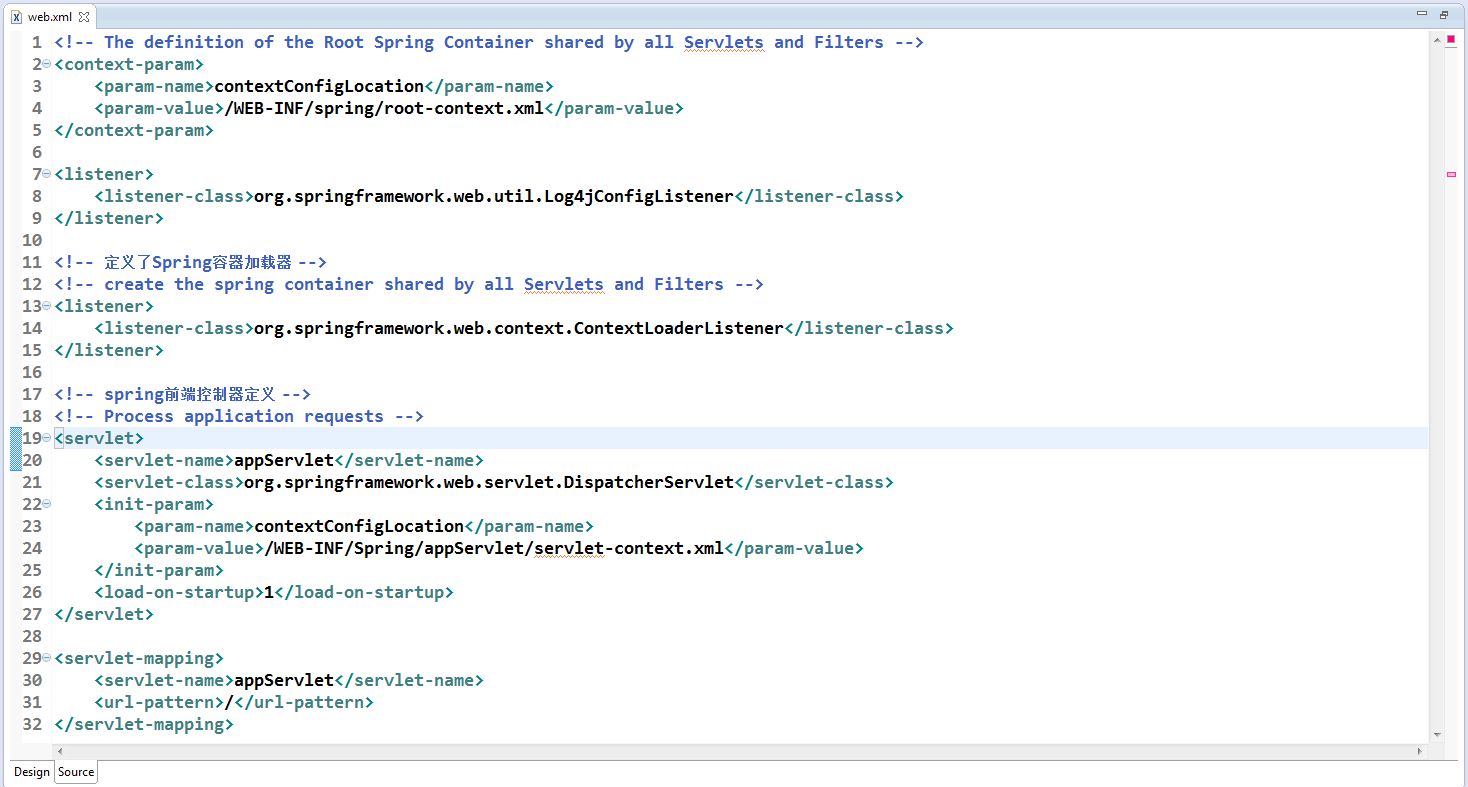
****

**上图展示了web容器初始化的过程，其官方文档给出了这样的描述：**

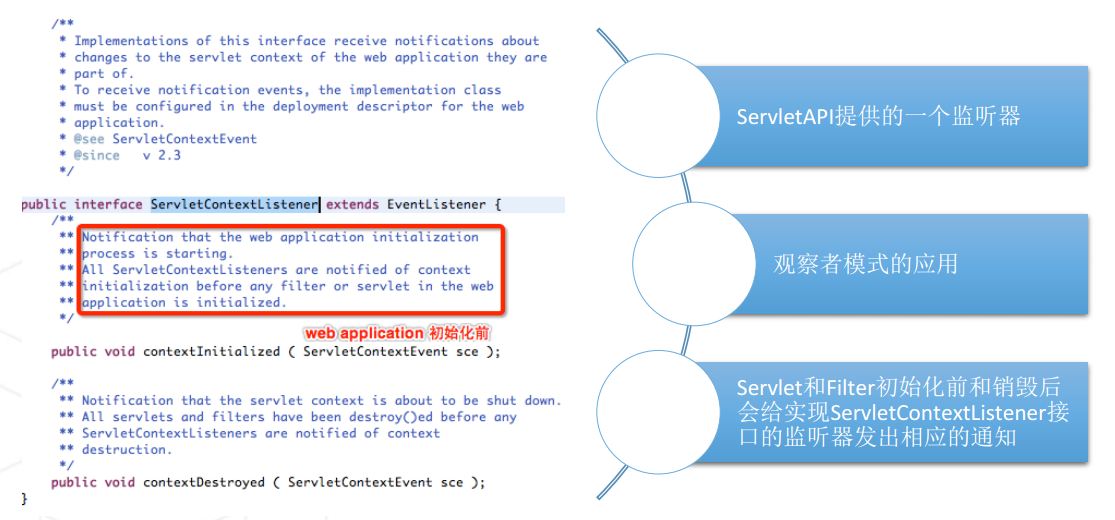
**When a web application is deployed into a container, the following steps must be performed, in this order, before the web application begins processing client requests.**

1. **Instantiate an instance of each event listener identified by a <listener> element in the deployment descriptor.**
2. **For instantiated listener instances that implement ServletContextListener, call the contextInitialized() method.**
3. **Instantiate an instance of each filter identified by a <filter> element in the deployment descriptor and call each filter instance's init() method.**
4. **Instantiate an instance of each servlet identified by a <servlet> element that includes a <load-on-startup> element in the order defined by the load-on-startup element values, and call each servlet instance's init() method.**

**二、SpringMVC中web.xml的配置**

****

**上图是截取的web.xml中的配置，在<listener>标签中定义了spring容器加载器；在<servlet>标签中定义了spring前端控制器。**

****

**上图是源码中接口ServletContextListener的定义，可以看到在其注释中指明：servlet和Filter初始化前和销毁后，都会给实现了servletContextListener接口的监听器发出相应的通知。**

****

**上面是类ContextLoadListener的定义，它实现了上面的servletContextListener。这里用到了代理模式，简单的代理了ContextLoader类。ContextLoadListener类用来创建Spring application context，并且将application context注册到servletContext里面去。**

****

**结合上面的WEB容器启动的过程，以及接口ServletContextListener和类ContextLoadListener。我们知道：**

**在 Servlet API中有一个ServletContextListener接口，它能够监听ServletContext对象的生命周期，实际上就是监听Web应用的生命周期。当Servlet容器启动或终止Web应用时，会触发ServletContextEvent事件，该事件由ServletContextListener来处理。在ServletContextListener接口中定义了处理ServletContextEvent 事件的两个方法contextInitialized()和contextDestroyed()。**

**ContextLoaderListener监听器的作用就是启动Web容器时，自动装配ApplicationContext的配置信息。因为它实现了ServletContextListener这个接口，在web.xml配置了这个监听器，启动容器时，就会默认执行它实现的方法。由于在ContextLoaderListener中关联了ContextLoader这个类，所以整个加载配置过程由ContextLoader来完成。**

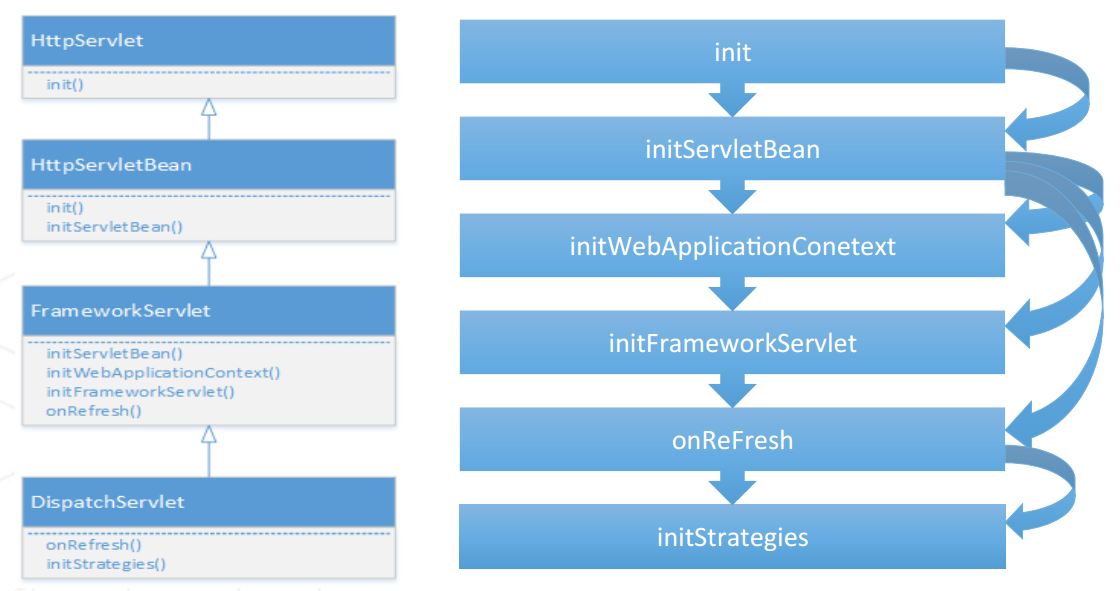
****

**上面是initWebApplicationContext的过程，方法名称即是其含义。方法中首先创建了WebApplicationContext，配置并且刷新实例化整个SpringApplicationContext中的Bean。因此，如果我们的Bean配置出错的话，在容器启动的时候，会抛异常出来的。**

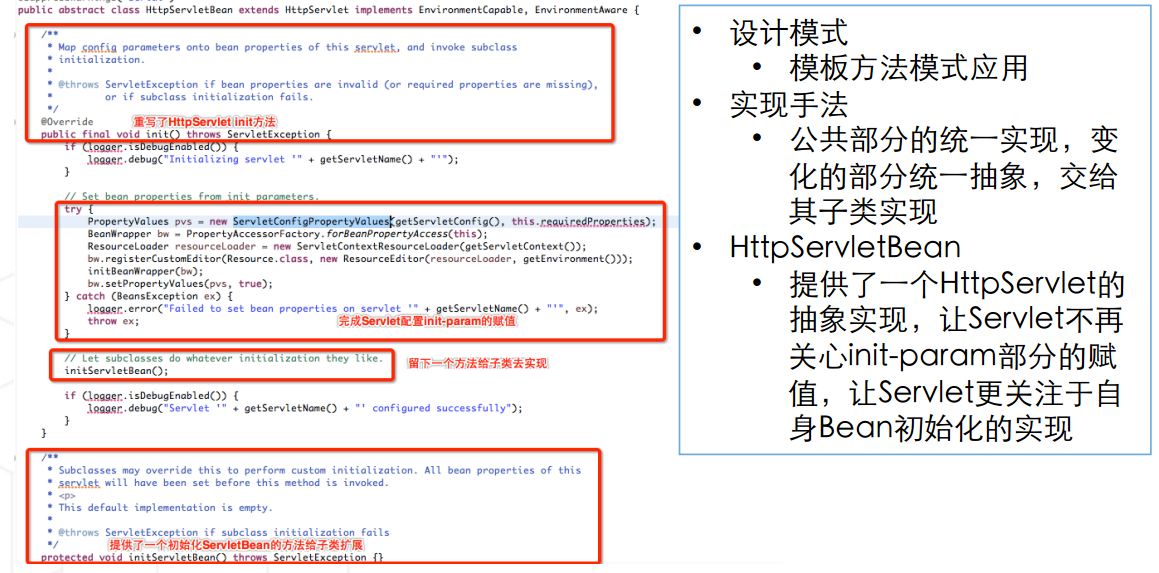
**综上，ContextLoaderListener类起着至关重要的作用。它读取web.xml中配置的context-param中的配置文件，提前在web容器初始化前准备业务对应的Application context;将创建好的Application context放置于ServletContext中，为springMVC部分的初始化做好准备。**

**三、DispatchServlet初始化**

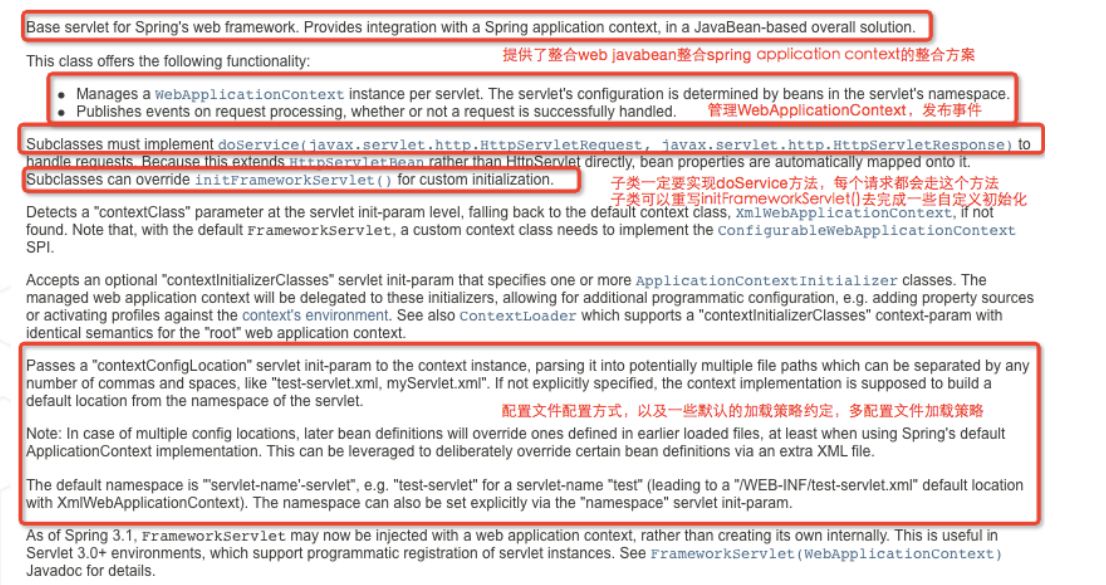
**在SpringMVC架构中，DispatchServlet负责请求分发，起到控制器的作用。下面详细来解释说明：**

****

**DispatchServlet名如其义，它的本质上是一个Servlet。从上面图可以看到，下层的子类不断的对HttpServlet父类进行方法扩展。**

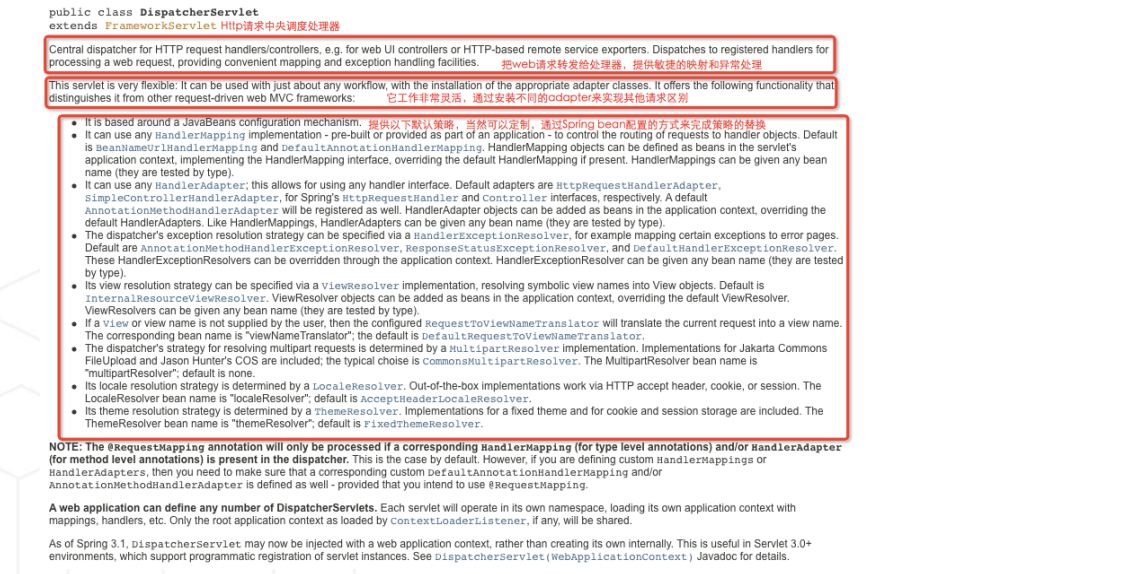
****

**上图是抽象类HttpServletBean的实现，我们知道HttpServlet有两大核心方法：init()和service()方法。HttpServletBean重写了init()方法，在这部分，我们可以看到其实现思路：公共的部分统一来实现，变化的部分统一来抽象，交给其子类来实现，故用了abstract class来修饰类名。此外，HttpServletBean提供了一个HttpServlet的抽象实现，使的Servlet不再关心init-param部分的赋值，让servlet更关注于自身Bean初始化的实现。**

****

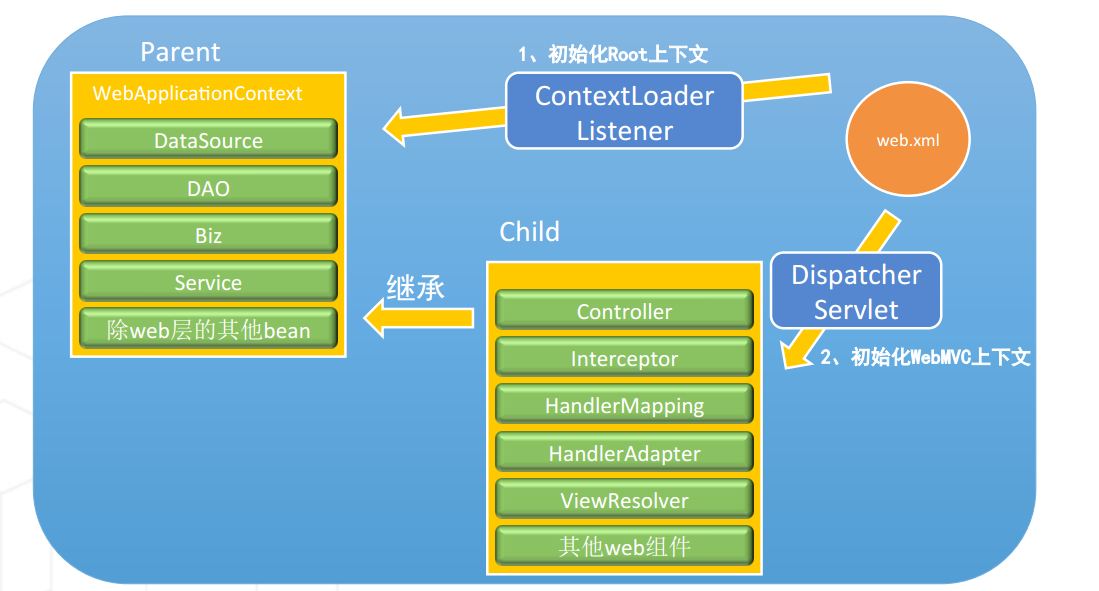
****

**上图是FrameworkServlet的官方定义， 它提供了整合web javabean和spring application context的整合方案。那么它是如何实现的呢？在源码中我们可以看到通过执行initWebApplicationContext()方法和initFrameworkServlet()方法实现。**

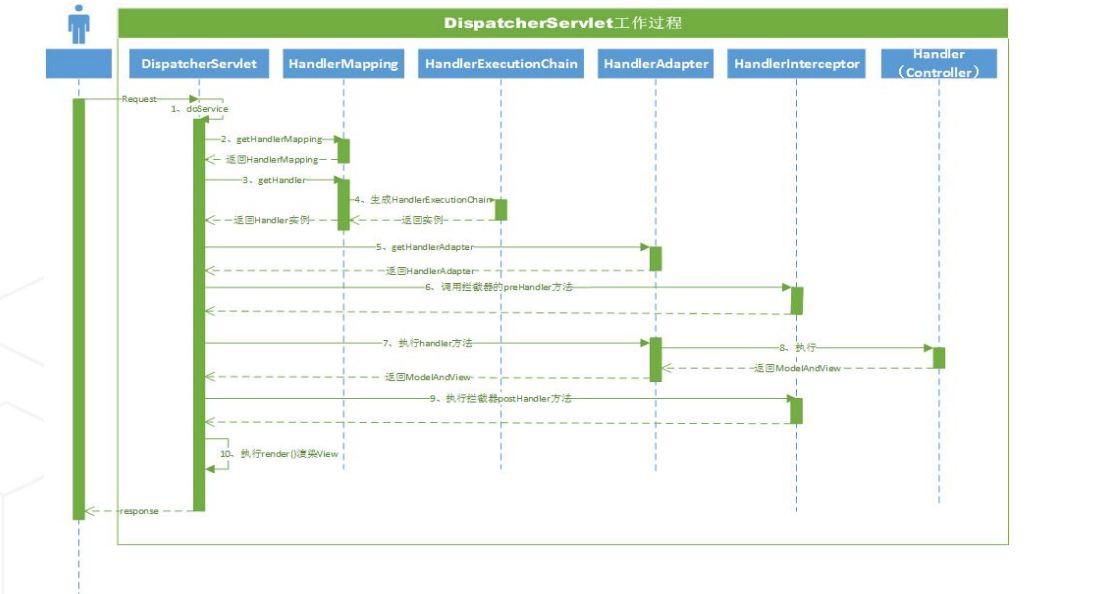
****

****

**DispatchServlet是HTTP请求的中央调度处理器，它将web请求转发给controller层处理，它提供了敏捷的映射和异常处理机制。DispatchServlet转发请求的核心代码在doService()方法中实现，详细代码参照图上。**

****

**上图是DispatchServlet类和ContextLoaderListener类的关系图。首先，用ContextLoaderListener初始化上下文，接着使用DispatchServlet来初始化WebMVC的上下文。**

****

**上图是DispatchServlet的工作流程图，作为HTTP请求的中央控制器，它在SpringMVC中起着分发请求的作用。下面总结了DispatchServlet设计的一些特点总结。**

****

**四、请求流程**

