spring的启动过程：

1. 首先，对于一个web应用，其部署在web容器中，web容器提供其一个全局的上下文环境，这个上下文就是ServletContext，其为后面的spring IoC容器提供宿主环境；
2. 其次，在web.xml中会提供有contextLoaderListener。在web容器启动时，会触发容器初始化事件，此时contextLoaderListener会监听到这个事件，其contextInitialized方法会被调用，在这个方法中，spring会初始化一个启动上下文，这个上下文被称为根上下文，即WebApplicationContext，这是一个接口类，确切的说，其实际的实现类是XmlWebApplicationContext。这个就是spring的IoC容器，其对应的Bean定义的配置由web.xml中的context-param标签指定。在这个IoC容器初始化完毕后，spring以WebApplicationContext.ROOT*WEB*APPLICATION*CONTEXT*ATTRIBUTE为属性Key，将其存储到ServletContext中，便于获取；
3. 再次，contextLoaderListener监听器初始化完毕后，开始初始化web.xml中配置的Servlet，这个servlet可以配置多个，以最常见的DispatcherServlet为例，这个servlet实际上是一个标准的前端控制器，用以转发、匹配、处理每个servlet请求。DispatcherServlet上下文在初始化的时候会建立自己的IoC上下文，用以持有spring mvc相关的bean。在建立DispatcherServlet自己的IoC上下文时，会利用WebApplicationContext.ROOT*WEB*APPLICATION*CONTEXT*ATTRIBUTE先从ServletContext中获取之前的根上下文(即WebApplicationContext)作为自己上下文的parent上下文。有了这个parent上下文之后，再初始化自己持有的上下文。这个DispatcherServlet初始化自己上下文的工作在其initStrategies方法中可以看到，大概的工作就是初始化处理器映射、视图解析等。这个servlet自己持有的上下文默认实现类也是xmlWebApplicationContext。初始化完毕后，spring以与servlet的名字相关(此处不是简单的以servlet名为Key，而是通过一些转换，具体可自行查看源码)的属性为属性Key，也将其存到ServletContext中，以便后续使用。这样每个servlet就持有自己的上下文，即拥有自己独立的bean空间，同时各个servlet共享相同的bean，即根上下文(第2步中初始化的上下文)定义的那些bean。