11 | 总结：从Tomcat和Jetty中提炼组件化设计规范

2019-06-04 李号双

深入拆解Tomcat & Jetty 进入课程 >

讲述：李号双
时长 08:19 大小 7.63M

在当今的互联网时代，我们每个人获取信息的机会基本上都是平等的，但是为什么有些人对信息理解得更深，并且有自己独到的见解呢？我认为是因为他们养成了思考和总结的好习惯。当我们学习一门技术的时候，如果可以勤于思考、善于总结，可以帮助我们看到现象背后更本质的东西，让我们在成长之路上更快“脱颖而出”。

我们经常谈敏捷、快速迭代和重构，这些都是为了应对需求的快速变化，也因此我们在开始设计一个系统时就要考虑可扩展性。那究竟该怎样设计才能适应变化呢？或者要设计成什么样后面才能以最小的成本进行重构呢？今天我来总结一些 Tomcat 和 Jetty 组件化的设计思想，或许从中我们可以得到一些启发。

组件化及可配置

Tomcat 和 Jetty 的整体架构都是基于组件的，你可以通过 XML 文件或者代码的方式来配置这些组件，比如我们可以在 server.xml 配置 Tomcat 的连接器以及容器组件。相应的，你也可以在 Jetty.xml 文件里组装 Jetty 的 Connector 组件，以及各种 Handler 组件。也就是说，**Tomcat 和 Jetty 提供了一堆积木，怎么搭建这些积木由你来决定**，你可以根据自己的需要灵活选择组件来搭建你的 Web 容器，并且也可以自定义组件，这样的设计为 Web 容器提供了深度可定制化。

那 Web 容器如何实现这种组件化设计呢？我认为有两个要点：

第一个是面向接口编程。我们需要对系统的功能按照“高内聚、低耦合”的原则进行拆分，每个组件都有相应的接口，组件之间通过接口通信，这样就可以方便地替换组件了。比如我们可以选择不同连接器类型，只要这些连接器组件实现同一个接口就行。

第二个是 Web 容器提供一个载体把组件组装在一起工作。组件的工作无非就是处理请求，因此容器通过责任链模式把请求依次交给组件去处理。对于用户来说，我只需要告诉 Web 容器由哪些组件来处理请求。把组件组织起来需要一个“管理者”，这就是为什么 Tomcat 和 Jetty 都有一个 Server 的概念，Server 就是组件的载体，Server 里包含了连接器组件和容器组件；容器还需要把请求交给各个子容器组件去处理，Tomcat 和 Jetty 都是责任链模式来实现的。

用户通过配置来组装组件，跟 Spring 中 Bean 的依赖注入相似。Spring 的用户可以通过配置文件或者注解的方式来组装 Bean，Bean 与 Bean 的依赖关系完全由用户自己来定义。这一点与 Web 容器**不同**，Web 容器中组件与组件之间的关系是固定的，比如 Tomcat 中 Engine 组件下有 Host 组件、Host 组件下有 Context 组件等，但你不能在 Host 组件里“注入”一个 Wrapper 组件，这是由于 Web 容器本身的功能来决定的。

组件的创建

由于组件是可以配置的，Web 容器在启动之前并不知道要创建哪些组件，也就是说，不能通过硬编码的方式来实例化这些组件，而是需要通过反射机制来动态地创建。具体来说，Web 容器不是通过 new 方法来实例化组件对象的，而是通过 Class.forName 来创建组件。无论哪种方式，在实例化一个类之前，Web 容器需要把组件类加载到 JVM，这就涉及一个类加载的问题，Web 容器设计了自己类加载器，我会在专栏后面的文章详细介绍 Tomcat 的类加载器。

Spring 也是通过反射机制来动态地实例化 Bean，那么它用到的类加载器是从哪里来的呢？Web 容器给每个 Web 应用创建了一个类加载器，Spring 用到的类加载器是 Web 容器传给它的。

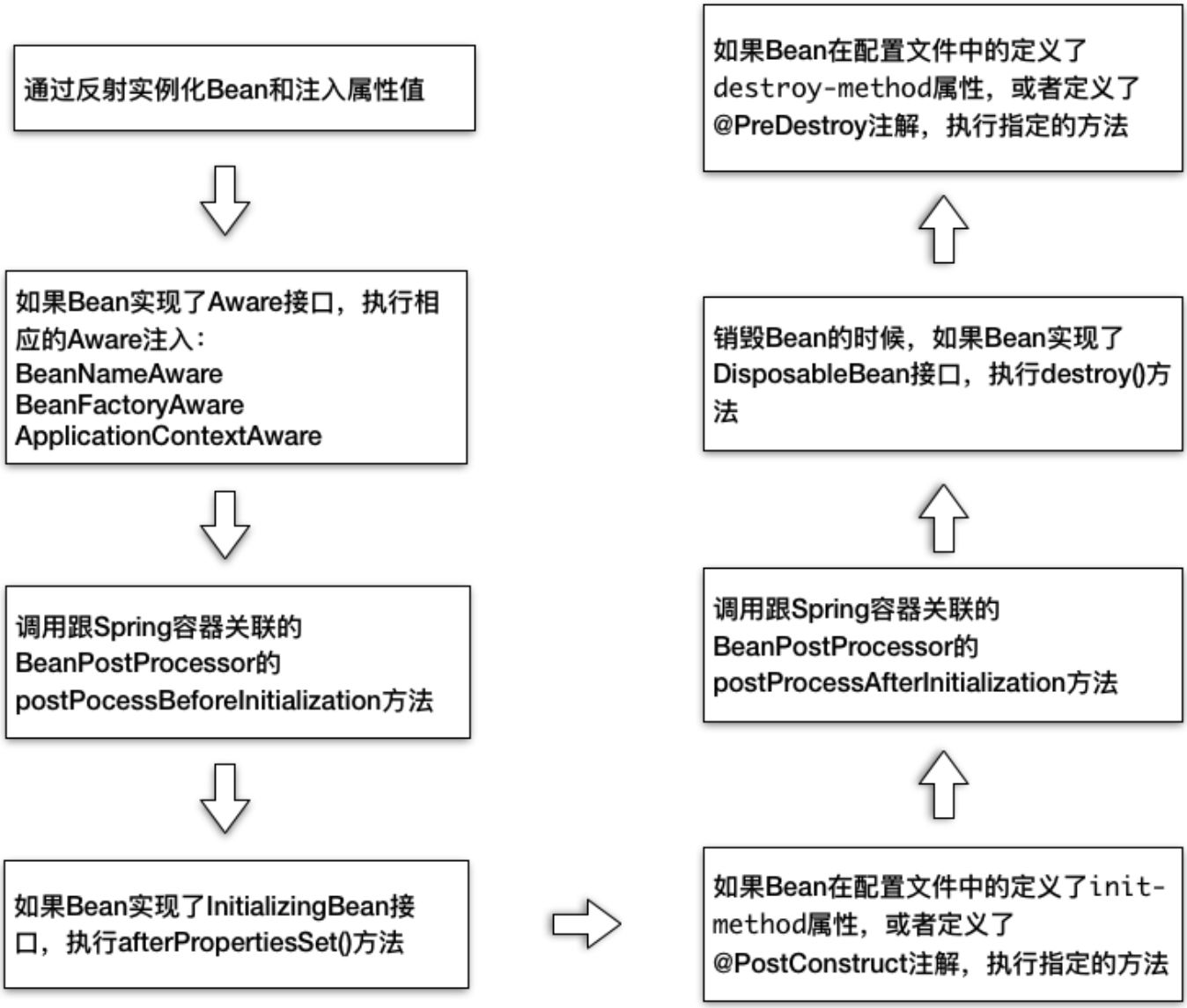
组件的生命周期管理

不同类型的组件具有父子层次关系，父组件处理请求后再把请求传递给某个子组件。你可能会感到疑惑，Jetty 的中 Handler 不是一条链吗，看上去像是平行关系？其实不然，Jetty 中的 Handler 也是分层次的，比如 WebAppContext 中包含 ServletHandler 和 SessionHandler。因此你也可以把 ContextHandler 和它所包含的 Handler 看作是父子关系。

而 Tomcat 通过容器的概念，把小容器放到大容器来实现父子关系，其实它们的本质都是一样的。这其实涉及如何统一管理这些组件，如何做到一键式启停。

Tomcat 和 Jetty 都采用了类似的办法来管理组件的生命周期，主要有两个要点，一是父组件负责子组件的创建、启停和销毁。这样只要启动最上层组件，整个 Web 容器就被启动起来了，也就实现了一键式启停；二是 Tomcat 和 Jetty 都定义了组件的生命周期状态，并且把组件状态的转变定义成一个事件，一个组件的状态变化会触发子组件的变化，比如 Host 容器的启动事件里会触发 Web 应用的扫描和加载，最终会在 Host 容器下创建相应的 Context 容器，而 Context 组件的启动事件又会触发 Servlet 的扫描，进而创建 Wrapper 组件。那么如何实现这种联动呢？答案是观察者模式。具体来说就是创建监听器去监听容器的状态变化，在监听器的方法里去实现相应的动作，这些监听器其实是组件生命周期过程中的“扩展点”。

Spring 也采用了类似的设计，Spring 给 Bean 生命周期状态提供了很多的“扩展点”。这些扩展点被定义成一个个接口，只要你的 Bean 实现了这些接口，Spring 就会负责调用这些接口，这样做的目的就是，当 Bean 的创建、初始化和销毁这些控制权交给 Spring 后，Spring 让你有机会在 Bean 的整个生命周期中执行你的逻辑。下面我通过一张图帮你理解 Spring Bean 的生命周期过程：



组件的骨架抽象类和模板模式

具体到组件的设计的与实现，Tomcat 和 Jetty 都大量采用了骨架抽象类和模板模式。比如说 Tomcat 中 ProtocolHandler 接口，ProtocolHandler 有抽象基类 AbstractProtocol，它实现了协议处理层的骨架和通用逻辑，而具体协议也有抽象基类，比如 HttpProtocol 和 AjpProtocol。对于 Jetty 来说，Handler 接口之下有 AbstractHandler，Connector 接口之下有 AbstractorConnector，这些抽象骨架类实现了一些通用逻辑，并且会定义一些抽象方法，这些抽象方法由子类实现，抽象骨架类调用抽象方法来实现骨架逻辑。

这是一个通用的设计规范，不管是 Web 容器还是 Spring，甚至 JDK 本身都到处使用这种设计，比如 Java 集合中的 AbstractSet、AbstractMap 等。 值得一提的是，从 Java 8 开始允许接口有 default 方法，这样我们可以把抽象骨架类的通用逻辑放到接口中去。

本期精华

今天我总结了 Tomcat 和 Jetty 的组件化设计，我们可以通过搭积木的方式来定制化自己的 Web 容器。Web 容器为了支持这种组件化设计，遵循了一些规范，比如面向接口编程，用“管理者”去组装这些组件，用反射的方式动态的创建组件、统一管理组件的生命周期，并且给组件生命状态的变化提供了扩展点，组件的具体实现一般遵循骨架抽象类和模板模式。

通过今天的学习，你会发现 Tomcat 和 Jetty 有很多共同点，并且 Spring 框架的设计也有不少相似的的地方，这正好说明了 Web 开发中有一些本质的东西是相通的，只要你深入理解了一个技术，也就是在一个点上突破了深度，再扩展广度就不是难事。并且我建议在学习一门技术的时候，可以回想一下之前学过的东西，是不是有相似的地方，有什么不同的地方，通过对比理解它们的本质，这样我们才能真正掌握这些技术背后的精髓。

课后思考

在我们的实际项目中，可能经常遇到改变需求，那如果采用组件化设计，当需求更改时是不是会有一些帮助呢？

不知道今天的内容你消化得如何？如果还有疑问，请大胆的在留言区提问，也欢迎你把你的课后思考和心得记录下来，与我和其他同学一起讨论。如果你觉得今天有所收获，欢迎你把它分享给你的朋友。

上一篇　10 | 比较：Jetty架构特点之Handler组件

精选留言 (7)

写留言

王超
2019-06-04

7

非常赞，这样的总结让人醍醐灌顶，触类旁通！期待后续精彩分享

展开

QQ怪
2019-06-04

1

非常赞，总结的非常到位，并且又对设计模式回温了下，感谢老师的分享！

展开

breezeQia...
2019-06-04

老师好，最近也在看 tomcat 的源码，对 tomcat 的 StandardService 类中的 executors 字段不太理解，这个东西有什么作用吗？

作者回复: executor是线程池，加了s表示一个service有多个线程池，一个Connector一个。

刘章周
2019-06-04

这些扩展点被定义成一个个接口，只要你的 Bean 实现了这些接口，Spring 就会负责调用这些接口，老师，spring是怎么判断这些接口被实现了，就调用接口的方法。
还有就是好多扩展原理都是创建一个类去实现接口，然后会自动调用接口的方法，比如springMVC的lan jie (敏感词)器，这个原理是怎么实现的呢？组件是怎么判断这个接口被实现了，就可以被调用？

展开

作者回复: Java里有个isinstanceof.

Royal
2019-06-04

您好，请问该栏目会涉及JAX-RS的内容么？我想请教下关于这块javax.ws.rs.container.AsyncResponse的内容

作者回复: 有相关的部分，异步Servlet。

往事随风，...
2019-06-04

怎么自己编写一个jetty 服务器，像spring boot 内嵌式的

展开

作者回复: https://github.com/jetty-project/embedded-jetty-jsp 这里有例子

朱晋君
2019-06-04

spring bean生命周期那张图，右下第二应该是postprocessAfterinitialization吧

作者回复: 是的是的，我更正一下，谢谢指出。