四、Tomcat中的类加载器与安全机制

类加载器

Java中的类遵循**按需加载**。

所谓类加载器，就是用于加载 Java 类到 Java 虚拟机中的组件，它负责读取 Java 字节码，并转换成 java.lang.Class 类的一个实例，使字节码.class 文件得以运行。一般类加载器负责根据一个指定的类找到对应的字节码，然后根据这些字节码定义一个 Java 类。另外，它还可以加载资源，包括图像文件和配置文件。

类加载器在实际使用中给我们带来的好处是，它可以使 Java 类动态地加载到 JVM 中并运行，即可在程序运行时再加载类，提供了很灵活的动态加载方式。

- 启动类加载器（Bootstrap ClassLoader）：加载对象是 Java 核心库，把一些核心的 Java 类加载进 JVM 中，这个加载器使用原生代码（C/C++）实现，并不是继承 java.lang.ClassLoader，它是**所有其他类加载器的最终父加载器，负责加载 <JAVA_HOME>/jre/lib 目录下 JVM 指定的类库**。其实它属于 JVM 整体的一部分，JVM 一启动就将这些指定的类加载到内存中，避免以后过多的 I/O 操作，提高系统的运行效率。**启动类加载器无法被 Java 程序直接使用**。
- 扩展类加载器（Extension ClassLoader）：加载的对象为 Java 的扩展库，即**加载 <JAVA_HOME>/jre/lib/ext 目录里面的类**。这个类由启动类加载器加载，但因为启动类加载器并非用 Java 实现，已经脱离了 Java 体系，所以如果尝试调用扩展类加载器的 getParent()方法获取父加载器会得到 null。然而，它的父类加载器是启动类加载器。
- 应用程序类加载器（Application ClassLoader）：亦叫系统类加载器（System ClassLoader），它负责**加载用户类路径（CLASSPATH）指定的类库**，如果程序没有自己定义类加载器，就默认使用应用

程序类加载器。**它也由启动类加载器加载，但它的父加载类被设置成了扩展类加载器**。如果要使用这个加载器，可通过 ClassLoader.getSystemClassLoader()获取。

双亲委派

双亲委派模型会在类加载器加载类时**首先委托给父类加载器加载，除非父类加载器不能加载才自己加载。**

这种模型要求，除了顶层的启动类加载器外，其他的类加载器都要有自己的父类加载器。假如有一个类要加载进来，一个类加载器并不会马上尝试自己将其加载，而是委派给父类加载器，父类加载器收到后又尝试委派给其父类加载器，以此类推，直到委派给启动类加载器，这样一层一层往上委派。只有当父类加载器反馈自己没法完成这个加载时，子加载器才会尝试自己加载。通过这个机制，保证了 Java 应用所使用的都是同一个版本的 Java 核心库的类，同时这个机制也保证了安全性。设想如果应用程序类加载器想要加载一个有破坏性的 java.lang.System 类，双亲委派模型会一层层向上委派，最终委派给启动类加载器，而启动类加载器检查到缓存中已经有了这个类，并不会再加载这个有破坏性的 System 类。

另外，类加载器还拥有全盘负责机制，即当一个类加载器加载一个类时，这个类所依赖的、引用的其他所有类都由这个类加载器加载，除非在程序中显式地指定另外一个类加载器加载。

在 Java 中，我们用完全匹配类名来标识一个类，即用包名和类名。而在 JVM 中，一个类由完全匹配类名和一个类加载器的实例 ID 作为唯一标识。也就是说，同一个虚拟机可以有两个包名、类名都相同的类，只要它们由两个不同的类加载器加载。当我们在 Java 中说两个类是否相等时，必须在针对同一个类加载器加载的前提下才有意义，否则，就算是同样的字节码，由不同的类加载器加载，这两个类也不是相等的。这种特征为我们提供了隔离机制，在 Tomcat 服务器中它是十分有用的。

URLClassLoader

我们在使用自定义类加载去加载类时，我们需要指明该去哪些资源中进行加载，所以JDK提供了URLClassLoader来方便我们使用，我们在创建URLClassLoader时需要传入一些URLs，然后在使用这个URLClassLoader加载类时就会从这些资源中去加载。

Tomcat中自定义的类加载器

Tomcat 拥有不同的自定义类加载器，以实现对各种资源库的控制。一般来说，Tomcat 主要用类加载器解决以下 4 个问题。

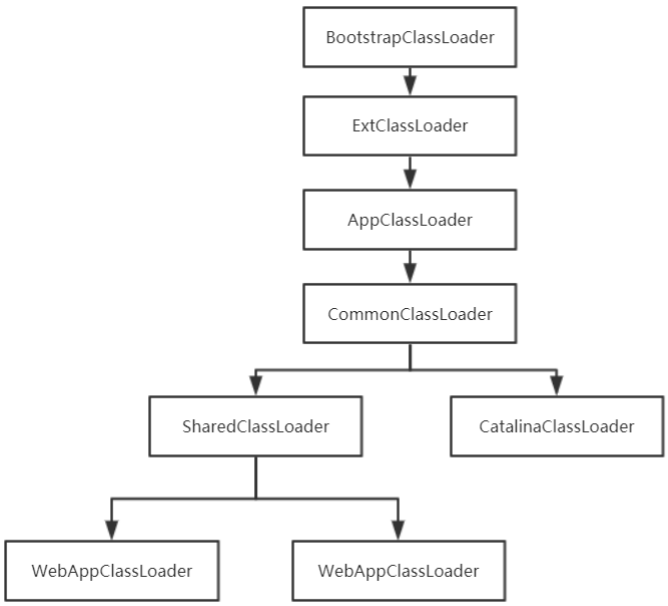
- 同一个Tomcat中，各个Web应用之间各自使用的Java类库要**互相隔离**。
- 同一个Tomcat中，各个Web应用之间可以**提供共享**的Java类库。
- 为了使Tomcat不受Web应用的影响，应该使服务器的类库与应用程序的类库互相独立。
- Tomcat支持热部署。

在 Tomcat中，最重要的一个类加载器是 Common 类加载器，它的父类加载器是应用程序类加载器，负责加载 $CATALINA_ BASE/lib、$CATALINA_HOME/lib 两个目录下所有的.class 文件与.jar 文件。

Tomcat中一般会有多个WebApp类加载器–WebAppClassLoader ，每个类加载器负责加载一个 Web 程序。它的父类加载器是Common类加载器。

由于每个 Web 应用都有自己的 WebApp 类加载器，很好地使多个 Web 应用程序之间互相隔离且能通过创建新的 WebApp类加载器达到热部署。这种类加载器结构能有效使 Tomcat 不受 Web 应用程序影响，而 Common 类加载器的存在使多个 Web 应用程序能够互相共享类库。

Tomcat中类加载器架构



安全机制

Tomcat中设置了了一些安全策略，默认的策略文件为conf/catalina.policy

Tomcat中设置了安全策略，规定了Tomcat在运行过程中拥有的权限，Tomcat管理者可以修改该权限，但是Tomcat中有一些类是必须能够被访问到的，所有Tomcat中在启动过程中会提前去加载这些类，如果发现没有对应的权限，那么将会启动失败。