类加载器

作者：xixiaoming
原文链接：https://ld246.com/article/1491987006350
来源网站：链滴

三种系统提供的类加载器

● 启动类加载器（Bootstrap）：这个类加载器负责将放在JAVA_HOME/lib目录下的类库记载到虚拟内存中，这个加载器无法被程序引用

● 扩展类加载器（Extension）：由sun.misc.Launcher$ExtClassLoader实现，负责加载JAVA_HOM/lib/ext目录或者java.ext.dirs指定的路径的类库

● 应用程序类加载器（Application）：由sun.misc.Launcher$AppClassLoader实现，负责加载用户路径上指定的类库（CLASSPATH指定的类库），一般情况下这个就是程序中默认的类加载器

加载路径简单描述：

BootStrap –>JRE/lib/rt.jar

ExtClassLoader–>JRE/lib/ext/*.jar

AppClassLoader–>CLASSPATH指定的所有jar或目录

双亲委派模型

如果一个类加载器收到了类加载的请求，首先不会自己去尝试加载这个类，而是把请求委派给父类加器去完成，如果父类加载器无法无法完成这个加载请求，子类加载器才会尝试自己去加载，这是推荐类加载器模型

能不能自己写一个类加载java.lang.System?

回答：类加载采用委托机制，总是保证父类优先与子类，也就是如果父类找到了你委托它加载的类，么它直接把加载完的字节码给你,由于BootStrap（启动类加载器）可以加载rt.jar下面有System这个类所以总是由BootStrap来加载System类,而不是用我们自己编写的类加载器来加载

测试代码

```
System.out.println(test.class.getClassLoader().getClass().getName());
// 打印出AppClassLoader,由AppClassLoader加载的
System.out.println(System.class.getClassLoader());
// 打印出null,由BootStrap加载的

// 测试3个类加载器的父子关系
ClassLoader loader = test.class.getClassLoader();
while(loader !=null) {
    System.out.println(loader.getClass().getName());
    loader = loader.getParent();
}
System.out.println(loader);
/*
 * 打印出AppClassLoader----的父类-->ExtClassLoader----的父类-->BootStrap,
 * BootStrap是最顶部的类加载
 */
```