JDK中提供了三个ClassLoader，根据层级从高到低为：

1. Bootstrap ClassLoader，主要加载JVM自身工作需要的类。
2. Extension ClassLoader，主要加载%JAVA_HOME%\lib\ext目录下的库类。
3. Application ClassLoader，主要加载Classpath指定的库类，一般情况下这是程序中的默认类加载器，也是**ClassLoader.getSystemClassLoader()** 的返回值。（这里的Classpath默认指的是环境变量中配置的Classpath，但是可以在执行Java命令的时候使用-cp 参数来修改当前程序使用的Classpath）

JVM加载类的实现方式，我们称为 **双亲委托模型**：

如果一个类加载器收到了类加载的请求，他首先不会自己去尝试加载这个类，而是把这个请求委托给自己的父加载器，每一层的类加载器都是如此，因此所有的类加载请求最终都应该传送到顶层的**Bootstrap ClassLoader**中，只有当父加载器反馈自己无法完成加载请求时，子加载器才会尝试自己加载。

双亲委托模型的重要用途是为了解决类载入过程中的安全性问题。

假设有一个开发者自己编写了一个名为***Java*.lang.Object**的类，想借此欺骗JVM。现在他要使用**自定义ClassLoader**来加载自己编写的***java.lang.Object***类。然而幸运的是，**双亲委托模型**不会让他成功。因为JVM会优先在**Bootstrap ClassLoader**的路径下找到***java.lang.Object***类，并载入它