## 1反射

JAVA反射机制是在运行状态中，对于任意一个类，都能够知道这个类的所有属性和方法；对于任意一个对象，都能够调用它的任意一个方法；这种动态获取的以及动态调用对象的方法的功能称为Java的反射机制。

三种方式获取class对象：

/第一种方式 通过对象**getClass**方法

Person person = new Person();

Class<?> class1 = person.getClass();

//第二种方式 通过**类的class属性**

class1 = Person.class;

try {

//第三种方式 通过**Class类的静态方法——forName()**来实现

class1 = Class.forName("com.whoislcj.reflectdemo.Person");

} catch (ClassNotFoundException e) {

e.printStackTrace();

}

Field[] allFields = class1.**getDeclaredFields()**;//获取class对象的所有属性

Field[] publicFields = class1.**getFields()**;//获取class对象的public属性

**class对象动态生成**

//第一种方式 Class对象调用newInstance()方法生成

Object obj = class1.newInstance();

//第二种方式 对象获得对应的Constructor对象，再通过该Constructor对象的newInstance()方法生成

Constructor<?> constructor = class1.getDeclaredConstructor(new Class[]{String.class});//获取指定声明构造函数

obj = constructor.newInstance(new Object[]{"lcj"});

应用：

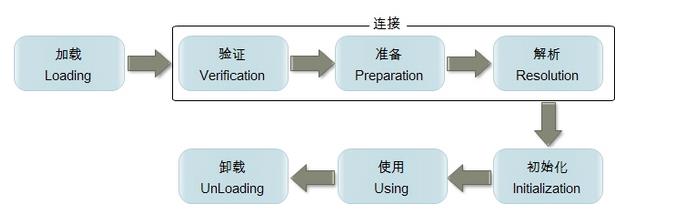
动态代理

Spring的AOP思想

## 2.类加载

Java虚拟机把描述类的数据从Class文件加载到内存，并对数据进行校验、转换解析和初始化，最终形成可以被虚拟机直接使用的Java类型，这就是虚拟机的加载机制。

它的整个生命周期包括了：加载（Loading）、验证（Verification）、准备（Preparation）、解析（Resolution）、初始化（Initialization）、使用（using）、和卸载（Unloading）七个阶段。其中验证、准备和解析三个部分统称为连接（Linking）



三个类加载器：

引导（Bootstrap）类加载器：负责将 <JAVA\_HOME>/lib路径下的核心类库或-Xbootclasspath参数指定的路径下的jar包加载到内存中

扩展（Extension）类加载器：负责加载<JAVA\_HOME>/lib/ext目录下或者由系统变量-Djava.ext.dir指定位路径中的类库

系统（System）类加载器（也称应用类加载器）：负责加载系统类路径java -classpath或-D java.class.path 指定路径下的类库

## 3.Java8新特性（十大）

### 1. 函数式接口和Lambda表达式

函数式接口是只包含一个方法的抽象接口，通过@FunctionalInterface注解（非必须），将一个接口标注为函数式接口。函数式接口是为Java 8中的lambda而设计的，lambda表达式的方法体其实就是函数接口的实现。

Lambda表达式语法结构如下：

(parameters) -> expression 或者 (parameters) -> {statements;}

lambda表达式使代码变得简单，代替了匿名内部类

new Thread(new Runnable(){

@Override

public void run() {

for (int i = 0; i < 10; i++)

System.out.println("Without Lambda Expression");

}}).start();

### 2.接口的默认方法

（default修饰）：接口里也可以写方法体了，实现该接口的类不再强制实现该方法，只需要在方法签名增加default签名并实现方法体；

接口可以定义**静态方法**，通过接口调用。实现类不需实现，也无法在实现类中直接调用。

### 3.方法引用

构造器引用。语法是Class::new，或者更一般的Class< T >::new，要求构造器方法是没有参数；

静态方法引用。语法是Class::static\_method，要求接受一个Class类型的参数；

特定类的任意对象方法引用。它的语法是Class::method。要求方法是没有参数的；

特定对象的方法引用，它的语法是instance::method。要求方法接受一个参数，与3不同的地方在于，3是在列表元素上分别调用方法，而4是在某个对象上调用方法，将列表元素作为参数传入；

### 4.重复注解

相同的注解在同一地方也可以声明多次。重复注解机制本身需要用@Repeatable注解。

### 5.扩展注解的支持

Java 8扩展了注解的上下文，几乎可以为任何东西添加注解，包括局部变量、泛型类、父类与接口的实现，连方法的异常也能添加注解

### 6.引入optional类

Java 8引入Optional类来防止空指针异常，Optional类最先是由Google的Guava项目引入的。Optional类实际上是个容器：它可以保存类型T的值，或者保存null。使用Optional类我们就不用显式进行空指针检查了

### 7.Stream API

Stream API是把真正的函数式编程风格引入到Java中。

### 8.Date/Time API

Java 8新的Date-Time API (JSR 310)受Joda-Time的影响，提供了新的java.time包，可以用来替代 java.util.Date和java.util.Calendar。一般会用到Clock、LocaleDate、LocalTime、LocaleDateTime、ZonedDateTime、Duration这些类，对于时间日期的改进还是非常不错的。

### 9.引入Javascript引擎Nashorn

Nashorn允许在JVM上开发运行JavaScript应用，允许Java与JavaScript相互调用。

### 10.Base64

对Base64编码的支持已经被加入到Java 8官方库中，这样不需要使用第三方库就可以进行Base64编码

### 其他

更好的类型推测机制：Java 8在类型推测方面有了很大的提高，这就使代码更整洁，不需要太多的强制类型转换了。

编译器优化：Java 8将方法的参数名加入了字节码中，这样在运行时通过反射就能获取到参数名，只需要在编译时使用-parameters参数。

并行（parallel）数组：支持对数组进行并行处理，主要是parallelSort()方法，它可以在多核机器上极大提高数组排序的速度。

并发（Concurrency）：在新增Stream机制与Lambda的基础之上，加入了一些新方法来支持聚集操作。

Nashorn引擎jjs：基于Nashorn引擎的命令行工具。它接受一些JavaScript源代码为参数，并且执行这些源代码。

类依赖分析器jdeps：可以显示Java类的包级别或类级别的依赖。

JVM的PermGen空间被移除：取代它的是Metaspace（JEP 122）

## 4.session

从打开一个浏览器访问某个站点，到关闭这个浏览器的整个过程，成为一次会话。会话技术就是记录这次会话中客户端的状态与数据的。会话技术是帮助服务器记住客户端状态（区分客户端）。

会话技术分为Cookie和Session：

Cookie：数据存储在客户端本地，减少服务器端的存储的压力，安全性不好，客户端可以清除cookie

Session：将数据存储到服务器端，安全性相对好，增加服务器的压力

Session技术是将数据存储在服务器端的技术，会为每个客户端都创建一块内存空间 存储客户的数据，但客户端需要每次都携带一个标识ID去服务器中寻找属于自己的内 存空间。所以说Session的实现是基于Cookie，Session需要借助于Cookie存储客 户的唯一性标识JSESSIONID

3．**Session对象的生命周期**（面试题/笔试题）

创建：第一次执行request.getSession()时创建

销毁：

1）服务器（非正常）关闭时

2）session过期/失效（默认30分钟）

问题：时间的起算点 从何时开始计算30分钟？

从不操作服务器端的资源开始计时

可以在工程的web.xml中进行配置

<session-config>

<session-timeout>30</session-timeout>

</session-config>

3）手动销毁session

session.invalidate();

作用范围：

默认在一次会话中，也就是说在，一次会话中任何资源公用一个session对象

面试题：浏览器关闭，session就销毁了？ 不对

## 5.servlet生命周期

(3)Servlet的生命周期（面试题）

1）Servlet何时创建

默认**第一次**访问servlet时创建该对象

2）Servlet何时销毁

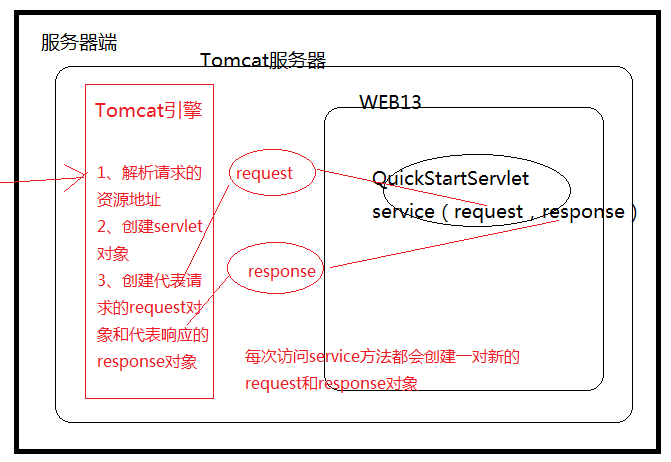
服务器关闭servlet就销毁了

3）每次访问**必然执行**的方法

service(ServletRequest req, ServletResponse res)方法

对XXXServlet进行了10次访问，init()，destory()，service()，doGet()，doPost() 一共执行力多少次？request对象创建几个？response创建几个？

（1，0，10，x,x）(10,10)



## 6.tomca

2．Web开发中常用的web应用服务器

1）weblogic：oracle公司的大型收费web服务器 支持全部javaEE规范

2）websphere：IBM公司的大型收费web服务器 支持全部的javaEE规范

3）Tomcat：Apache开源组织下的 开源免费的中小型的web应用服务器 支持 javaEE 中的 servlet 和 jsp规范t

## 7.排序算法（8个）