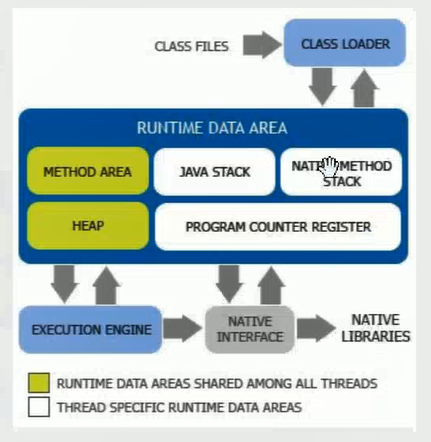
# JVM性能调优

Book book = new Book();

new Book();堆区---新生代---伊甸园区

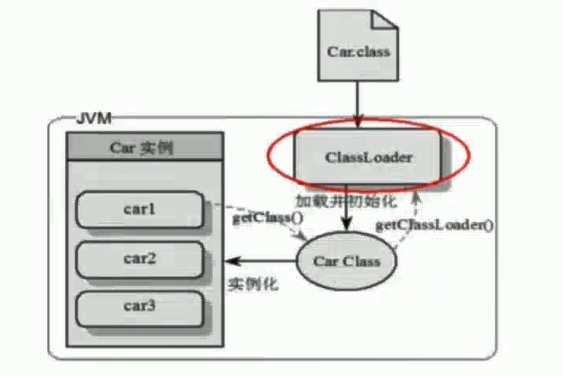
所有的Java文件最终都会被编译为class文件，通过类加载器classloader加载到运行数据的区域，该区域又分为方法区，栈区，堆区，

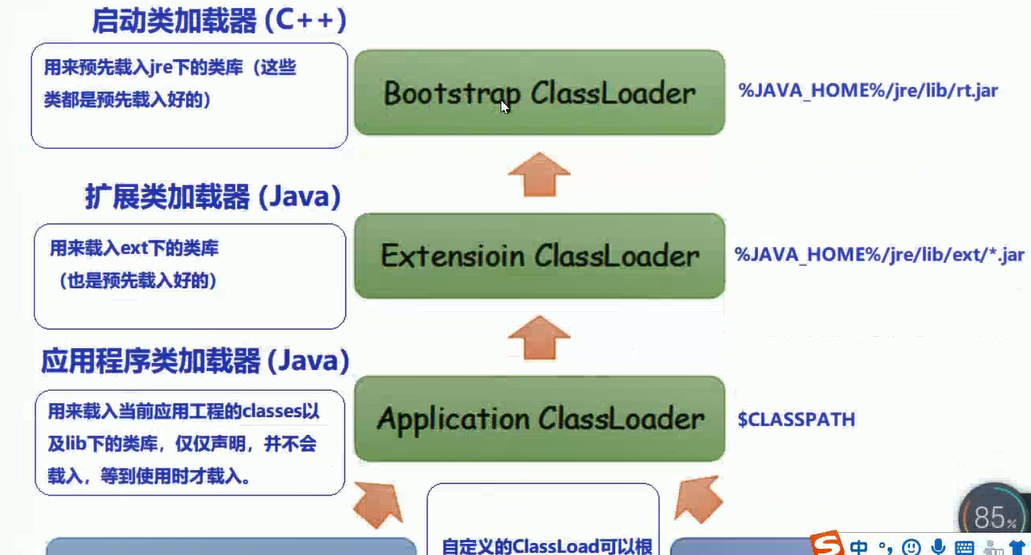


#### 1.class loader类加载器

负责加载class文件，class文件在开头有特定的文件提示，并且classLoader只负责class文件的加载，至于是否可以运行，则有Execution engine负责

类记载的机制：双亲委派机制（为了安全，防止重复加载）





#### 2.Java栈

后进先出，所以是个队列，不可能无限大，可能会栈内存溢出

StackOverFlowError :典型原因：方法无限递归

哪些东西会放到栈区：引用、基本数据类型，局部变量

#### 3.nativeMethodStack

本地方法栈，里面是一些底层的函数库，Java不能调用到，C去调用 Java jni c

#### 4.program counter register（程序注册计数器/PC寄存器）：

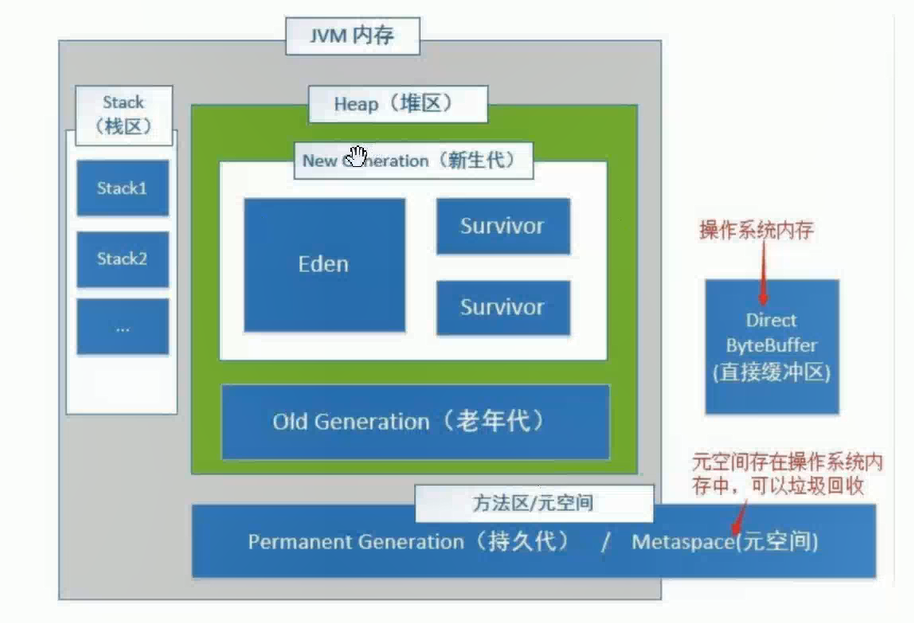
Java代码的执行顺序，没执行一行都会记录下来（空白行不会编译，但是也会统计）所占的内存空间可忽略不计

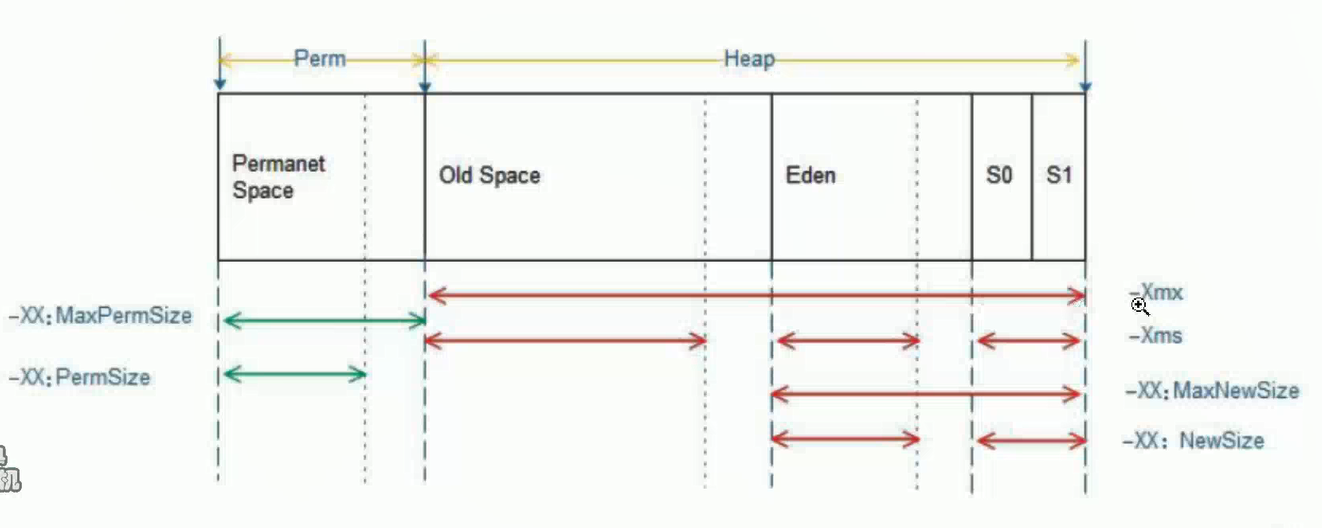
#### JVM调优，其实指的就是堆内存调优

堆：对象

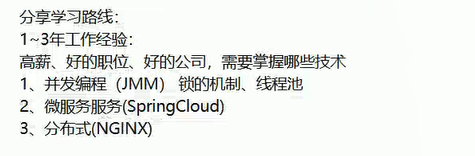
Heap堆区：分为新生代，老年代，持久代三个区域（JDK1.8后持久代叫元空间Metaspace），new的对象首先放在Eden新生代的伊甸园区，伊甸园区满了，会把对象移放到新生代的Survivor（Survivor from/Survivor To）区中的某一个区域

引用类型：强引用，软引用，弱引用，虚引用，强引用，就算JVM抛出OOM异常也不会被回收，OOM（OutOfMemoryError虚拟机内存不够）











# Mybatis

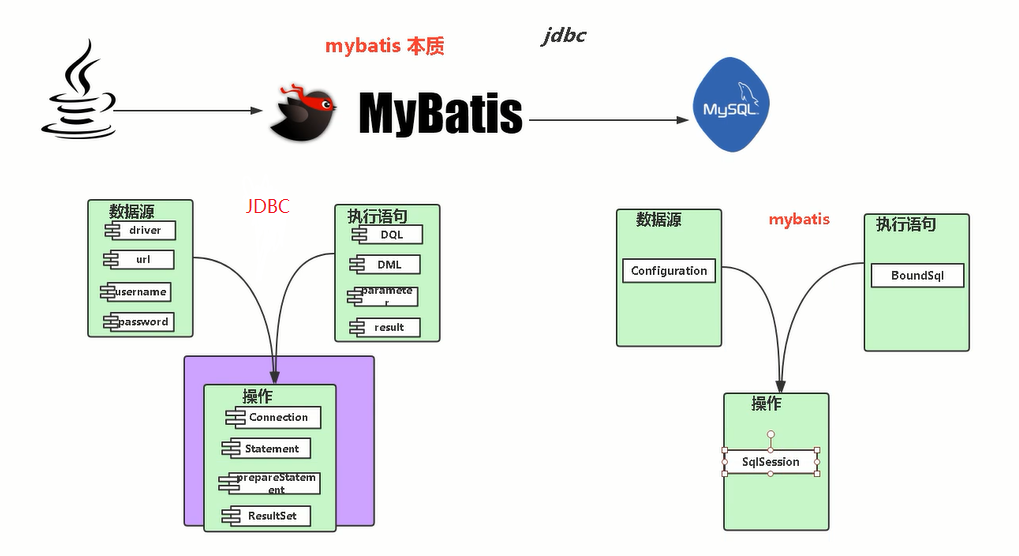
Mybatis前身叫ibatis，本是apache的一个开源项目，2010年这个项目有apache软件基金迁移到Google code，并且改名为mybatis

Mybatis是支持普通SQL查询，存储过程和高级映射的优秀持久层框架，半自动ORM框架

Mybatis消除了几乎所有的JDBC代码和参数的手工设置以及结果集的检索

Mybatis使用简单的XML或注解用于配置和原始映射，将接口和Java的POJO（Plan Old Java Object，普通Java对象）映射成数据库中的记录

Mybatis的本质：解决我们相关的Java和MySQL之间的协调作用，也就是说Java里面可以通过POJO这种面向对象的形式拿到MySQL中的SQL



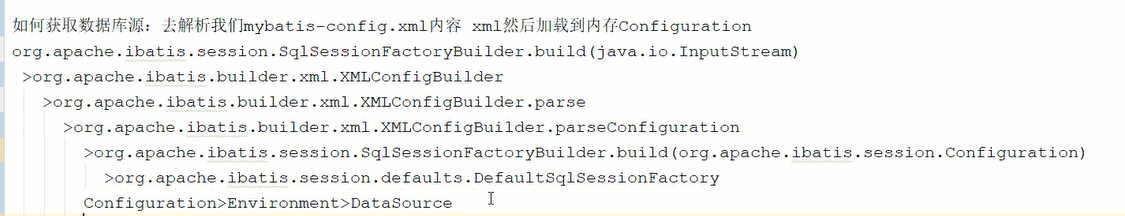
Mybatis加载mapper的方式有四种：

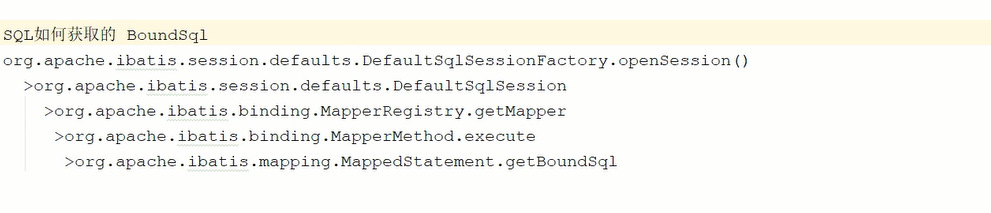
<mapper resource=”mybatis/UserMapper.xml”></mapper>

<mapper class = “”></mapper>

<mapper url = “”></mapper>

<package name = “” ></mapper> 根据包名







Java动态代理：

Java.lang.reflect.InvocatonHandler

实现意义：不改变被带离的对象的源码的情况下，对这个被代理的对象方法增强

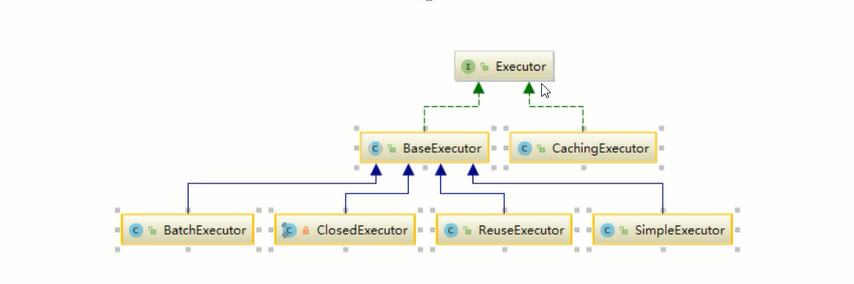
mybatis底层用的是JDBC

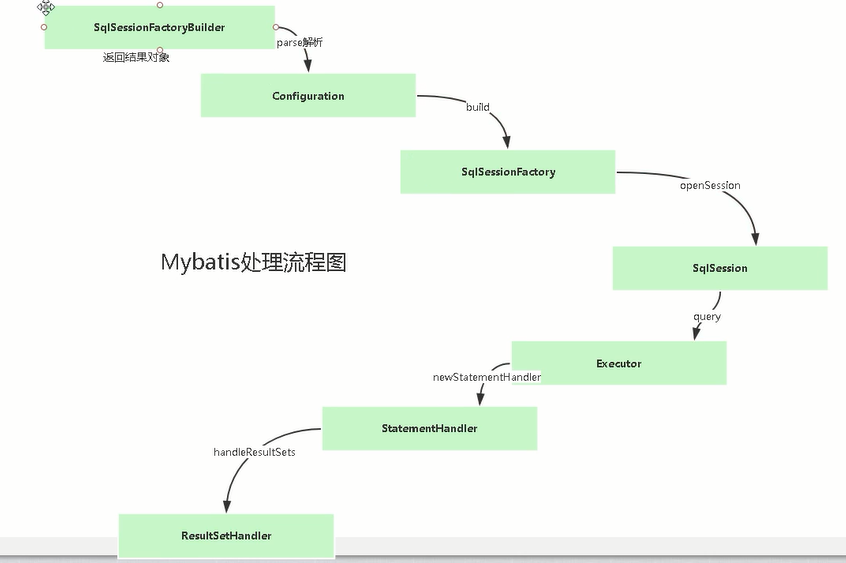
preparedStatement和statement的区别在于preparedStatement能够防止SQL注入

Mybatis的以及缓存默认是开启的

cacheEnable默认位true也就是一级缓存默认是开启的，mybatis自带一级缓存（session级别的缓存），但是这个一级缓存在spring中是失效的，因为不用spring的时候，我们的session要自己手动去关，而用了spring之后，我们的session在请求结束后，spring自动就关闭session，所以一级缓存也就失效了

mybatis的执行器类图关系





## Mybatis的运行原理：

1. 首先会进行初始化，初始化mapper，初始化数据库连接， 初始化mybatis需要的一些类，比如SqlSessionFactory，再来初始化mapper里面的SQL和Id，把它们放到一个叫作mappedStatements的Map对象当中，还会去扫描包，mybatis会根据我们的包名得到项目名，得到class所在文件夹的名称，通过文件夹得到里面的文件名，通过 反射把我们的类加载到内存中，就完成了mapper的扫描;
2. 底层的调用，拿到这些mapper后，通过SqlSession来调用的，通过调用selectList()方法，selectList()方法调用Execute接口，Execute有cacheExecute，defaultExecute， simpleExecute，mybatis默认用的cacheExecute，cacheExecute底层调用的是JDBC的preparedStatement执行的Executor语句

# SpringMVC源码分析

在springMVC里面写一个controller实现一个控制器的方式有三种：

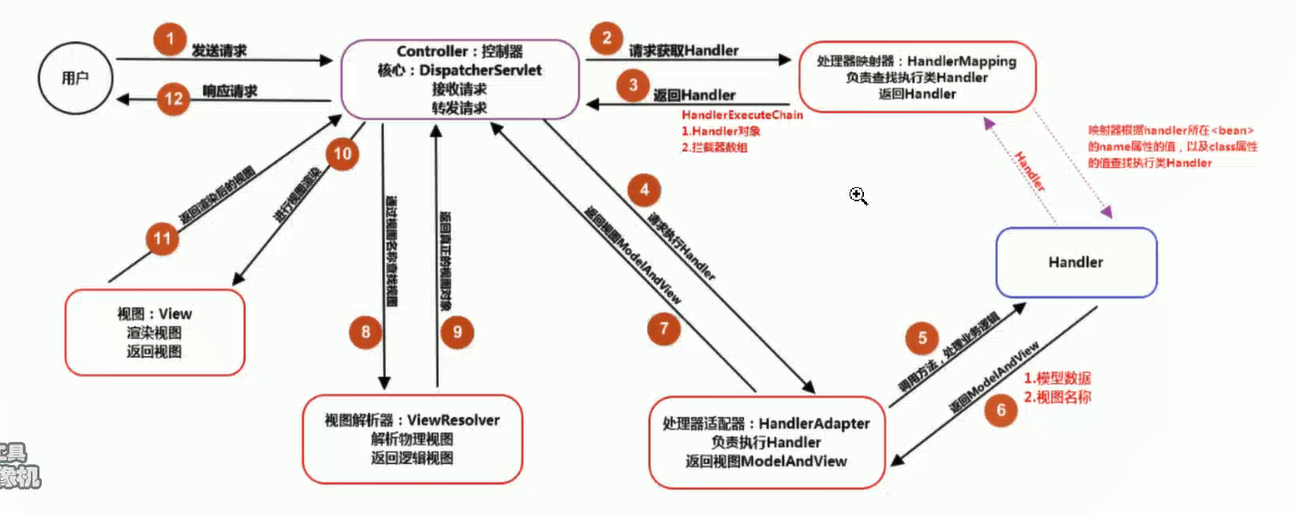
1. 实现Controller接口，重写handleRequest()方法，在配置文件中配置bean把控制器交给spring容器管理
2. 使用@Controller注解，xmlns:congtext=””添加约束，

使用<context:component-scan base-package=””></context:component-scan>配置包扫描，使用<mvc：annotation-driven></mvc:annotation-driven>开启springMVC注解支持

1. 使用XML

虽然使用注解的方式比起xml配置的方式使用起来更加简单，但是SpringMVC的底层代码，使用注解的方式当地曾去调用getHandlerAdaptor()方法时其实比使用xml配置的方法复杂的多。SpringMVC底层已经封装定义好乐三个HandlerAdaptor，和两个HandlerMapping

在 使用xml配置来实现控制器的时候，底层匹配成功后，其实只做了强转获取到对应的控制器Controller，二使用注解的方式，底层就不是强转了，而是做了一些列的操作，最终匹配到对应的控制器



SpringMVC简单执行原理：





# Sprig之AOP

什么时AOP

与OOP相比，面向切面，传统的OOP开发中的代码逻辑是至上而下的，在这些至上而下的过程中会产生一些横切性的问题，这些横切性的问题和我们的主页务逻辑关系不大，会散落在代码的各个地方，造成难以维护，AOP的编程思想就是把业务逻辑和横切的问题进行分离，从而达到解耦的目的，使代码的重用性和开发效率高

AOP的应用场景：  
1.日志的记录

2.权限验证

3.效率查询

4.事务管理。。。。

AOP的底层技术实现：

JDK动态代理和CGLIB动态代理，spring在执行的过程中到底使用JDK动态代理还是CGLIB动态代理是有条件的，首先看有没有接口， 并且看配置proxyTargetClass的值，默认位false（1）如果有接口并且proxyTargetClass为false的话，spring将使用JDK的动态代理；

（2）如果proxyTargetClass为true的话，不管有没有接口spring都会使用CGLIB的动态代理；

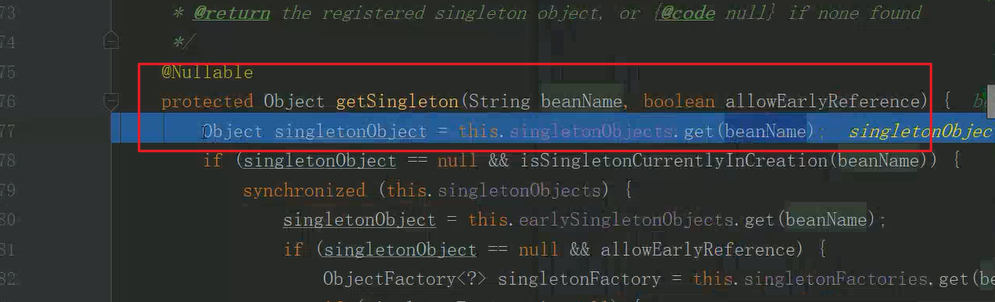
（3）如果没有接口，并且proxyTargetClass为false的话，spring将会用CGLIB的动态代理；

编译时期的织入还是运行时期的织？

答：两者都是在运行时期织入的

初始化时期织入还是获取对象时期织入？

答：通过源码分析，可以知道是在初始化时期织入的



SpringAOP和AspectJ的关系

@AspectJ support ----------------->利用aspectj的注解

Schema-based AOP support --------->xml

证明：spring，通过源码分析，我们可以知道spring底层使用的是JDK或者CGLIB来完成的代理，并且在官网给出了aspectJ的文档，和springAOP是不同的

SpringAOP中的术语：

1. aspect切面：一定要给spring去管理
2. join point连接点：目标对象中的方法称为连接点
3. pointcut切点：切点表示连接点的集合
4. weaving织入：被代理逻辑加入到目标对象中的过程叫做织入
5. target object目标对象：
6. AOP proxy代理对象：
7. advice通知：

通知类型:

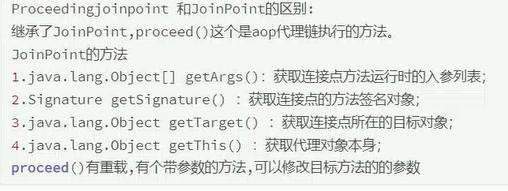
Before

After

After throwing

After (finally)

Around adice



AOP是我们最终要达成的目的，springAOP是具体的实现，aspectJ也能实现AOP，springAOP借助AspectJ的语法来实现AOP

springAOP是实现切面编程、

1. 导入spring相关的jar包，并导入aspectJ的依赖
2. 定义一个切面类，这个类必须交给spring容器管理，并使用@Aspect注解来声明
3. 声明一个pointcut,在切面类中定义方法，里面不需要任何实现，他仅仅是一个载体， 并使用@Pointcut注解来声明， 用来描述有哪些方法，那些类@Pointcut(“”)注解中使用了aspectJ表达式和springEL表达式的结合，用来描述我们定义的切点要放到那些类，哪些方法当中



1. 声明一个通知，通知要去关联链接点（通过注解去找声明的切点，通过切点找到切点下的连接点），连接点就是一个方法（切点是连结点/方法的集合），再通知里面嵌入逻辑；通过@Before @After。。等注解，来声明是前置通知还是后置通知，然后把通知织入到连接点的适当位置

# 哈希算法与HashMap

一个优秀的哈希算法有什么特点：

1. 快速性：速度快，效率高
2. 不可逆性
3. 敏感性
4. 低碰撞性

速度、空间、碰撞性进行取舍，不同的哈希算法侧重点不同

Java中的Hash算法

1. HashMap，侧重点是速度
2. Object.hashCode,直接获取内存地址
3. Integer.hashCode,直接返回的intValue
4. String.hashCode,根据字符串内容生成hashCode, 字符串内容一样则hashCode也一样

哈希算法的用途

1. 哈希查找，哈希表
2. 秒传
3. HashMap
4. 加解密，MD5，SHA
5. Git
6. 比特币，区块链

为什么在重写equals的时候，必须要重写hashCode方法？

# 理解restfull

RESTful架构，就是目前最流行的一种互联网软件架构。它结构清晰、符合标准、易

于理解、扩展方便，所以正得到越来越多网站的采用。REST这个词，是Roy Thomas

Fielding在他2000年的博士论文中提出的 .

REST 是Representational State Transfer的缩写，翻译是”表现层状态转化”。 可以

总结为一句话：REST是所有Web应用都应该遵守的架构设计指导原则。

面向资源是REST最明显的特征，对于同一个资源的一组不同的操作。资源是服务器

上一个可命名的抽象概念，资源是以名词为核心来组织的，首先关注的是名词。REST要

求，必须通过统一的接口来对资源执行各种操作。对于每个资源只能执行一组有限的操

作。

7个HTTP方法：GET/POST/PUT/DELETE/PATCH/HEAD/OPTIONS

## 幂等的意思：

在操作成功的前提条件下，不会对数据库造成额外不好的影响，

get是安全与幂等的，因为我们用get方法用来做查询的，查询只是查询数据，并不会给数据库种的数据带来不好的影响，一般查询的时候甚至可以不需要事务，所以不会出现数据库事务的并发问题：脏读、不可重复读、幻读，所以Get是安全且幂等的；

Post、put、delete只要涉及到增、删、改都必须要有事务的控制，涉及到事务就有可能会出现事务的并发问题，所以它是不安全的；Post也不幂等：比如说现在要插入一条记录，那么就往数据库中就新增了一条记录，但是因为网络延迟等等的一些原因，做了重复操作，又进了一条记录，两次都成功了，但是对数据库就造成了不好的影响，因为本质上我们只想插入一条记录，实际上却插入了两条内容完全一样的记录；比如说要下单，只需要下一个订单，但是因为网络延迟的问题，第一次提交的时候，可能已经发出去了，返回的消息没返回，可能就认为没有成功，又提交了一次则又做了一次保存操作，那么同一次保存操作内容是一摸一样的，id却不一样了，这种情况就是不幂等了

幂等：说白了就是我们每次操作数据库，对数据库中的数据操作受影响的记录条数是否与操作次数保持一致

## GET

安全且幂等

获取表示

变更时获取表示（缓存）

200（OK） - 表示已在响应中发出

204（无内容） - 资源有空表示

301（Moved Permanently） - 资源的URI已被更新

303（See Other） - 其他（如，负载均衡）

304（not modified）- 资源未更改（缓存）

400 （bad request）- 指代坏请求（如，参数错误）

404 （not found）- 资源不存在

406 （not acceptable）- 服务端不支持所需表示

500 （internal server error）- 通用错误响应

503 （Service Unavailable）- 服务端当前无法处理请求

## POST

不安全且不幂等

使用服务端管理的（自动产生）的实例号创建资源

创建子资源

北京市昌平区建材城西路金燕龙办公楼一层 电话：400-618-9090

部分更新资源

如果没有被修改，则不过更新资源（乐观锁）

200（OK）- 如果现有资源已被更改

201（created）- 如果新资源被创建

202（accepted）- 已接受处理请求但尚未完成（异步处理）

301（Moved Permanently）- 资源的URI被更新

303（See Other）- 其他（如，负载均衡）

400（bad request）- 指代坏请求

404 （not found）- 资源不存在

406 （not acceptable）- 服务端不支持所需表示

409 （conflict）- 通用冲突

412 （Precondition Failed）- 前置条件失败（如执行条件更新时的冲突）

415 （unsupported media type）- 接受到的表示不受支持

500 （internal server error）- 通用错误响应

503 （Service Unavailable）- 服务当前无法处理请求

## PUT

不安全但幂等

用客户端管理的实例号创建一个资源

通过替换的方式更新资源

如果未被修改，则更新资源（乐观锁）

200 （OK）- 如果已存在资源被更改

201 （created）- 如果新资源被创建

301（Moved Permanently）- 资源的URI已更改

303 （See Other）- 其他（如，负载均衡）

400 （bad request）- 指代坏请求

404 （not found）- 资源不存在

406 （not acceptable）- 服务端不支持所需表示

409 （conflict）- 通用冲突

412 （Precondition Failed）- 前置条件失败（如执行条件更新时的冲突）

415 （unsupported media type）- 接受到的表示不受支持

500 （internal server error）- 通用错误响应

503 （Service Unavailable）- 服务当前无法处理请求

## DELETE

不安全但幂等

删除资源

200 （OK）- 资源已被删除

301 （Moved Permanently）- 资源的URI已更改

303 （See Other）- 其他，如负载均衡

400 （bad request）- 指代坏请求

404 （not found）- 资源不存在

409 （conflict）- 通用冲突

500 （internal server error）- 通用错误响应

503 （Service Unavailable）- 服务端当前无法处理请求

# 数据库笔试常考SQL练习