Mybatis 使用xml 和注解将 pojo plain old java object releclt to database 需要自己写sql

属于持久层ORM框架

1. 持久层: 讲内存中对象数据,转移到数据库中的过程持久层

Mybatis Hibernate Spring-jpa

MyBatis supports declarative data caching. A statement can be marked as cacheable so any data retrieved from the database will be stored in a cache and future executions of that statement will retrieve the cached data instead hitting the database. MyBatis provides a default cache implementation based on a Java HashMap and default connectors for integrating with: [OSCache](https://en.wikipedia.org/w/index.php?title=OSCache&action=edit&redlink=1" \o "OSCache (page does not exist)), [Ehcache](https://en.wikipedia.org/wiki/Ehcache" \o "Ehcache), [Hazelcast](https://en.wikipedia.org/wiki/Hazelcast" \o "Hazelcast) and [Memcached](https://en.wikipedia.org/wiki/Memcached" \o "Memcached). It provides an [API](https://en.wikipedia.org/wiki/API" \o "API) to plug other cache implementations.

1. mybatis 导入jar 包
2. 新建xml 输入数据库配置信息
3. 配置java bean 和 sql 的Mapper 位置
4. 配置sqlsession xml 即要map的bean和sql
5. 返回值的类要编写 比如要dep要有实体entity dept 的类
6. 用配置信心的xml 的inputstream创建session会话
7. 打开会话
8. Session.selectList(传入4)
9. Mybatis sql语句是prestatement
10. select的id必须和接口的方法名一致
11. resultType和接口返回值的泛型一致
12. 没有返回值就不写
13. 接口映射 接口名 一致
14. 方法名和mapper的xml 中的 id一致 返回类型一致
15. 入参类型可以是基本类型 ，可以是包装类，String,javabean

List 数组，map

Map入参xml对应的是key 传递的是value

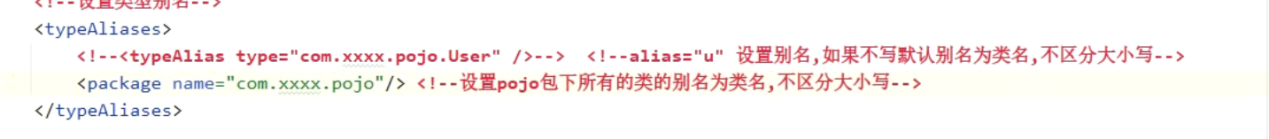
Where 多个名字 可以for each (collection=“list” ,item=“”



接口绑定的方法中也要是list



Db.properties 配置再mybatis.xml最上面 然后下面用${dollar}匹配

事务默认需要手动提交/也可以按照下面来设置为auto commit

loglevel fatal error warn info debug

级别高的才会被打印 LOGFILE 才会有file输出

# Set root category priority to INFO and its only appender to CONSOLE.

log4j.rootCategory=INFO, CONSOLE, LOGFILE

# log4j.rootCategory=INFO, CONSOLE, LOGFILE

#单独设置要进行sql 语句的级别为debug 级别

#set current package to debug mode

log4j.logger.com.Neil.mappers=DEBUG

# CONSOLE is set to be a ConsoleAppender using a PatternLayout.

log4j.appender.CONSOLE=org.apache.log4j.ConsoleAppender

log4j.appender.CONSOLE.layout=org.apache.log4j.PatternLayout

log4j.appender.CONSOLE.layout.ConversionPattern=- %m %c %l %d{yyyy-MMdd HH:mm:ss}%n

# LOGFILE is set to be a File appender using a PatternLayout.

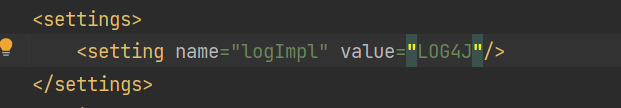
log4j.appender.LOGFILE=org.apache.log4j.FileAppender

log4j.appender.LOGFILE.File=G:/test.log

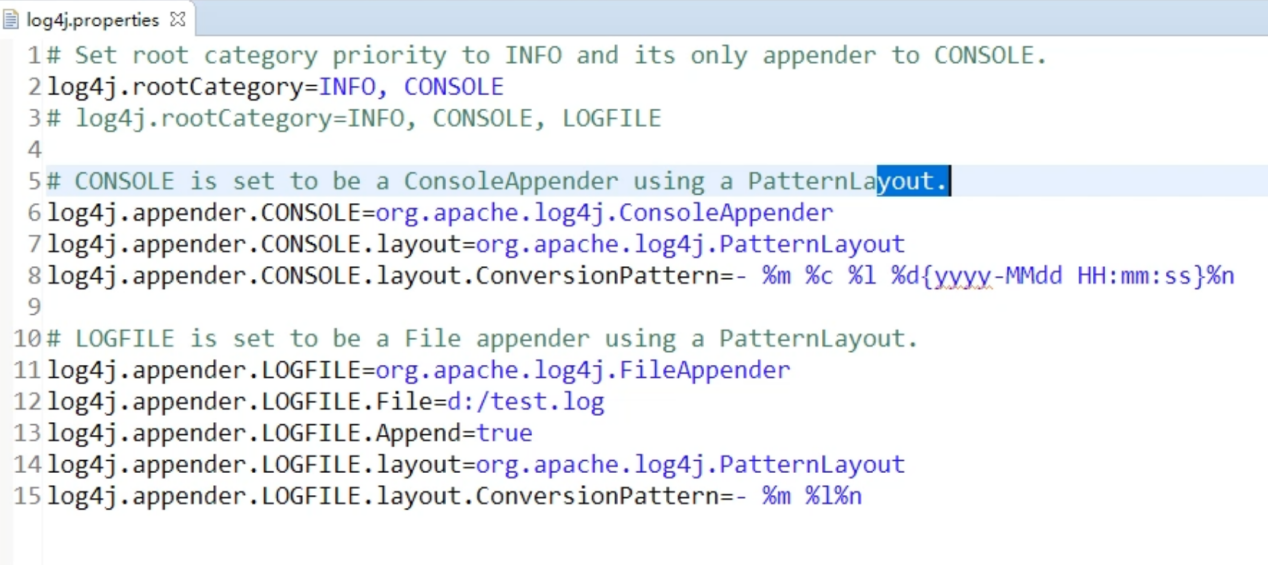
log4j.appender.LOGFILE.Append=true

log4j.appender.LOGFILE.layout=org.apache.log4j.PatternLayout

log4j.appender.LOGFILE.layout.ConversionPattern=- %m %l %d{yyyy-MMdd HH:mm:ss}%n







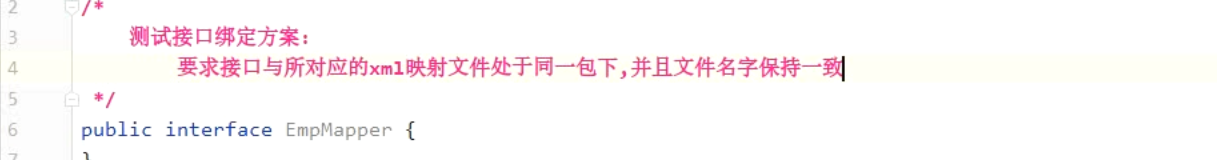
Resulttype为map 返会一个元组 key 为属性 只能接受一个元组

查询用selectOne 方法

Map list<map> as resulttype 在多表联查的时候可以使用



根据业务设计表的时候 如何设计 减少orm 的代码量



# 测试接口绑定方案：

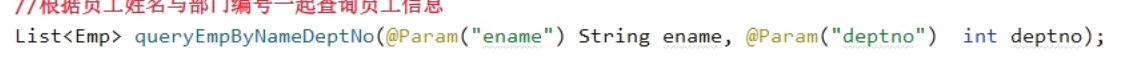
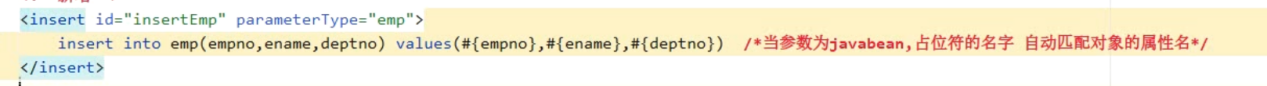
1.要求接口与所对应的xml映射文件处于同一包下,并且文件名字保持一致

2.sq1映射文件的命名空间namespace 要求设置为所对应的接口的权限定名 包名.类名 com.xxxx.mapper.EmpMapper

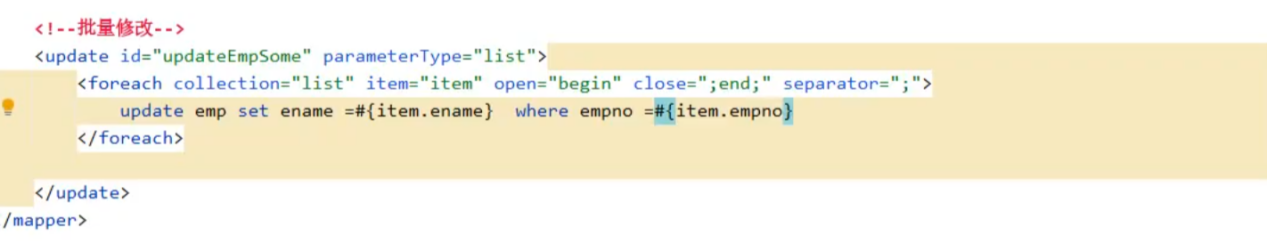
3.sq1映射文件中sq1语句标签的id值要与对应的借口中的对应的抽象方法的方法名称保持一致

 \



在方法中使用注解 

批量修改 begin end 类似存储过程 separator 是sql 的语句分割 所以结尾需要写上；end；



批量插入



Mybatis 占位符

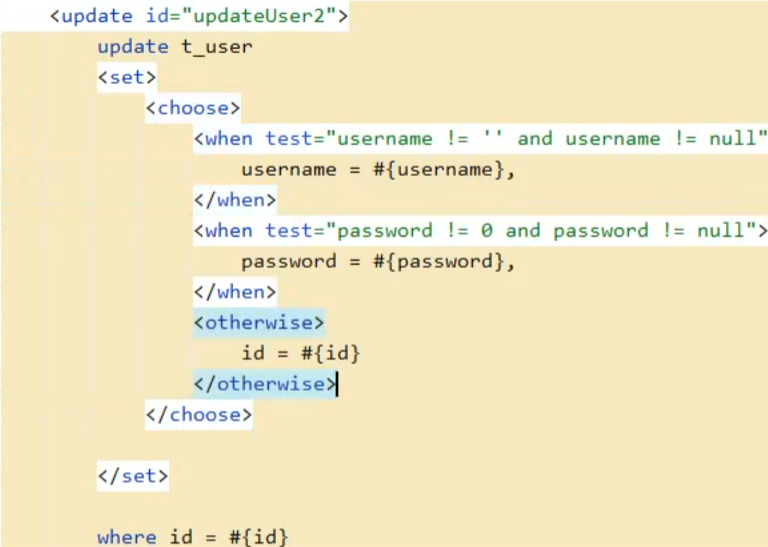
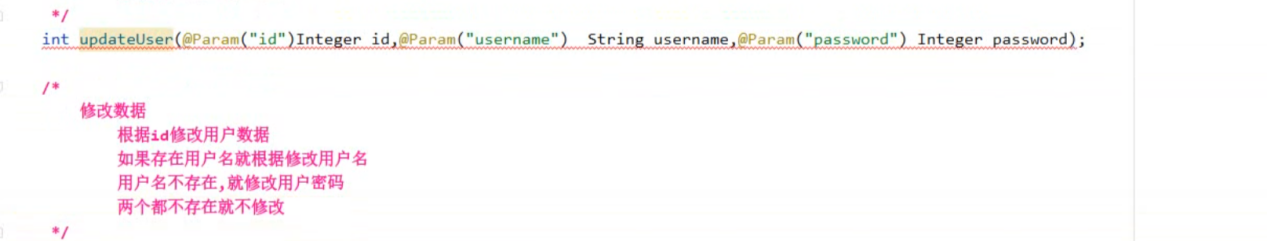


p

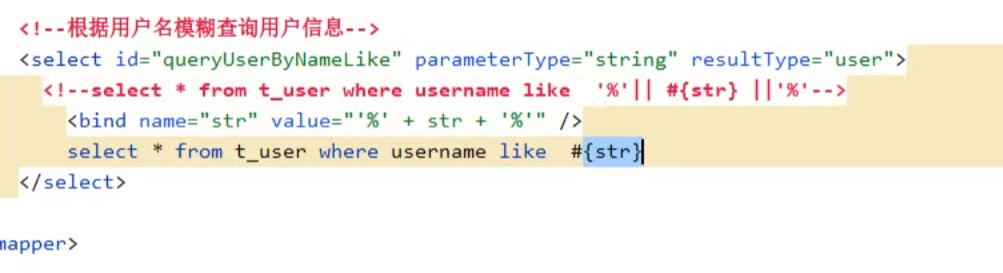




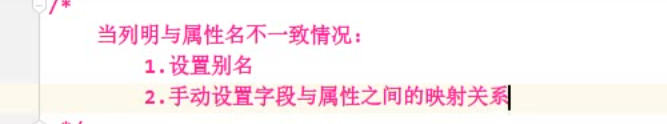
存在才修改choose中选一个when



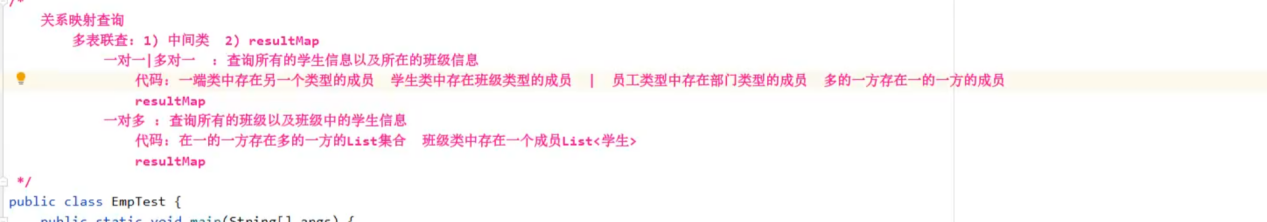
拼接字符 的 sql 语句



当pojo 类字段与表列名不一致





多表联查 使用resultmap

查询emp 所在的dept



一个类中存在另一个类的对象的容器

# Mybatis 源码

Select \* from emp left join dept on emp.deptno=dept.deptno where empno = ?

new SqlSessionFactoryBuilder().build(is);

XPathParser把xml 解析成document 对象 node 节点

Parse.parser 中evalNode 中进行parser.parse()$ propertis 对象进行解析

返回XNode

Xnode 像dom

ParseConfiguration 解析

返回configuration文件

传给sqlsessionFactory

Open session

返回一个defaultsqlsession

Congiguration.getEnvironment

Transactionfactory new transaction

创建Excutor 接口 执行器

会话没有建立连接

获取mapper对象

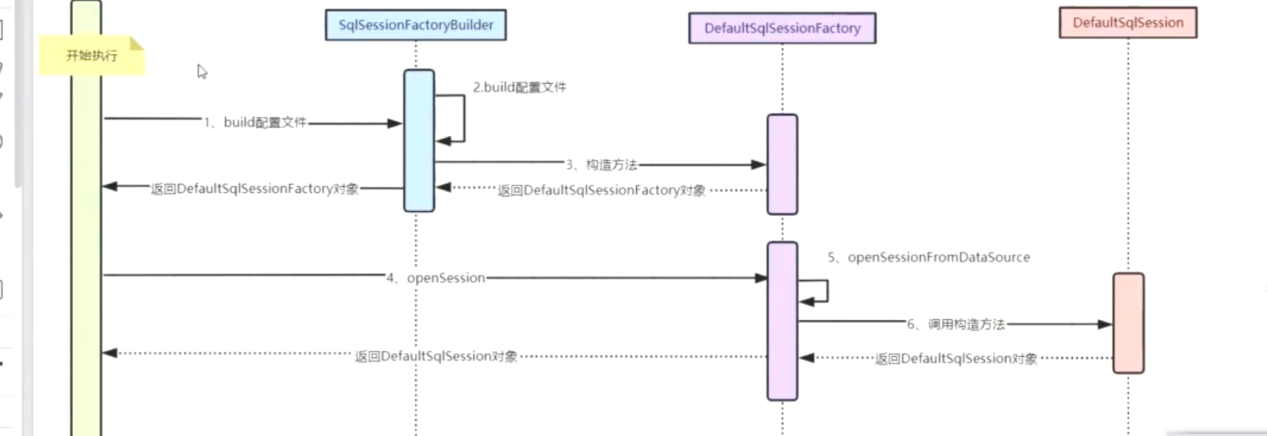
把xml 里的mapper 类放进hashmap knowmappers 中

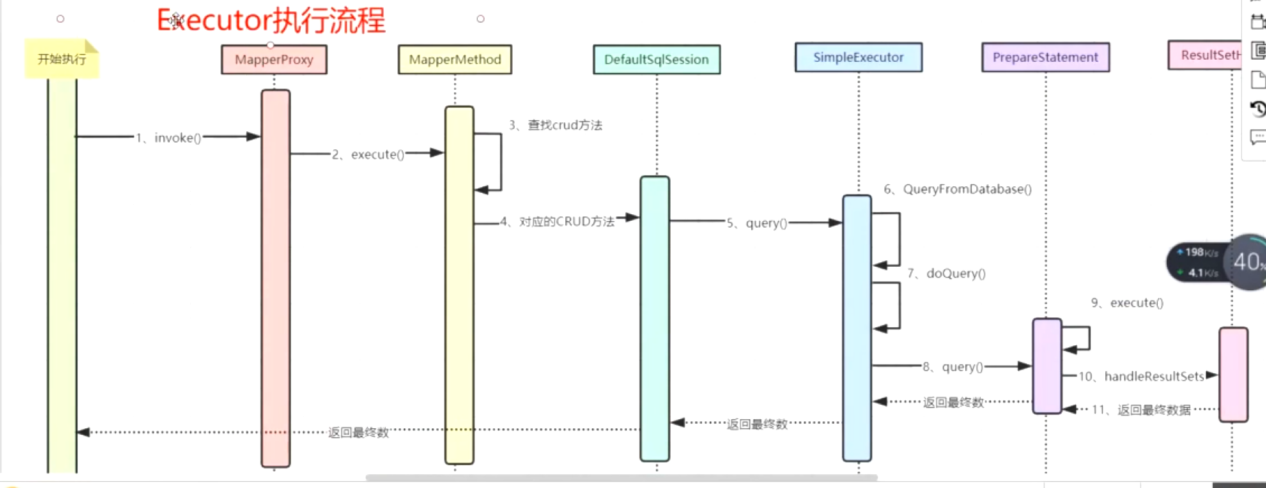
Knownmapper 转成 mappperproxyFactory

MapperproxyFactory.newInstance

返回mapper proxy 代理对象

使用jdk 动态代理





Mapper.select...

Invoke pojo 方法 判断有无缓存

一级缓存

二级缓存 enable cashe 标签

三级缓存

Execute 方法中判断crud 方法

判断pojo类方法的返回值

决定用excuteForMap excuteForMany executeFor。。。。

convertArgsToSqlCommanParam解析参数到sql 语句中

返回paramNameResolver.getNamedParams

判断参数有无注解

多值参数 参数放入map entry 中 param1 param2

带着方法名称进入到SelectOne

带着statement，parameter执行selectLIst方法

New mapperstatement

Complete 验证

Buildallstaztement

非空合法性验证（缓存，statement）

返回mapperstatement (和jdbc 一样 参数包括 preparedstatement

Sql source）文件

Executor.query boundsql

Create cachekey

根据hashkey 判断sql 语句产生一个唯一的key

把result 和key 放入缓存中去

如果key 存在

获取缓存 tcm 事务缓存管理获 取缓存 如果缓存不为空 判断是否需要刷出缓存 二级缓存（session 关闭之后生效）

如果resulthandler 为空 如果querystack == 0, 刷出并清空缓存

Querystack！= 0 使用一级缓存

两个重复语句使用一级缓存

从localcashe perpetual cashe ( 一级缓存） 中取得列表

否则从不为空则从数据库中取 就把key 放入缓存

查询 jdbc connection

Prestatement resultmap 结果封装

resultsetMapper()

Tcm是二级缓存 tcm.commit() 后二级缓存才有效

先查二级缓存 ，再查一级缓存

public static void main(String[] args) {

try (SqlSession sqlSession = sqlSessionFactory.openSession()) {

UserMapper userMapper = sqlSession.getMapper(UserMapper.class);// <1>

User user = userMapper.selectById(1);

System.out.println("User : " + user);

}

}

在前面的例子中，我们使用sqlSession.getMapper()方法获取UserMapper对象，实际上这里我们是获取了UserMapper接口的代理类，然后再由代理类执行方法。那么这个代理类是如何生成的呢？在探究动态代理类如何生成之前，我们先来看下SqlSessionFactory工厂的创建过程做了哪些准备工作，比如说mybatis-config配置文件是如何读取的，映射器文件是如何读取的？

https://segmentfault.com/a/1190000019130222



# Mybatis 查询结果Result map 和list

Mybatis

一条记录 返回是map

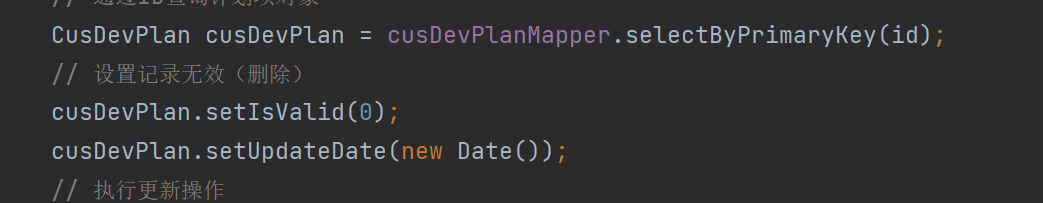
多条记录 list<map>

Mapper返回类型 以mapper 接口的方法的返回类型为主

Xml 文件中的resultMap 是指单行记录中属性值vo 包中的对象属性一致

mapper的方法可以直接返回对象





Mybatis

一条记录 返回是map

多条记录 list<map>

映射结果集 就是 list<java bean>

Mapper返回类型 以mapper 接口的方法的返回类型为主

Xml 文件中的resultMap 是指单行记录中属性值vo 类中的object属性一致

# Pojo类别名



# FreeMybatisplugin x

# 增删改查

Insert 返回的是一个int 类型 1 插入成功 0 插入失败

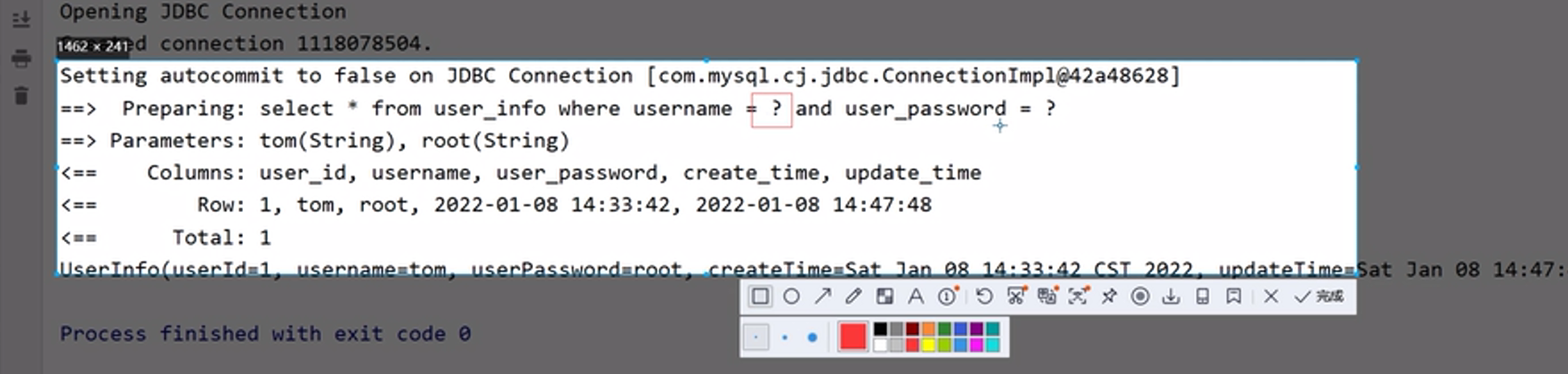
只有传入 传出 类 才需要指定类型

Mybati 多个参数 需要用注解

7

#和$的区别

#编译成问号



$直接编译成