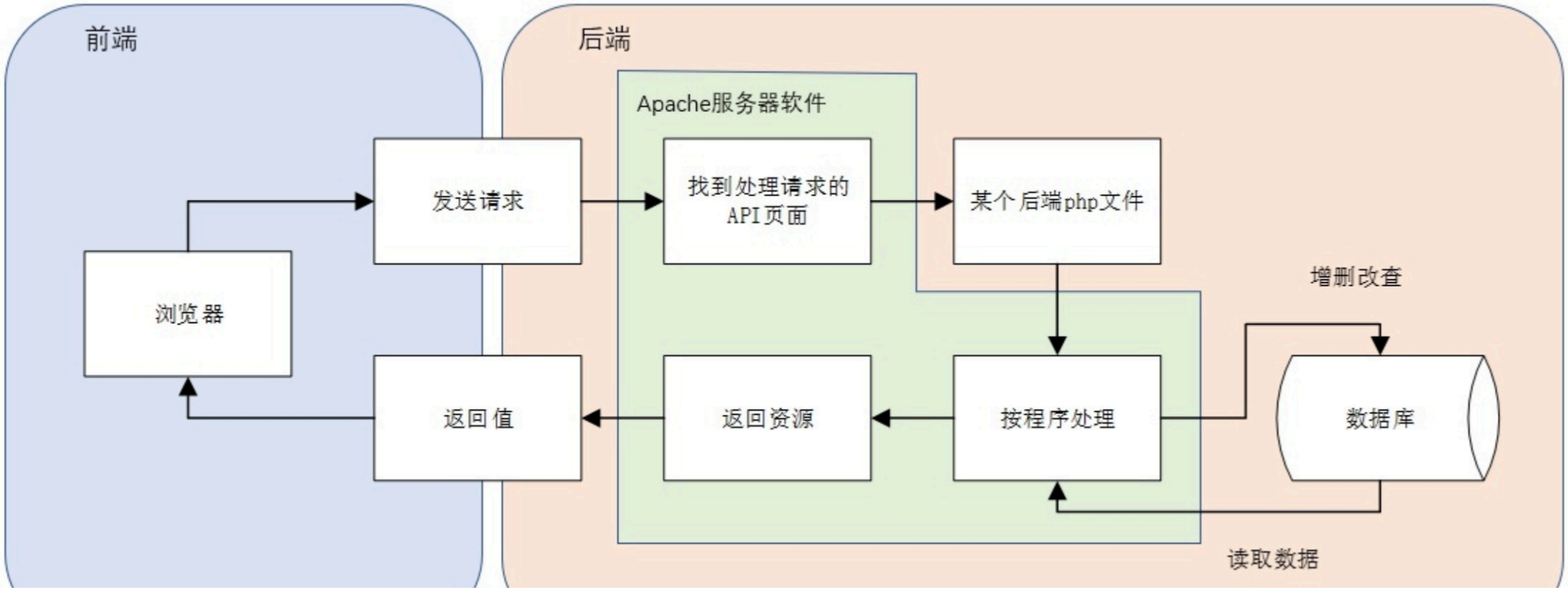
SpringBoot入门

张钊铭 博客 515code.com

先抛开SpringBoot,什么是前端和后端



什么是SpringBoot

Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Spring Boot致力于在蓬勃发展的快速应用开发领域(rapid application development)成为领导者。

“约定大于配置” 、让Spring应用开发变得更简单

传统Spring应用开发流程

- 1 配置环境
- 2 创建工程
- 3 构建目录结构
- 4 设置组件参数
- 5 配置Web容器
- 6 组件依赖管理
- 7 业务开发
- 8 测试与构建
- 9 手动部署
- 10 运维与监控

SpringBoot应用开发流程

- 1 配置环境
- 2 Spring Initializr
- 3 配置参数(可选)
- 4 自动部署
- 5 自动构建
- 6 业务开发
- 7 运维与监控

SpringBoot特点

- 极低的学习成本
- 化繁为简，简化配置
- 与分布式架构和云计算天然集成
- 微服务的入门级微框架

构建一个简单的SpringBoot项目

- 你需要用到以下知识/工具

- 1.Java基础知识
- 2.Maven项目管理工具
- 3.JDK1.8　/　Intelliji IDEA Ultimate (旗舰版)
- 4.Spring注解

SpringBoot目录结构

- /src/main
  项目根目录
  /java
  Java源代码目录
  /resources
  资源目录
  /resources/static
  静态资源目录
  /resources/templates
  表示层页面目录
  /resources/application.properties
  Spring Boot配置文件
  /test
  测试文件目录

SpringBoot中有哪些约定呢？

- 1.Maven的目录结构。默认有resources文件夹,存放资源配置文件。 src-main-resources, src-main-java。默认的编译生成的类都在target文件夹下面

- 2.spring boot默认的配置文件必须是，也只能是application.命名的yml文件或者properties文件，且唯一

- 3.application.yml中默认属性。数据库连接信息必须是以spring: datasource: 为前缀；多环境配置。该属性可以根据运行环境自动读取不同的配置文件；端口号、请求路径等

SpringBoot入口类

- 入口类命名通常以*Application结尾。
- 入口类上增加@SpringBootApplication注解
- 利用SpringApplication.run()方法启动应用

SpringBoot的常用配置

配置名称	默认值	描述
server.port	80	端口号
server.servlet.context-path	/	设置应用上下文
logging.file	无	日志文件输出路径
logging.level	info	最低日志输出级别
debug	false	开启/关闭调试模式
spring.datasource.*		与数据库相关的设置
...		...

SpringBoot注解简介

- 什么是注解？有哪些常用注解？

- 从JDK5开始,Java增加对元数据的支持，也就是注解，注解与注释是有一定区别的，可以把注解理解为代码里的特殊标记，这些标记可以在编译，类加载，运行时被读取，并执行相应的处理。通过注解开发人员可以在不改变原有代码和逻辑的情况下在源代码中嵌入补充信息。

```
@Override
public boolean supportsParameter(
    System.out.println("---------
```

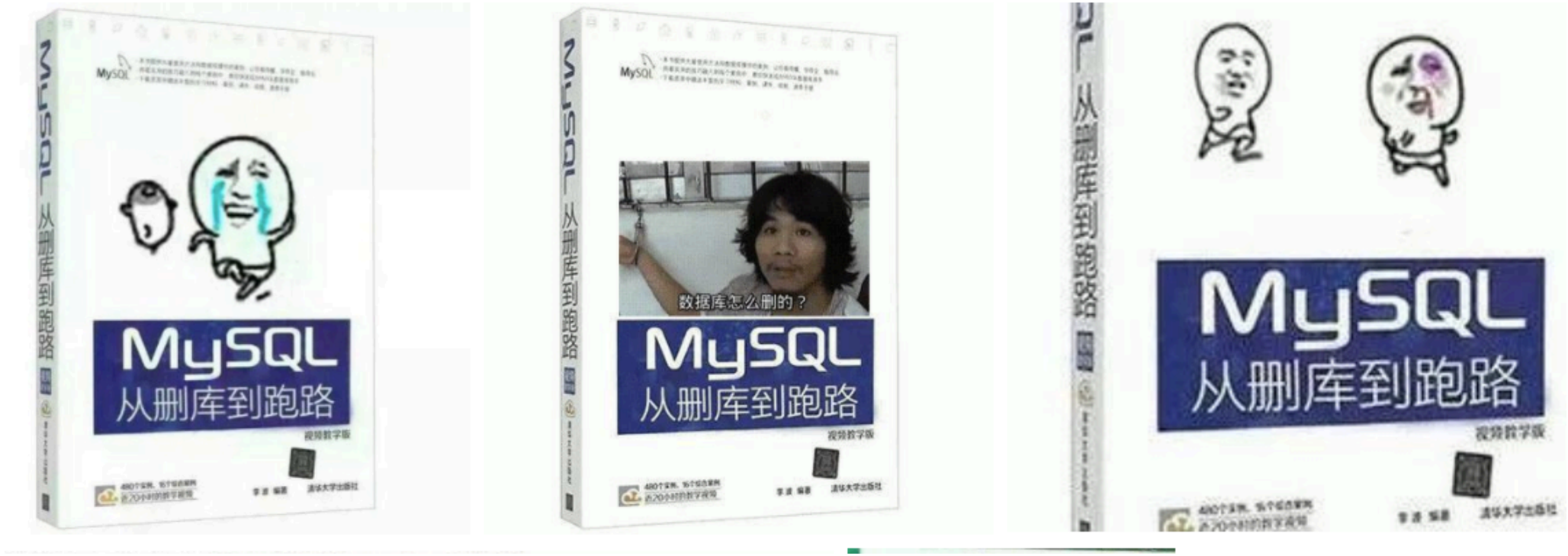
SpringBoot启动流程

- 第一步：加载配置文件 application.properties
- 第二步：自动装配

ArtifactId	描述
spring-boot-starter-web	增加Web支持
spring-boot-starter-data-jpa	对JPA支持，集成Hibernate
spring-boot-starter-logging	增加logback日志的支持
spring-boot-starter-test	集成JUnit单元测试框架

- 第三步：加载组件 @Repository @Service @Controller @Entity …
- 第四步：应用初始化

数据库操作



什么是数据库？

- 数据库是“按照数据结构来组织、存储和管理数据的仓库”。是一个长期存储在计算机内的、有组织的、可共享的、统一管理的大量数据的集合。
- 数据库是以一定方式储存在一起、能与多个用户共享、具有尽可能小的冗余度、与应用程序彼此独立的数据集合，可视为电子化的文件柜——存储电子文件的处所，用户可以对文件中的数据进行新增、查询、更新、删除等操作。

MySQL简介

- MySQL是一个关系型数据库管理系统
- 关系数据库将数据保存在不同的表中，而不是将所有数据放在一个大仓库内，这样就增加了速度并提高了灵活性。
- MySQL所使用的 SQL 语言是用于访问数据库的最常用标准化语言。
- MySQL 软件采用了双授权政策，分为社区版和商业版，由于其体积小、速度快、总体拥有成本低，尤其是开放这一特点，一般中小型网站的开发都选择 MySQL 作为网站数据库。

MySQL的简单使用

- MySQL常用指令　https://www.515code.com/posts/fm1u9eli/
- MySQL时间戳　https://www.515code.com/posts/sz04t8aa/

MySQL语句规范

- 关键字与函数名全部大写
- 数据库名、表名、列名全部小写
- SQL语句必须以分号结尾

AUTO_INCREMENT

- 自动编号
- 默认起始值为1，增量为1

PRIMARY KEY

- 主键约束
- 每张表只能有一个主键
- 记录唯一
- 自动NOT NULL

UNIQUE KEY

- 唯一约束保证记录唯一
- 字段可以为空
- 一张表可存在多个唯一约束

SpringBoot的数据库操作

- 使用Spring-Data-Jpa
- JPA(Java Persistence API)定义了一系列对象持久化标准
- 下面来设计一个RESTful API

请求方式	路径	功能
GET	/515code/students	获取所有学生信息
POST	/515code/students	创建一条学生信息
GET	/515code/students/id	通过学号查询学生
PUT	/515code/students/id	通过学号更新学生

事务

- 数据库事务指的是单个逻辑工作单元执行一系列的操作，要么完全执行，要么完全不执行。

- @Transactional注解

小结

- Maven的使用方法
- SpringBoot的配置与部分注解使用
- MySQL的数据库、表常用操作等
- MySQL的自动增长、主键约束、唯一约束、默认约束
- SpringBoot连接数据库和操作
- Restful API / 数据库事务

课后小作业

- 复习SpringBoot的配置与注解的使用
- 复习MySQL的常用sql语句

- 设计并创建一个数据库、表，包含以下字段，
  并自行编写一套Restful API接口进行测试

添加志愿选项：包括收集学生姓名、学号、专业、班级、联系方式、志愿一、志愿二、申请理由。

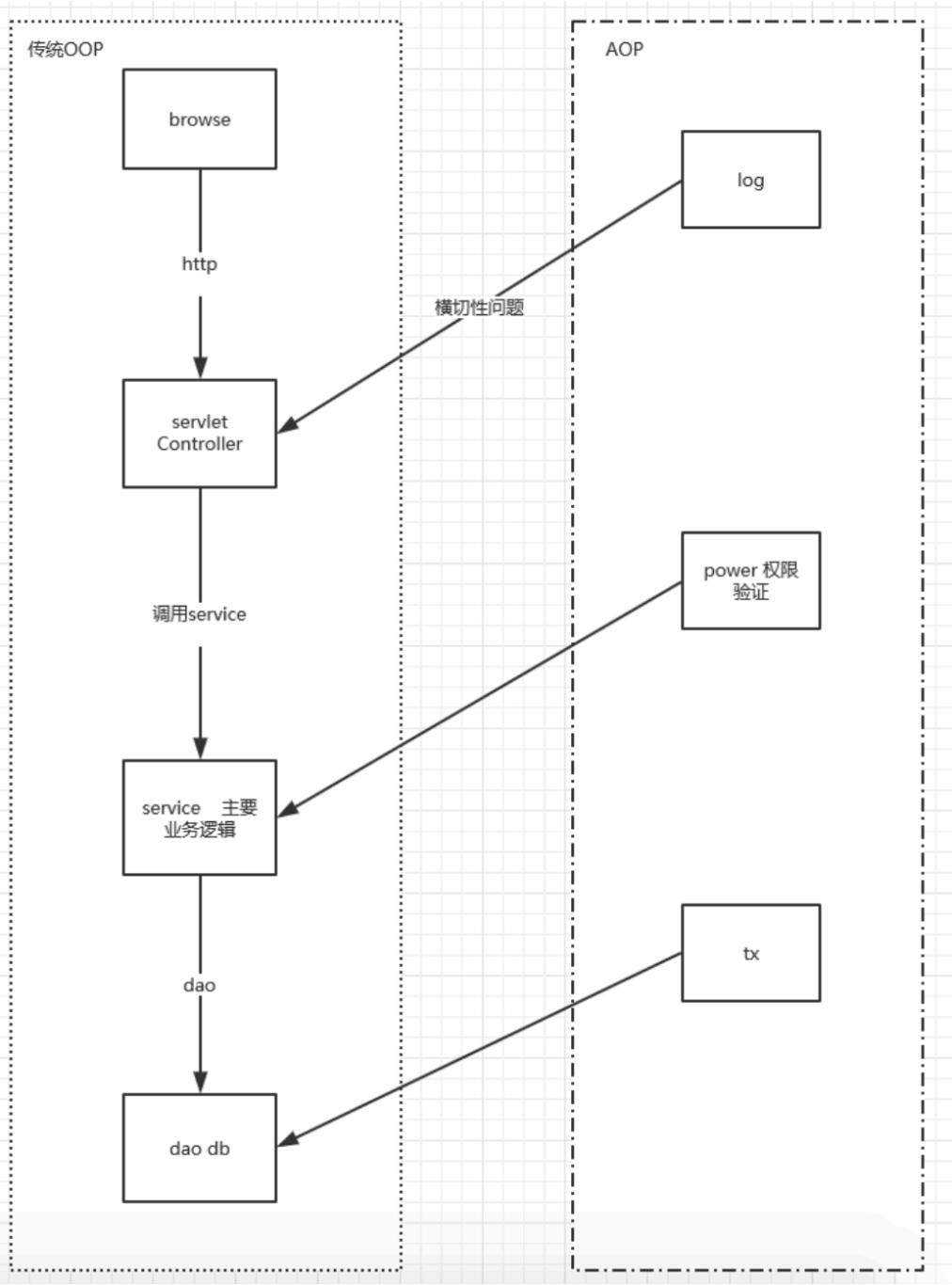
单元测试

- 单元测试（unit testing），是指对软件中的最小可测试单元进行检查和验证。对于单元测试中单元的含义，一般来说，要根据实际情况去判定其具体含义，如C语言中单元指一个函数，Java里单元指一个类，图形化的软件中可以指一个窗口或一个菜单等。

- <!--测试工具-->
  <dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <scope>test</scope>
  </dependency>

Spring AOP 面向切面编程

- AOP（Aspect Oriented Programming）称为面向切面编程，在程序开发中主要用来解决一些系统层面上的问题，比如日志，事务，权限等待
- 它是一種編程範式，不是編程語言



为什么使用AOP

- 解决分离问题
  - 水平分离：展示层->服务层->持久层
  - 垂直分离：模块划分（如订单、库存）
  - 切面分离：分离功能性需求和非功能性需求
- 好处
  - 集中处理某一关注点
  - 增强代码可维护性

- 应用场景列举
  - 权限控制
  - 缓存控制
  - 审计日志
  - 性能监控
  - 异常处理
  - 分布式追踪…

AOP术语

- 1.target：目标类，即需要被代理的类。例如：UserService
- 2.Joinpoint(连接点):所谓连接点是指那些可能被拦截到的方法。例如：所有的方法
- 3.PointCut 切入点：已经被增强的连接点。例如：addUser()
- 4.advice 通知/增强，增强代码。例如：after、before
- 5.Weaving(织入):是指把增强advice应用到目标对象target来创建新的代理对象proxy的过程.
- 6.proxy 代理类
- 7.Aspect(切面): 是切入点pointcut和通知advice的结合

常见通知 五种Advice

前置通知:在我们执行目标方法之前运行(@Before)

后置通知:在我们目标方法运行结束之后 ,不管有没有异常(@After)

返回通知:在我们的目标方法正常返回值后运行(@AfterReturning)

异常通知:在我们的目标方法出现异常后运行(@AfterThrowing)

环绕通知:动态代理, 需要手动执行joinPoint.procced()(其实就是执行我们的目标方法执行之前相当于前置通知, 执行之后就相当于我们后置通知(@Around)

https://www.515code.com/posts/egs95cdu/

开始AOP实践

- <dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-aop</artifactId>
  </dependency>

声明注解进行拦截

- @annotation方式，具体可以查看lesson4源码CheckUserAspect类

用切面表达式进行拦截

- 切面表达式理解及用法

https://www.515code.com/posts/egs95cdu/#%E5%9B%9Bspringaop%E4%BD%BF%E7%94%A8%E8%AF%A6%E8%A7%A3

Java日志

- 日志：保存描述系统运行状态的信息
- 日志框架：能实现日志输出的工具包
- 日志框架的功能：定制输出目标、格式、选择性输出
- 常见的日志框架：
  - 日志门面：是日志实现的抽象层。如JCL 、SLF4j
  - 日志实现：具体的日志功能的实现。如JUL、log4j、log4j2、logback

- https://www.515code.com/posts/khsh8lxz/

Java日志

- slf4j是Java的一个日志门面，实现了日志框架一些通用的api;
- logback是具体的日志框架。它和log4j是同一个作者，他是为了解决log4j存在的问题而开发的新的日志框架。
- slf4j和logback可以简单的看作jdbc和其具体数据库的JDBC的jar包的关系。
- 推荐使用slf4j，而不是直接使用logback。

数据库连接补充

- @Entity:每个持久化POJO类都是一个实体Bean,用于映射数据库表.
  @Table:生声明此对象映射到数据库的数据表,该注释不是必须的,如果没有,系统会使用默认值(实体类的短名).
  @Id:用于指定表的主键.
  @GeneratedValue:默认使用主键生成方式为自增，hibernate会自动生成一个名为HIBERNATE_SEQUENCE的序列.
  @Column:用来映射属性名和字段名，没有该注解的时候,hibernate会自动根据属性名字生成数据表的字段名。如属性name映射生成字段NAME；多字母属性如usertName会自动映射为USER_NAME。

Hibernate命名策略

- 无修改命名
  - spring.jpa.hibernate.naming.physical-strategy=org.hibernate.boot.model.naming.PhysicalNamingStrategyStandardImpl
- 遇到大写字母 加”_”的命名
  - spring.jpa.hibernate.naming.physical-strategy=org.springframework.boot.orm.jpa.hibernate.SpringPhysicalNamingStrategy

自定义SQL语句

- https://www.515code.com/posts/pjqr0a78/#query%E6%B3%A8%E8%A7%A3%E4%BD%BF%E7%94%A8

- 不建议在jpa中自定义SQL语句

小结

- AOP 面向切面编程
- SpringBoot中日志的使用
- Java 快速生成get set方法注解 @Data
- 数据库补充 – Hibernate命名策略与自定义SQL

- 课后作业：了解API文档，编写一份API文档并进行项目开发（最好能和前端一起配合）