1. 规约
2. 命名不以下划线开始结束 不用中英文混输不用拼音（先想好一个专业的功能名，用翻译软件）， 大小驼峰 常量大写，下划分隔

抽象类用abstract或者base开头，异常类用exception结尾

数组 类型和括号一起写 int[] 无空格

布尔类型变量不用is

不用不规范的缩写

用了设计模式就标出来 类名中 使用工厂 代理 监听者等

接口 属性和方法不加修饰符 public

借口和实现类：服务 服务实现 Service ServiceImpl 能力接口 -able

枚举就是常量类 成员大写 下划线

Service/DAO层：获取单个对象get 获取多个对象list 获取统计值count 插入save insert删除 remove delete- 修改update

领域模型“ 数据对象 表名+DO 数据传输对象 业务+DTO 展示对象 网页+VO

常量，不能出现魔法值（未赋给变量的直接数） 常量应该归类 跨应用共享 工程内共享包内共享 类内共享

Static final

Long 类型 用大写L结尾

1. 格式

空括号可不换

{} 需要分成单独行

if for () 括号之间空格

运算符左右空格

强制四个空格缩进

参数间空格

单行120

语句逻辑块之间空行

1. OOP

使用静态变量不用创建对象

重写父类方法必须加@Override

尽量不用可变参数

对外的接口方法不允许改名字，废弃时加@deprecated说明

Equals（）容易空指针异常，需要用常量的equals 或者确定有值 “a”.equals()

相同类型包装类，用equals比较

局部变量用基本类型 pojo类属性必须用包装类型 rpc方法返回值和参数必须用包装类

型

Do、dto/vo 等pojo不设定属性默认值

serialVersionUID

构造方法中不写业务逻辑，写在init()方法中

POJO类必须有toString 异常时，toString打印属性值

索引访问 split数组 检查最后一个分隔符 arr.length

同名方法 和多个构造方法放在一起

方法定义： 公有——保护——私有——getter,setter get set中不加业务逻辑（条件判断）

多声明final

Object.clone是浅拷贝，少用

控制： 工具类，工具方法应该私有化 单例模式禁止new构造方法private

常量final 禁止继承属性，方法用private 可以继承用protected 但不能外部访问

1. 集合 hasCode同时重写equals

集合转数组 toArray(arr) 大小list.size

数组转集合 Arrays.asList() 适配器 没有集合方法

Foreach中不能remove add 需要删除，使用迭代器

用Set去重

1. 并发 单例对象必须是线程安全 单例对象的方法也应该线程安全

线程名字需要有意义

用线程池提供，不显式创建

ThreadPoolExecutor去创建

SimpleDateFormat 是线程不安全的类 使用DateUtils

锁消耗性能，能锁区块就不锁整个方法 能锁对象就不锁类

两个线程对 多个资源 表 对象加锁 加锁顺序一致 不然死锁

并发修改同一数据 在应用层或者缓存加锁 或者在数据库用乐观锁

Volatile解决多线程内存不可见 一写多读 多写无法保证线程安全

1. 控制语句

Switch 必须有default 即使空 用break或者return终止，如果不终止就需要注释

If else 必须用大括号，不用单行语句

少用else 用if 取代else if else超过三层用状态模式（多态）

不要用复杂的条件逻辑 赋值给一个变量 或者一个函数

循环中移除复杂操作到外面

必须进行参数校验： 调用少的 ，必须稳定的 ，时间开销大的，敏感权限函数，对外开放的函数,api,http,rpc

Private函数在确认参数没问题时可以不校验

底层方法一般不较验 如dao 在之前就过滤掉

循环使用的函数最好不要在内部检查参数，在外部就行了

1. 注释 类 类属性 类方法用javadoc规范注释

抽象 接口方法也用javadoc

类创建者

方法内部用单行注释

或者多行

代码和注释同步修改

注释掉的代码加上说明

注释表示设计思想代码逻辑 业务含义

命名自注释

1. 其他

获取当前毫秒 system.currentimemills() 不用getTime()

1. 异常

稳定代码不需要try catch之后必须处理

对于对象需要进行npe判断： 进行null判断，避免空指针， 所有调用方法返回对象的时候都应该判断 ：返回包装类型 查询数据库null 集合本身和集合内的元素 session中的数据检查 级联调用，可能有空对象

Finially中关闭资源对象，流对象

加入栈信息

抛出异常还是错误代码

多使用业界定义的业务自定义异常，不要用通用异常

不应该复制粘贴代码 ，而应该复用

九、日志

十、表： 是否 is\_xxx 表名，字段名 不用数字开头，不用复数，不\_3\_

不许用sql保留字

唯一索引 uk\_字段名 普通索引 idx\_字段 unique key index

小数类型用decimal 不许用float double

存储的字符串长度近似（差距不算大） 使用定长char

Vchar 长度超过5000 应该用text

必须有 id gmt\_create gmt\_modified 创建时间 修改时间 unsigned bigint 无符号大整数自增步长1主键 datetime

表命名 ：业务名称\_表作用

库名=应用名

字段可以冗余 ，但不能是text 和总是修改的字段

>500万行或者表超过2g 才分库分表

字符长度 应该尽可能小，能用tiny int 就用 small int big int

对于查询建立索引：

如果字段是唯一的，不重复，就建成唯一索引

Vchar索引需要指定长度

超过三个表禁止联查

写页面搜索逻辑，禁止左模糊或者全模糊，索引用不了

Order by 的时候，利用已排 索引的顺序

优化达到range（范围检索） ref（普通索引） consts(id或者是唯一索引)

组合索引 区分度高的放左边

SQL

Count(\*) 统计所有行 conut(列) 不统计为null的行数

Count(distinct 列 ) 统计为null的行数

Sum(列) 如果全为null返回null

用ISNULL（） 判断某项是否null null与任何值的比较都是null，不能用判断句

分页查询逻辑：count为0直接返回

不允许使用外键和级联更新 修改一个字段，同时修改另一个表的外键

禁止存储过程

删除和修改之前先select 确认无误才更新

避免in操作 减少In 的元素个数

表查询不用\*

Pojo的布尔属性不用加is

Hashmap.put 不允许直接输出hashmap和hashtable的结果集

更新表记录的同时更新 修改时间

修改，查询的时候，不建议一次修改过多字段

使用事务考虑回滚方案 缓存回滚、搜索引擎回滚、消息补偿、统计修正

工程规约 分层 上层依赖下层

开放接口：rpc接口 或者web的http接口

显示层 velocity js jsp等

Web层 ：转发访问，校验参数

Service:业务逻辑