**三方件使用**

**类型转换：**用MapStruct(getter/setter)进行对象属性复制; beanUtils使用反射性能差，会OOM

**Json转换：**用JackSon;FastJson经常出现bug,不允许使用

**不考虑线程安全的场景中**，用StringBuilder,不用StringBuffer。

**集合判空**：用CollectionUtils.isEmpty(xxx)

**数组判空**：ArraysUtils.isEmpty()

**字符串判空**：StringUtils.isEmpty()，StringUtils.isBlank()

**返回单个元素的列表**：用Collections.singletonList()长度不可变，减少内存，不用Arrays.asList()

**返回空集合**：用Collections.emptyXXX，避免不必要的对象创建，每次返回都是同个空列表；不用new ArrayList

**Replace匹配**：用StringUtils的replace(无正则)；string中的replace,replaceAll,split用了正则，每次调用都会预编译，性能慢(用Pattern.compile避免)

**优先使用Apache Common,Guava工具类**

**日期时间用JDK8中的LocalDate、LocalTime、LocalDateTime，**不用Date,Calendar，SimpleDateFormat(比较繁琐)

**规约**

接口返回用return ApiResponse.success()； 无返回的返回参数写ApiResponse<Void>

基本数据类型都要用包装类

update/delete要返回变更/删除的条目数

add/create/save返回保存的对象(必须带主键ID)

使用全局唯一线程池，避免重复创建线程池

核心线程数：CPU密集型(核数+1) 、 IO密集型(核数\*2)

日志打印时，通过占位符拼接字符串

线程池中线程使用完ThreadLocal后要手动清除threadlocal中的内容

必须有无参构造函数(序列化需要)

选择合适的日志等级，避免日志泛滥

debug:调试定位，耗时，关键参数(id)，入参，返回值

info:关键逻辑/执行节点，定时任务，消息处理，方法出入口

warn:非预期情况

error:普通错误

fatal:致命错误

Redis中的缓存数据要有过期时间,避免造成Redis内存溢出

工具类中的都得是静态方法

数据库事务要最小化

金额计算要用BigDecimal

BigDecimal必须显式指定保留位数和舍入方法，用compareTo进行比较

Future.get()必须指定超时时间，避免无限阻塞主线程

Future task =CompletableFuture.supplyAsync(()->return getPrime(10000),asyncLineExecutor);

long primeCount =task.get(60,TimeUnit.SECONDS)

**禁止**

**禁止JSON序列化/反序列化**，大量反射影响性能(JSON.toJSONString,JSON.parseArray)

**禁止实现Serializable接口**

**禁止循环调用**数据库操作/外部接口/日志打印

**禁打印异常堆栈用**System.out/e.printStackTrace，要写成LOGGER.error("xx",e);

**禁止循环中打印**info以上级别日志

**禁止日志中打印**对象Json序列化后转成string

**禁止代码中出现密码/口令**

**坏味道，建议不要有**

不修改入参(情节依赖)

方法入参不超过5个

**命名规范**

返回布尔类型，方法命名为is/hasXXX

填充属性，方法命名为：populateXXX

包名小写，且只有一个单词(个别场景允许两个单词)

get/query/find都是一个意思，统一用get

接口层，应用层，领域层命名为getX,modifyX,addX,saveX,removeX，getXXXByQuery()

DAO层方法命名为insertXXX,deleteXXX,selectXXX

枚举名全大写，单词间下划线隔开

缩写全用大写：DAO,DTO,VO

**DDD领域驱动模型**

dto/vo不能继承entity

**Sql规范**

批量插入/更新要使用batchAdd/batchUpdate

关联对象<=3

子查询<=1

in字句<=100

union<=2

数据库事务最小化，禁止事务中调用外围服务，如事务内发mq(回滚后，mq不会撤回，要将方法分成两个)

调用自身方法时，禁止绕过Transactional proxy,用self注入自身来规避

禁止xml做一刀切事务，用@Transactional定义事务范围

为保证ACID,所有修改操作都要加事务注解

Select查询禁止返回大对象(<=1000)，需要分页，limit兜底

**单元测试**

类命名：被测试类名Test

方法命名：test{methodName}\_{场景}

方法上加@DisplayName

必须有断言，assertEqual,assertBoolean

用assert/verify进行断言

独立可执行，一个方法只测试一个场景

**DDD架构**

**Interface(接口层)**:出参为包装类型，只做出入参定义和校验，mq接收，DTO转entity,query,vo

**Application(核心层-应用服务层)**:定义主要任务，编排领域层来处理，多聚合，不允许有DTO

**Domain(核心层-领域层)**:主要业务功能，单聚合，不允许有DTO

**Infrastructure(基础设施层)**:为应用层传递消息，为核心层提供持久化，调用外部接口Delegate,XXXRepository的具体实现，dao层实现

**Common层**(util公共组件):可以被任何其他层使用

**maven约束依赖关系**(避免引用到错误层级的类，如接口层调用domain层),

interface层只允许依赖applicatoin层

Application只依赖domain,不直接依赖infrastructure层，形成依赖倒置(application,domain只定义XXXRepository,具体实现放在infrastructure层)

domain不直接依赖infrastructure层，形成依赖倒置

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| mave子模块 | Package命名 | 包含内容 | 类命名 | 入参 |
| start启动类 | Start | 启动类 | XXstract |  |
| Interface  (接口层) | interface | converter  dto  api  impl  consumer(mq接收方接口) | XXDTOToEntityConverter  XXEntityToDTOConverter  XXQueryDTOToQueryConverter  XXAPI  XXAPIImpl | httpRequest  queryDTO  DTO |
| Application  (应用层) | application | api  impl | XXAppService  XXAppServiceImpl | entity  vo  query |
| Domain  (核心层) | domain | constants  converter(entity-vo)  entity  query  repository(接口)  service(api,impl)  vo  producer(mq发送方接口)  validator(校验类)  delegate(调用外部服务的接口)  util | XXConstans  XX  XXQuery  XXRepo  XXService  XXVO  XXServiceImpl  XXValidator  XXDelegate  utils |  |
| Infrustructure  (基础设施层) | infrustructure | dao  repository的实现类  cache  producerImpl(mq发送方接口实现类)  delegate(对外调用的服务接口实现类) | XXDAO  XXRepoImpl  DataSourceConfiguration  XXDelegateImpl |  |