Java 开发规范

开发规范

此规范完全参照阿里的开发手册，以下只是罗列出来的重点，**并不是全部**。所以规范还是需要大家阅读文末附上的PDF。

强烈建议大家细读PDF文件，很多常见的JDK的坑都有规避手段，避免一些常见的低级错误。

此外，文末也给出了IDE的相关插件，有助于在开发过程中规避问题。此规范部分规则也有sonar的集成，后续也会推动QA把大家code纳入扫描范围。

命名风格

- 类名使用UpperCamelCase风格，但以下情形例外：DO / BO / DTO / VO / AO / PO / UID等
- 方法名、参数名、成员变量、局部变量都统一使用lowerCamelCase风格，必须遵从驼峰形式。
- 代码中的命名均不能以下划线或美元符号开始，也不能以下划线或美元符号结束。
- 常量命名全部大写，单词间用下划线隔开，力求语义表达完整清楚，不要嫌名字长。
- 抽象类命名使用Abstract或Base开头；异常类命名使用Exception结尾；测试类命名以它要测试的类的名称开始，以Test结尾。
- POJO类中布尔类型的变量，都不要加is前缀，否则部分框架解析会引起序列化错误。
- 杜绝完全不规范的缩写，避免望文不知义。
- 对于Service和DAO类，基于SOA的理念，暴露出来的服务一定是接口，内部的实现类用Impl的后缀与接口区别。

代码格式

- 采用4个空格缩进，禁止使用tab字符
- 单行字符数限不超过 120 个，超出需要换行。
- IDE的text file encoding设置为UTF-8; IDE中文件的换行符使用Unix格式

编码

- 不允许任何魔法值（即未经预先定义的常量）直接出现在代码中。
- 避免通过一个类的对象引用访问此类的静态变量或静态方法，无谓增加编译器解析成本，直接用类名来访问即可。
- 所有的覆写方法，必须加@Override注解。
- 外部正在调用或者二方库依赖的接口，不允许修改方法签名，避免对接口调用方产生影响。接口过时必须加@Deprecated注解，并清晰地说明采用的新接口或者新服务是什么。
- 不能使用过时的类或方法。
- 定义DO/DTO/VO等POJO类时，不要设定任何属性默认值
- 序列化类新增属性时，请不要修改serialVersionUID字段，避免反序列失败；如果完全不兼容升级，避免反序列化混乱，那么请修改serialVersionUID值。
- 构造方法里面禁止加入任何业务逻辑，如果有初始化逻辑，请放在init方法中。
- POJO类必须写toString方法。
- 禁止在 POJO类中，同时存在对应属性 xxx的 isXxx()和 getXxx()方法。
- 因为Set存储的是不重复的对象，依据hashCode和equals进行判断，所以Set存储的对象必须重写这两个方法。自定义对象作为Map的键，同理也必须重写hashCode和equals。
- 避免出现重复的代码（Don’t Repeat Yourself）

异常

- Java 类库中定义的可以通过预检查方式规避的RuntimeException异常不应该通过catch 的方式来处理，比如：NullPointerException，IndexOutOfBoundsException等等。
- 异常不要用来做流程控制，条件控制。
- 捕获异常是为了处理它，不要捕获了却什么都不处理而抛弃之，如果不想处理它，请将该异常抛给它的调用者。最外层的业务使用者，必须处理异常，将其转化为用户可以理解的内容。
- finally块必须对资源对象、流对象进行关闭，有异常也要做try-catch。

- 方法的返回值可以为null，不强制返回空集合，或者空对象等，必须添加注释充分说明什么情况下会返回null值。

日志

- 应用中不可直接使用日志系统（Log4j、Logback）中的API，而应依赖使用日志框架SLF4J中的API，使用门面模式的日志框架，有利于维护和各个类的日志处理方式统一。
- 避免重复打印日志，引起混乱、浪费磁盘空间，务必在log4j.xml中设置additivity=false。
- 异常信息应该包括两类信息：案发现场信息和异常堆栈信息。如果不处理，那么通过关键字throws往上抛出。
- 谨慎地记录日志。生产环境禁止输出debug日志；有选择地输出info日志。
- 可以使用 warn日志级别来记录用户输入参数错误的情况，避免投诉时无所适 从。如非必要，请不在此场景打出 error级别，避免频繁报警。 级

注释

- 好的命名、代码结构是自解释的，注释力求精简准确、表达到位。避免出现注释的一个极端：过多过滥的注释，代码的逻辑一旦修改，修改注释是相当大的负担。
- 类、类属性、类方法的注释必须使用Javadoc规范，使用/**内容*/格式，不得使用// xxx方式。
- 所有的抽象方法（包括接口中的方法）必须要用Javadoc注释、除了返回值、参数、异常说明外，还必须指出该方法做什么事情，实现什么功能。
- 所有的类都必须添加创建者和创建日期。
- 方法内部单行注释，在被注释语句上方另起一行，使用//注释。方法内部多行注释使用/* */注释，注意与代码对齐。
- 所有的枚举类型字段必须要有注释，说明每个数据项的用途。

单元测试

- 单元测试应该是全自动执行的，并且非交互式的。测试用例通常是被定期执行的，执行过程必须完全自动化才有意义。输出结果需要人工检查的测试不是一个好的单元测试。
- 保持单元测试的独立性。为了保证单元测试稳定可靠且便于维护，单元测试用例之间决不能互相调用，也不能依赖执行的先后次序。
- 单元测试是可以重复执行的，不能受到外界环境的影响。
- 核心业务、核心应用、核心模块的增量代码确保单元测试通过。
- 对于不可测的代码建议做必要的重构，使代码变得可测。

安全

- 隶属于用户个人的页面或者功能必须进行权限控制校验。
- 用户敏感数据禁止直接展示，必须对展示数据进行脱敏。
- 用户输入的SQL参数严格使用参数绑定或者METADATA字段值限定，防止SQL注入，禁止字符串拼接SQL访问数据库。
- 用户请求传入的任何参数必须做有效性验证。
- 表单、AJAX提交必须执行CSRF安全验证。

工具

IDE插件

插件安装流程在Git主页写的非常清楚，大家自行按照步骤进行安装即可。

idea

https://github.com/alibaba/p3c/tree/master/idea-plugin

eclipse

https://github.com/alibaba/p3c/tree/master/eclipse-plugin

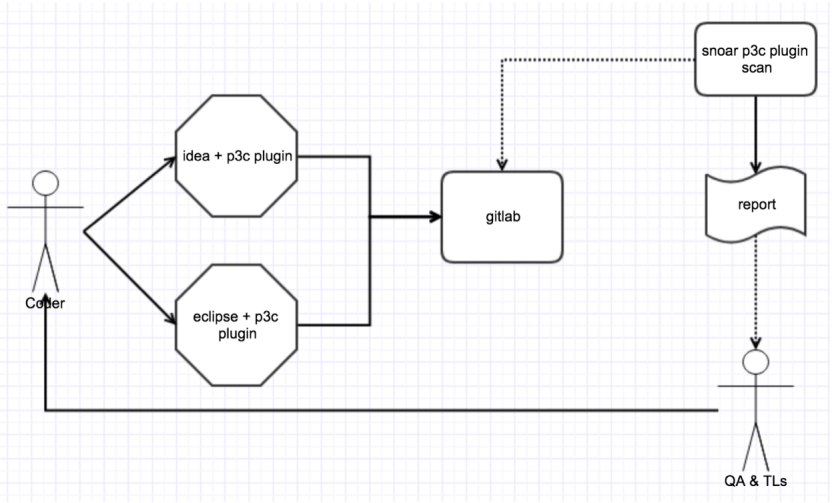
代码自动扫描

http://sonar-ci.envisioncn.com + https://github.com/mrprince/sonar-p3c-pmd

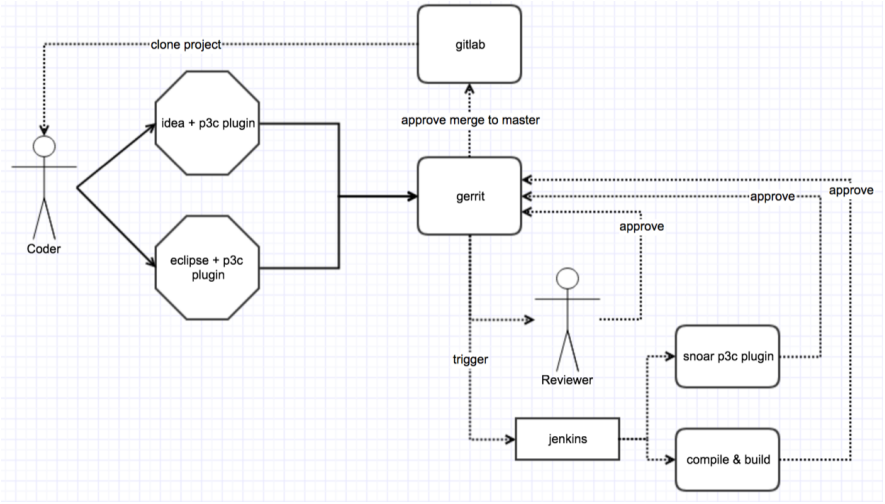
ref: https://www.jianshu.com/p/b849175dd38b

流程示意图

短期开展计划



长期开展计划


gitlab
clone project
idea + p3c plugin
approve merge to master
Coder
gerrit
approve
approve
approve
eclipse + p3c plugin
Reviewer
trigger
snoar p3c plugin
jenkins
compile & build