

代码审查指南

版本	修订内容	日期	修订人

目录

1	术语.....	3
2	预期的读者.....	3
3	编写目的.....	3
4	CodeReview 的目标.....	4
5	CodeReview 的前提.....	4
5.1	制定开发规范.....	4
5.2	完善设计文档.....	4
5.3	Checklist 达成共识.....	4
6	CodeReview 的步骤.....	5
6.1	Committer 选择相关人员作为 CL 的 Reviewer.....	5
6.2	Committer 选择合适的时机提交 CL.....	5
6.3	Reviewer 处理 CL 请求.....	5
6.4	Committer 根据 Reviewer 意见进行处理.....	5
7	完成 CodeReview 后的工作.....	5
7.1	阶段总结.....	5
7.2	整合存档.....	6
8	对 Reviewer 的要求.....	6
8.1	关注每一行.....	6
8.2	关注上下文.....	6
8.3	发现优秀的细节.....	6
8.4	通过 review 进行指导.....	6
8.5	不要过度追求完美.....	6
9	对 commiter 的要求.....	7
9.1	写好 CL 的描述.....	7
9.2	提交小而精的 CL.....	7
9.3	正确处理 reviewer 的评论.....	8
10	Checklist.....	8
10.1	设计.....	8
10.2	功能.....	8
10.3	复杂度.....	9
10.4	测试.....	9
10.5	命名.....	9
10.6	注释.....	9
10.7	风格.....	9
10.8	文档.....	9
11	icode 操作说明.....	10
11.1	开启 icode 的 CodeReview 流程.....	10
11.2	Committer 发起 CL (CR).....	10
11.3	Reviewer 进行 CodeReview.....	10

1 术语

术语	解释
CodeReview	代码审查
CL	Changelist 即“变更列表”，表示已提交到版本控制或正在进行代码审查的自包含更改，也叫 change 或 patch
Reviewer	有资格进行代码审查的人，负责进行 code review 工作的个人
Committer	提交 CL 的人，一般为需求开发人员
Checklist	检查列表，即 Reviewer 在 review 过程中需要检查的要点

2 预期的读者

本文档预期的读者为 Committer、Reviewer、及其干系人

3 编写目的

本文档编写目的如下：

- 1、 说明 CodeReview 的作用及如何进行 CodeReview;
- 2、 明确 Committer 和 Reviewer 职责;
- 3、 指导相关人员正确开展 CodeReview 工作

4 CodeReview 的目标

- 1、 提高代码质量，及早发现潜在缺陷，降低修改/弥补缺陷的成本
- 2、 促进团队内部知识共享，提高团队整体水平
- 3、 帮助更多的人理解系统
- 4、 传递知识，实现人员互备
- 5、 鼓励相互学习对方的长处和优点

5 CodeReview 的前提

5.1 制定开发规范

以《阿里巴巴集团 java 开发规范》为主体，制定代开发规范，包括异常处理、数据安全、权限控制、日志输出、熔断处理等。

5.2 完善设计文档

Comminter 应依据《设计文档规范》进行设计文档编写，在相应模块代码被进行评审时，reviewer 应参照设计文档进行检查，并对代码实现与设计方案实现进行核对。

5.3 Checklist 达成共识

6 CodeReview 的步骤

6.1 Committer 选择相关人员作为 CL 的 Reviewer

Reviewer 可从以下人员中选择：

- 1、 参与需求评审者的高级开发人员
- 2、 参与技术方案评审的高级开发人员
- 3、 与本次 CL 相关的开发人员

6.2 Committer 选择合适的时机提交 CL

Committer 每次完成单一功能后，依照[对 commiter 的要求](#)提交 CL。

6.3 Reviewer 处理 CL 请求

Reviewer 应依照[对 Reviewer 的要求](#)尽快响应 CL 请求，一般不应超过 6 小时，同时填写 CodeReview 跟踪文档。

6.4 Committer 根据 Reviewer 意见进行处理

根据 Reviewer 反馈的问题，如 CL 过大，复杂度过高等问题进行代码调整，并重复以上步骤，直至审核通过。

7 完成 CodeReview 后的工作

7.1 阶段总结

团队对每周进行的 CodeReview 进行回顾、总结分析并持续优化，对于有代

表性的问题，需在每周的分享会中进行分享。

7.2 整合存档

定期对团队的 review 记录进行整合并存档，并对持续存在的问题进行复盘分析。

8 对 Reviewer 的要求

8.1 关注每一行

8.1.1 需要关注每个类、函数和代码块，不能假设这些代码与其命名相匹配

8.1.2 若代码十分难以理解，需要线下沟通

8.1.3 对于审查者而言，如果对 review 代码没有信心，尤其是设计并发、安全等问题时，需要向上反馈

8.2 关注上下文

8.2.1 关注细节，充分考察局部代码变更的上下文

8.2.2 站在系统的角度上考虑变更对系统的影响

8.3 发现优秀的细节

8.3.1 不要吝啬赞美，对精巧的实现给予肯定

8.4 通过 review 进行指导

8.4.1 通过 review 尽量帮助开发人员掌握新的知识

8.5 不要过度追求完美

- 8.5.1 平衡变更的重要性与实际进展
- 8.5.2 是否通过 review 的标准是代码是否使代码库的整体质量降低
- 8.5.3 对不是很重要的评论增加“建议标识”

9 对 commiter 的要求

9.1 写好 CL 的描述

- 9.1.1 整个 CL 描述本次进行了哪些更改以及为何更改
- 9.1.2 首行描述本次 CL 是做什么的简短摘要，需要是完整的句子，一般以动词开头
- 9.1.3 正文部分需要简要描述正在解决的问题以及为什么要这样解决
- 9.1.4 正文部分需要描述当前解决方案的缺点或折中处理，同时附加相关背景信息的链接

9.2 提交小而精的 CL

- 9.2.1 reviewer 可以拒绝过大的 CL，所以 commiter 需要提交小而精的 CL
 - a) 删除文件视实际情况可不视为大 CL
 - b) 自动生成的代码原则上不视为大 CL
- 9.2.2 CL 大小的判断规则需要参考变更的扩散程度
 - a) 对于少量文件（小于 5 个文件）变更可以在每个文件内包含更多变更行
 - b) 超过 20 个文件的变更，尽管总行数较小，仍然不适合作为单次 CL 提交
- 9.2.3 一次 CL 应该只解决了一件事，一般是全部功能的一部分，而不是完整功能
 - a) 移动、重命名等操作需要与修复相关类的改动分离，即不将移动、重命名的操作与业务变更放在同一个 CL 中
- 9.2.4 如果 CL 只新增了 API 功能，但是没有外部调用场景，需要提交该 API 的具体使用方法

9.2.5 由业务代码变更而导致测试代码变更时, 应将二者放入同一个 CL 中, 但如果仅变更了测试代码 (如重构或新增测试用例) 则测试变更可以单独构成 CL

9.2.6 避免因为拆分小 CL 而影响系统构建

9.2.7 若 CL 经过慎重处理后仍不能达到合适的大小, 需要线下沟通

9.3 正确处理 reviewer 的评论

9.3.1 保持自信, 同时要相信 reviewer 的判断, 并谨慎的审视对方的意见

a) 相信 reviewer 是为了系统的整体考虑, 而不是针对具体的人

b) 技术处于初级水平时虚心听取意见

c) 技术处于高级时要从客观的角度考虑问题

9.3.2 面对质疑时保持冷静

9.3.3 减少在 CodeReview 系统中回复 reviewer 的评论, 用代码或注释进行回应, 因为代码文件本身会被更多人浏览

9.3.4 有则改之无则加勉, 认真对 reviewer 的评论, 考虑其是否正确, 并达成共识

10 Checklist

Checklist 指明了团队核心关注的要素, 作为进行 CodeReview 的审查依据, 并进行不断的更新。

Checklist 中的内容并不完能全覆盖开发规范中所有内容, 只会列出非常重要的检查点。

10.1 设计

10.1.1 代码的逻辑是否符合设计文档

10.2 功能

- 10.2.1 代码是否完全实现了设计文档中提出的功能需求
- 10.2.2 代码是否存在并发问题
- 10.2.3 代码是否正确的打印了日志
- 10.2.4 代码是否接入了熔断系统并正确的处理了相关逻辑

10.3复杂度

- 10.3.1 代码涉及的类或方法是否过于复杂（圈复杂度）
- 10.3.2 代码是否存在过度设计

10.4测试

- 10.4.1 代码是否已按照设计文档进行了自测和 Debug
- 10.4.2 代码是否已执行完成单元测试并执行通过

10.5命名

- 10.5.1 代码的函数、类、变量是否具备良好的规则，能够准确的表达其代表的含义

10.6注释

- 10.6.1 注释是否解释了注释目标为什么如此做而不是做了什么

10.7风格

- 10.7.1 代码中使用的格式、符号、结构等风格是否保持一致（采用 code style 插件）

10.8文档

- 10.8.1 代码变更后是否变更了相关文档

11 icode 操作说明

11.1开启 icode 的 CodeReview 流程



11.2Committer 发起 CL （CR）

发起CR

* 标题 54,1

概述

* 评审人 于飞 刘贺

一键添加

* 类型 ☐ 全量 ☒ 增量

	产品包	创建人	创建时间	信息
<input checked="" type="checkbox"/>	4.0.54.2	刘贺	2020-01-14 18:43:19	edbe99992ac8a510e3b64fb345f5ed3d9e64e950
<input checked="" type="checkbox"/>	4.0.54.1	刘贺	2020-01-14 18:36:14	4f5a9e753df93f08ee6c616468a2935cca62e3e0
<input type="checkbox"/>	4.0.54.0	任勇达	2020-01-14 17:19:50	0c42ba612876be8174c12512ecde65c3b5371e8a

取消 确定

11.3Reviewer 进行 CodeReview

