基于Git的工程规范

修订历史

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 时间 | 作者 | 描述 |
| 0.1 | 2020.07.21 | 郑宇真 | 为了更好的工程式的协作开发体验，故特此提交工程规范 |

本文主要为指导工程开发过程中的分支设置、代码合并规范。

# 1. 分支设置

对于工程开发阶段的仓库，原则上可以分为以下几种分支类型：

1. 用于发布稳定版的master分支

2. 用于汇总功能开发的develop分支

3. 用于快速开发某个具体的功能task分支

4. 用于修复当前已发布代码中的bug的hotfix分支

对于上述分支类型，应当遵循以下规则。

## 1.1 master分支

该分支的用于发布**稳定版**的源码，其命名应当为“master”。 该分支不可直接提交commit，且只能合并由develop分支或hotfix分支提出的合并请求（Pull Request, PR）。稳定版意味着：

* 每个PR之后的代码都是可以稳定运行的，相当于一个已成型的产品；
* 每个PR之后的代码明确对应着一个不同的版本号；
* 每个PR之后的代码利用一个版本标签标志其版本号；
* 对应生成的可执行文件需要挂在网站的“版本发布”页面下。

## 1.2 develop分支

该分支用于汇总各项功能开发，是单个功能的汇总，也是master版本的基础，其命名应当为“develop”。 该分支不可直接提交commit，仅能合并由task分支提出的PR，或hotfix分支提出的PR。

## 1.3 task分支

该分支是为了实现和修改新特性而建立的分支，命名以“task-”开头。可以具有较为灵活的命名，并且可以同时存在多个。原则上每个分支仅允许当前任务的开发人员进行提交。

分支的建立与删除。task分支以develop分支为基础新建，仅可合并到develop分支中。完成功能的开发和自测后应当及时提交到develop的合并申请。合并后删除。如果开发过程中发现该功能是一次失败的探索，可直接删除该分支。

## 1.4 hotfix分支

该分支用于修复已发布代码(develop/master)的bug，具有较高的优先级，命名应当以“hotfix-”开头并在分支名中体现分支修复的问题信息，可以同时存在多个。该分支应当以master或者develop分支为基础创建，且仅可合并入其创建的基础分支。合并后删除。

# 2 . 代码合并规范

工程人员的代码合并可分为三种：新功能添加、分支修复以及稳定版本发布。

## 2.1 新功能添加

新特性添加指从task分支提出针对develop分支的PR。开发人员应当遵循以下步骤：

1. 功能自测

2. 用例自测

3. 提交合并申请

4. 根据反馈结果完成或改进此次合并。

对于审核人，应当遵循以下步骤：

1. 收到PR后查看申请信息，主要审核内容为：功能自测的用例是否完善，提交修改文件与信息是否对应，若有必要同时检查代码是否合规。

2. 根据审核结果尽快在合并申请中给出合并意见，并对不合规的申请指出修改意见。

3. 审核人全部同意后方可合并入develop分支。

若开发人员提交合并申请后审核人长时间未给出回应，应主动联系审核人进行处理，同理审核人给出修改意见后，开发人员长时间未给出回应也应当主动联系。

对于微小功能或联系紧密的功能，为了更高效的进行开发工作，允许用单个分支同时实现少数多个功能的开发。

当添加新功能时发现工程其他功能出现问题时，应当联系对应的开发人员，要求其在develop分支上建立hotfix分支进行修复，待bug修复后再次拉取develop分支测试新特性，不可在自己的开发分支上直接修复其他功能的bug。

若当前无稳定测试用例集则仅进行功能测试。

## 2.2 分支修复

已完成bug修复的hotfix分支应当并入master以及develop分支。分支修复后应当尽快提出PR。对于开发人员应当遵循以下步骤：

1. 功能自测

2. 用例自测

3. 提交合并申请：提交合并申请需要制定一名或以上审核人。提交信息应当包括：a. bug产生的原因及修复方式（可指向描述该bug的issue）；b. 本次修复修改文件；c. 功能和用例的测试情况。

4. 根据反馈结果完成或改进此次合并。

审核人审核步骤参考上一节。

单个hotfix分支仅用于修复单个bug。为增加稳定性，禁止在同一分支引入不同代码修复多个bug（同一代码修改可解决多个bug的情况除外）。

若当前无稳定测试用例集则仅进行功能测试。

来自于master的hotfix分支，修复后应同时合并到develop分支

## 2.3 版本发布

版本发布分为开发版和稳定版。

* 若某个开发版经过足够的测试之后，仍然比较稳定，则可将基于该预开发针对master分支提交PR，在master分支合并该PR后，得到一个稳定版。每个稳定版对应一个稳定版本编号。