## Git使用规范

|  |  |  |  |
| --- | --- | --- | --- |
| **版本号** | **作者** | **更新日期** | **备注** |
| 1.0.0 | 徐良永 | 2018.5.11 |  |
| 1.1.0 | 徐良永 | 2018.5.16 |  |

# 第一章 GIT分支定义

1.【**master**】

master分支，这个分支是最近发布到生成环境的代码，最近发布的release。这个分支只能从其他分支合并，不能在这个分支直接修改。

2.【**develop**】

develop分支是我们的主开发分支，包含所有要发布到下一个 release的代码，develop分支主要合并其他分支，如feature分支

1. 【**feature**】

feature分支，主要用来开发一个新的功能，一旦开发完成合并 到develop 分支， 进入下一个release

1. 【**release**】

release分支，当你要发布一个新版本的时候，就创建一个release 分支， 在分支上做测试、修改bug。我们基于develop分支创建release分支，完 成 release（上线）后，合并到master分支和develop分支。

1. 【**hotfix**】

hotfix分支，当我们在生产环境发现bug时，需要创建一个hotfix，完成hotfix后，合并到master分支和develop分支

# GIT使用规范

## 一．Git 客户端

1. 【推荐】使用git base客户端

## master分支规约

1. 【强制】master分支用于发布版本，非常稳定，一般不接受提交，只能从 develop分支merge
2. 【推荐】所有在master分支上的commit应该是tag

## develop分支规约

1. 【强制】develop分支用于开发，用于接受提交。开发完一个版本，有项目负责人将develop分支merge到master分支
2. 【强制】禁止把自己的分支push到远程仓库
3. 【强制】每天开发前pull develop分支

## 本地分支和远程分支规约

1. 【强制】本地存在的分支：master， develop，自己的分支（例如Tom，Jim，feature1， bug1）。其中master和develop分支需要和远程同步。自己的分支不需要同步，在自己的分支开发完后，merge到develop，然后将本地develop提交到远程develop。
2. 【推荐】自己创建的本地分支，merge到develop分支后，若无冲突问题，立即删除

## Release分支规约

1. 【强制】release分支基于develop分支创建。创建后，可以在release分支 上测试、修复bug等
2. 【强制】决不能在release分支上开发新功能

3.【强制】发布release分支时，合并release到master和develop，同事在 master分支上打tag记住release版本号，然后删除release分支

## Feature分支规约

1. 【强制】feature分支做完后必须合并到develop分支
2. 【强制】若多人合作开发一个feature，要推送到远程仓库；若只一人，就留 在本地吧
3. 【推荐】feature分支合并到develop分支后一般会删除

## Hotfix分支规约

1. 【强制】hotfix分支基于master分支创建
2. 【强制】hotfix开发完成后需要合并到master分支和develop分支，同时在 master分支上打一个tag