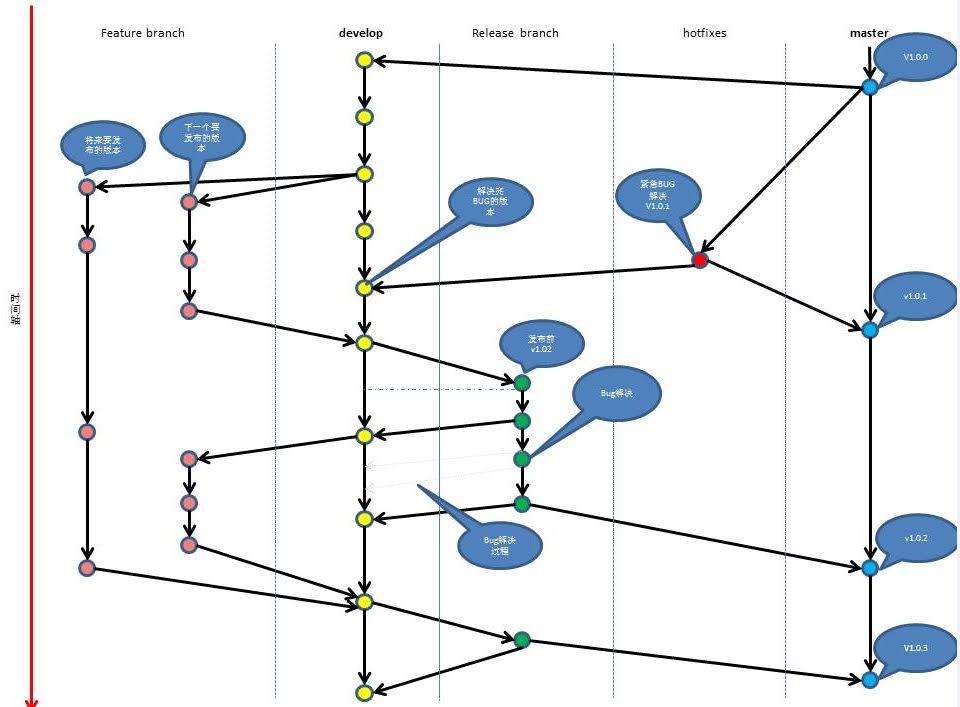
# 简介



项目版本号：

1.0.0 第一个数字代表改版与重构，第二个数字代表新的大功能需求更新，第三个数字代表小需求及BUG修复的版本

# 项目中长期存在的两个分支

master: 主分支，负责记录上线版本的迭代，该分支的代码与线上的代码完全一致。

develop：开发分支，该分支记录相对稳定的版本，所有的的feature分支与bugfix分支都从该分支创建。

# 其他分支为短期分支，其完成功能开发之后进行相应删除

* + feature版本号: 功能分支（如feature1.6.0）,用于开发新的功能，不同的功能版本开号创建不同的功能分支，功能分支开完完成并自测与测试通过后，需要合并到develop分支，之后删除此分支。
  + bufix版本号：bug修复分支（如bugfix1.6.1），用于修复不紧急BUG，普通BUG均需要创建bugfix分支开发，开发完成自测与测试测试没有问题后，合并到develop分支，之后删除此分支
  + release版本号:发布分支（如release1.6.0,release1.6.1），用于代码上线准备，该分支从develop分支创建，创建后由运维同学发布到测试环境进行测试，测试过程中发现BUG需要开发人员在该release分支进行BUG修复。所有BUG修复完成后，在上线前，需要合并该release分支到master分支和develop分支。
  + hotfix版本号:紧急bug修复分支，该分支只有在紧急情况使用,从master分支创建,用于紧急修复线上BUG。修复完成后，需要合并该分支到master分支以便上线，同时需要再合并到develop.

# 操作步骤

第一步

开发人员接到任务，

如果是功能开发，会以develop为基础创建feature1.6.0

如果是bug修复或小需求，会以develop为基础创建 bufix1.6.1

bugfix的版本永远小于feature的版本

基本操作命令：

git checkout -b <feature\*\*\*> <develop>

第二步

开发差不多了或者任务穿插时

git status

git add

git commit

第三步

提交几次后，感觉差不多了，让测试在本地开发环境进行测试。此处涉及多人合作，多人合作请将对应分支上传到服务器，测合并到当前需求主要负责开发人开发机上进行测试。

第四步

测试在本地测试通过了。就可以合并到develop了

git pull origin develop //先拉取，因为可能别人已经提交过代码

git checkout develop

git merge <feature\*\*\*> //合并到develop中

git push origin develop

git branch –d <feature\*\*\*>

第五步

运维人员介入在测试机上拉个realse待发布

git checkout -b <realse\*\*\*> develop

创建这个分支相当于测试环境修改，改好后就需要跟新master和develop，然后删除分支。

git checkout master

git merge release\*\*\*

git push origin master

git checkout develop

git merge release\*\*\*

git push origin develop

git branch –d release\*\*\*\* //删除本地

第六步

打开TAG追踪

git tag -a v\*\*\* -m 'xxxxxx'

git push --tags

运维对外发布到线上

第七步

线上环境发现紧急bug

git checkout -b hotfi\*\*\* master//从master分支上新建分支

改完BUG后

git checkout master

git merge hotfix\*\*\*

git push origin master

合并到develop

git checkout develop

git merge hotfix\*\*\*

git push origin develop

git branch -d hotfix\*\*\*