GitFlow工作流

git学习链接

分支说明

master分支

主分支，任何项目都必须有个这个分支，对项目进行正式发布版本的操作必须在该分支上进行。

develop分支

开发分支，从master分支上检出。团队成员一般不会直接更改该分支，而是分别从该分支检出自己的feature分支，开发完成后将feature分支上的改动merge回develop分支。

feature分支

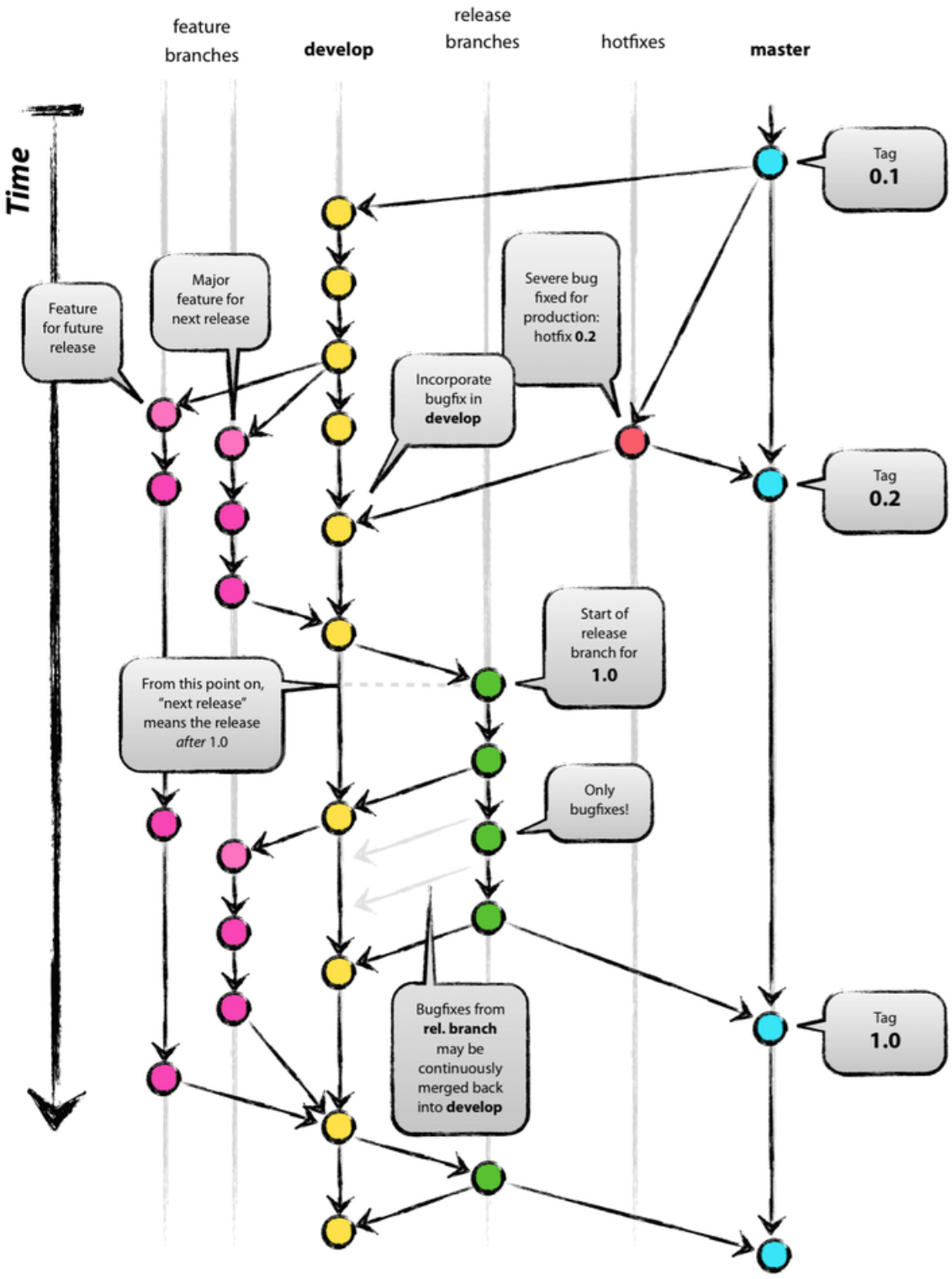
功能分支，从develop分支上检出。团队成员中每个人都维护一个自己的feature分支，并进行开发工作，开发完成后将此分支merge回develop分支。此分支一般用来开发新功能或进行项目维护等。

fix分支

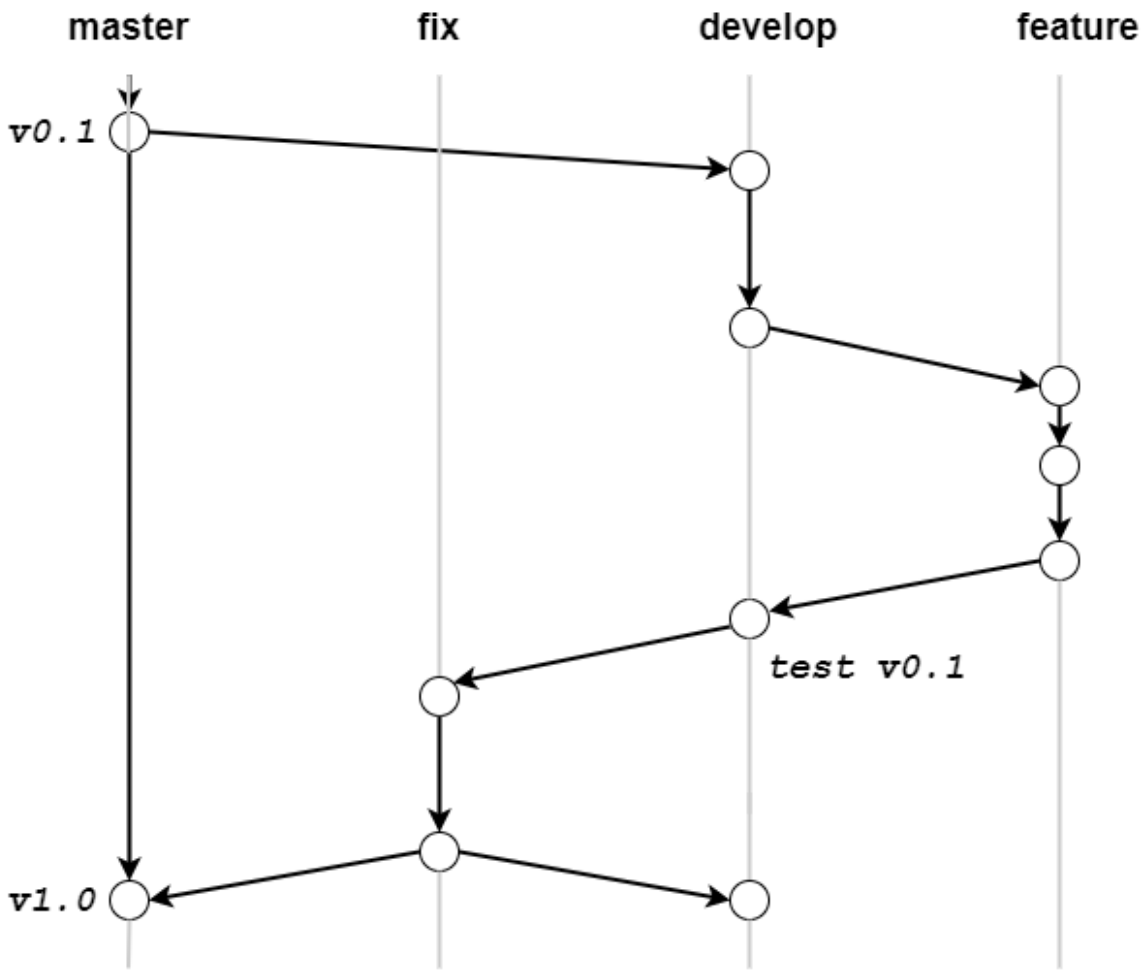
修复分支，从develop分支上检出。用作bug修复，bug修复完成需merge回develop分支，并将其删除。所以该分支属于临时性分支。

工作流程

经典工作流


feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release after 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

简化工作流



提交规范

git commit message应该清晰明了，说明本次提交的目的，具体做了什么操作，每次提交，Commit message 都包括三个部分：header，body ，其中，header 是必需的，body 可以省略。

```
<type>: <subject>

<body>
```

Header

Header部分只有一行，包括：`type`（必需）和`subject`（必需）。

type

用于说明 commit 的类别,至允许使用下面的标识：

- feat：新功能（feature）
- fix：修补bug
- docs：文档（documentation）修改
- style： 格式（不影响代码运行的变动）
- refactor：重构（即不是新增功能，也不是修改bug的代码变动）
- test：增加测试
- add：添加内容

subject

用于commit简短描述，不超过50个字符。

其他注意事项：

- 以动词开头，使用第一人称现在时，比如change，而不是changed或changes
- 第一个字母小写，可以是中文
- 结尾不加句号（.）

body

Body 部分是对本次 commit 的详细描述，可以分成多行。下面是一个范例。

```
More detailed explanatory text, if necessary.  Wrap it to
about 72 characters or so.

Further paragraphs come after blank lines.

- Bullet points are okay, too
- Use a hanging indent
```

有两个注意点：

- 使用第一人称现在时，比如使用change而不是changed或changes。
- 永远别忘了第2行是空行
- 应该说明代码变动的动机，以及与以前行为的对比。

示例说明

关联项目或克隆项目

```
cd existing_folder
git init
git remote add origin
ssh://git@10.100.28.244:8122/byrj/SW_Requirement/gitflow.git
```

```
git clone ssh://git@10.100.28.244:8122/byrj/SW_Requirement/gitflow.git
```

打标签与提交

```
touch readme.md
git add readme.md
git commit -m "demo first push"
git tag -a v0.1 -m "version v0.1 "  #打标签
git push origin master   #提交到远程主分支仓库
git push origin --tags  #提交所以tag标签 也可git push origin v0.1
```

提交完成后，通过git log显示如下：

```
git log --pretty=oneline --abbrev-commit --graph ##显示git log关键分支信息
```

```
* 93aaf19 (HEAD -> master, tag: v0.1, origin/master) demo first push
```

开发分支新建与关联

```
git branch develop master    # 从master分支上新建develop分支
git checkout develop    # 检出develop分支
# 此处可进行功能开发，并add和commit到develop分支
git push origin develop    # 推送develop分支到远端的origin/develop
```

提交完成后，通过git log显示如下：

```
* f5587aa (HEAD -> develop, origin/develop) docs:添加test.c文件
* 93aaf19 (tag: v0.1, origin/master, master) demo first push
```

远程分支上需要保持两个分支：master和develop。目的是为以后团队协作更新develop分支做准备。

功能分支新建与合并

本地仓库从develop分支检出feature分支，不断在feature分支开发新功能、合并至develop分支，然后push至远程develop分支

```
git branch feature_add develop  # 从develop分支上新建feature_add分支
git checkout feature_add  # 检出feature_add分支
# 此处可进行功能开发，并add和commit到feature_add分支
git checkout develop    # 检出develop分支
git merge --no-ff feature_add  # feature_add合并到develop分支
git push origin develop    # 推送develop分支到远端的origin/develop
git branch -d feature_add  #删除feature_add功能分支
```

develop分支所有功能开发完成后，在develop分支打测试(test-v0.1)标签发布测试版本

```
git tag -a test-v0.1 -m "test version v0.1 "
git push origin test-v0.1
```

提交完成后，通过git log显示如下：

```
*   ffc32c6 (HEAD -> develop, tag: test-v0.1, origin/develop) 完成了加法功能，合并至develop分支
|\
| * c8cb61c (feature_add) feature:完成add加法功能
|/
* f5587aa docs:添加test.c文件
* 93aaf19 (tag: v0.1, origin/master, master) demo first push
```

修复分支测试与发布

在develop分支发布测试版本后，需在该版本基础上新建fix分支用于修复BUG，之后合并到develop分支。该阶段为持续修复BUG、发布测试版本阶段，直接测试版本稳定。

```
git branch fix develop
git checkout fix
# 此处可进行BUG修复开发，并add和commit到fix分支
git checkout develop
git merge --no-ff fix
git tag -a test-v0.2 -m "test version v0.2"
git push origin test-v0.2 #第二次测试版本打标签，发布
#重复上面fix分支修复操作
```

```
*   f5b37ae (HEAD -> develop, origin/develop) 修改GUG，合并到develop分支
|\
| * 8136152 (fix) 修改add接口调用
* |   6ec1afe (tag: test-v0.2) 测试版本test version 0.1修复一个bug
|\ \
| |/
| * 53ffa65 docs:修改readme.md描述
| * f451828 fix:修改add操作
|/
*   ffc32c6 (tag: test-v0.1) 完成了加法功能，合并至开发分支
|\
| * c8cb61c feature:完成add加法功能
|/
* f5587aa docs:添加test.c文件
* 93aaf19 (tag: v0.1, origin/master, master) demo first push
```

主分支合并完成发布

当develop分支多次发布和测试得到稳定版本后，需将develop分支合并至master主分支，并打标签，完成整个开发流程

```
git checkout master
git merge --no-ff develop
git tag -a v1.0 -m "v1.0"
git push origin master
git push origin v1.0
```

提交完成后，通过git log显示如下：

```
*   3fddea8 (HEAD -> master, tag: v1.0, origin/master) 完成开发，开发分支合并到主分支
|\
| *   f5b37ae (origin/develop, develop) 修改GUG，合并到develop分支
| |\
| | * 8136152 (fix) 修改add接口调用
| * |   6ec1afe (tag: test-v0.2) 测试版本test version 0.1修复一个bug
| |\ \
| | |/
| | * 53ffa65 docs:修改readme.md描述
| | * f451828 fix:修改add操作
| |/
| *   ffc32c6 (tag: test-v0.1) 完成了加法功能，合并至开发分支
| |\
| | * c8cb61c feature:完成add加法功能
| |/
| * f5587aa docs:添加test.c文件
|/
```

```
* 93aaf19 (tag: v0.1) demo first push
```