基于 Git Flow 的开发流程

Git Flow 分支模型

master 分支 => 正式环境（最为稳定功能最为完整）

test 分支（release分支） => 测试环境（发布定期要上线的功能）

dev 分支 => 团队协作的发开环境（功能最新最全的分支）

hotfix 分支 => 修复线上代码的 bug

feature分支 => 某个功能点正在开发阶段

**开发阶段**

创建一个对应的 feature 分支，拉取 dev 分支代码，进行开发

功能开发完并自测之后，先切换到 develop 分支将最新的代码拉取下来，再把 feature 分支的代码合并进来。有冲突解决冲突。

如果管理比较严格的话可能会有code-review，那么此时你应该提交一个pull request请求，告诉项目管理员，请求代码审核，审核通过后由管理员进行合并。

**测试环境**

完成新功能后，dev 分支进行自测，自测通过merge到 test 分支，测试人员进行测试

如有bug：创建hotfix 分支（修复后先合dev分支进行二次测试，没问题再合并到test分支 ）

**发布上线**

项目管理员将Test分支提交到 master分支，并部署

**上线**

线上环境出现 bug：创建 hotfix+date 分支进行修改（修复后合并到 master分支），大bug：版本回滚

分支命名

参考：

feature——按照功能点（而不是需求）命名；

test——用发布时间命名，可以加上适当的前缀；

Hotfix+date——反应问题 的 issue 编号或 bug 性质或时间等。

****各分支关系****：  
[](https://camo.githubusercontent.com/efdcb5a013cfb015848b68bd777cdb7f77176e3c/687474703a2f2f646976696f2e71696e6975646e2e636f6d2f466d48594e4f3365794e354d6a517159357254526a78754177554434)