Git 最佳实践 & Git flow介绍

刘冰心　2016.06
Bingxin.liu@zhaopin.com.cn

“ Nobody actually creates perfect code the first time around, except me. But there's only one of me ”

- Linus Torvalds (25 May 2007)

- Git 前世今生
- Git 设计哲学
- Git 常用命令及GUI
- Git 最佳实践
- Git demo
- Git flow 介绍
- Git flow demo

Git 前世今生

- 作者：Linus
- linux的开发管理 2002之前 diff patch命令
- 2002之后 bitkeeper
- 2005之后 bitkeeper撤出 需要一款新的VCS
  - 速度
  - 简单的设计
  - 对非线性开发模式的强力支持（允许成千上万个并行开发的分支）
  - 完全分布式
  - 有能力高效管理类似 Linux 内核一样的超大规模项目（速度和数据量）
- Git != github != gitlab

Git 设计哲学

- 简单
  - 核心只有3个对象 blob、tree、commit
    - blob和tree之间通过hash key关联，作为唯一协议
      tree和blob或自己关联
      commit和tree关联，和blob不直接关联
- 分布式
- 保存快照 而非差异比较

Git 常用命令

- 基础
  - clone
  - add
  - commit
  - pull
  - push
  - log
- 分支
  - branch
  - checkout
  - merge
- 高级
  - rebase*
  - reset
  - tag

Git 常用命令及GUI

- 命令行
- GUI客户端
  - sourcetree（跨平台，集成gitflow）
  - tortoisegit（windows only）

Git 最佳实践

- 单一提交 - 一个提交只做一件事
- 快速提交 - 有助于每次提交的东西很少 减少冲突
- 不要提交半成品 - 并不是要求把大功能完整实现好之后再提交，而是要分解功能
- 提交之前的测试
- 提交描述
- 使用分支 – 即使滥用也强过不用

Git 最佳实践

- .gitignore 忽略文件
  - 忽略可从外部获取的库，eg. NPM, nuget, maven
  - 忽略最终build的文件，eg. *.DLL, *.EXE, gulp 后的html js
  - 忽略IDE的配置文件，eg. .idea, .vscode, *.suo
  - 忽略日志
  - 忽略测试文件

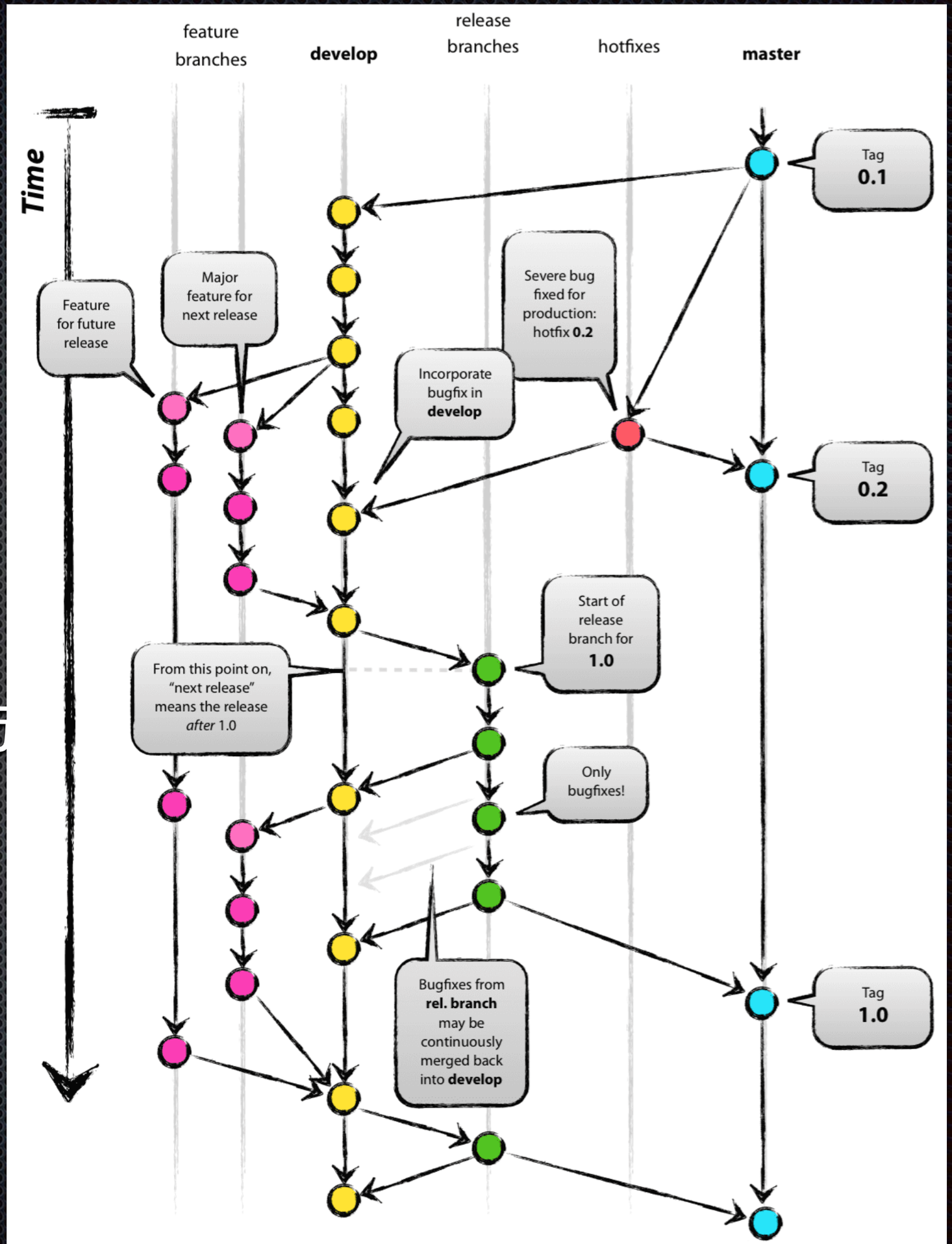
Git 最佳实践

- commit 信息的填写
  - 不要附加多余的信息，例如Author，Date，Email
  - 不要使用含糊的或者概括的描述。修复BUG “修复用户登录后无故崩溃的BUG”或者“修复ISSUE 553”
  - 不能用一句话描述你的提交，很有可能说明你的提交在做不止一件事情

Git demo

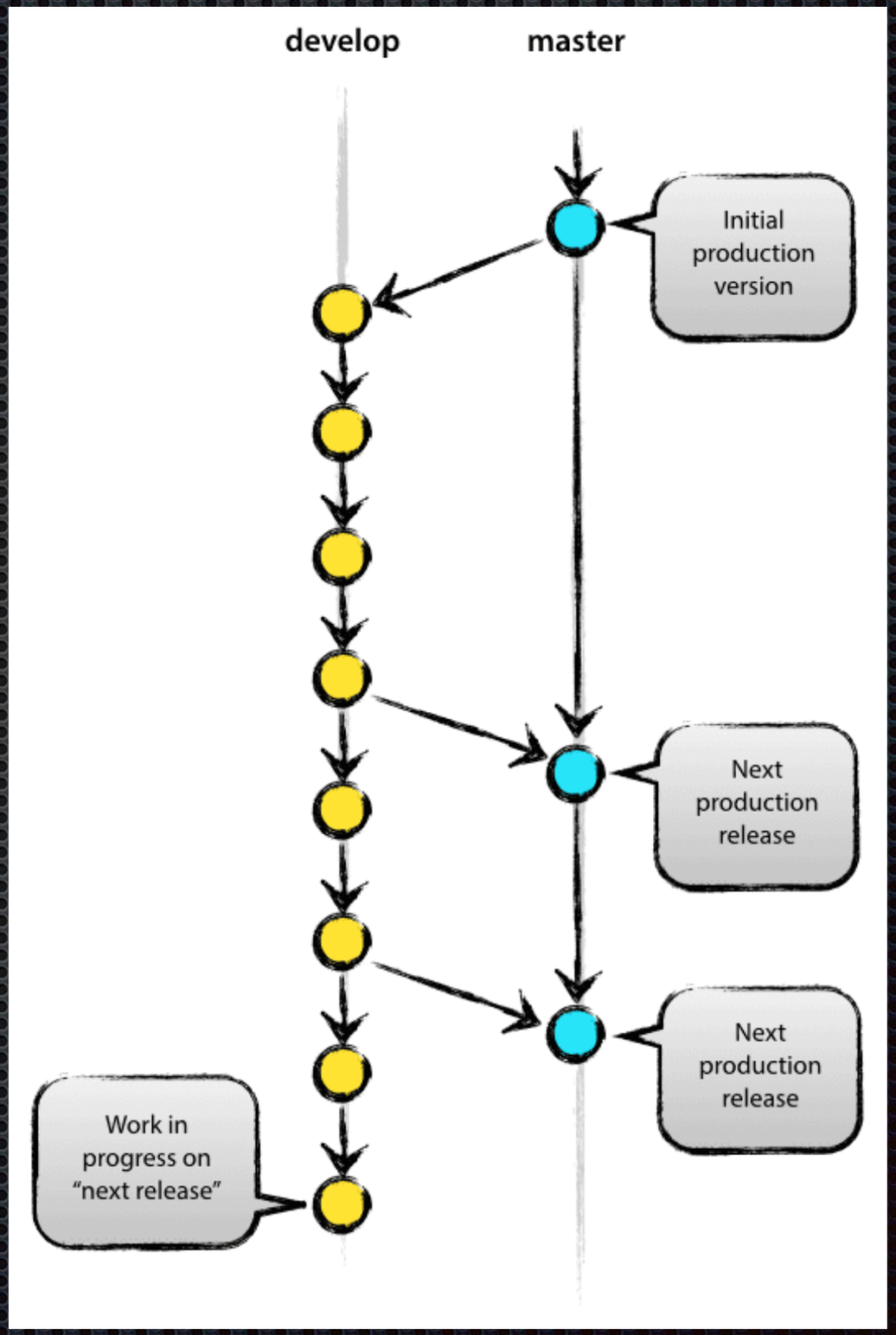
Git flow

- 提出者：nvie
- 是一种Git分支模型 方
- 衍生出一套相应工具



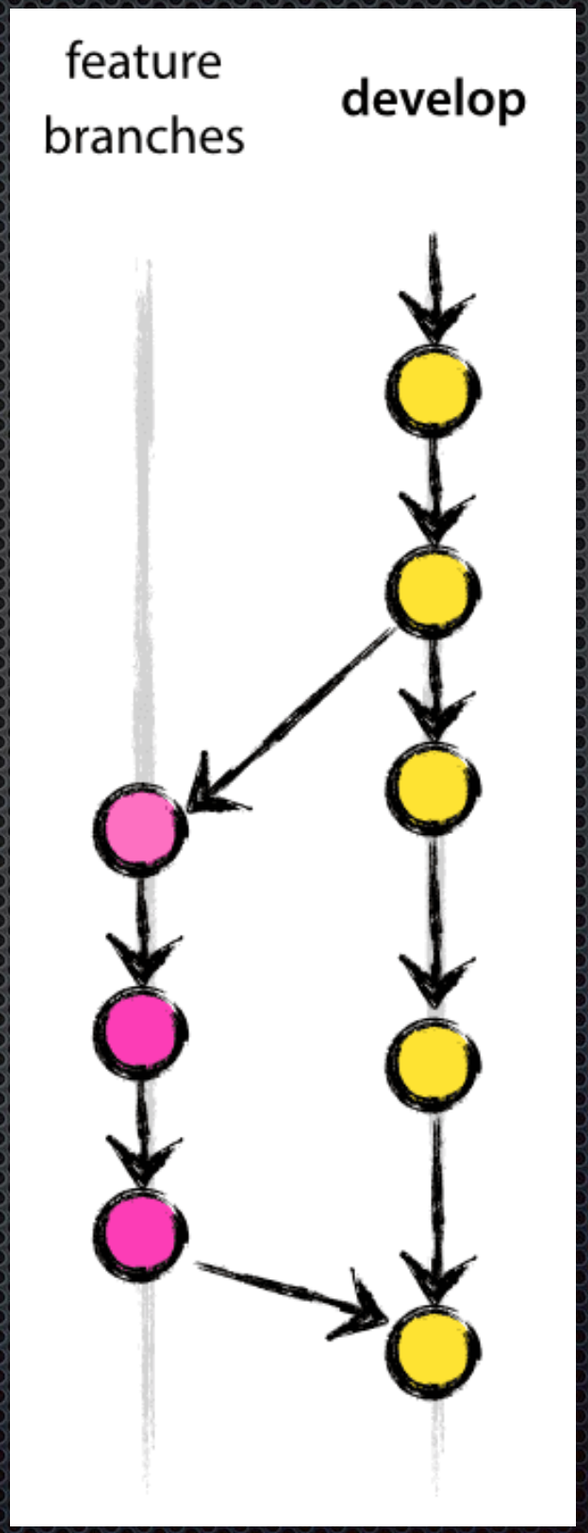
Git flow

- 主要分支
  - master - 随时可供在生产环境中部署的代码
  - develop - 日常开发分支，可以进行每日发布的代码



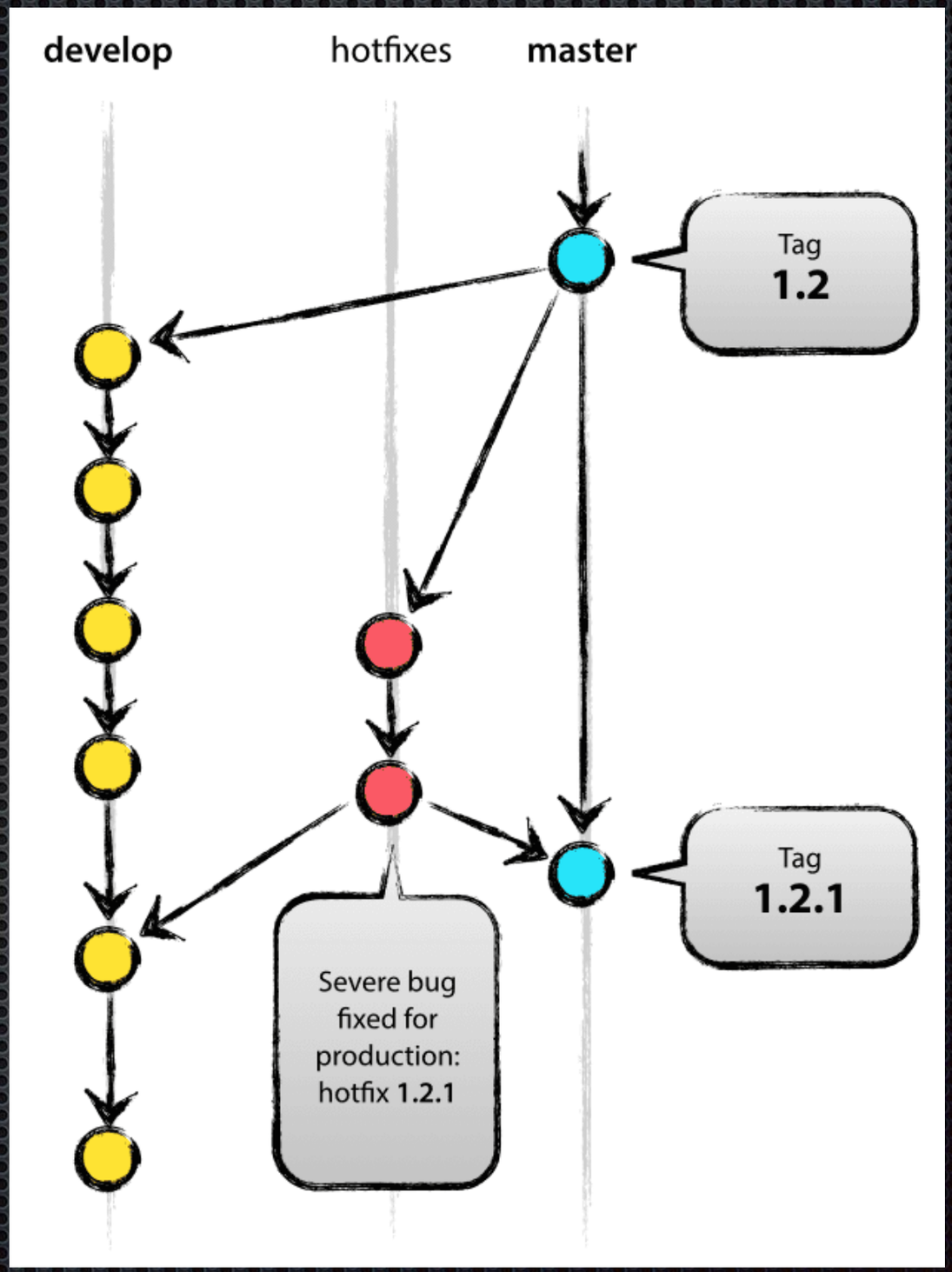
Git flow

- feature分支



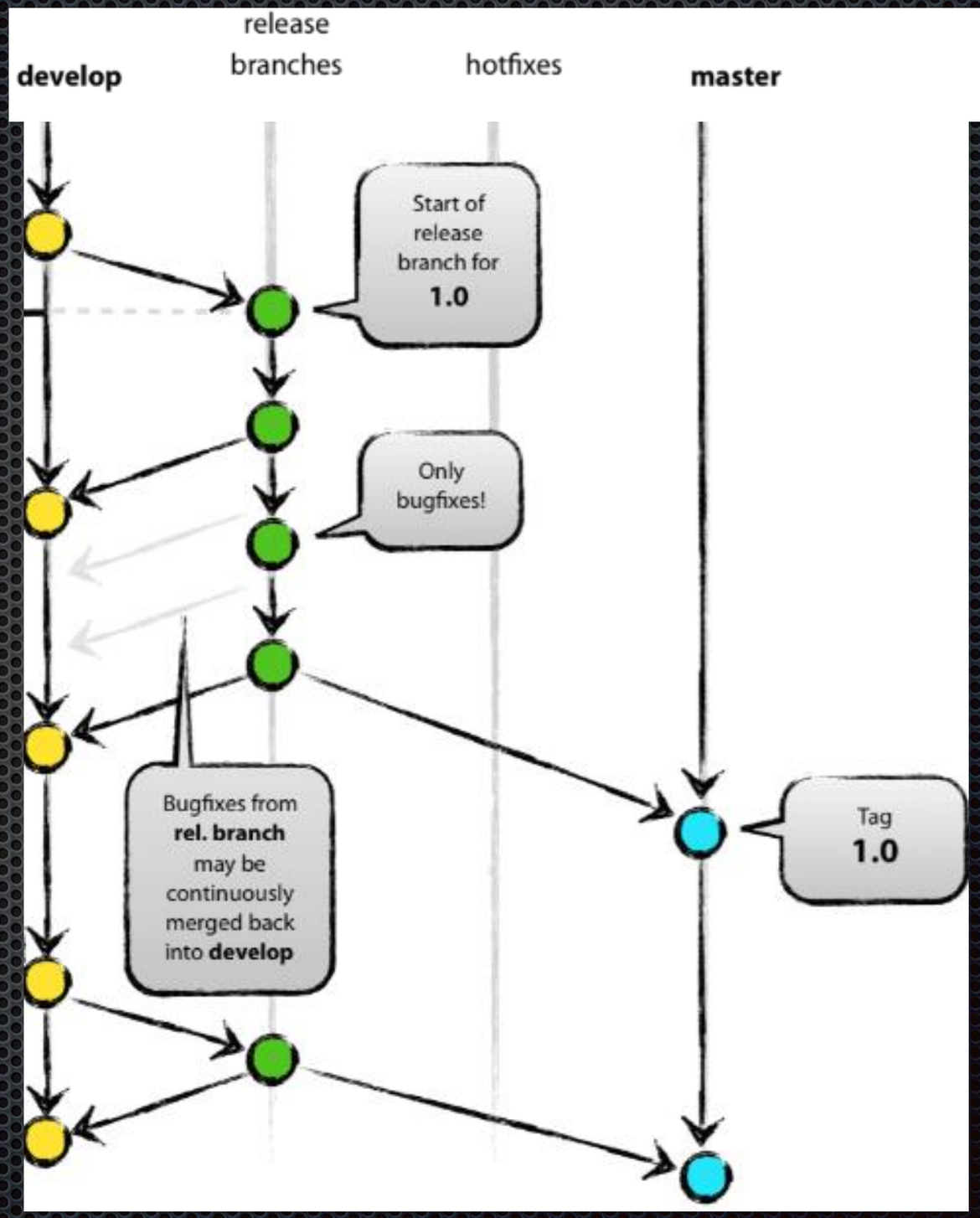
Git flow

- hotfix分支



Git flow

- release分支



Git flow demo

- 引用 & 参考
  - Pro git book https://git-scm.com/book/zh/
  - gitflow http://nvie.com/posts/a-successful-git-branching-model/

Q & A

Thanks