使用 Git 让团队开发更有效率、更规范

Git 和基于 Git 的开发流程的介绍

Cai Mingshi / i@unoiou.com / 2018/06/08

为什么使用 Git?

简易的版本控制

权限管理

代码分享

项目管理

去中心化

随时随地更新代码

团队协作

保存快照

Git 简介

Git 存储区域和状态

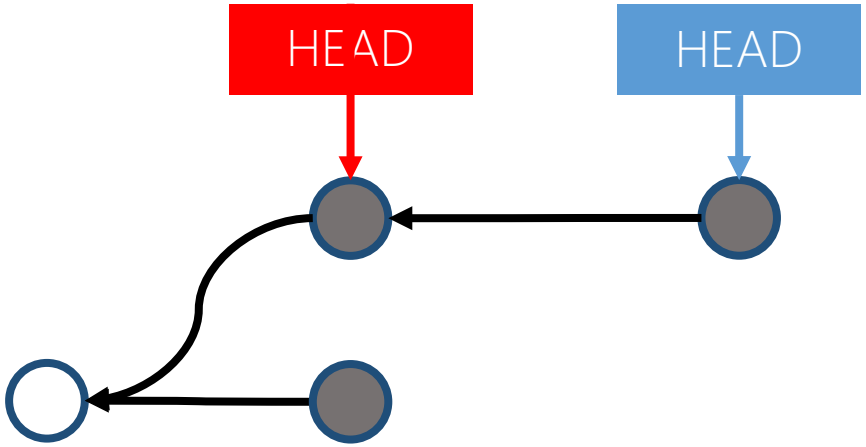
Git 简介

`HEAD`

对一个特定的 branch / commit / file 的引用，头指针

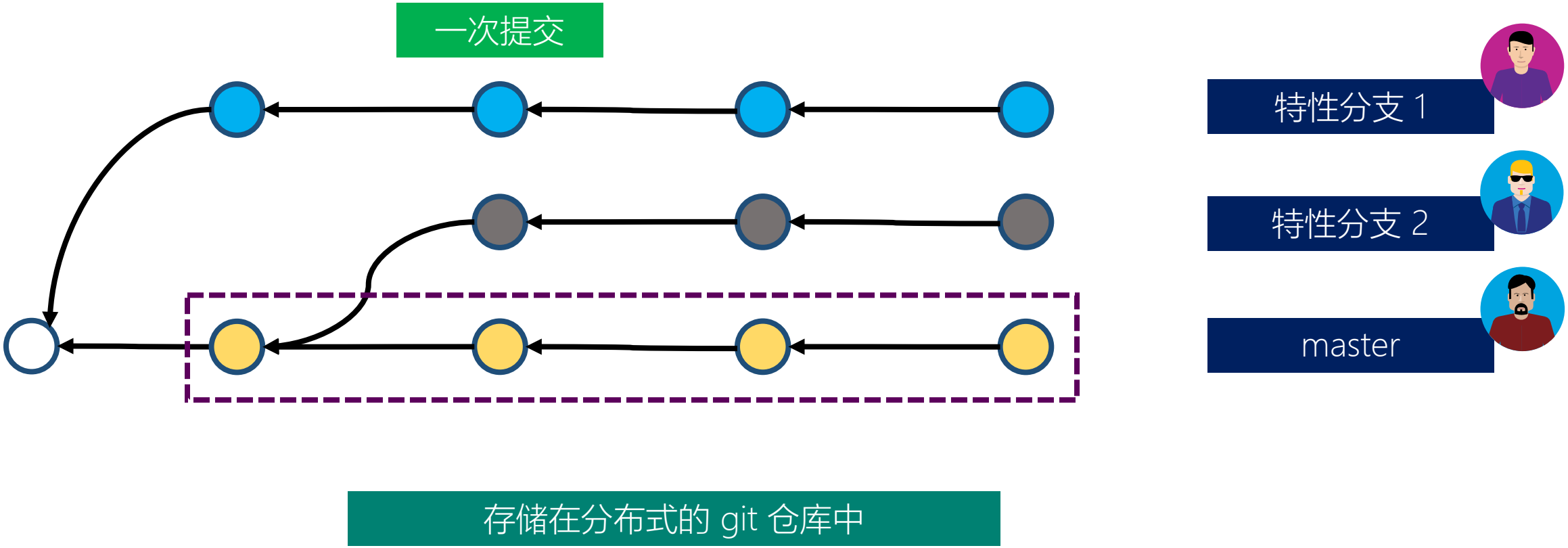
detached HEAD state

在 git 命令中作为参数



Git 简介

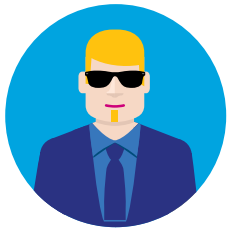
Git 分支、提交、仓库...



Git 工作流：集成管理者工作流

私有个人仓库

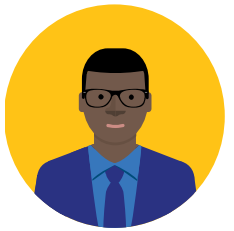
公开仓库



私有个人仓库

公开仓库

项目主仓库



管理者

私有个人仓库

公开仓库

Git 不仅能处理这些内容

代码仓库 / Repository

```
git init

git init --bare –shared /your/git/path/project.git

git clone git@git.contoso.com:remote-directory/project-name.git

git remote add repository-name remote-repository-ssh-addr

git status
```

Git 不仅能处理这些内容

文件

```
git add /direcotry/untracked-file-name.file-type

git add /direcotry/modified-file-name.file-type

git stash

git rm --cached file-name-or-file-path

vim .gitignore

# .gitignore
/.vscode/
*.orig
.pycharm
```

Git 不仅能处理这些内容

文件

```
git reset HEAD file-name
# or
git reset file-name

# to specific commit
git checkout commit-hash -- file-name

# about `--`

# to last commit without hash
git checkout -- file-name
```

Git 不仅能处理这些内容

提交 / Commit

```
git add file-name
# git add .
git commit –m “your commit msg”
```

```
git rebase [branch-name] [–i]
```

Git 不仅能处理这些内容

分支 / Branch

```
git branch new-branch-name

git checkout new-branch-name

git checkout -b new-branch-name

git branch -d branch-name

git push remote-repository-name : remote-branch-name

git branch -m branch-old-name branch-new-name

git branch –a
```

Git 不仅能处理这些内容

分支 / Branch

```
# to merge f-b-1 into c-b
git checkout current-branch
git merge feature-branch-1

git checkout current-branch
git rebase feature-branch-1
```

```
git pull remote-repository-name remote-branch-name : local-branch-name

git push remote-repository-name local-branch-name : remote-branch-name
```

“origin”

Git 不仅能处理这些内容

版本控制 / Version Control

```
git reset HEAD^

git reset HEAD~233

# reset parameters
git reset [parameter] (HEAD | Commit-Hash)

# --mixed

# --soft

# --hard

git log [parameters]
git log –graph
git log -23
```

克服 Git 中的最常见的阻碍

解决代码冲突

merge

```
git checkout master
git fetch
git checkout feature-branch

git merge master

git merge

git add /directory/conflicted-code.file-type
git commit -m "solve conflicts"

git push
```

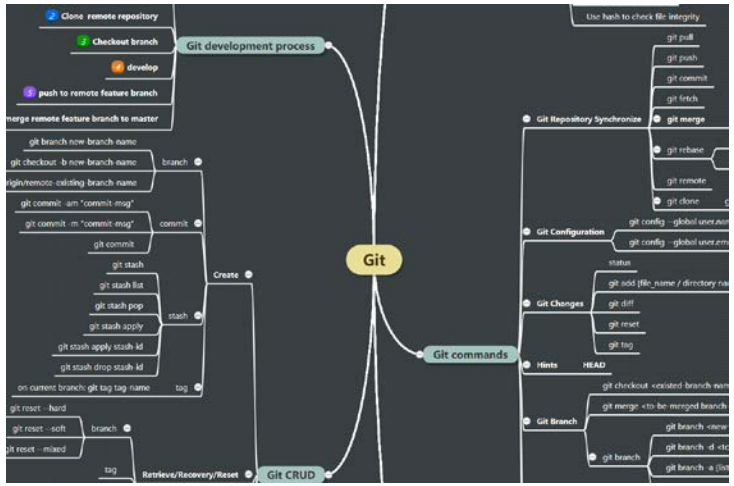
克服 Git 中的最常见的阻碍

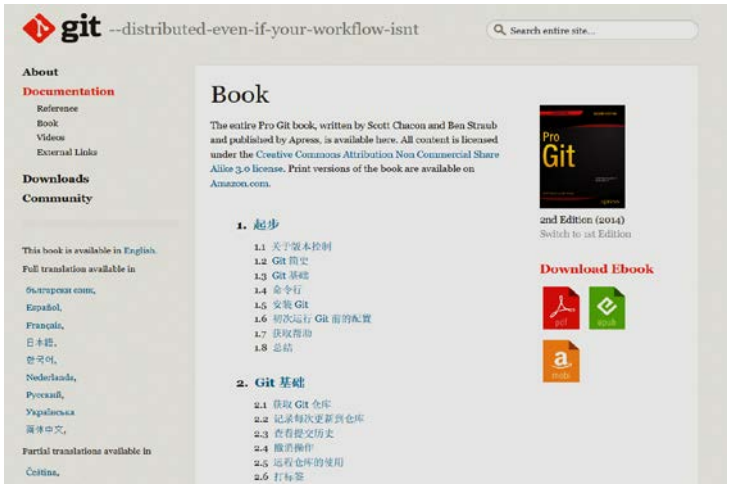
解决代码冲突

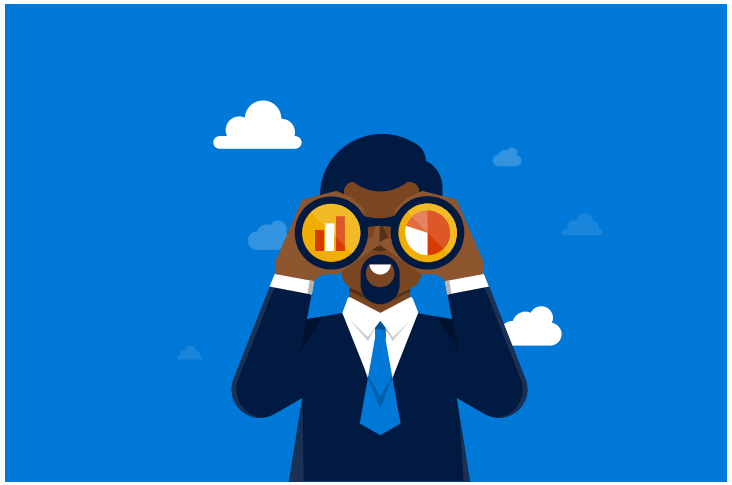
rebase

```
git checkout master
git fetch
git checkout feature-branch

git rebase master

git commit -am "solve conflicts"
git rebase --continue

git push
```

进一步了解 Git







Git 脑图

Pro Git Book

Githug

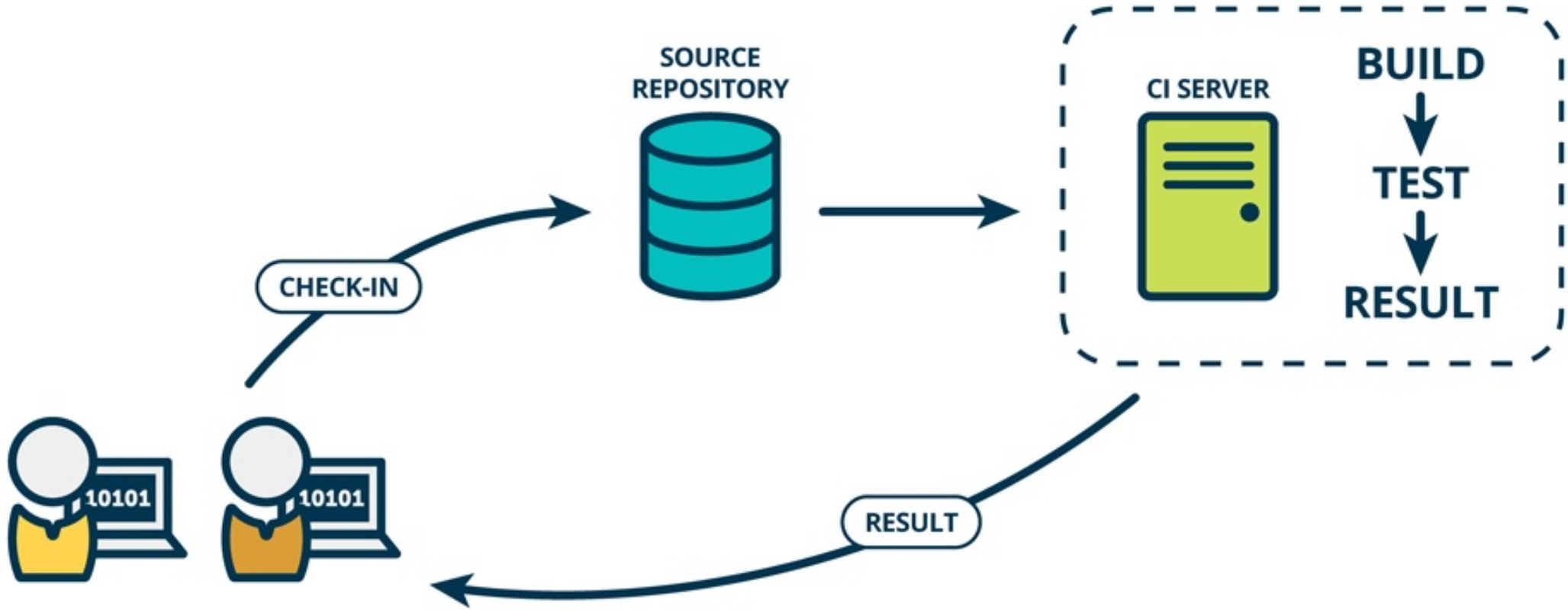
http://t.im/naotu

http://t.im/gitbook

http://t.im/githug

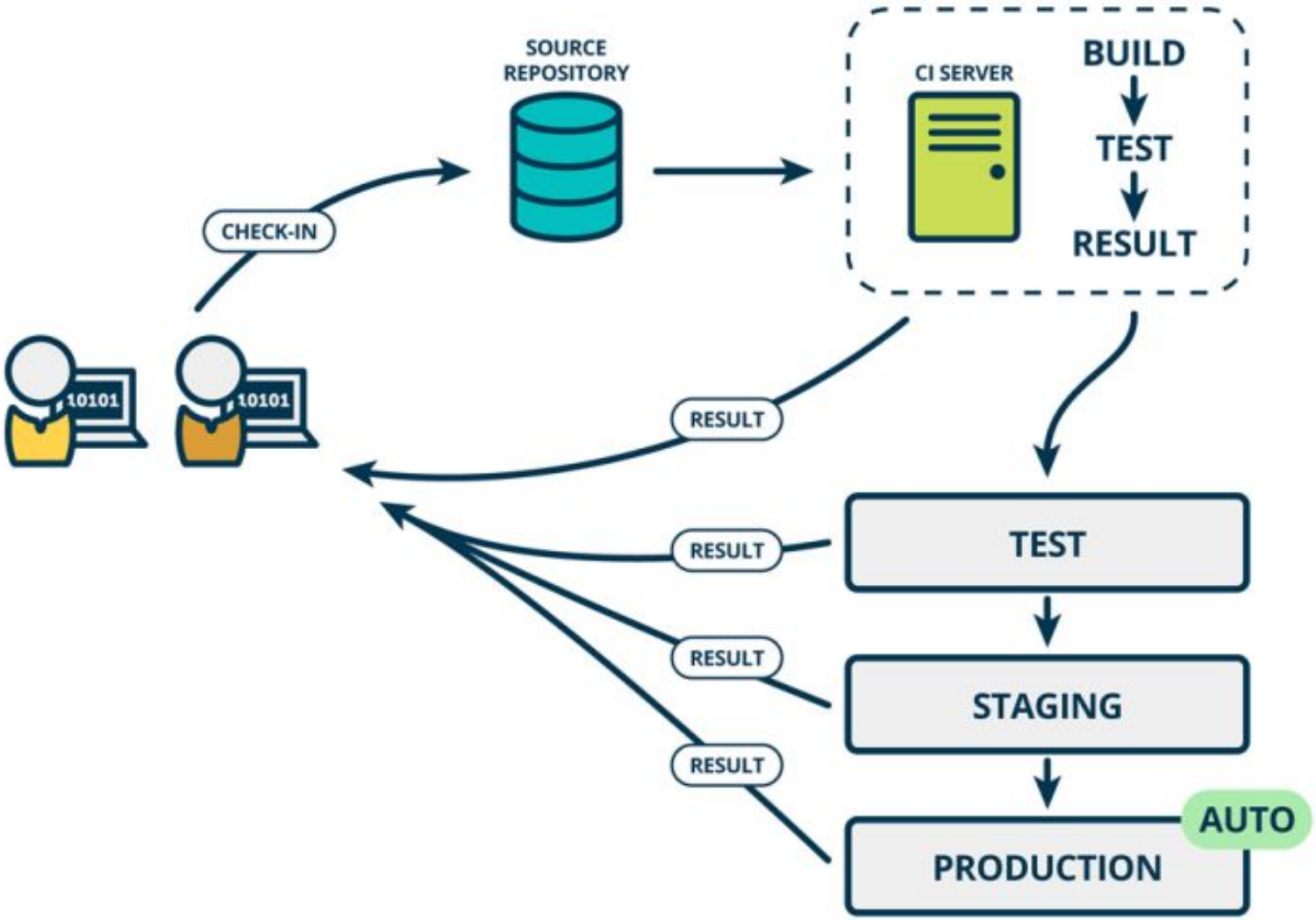
配合 CI/CD 工具， Git 更能简化开发

持续集成(Continuous Integration)



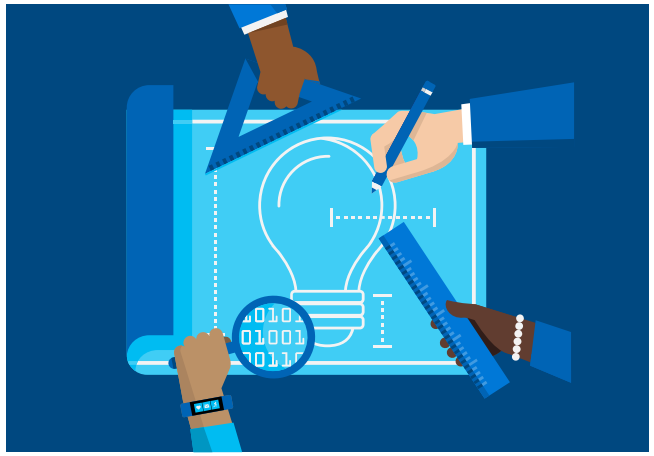
配合 CI/CD 工具， Git 更能简化开发

持续部署(Continuous Deployment)

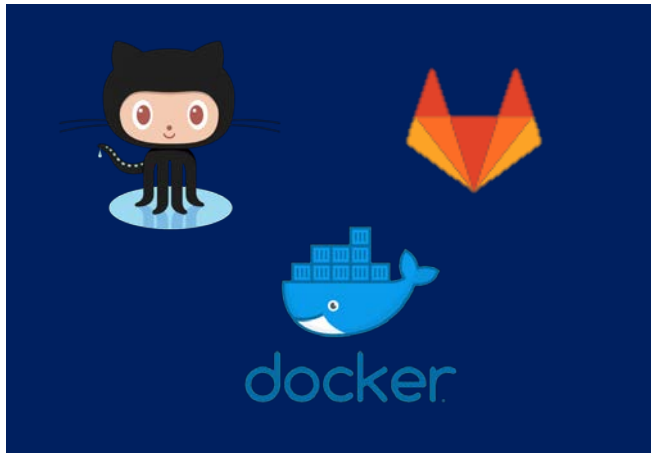


我们的开发流程：测试驱动、持续集成、DevOps

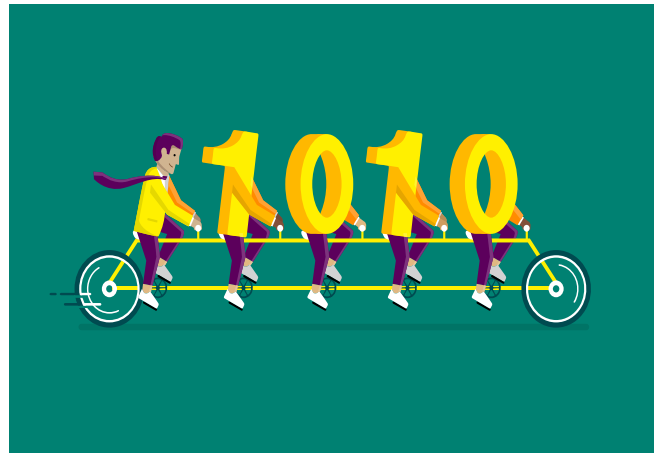
需求文档 / 测试用例 / 任务管理



代码仓库 / 自动化测试环境



任务分解 Issues



根据 Issues 创建分支

团队协作开发

持续集成 / 持续部署

拥抱 Git 以创造更多

Cai Mingshi / i@unoiou.com / 2018/06/08