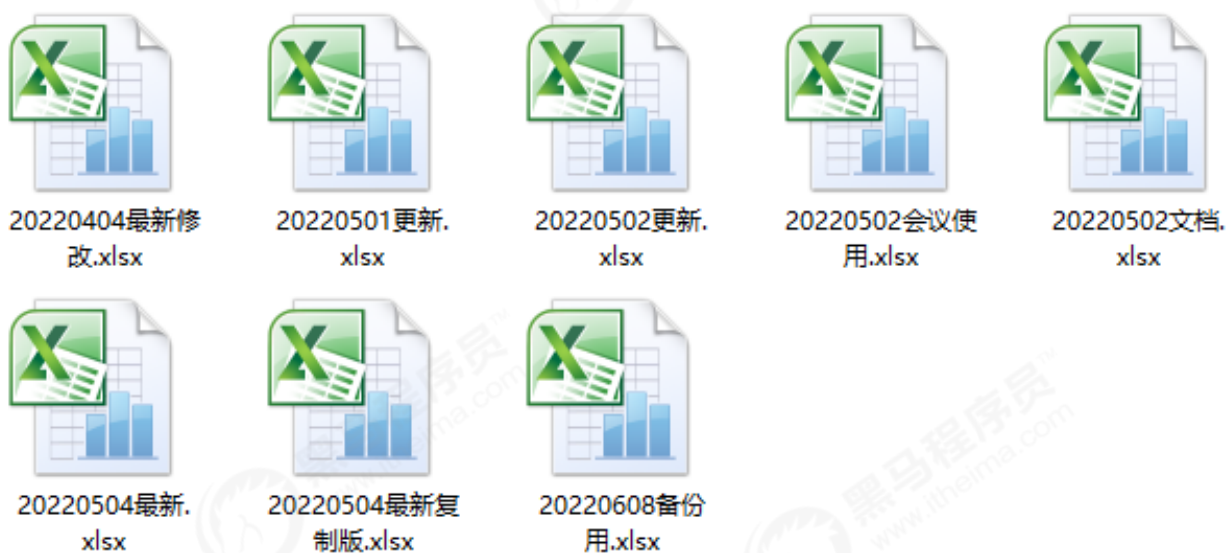


# 1. Git基础

## 1.1 版本管理

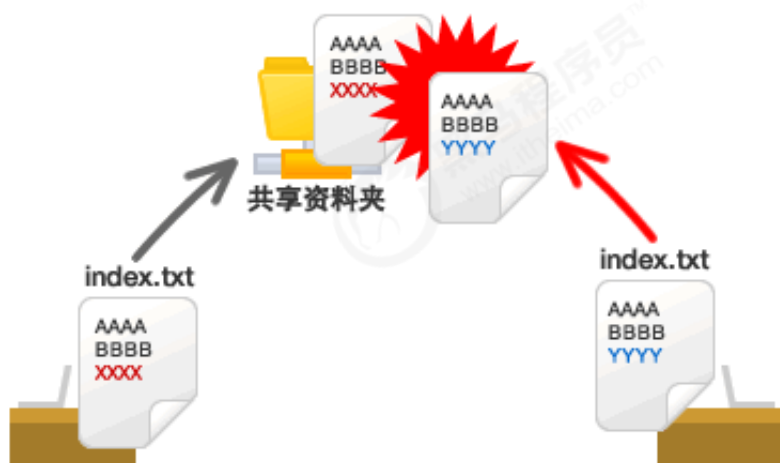
### 1.1.1 什么是版本管理

版本管理是一种记录文件变化的方式，以便将来查阅特定版本的文件内容。



### 1.1.2 人为维护文档版本的问题

1. 文档数量多且命名不清晰导致文档版本混乱
2. 每次编辑文档需要复制，不方便
3. 多人同时编辑同一个文档，容易产生覆盖



# 1.2 Git 是什么

Git是一个版本管理控制系统（缩写VCS），它可以在任何时间点，将文档的状态作为更新记录保存起来，也可以在任何时间点，将更新记录恢复回来。



# 1.3 Git 安装

[下载地址](#)

在安装的过程中，所有选项使用默认值即可。

# 1.4 Git 基本工作流程

git仓库	暂存区	工作目录
用于存放提交记录	临时存放被修改文件	被Git管理的项目目录



## 1.5 Git 的使用

### 1.5.1 Git 使用前配置

在使用 git 前，需要告诉 git 你是谁，在向 git 仓库中提交时需要用到。

1. 配置提交人姓名: `git config --global user.name 提交人姓名`
2. 配置提交人姓名: `git config --global user.email 提交人邮箱`
3. 查看git配置信息: `git config --list`

注意

1. 如果要对配置信息进行修改，重复上述命令即可。
2. 配置只需要执行一次。

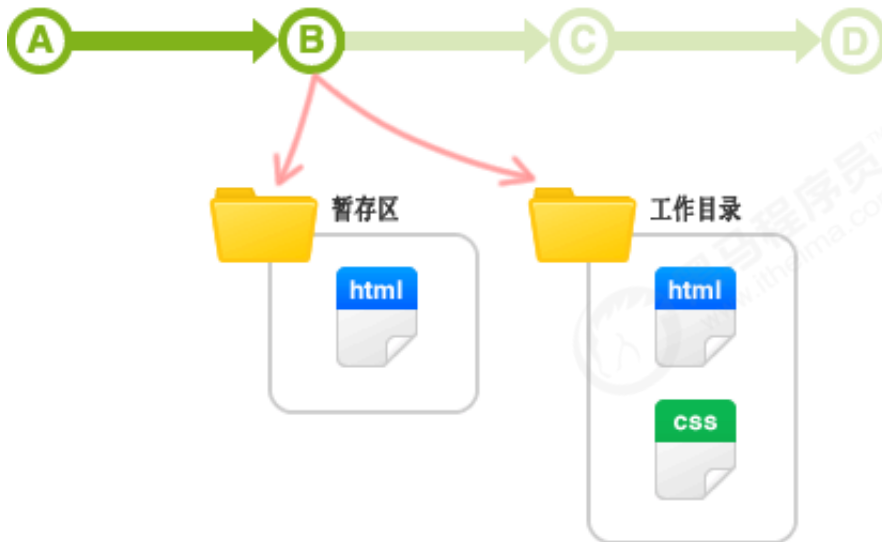
### 1.5.2 提交步骤

1. `git init` 初始化git仓库
2. `git status` 查看文件状态
3. `git add 文件列表` 追踪文件
4. `git commit -m 提交信息` 向仓库中提交代码
5. `git log` 查看提交记录

### 1.5.3 撤销

- 用暂存区中的文件覆盖工作目录中的文件: `git checkout 文件`
- 将文件从暂存区中删除: `git rm --cached 文件`

- 将 git 仓库中指定的更新记录恢复出来，并且覆盖暂存区和工作目录：`git rest --hard commitID`

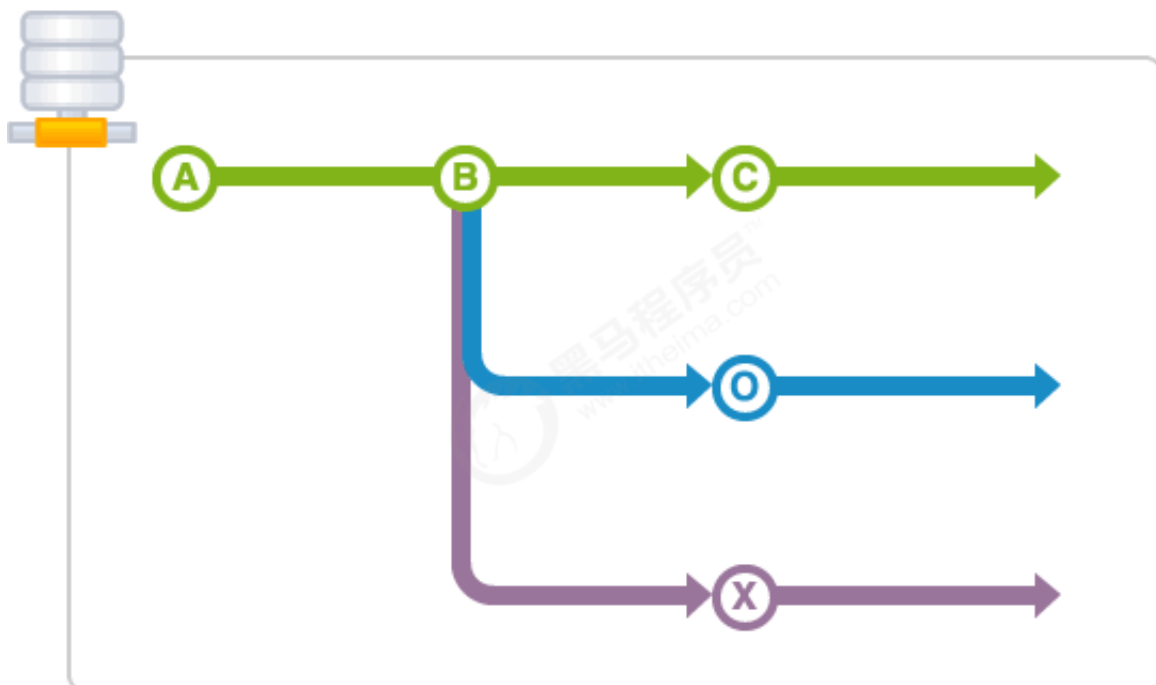


## 2. Git进阶

### 2.1 分支

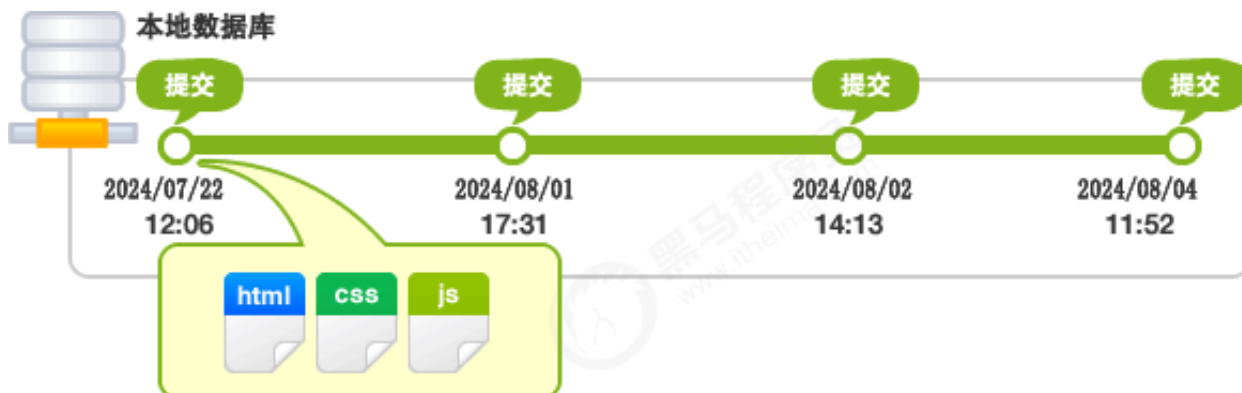
为了便于理解，大家暂时可以认为分支就是当前工作目录中代码的一份副本。

使用分支，可以让我们从开发主线上分离出来，以免影响开发主线。

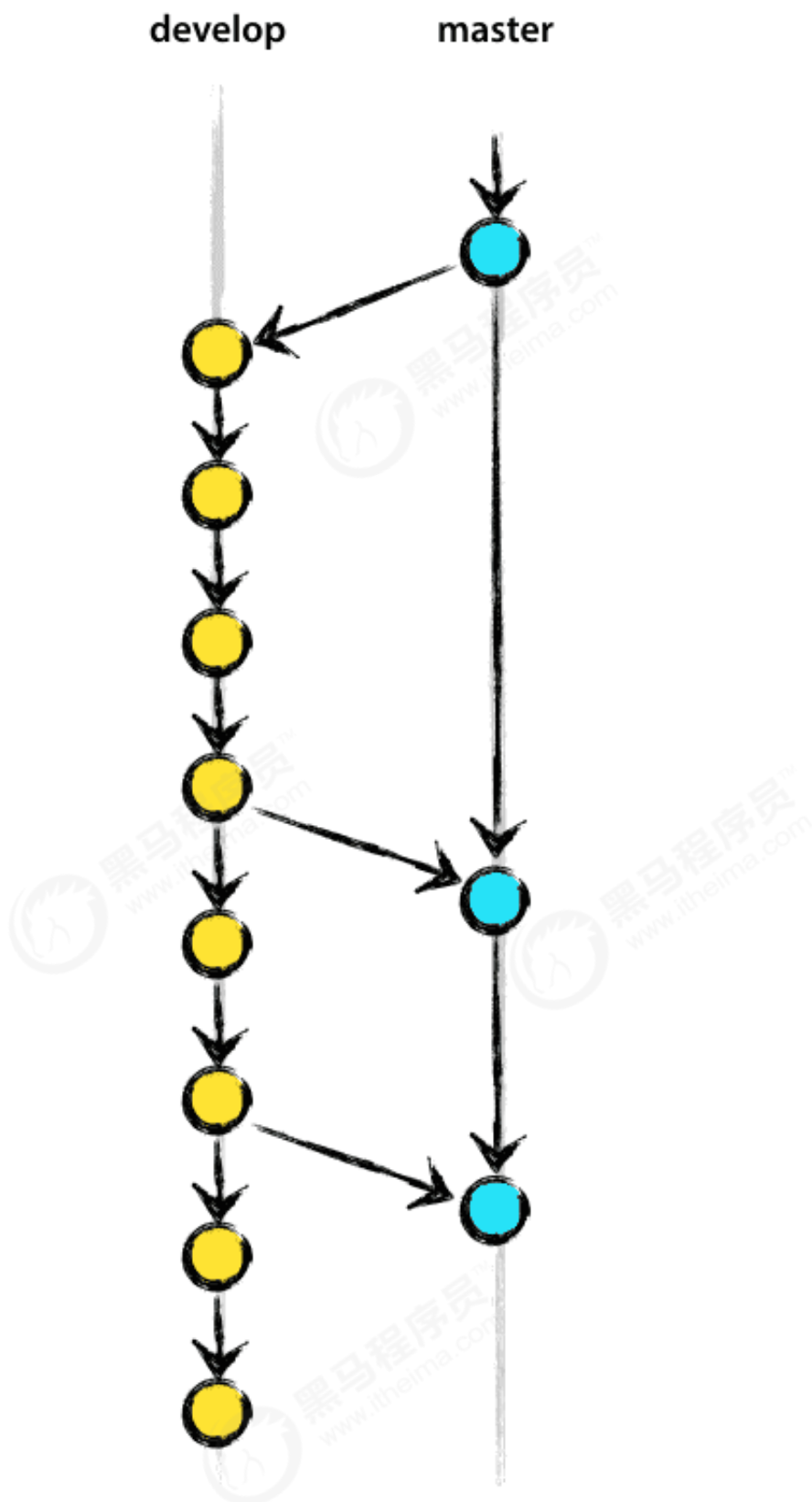


### 2.1.1 分支细分

1. 主分支（master）：第一次向 git 仓库中提交更新记录时自动产生的一个分支。

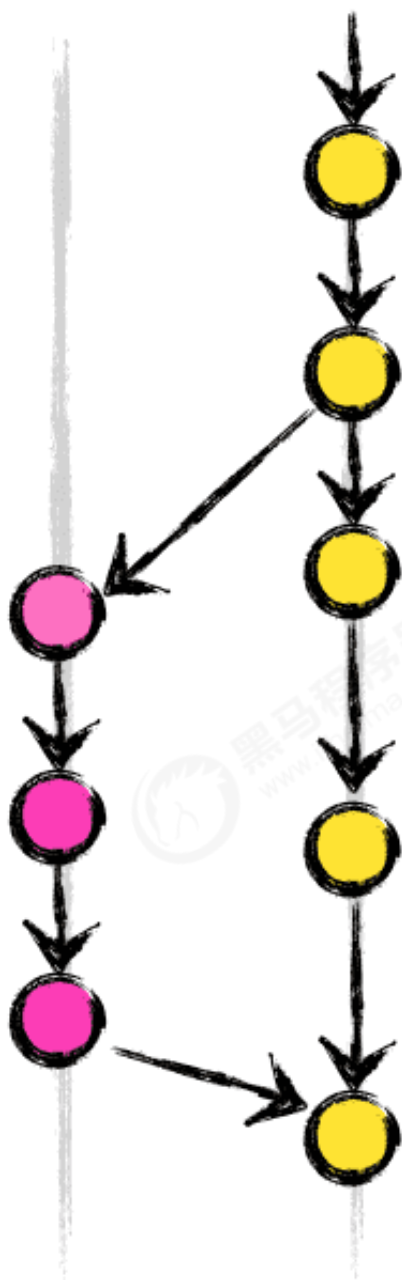


1. 、开发分支（develop）：作为开发的分支，基于 master 分支创建。



1. 功能分支 (feature)：作为开发具体功能的分支，基于开发分支创建

feature  
branches      develop



功能分支 -> 开发分支 -> 主分支

## 2.1.2 分支命令

- `git branch` 查看分支
- `git branch 分支名称` 创建分支
- `git checkout 分支名称` 切换分支

- `git merge 来源分支` 合并分支
- `git branch -d 分支名称` 删除分支（分支被合并后才允许删除）（-D 强制删除）

## 2.2 暂时保存更改

在git中，可以暂时提取分支上所有的改动并存储，让开发人员得到一个干净的工作副本，临时转向其他工作。

使用场景：分支临时切换

- 存储临时改动： `git stash`
- 恢复改动： `git stash pop`

## 3. Github

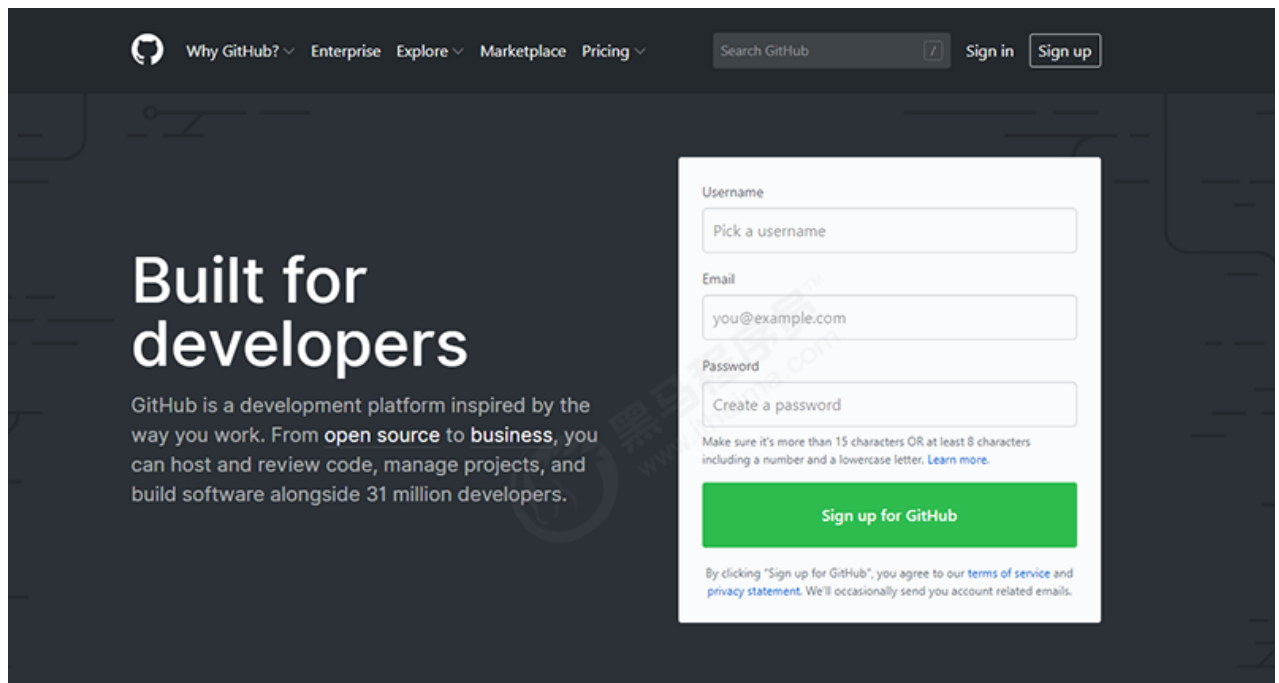
在版本控制系统中，大约90%的操作都是在本地仓库中进行的：暂存，提交，查看状态或者历史记录等等。除此之外，如果仅仅只有你一个人在这个项目里工作，你永远没有机会需要设置一个远程仓库。

只有当你需要和你的开发团队共享数据时，设置一个远程仓库才有意义。你可以把它想象成一个“文件管理服务器”，利用这个服务器可以与开发团队的其他成员进行数据交换。

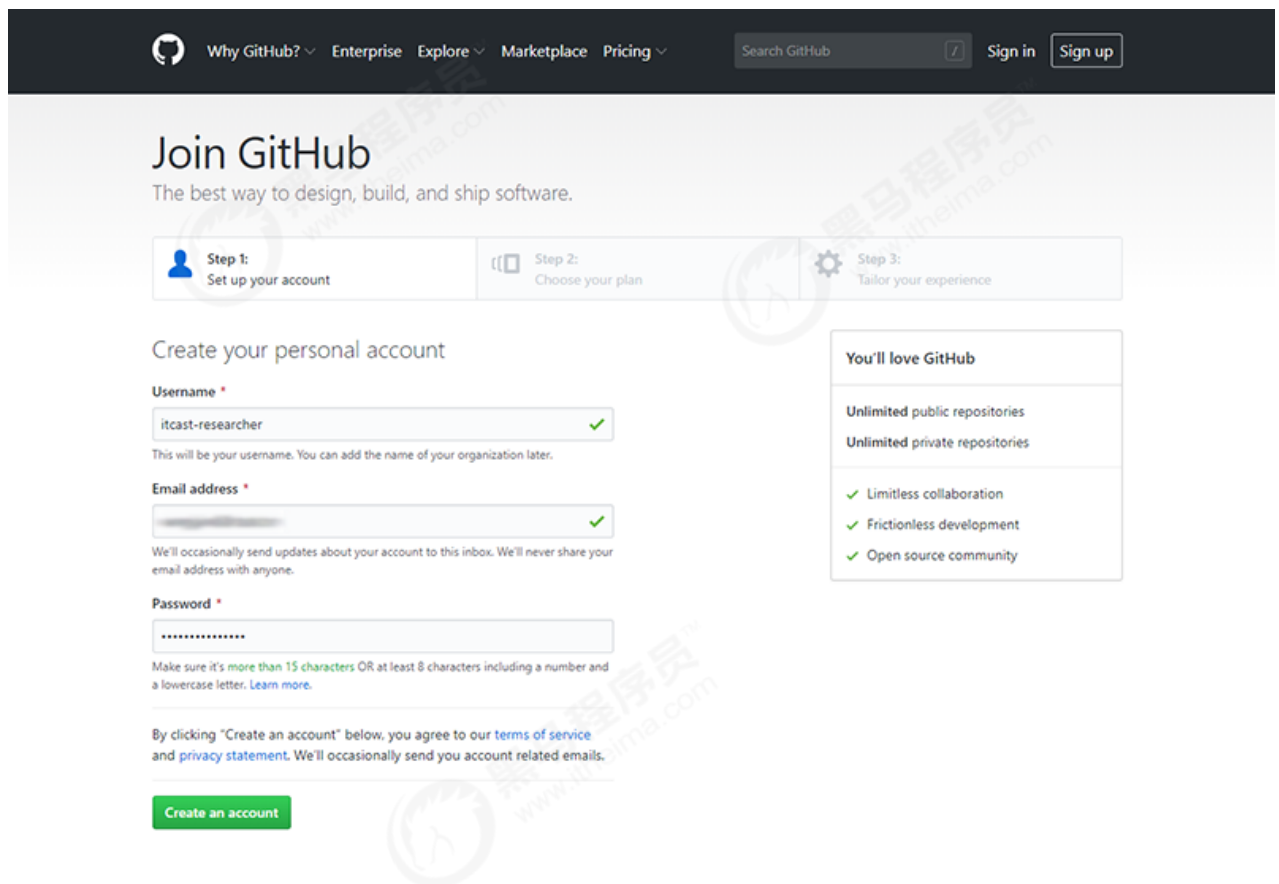
### 3.1 注册

1. 访问[github](https://github.com)首页，点击 Sign up 连接。（注册）





## 2. 填写用户名、邮箱地址、GitHub登陆密码



## 3. 选择计划

# Welcome to GitHub

You're a few steps away from building better software, @itcast-researcher.

✓ Completed

Set up your account

🔧 Step 2:

Choose your plan

⚙️ Step 3:

Personalize your experience

## Choose your plan

With tools developers love and the world's largest open source community, there's no wrong choice.

✓

Free

The basics of GitHub for every developer

\$0

per month

Includes:

∞ Unlimited public and private repositories

✓ 3 collaborators for private repositories

✓ Issues and bug tracking

✓ Project management

Are you a student? Get access to the best developer tools for free with the [GitHub Student Developer Pack](#).

🔧

Pro

Pro tools for developers with advanced requirements

\$7

per month

[\(view in HKD\)](#)

Includes:

∞ Unlimited public and private repositories

∞ Unlimited collaborators

✓ Issues and bug tracking

✓ Project management

✓ [Advanced tools and insights](#)

- ☐ **Help me set up an organization next**  
Organizations are separate from personal accounts and are best suited for businesses who need to manage permissions for many employees. [Learn more about organizations](#)
- ☐ **Send me updates on GitHub news, offers, and events**  
Unsubscribe anytime in your email preferences. [Learn more](#)

Continue

## 4. 填写 GitHub 问题

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

# Welcome to GitHub

You'll find endless opportunities to learn, code, and create, @itcast-researcher.

✓ Completed  
Set up a personal account

🔧 Step 2:  
Choose your plan

⚙️ Step 3:  
Tailor your experience

How would you describe your level of programming experience?

☐ Totally new to programming ☐ Somewhat experienced ☐ Very experienced

What do you plan to use GitHub for? (check all that apply)

☐ Project Management ☐ Development ☐ Research  
☐ Design ☐ School projects ☐ Other (please specify)

Which is closest to how you would describe yourself?

☐ I'm a student ☐ I'm a professional ☐ I'm a hobbyist  
☐ Other (please specify)

What are you interested in?

e.g. tutorials, android, ruby, web-development, machine-learning, open-source


[Submit](#) [skip this step](#)

© 2019 GitHub, Inc. [Terms](#) [Privacy](#) [Security](#) [Status](#) [Help](#)

[Contact GitHub](#) [Pricing](#) [API](#) [Training](#) [Blog](#) [About](#)

## 5. 验证邮箱

[Pull requests](#) [Issues](#) [Marketplace](#) [Explore](#)

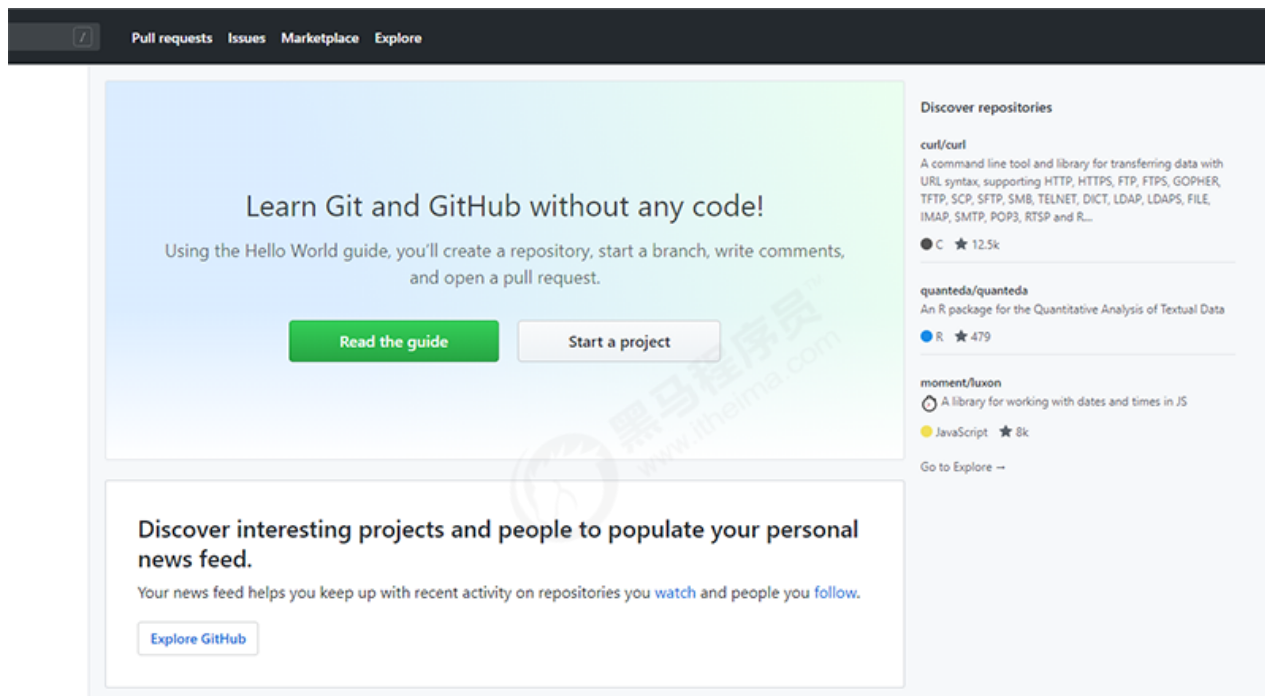


## Please verify your email address

Before you can contribute on GitHub, we need you to verify your email address.  
An email containing verification instructions was sent to wangjian@itcast.cn.

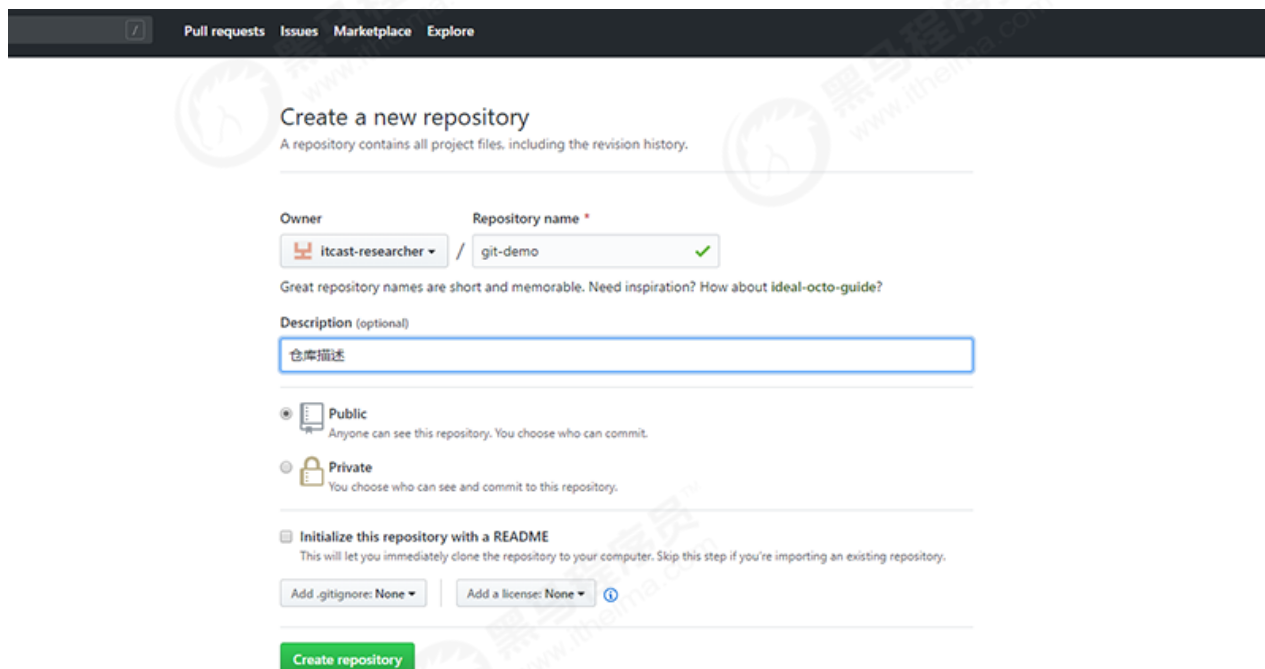
Didn't get the email? [Resend verification email](#) or [change your email settings](#).

## 6. GitHub 个人中心



## 3.2 创建仓库

### 1. 填写仓库基本信息



### 2. 将本地仓库推送到远程仓库

Navigation bar: Pull requests, Issues, Marketplace, Explore

Repository: itcast-researcher / git-demo

Buttons: Watch 0, Star 0, Fork 0

Menu: Code, Issues 0, Pull requests 0, Projects 0, Wiki, Insights, Settings

**Quick setup — if you've done this kind of thing before**

Set up in Desktop or **HTTPS** `https://github.com/itcast-researcher/git-demo.git`

Get started by creating a new file or uploading an existing file. We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

**...or create a new repository on the command line**

```
echo "# git-demo" >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin https://github.com/itcast-researcher/git-demo.git
git push -u origin master
```

**...or push an existing repository from the command line**

```
git remote add origin https://github.com/itcast-researcher/git-demo.git
git push -u origin master
```

**...or import code from another repository**

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

[Import code](#)

ProTip! Use the URL for this page when adding GitHub as a remote.

1. `git remote add` 远程仓库地址别名 远程仓库地址
2. `git push -u` 远程仓库地址或别名 本地分支名称:远程分支名称  
-u 记住推送地址及分支，下次推送只需要输入`git push`即可