## Git 与 GitHub 简介

### Git 与 GitHub 的来历

Linux 之父 Linus 在 1991 年创建开源的 Linux 操作系统之后，多年来依靠全世界广大热心志愿者的共同建设，经过长足发展，现已成为世界上最大的服务器系统。系统创建之初，代码贡献者将源码文件发送给 Linus，由其手动合并。这种方式维持多年后，代码量已经庞大到人工合并难以为继，于是深恶集中式版本控制系统的 Linus 选择了一个分布式商业版本控制系统 BitKeeper，不过 Linux 社区的建设者们可以免费使用它。BitKeeper 改变了 Linus 对版本控制的认识，同时 Linus 发现 BitKeeper 有一些不足，而且有个关键性的问题使之不能被广泛使用，就是不开源。

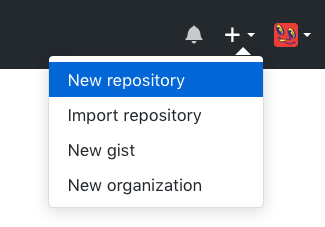
在 2005 年，BitKeeper 所在公司发现 Linux 社区有人企图破解它，BitKeeper 决定收回 Linux 社区的免费使用权。Linus 对此事调解数周无果，找遍了当时已知的各种版本控制系统，没有一个看上眼的，一怒之下决定自己搞一个。Linus 花了十天时间用 C 语言写好了一个开源的版本控制系统，就是著名的 Git。

2007 年旧金山三个年轻人觉得 Git 是个好东西，就搞了一个公司名字叫 GitHub，第二年上线了使用 Ruby 编写的同名网站 GitHub，这是一个基于 Git 的免费代码托管网站（有付费服务）。十年间，该网站迅速蹿红，击败了实力雄厚的 Google Code，成为全世界最受欢迎的代码托管网站。2018 年 6 月，GitHub 被财大气粗的 Microsoft 收购。2019 年 1 月 GitHub 宣布用户可以免费创建私有仓库。根据 2018 年 10 月的 GitHub 年度报告显示，目前有 3100 万开发者创建了 9600 万个项目仓库，有 210 万企业入驻。

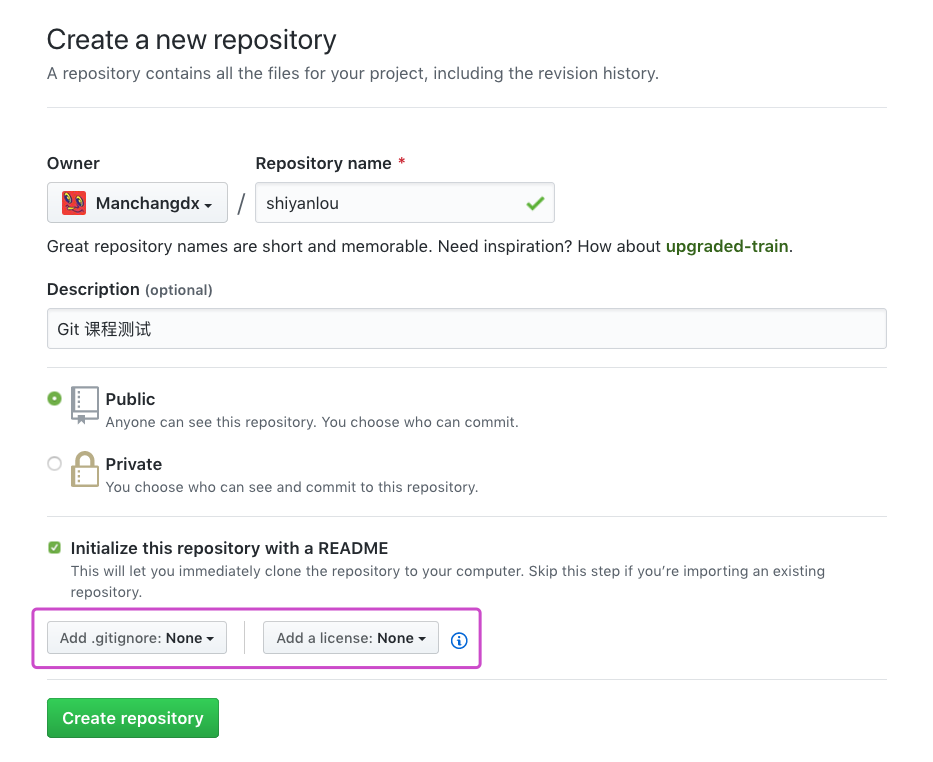
本课程将以图文的形式逐步讲解 GitHub 的使用以及 Git 实现版本控制。

### 在 GitHub 上创建仓库

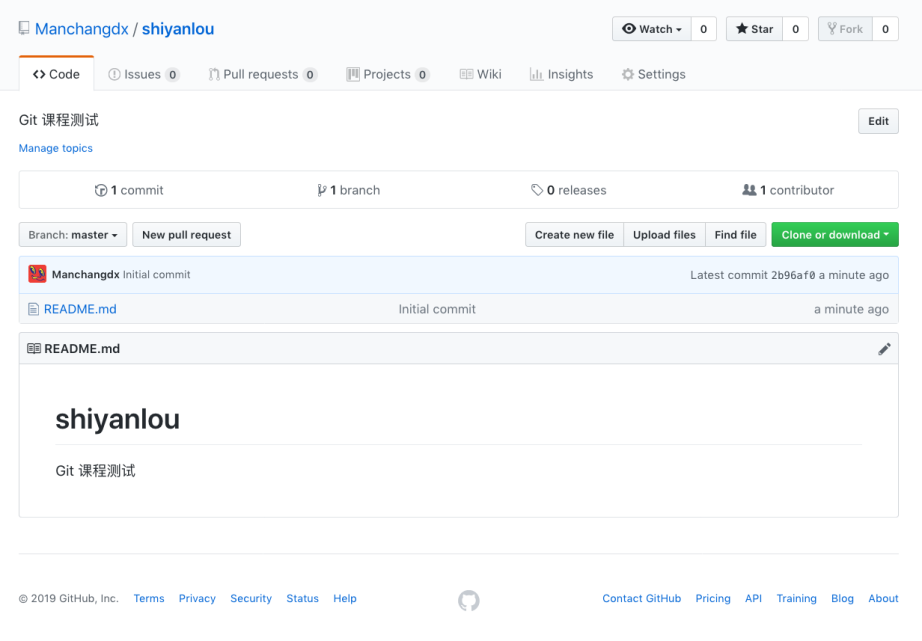
首先，打开 [GitHub](https://github.com/) 注册个人账户并登录。登录后，在个人主页的右上角点击 New repository 创建新的仓库：



打开页面如下图所示，填入相关信息。注意下图紫色框中有两个下拉按钮，左边的用来选择忽略文件，右边的用来选择所属协议，这两项可以不选，后面的课程会讲到。



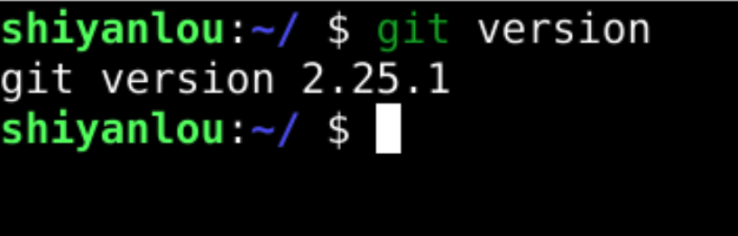
点击绿色按钮创建新的仓库，成功后自动跳转到新建仓库的主页面，如下图所示：



### 安装 Git

接下来，我们就要尝试使用这个仓库。

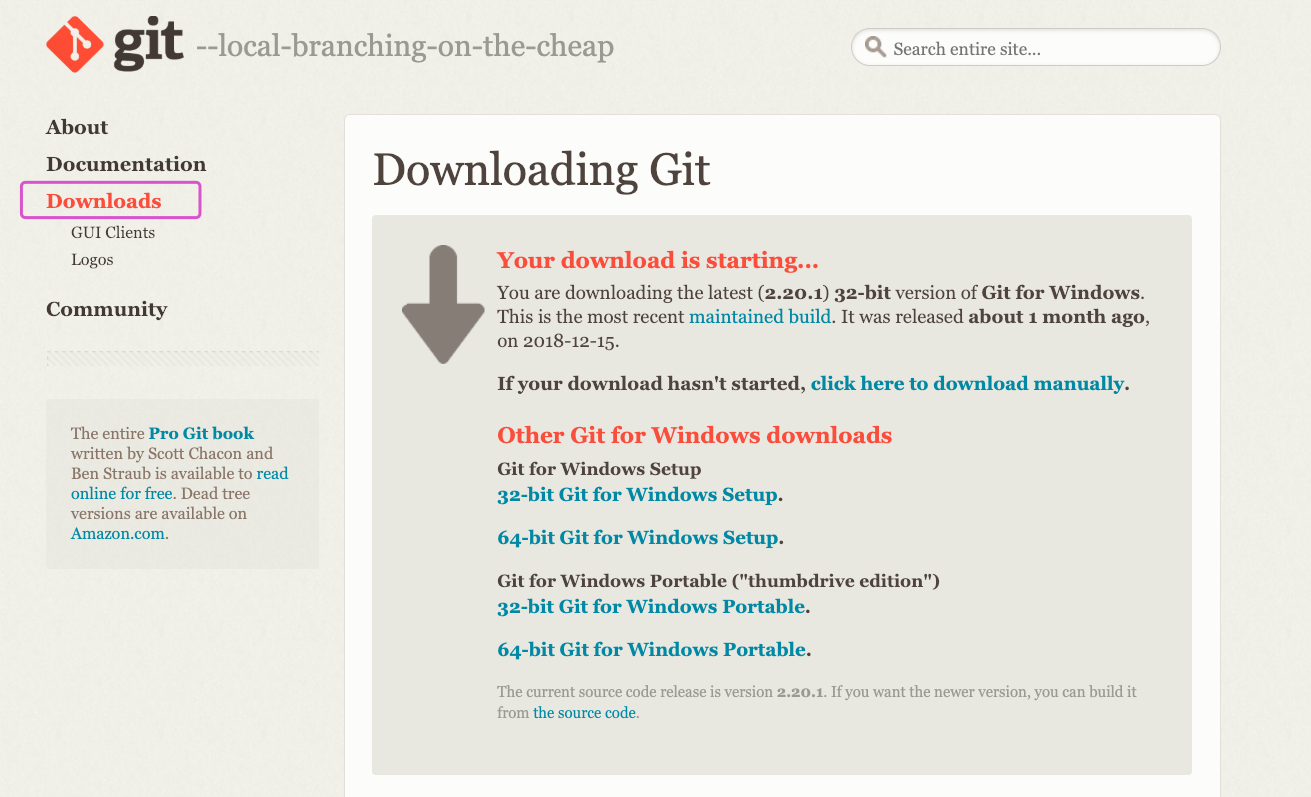
首先，打开实验环境。实验环境中内置了 Git 版本控制器，无需自己安装。打开终端使用 git version 命令查看版本：



\*注意：随着实验环境的升级 git 的版本可能会有变化，不过本课程所讲的功能在各个版本的使用中差别不大。

在 Windows 系统中可以安装 [Git for Windows 客户端](https://git-scm.com/download/win) ：

如遇网络问题无法下载可以访问 [华为云镜像站](https://mirrors.huaweicloud.com/home) 下载。



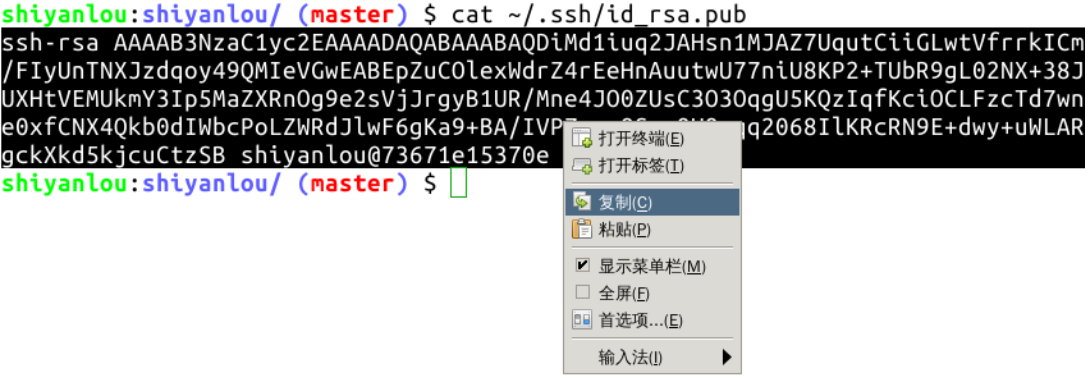
### 添加 SSH 关联授权

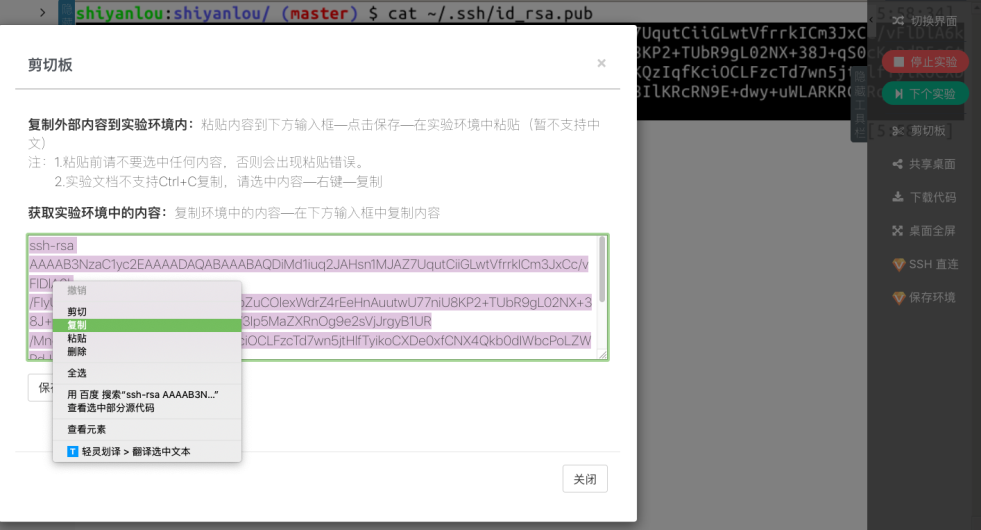
在 2021 年 8 月 GitHub 更新后，已经不再允许使用账户密码操作 GitHub，必须使用 SSH 密钥登陆。所以我们可以在系统中创建 SSH 公私钥，并将公钥放到 GitHub 指定位置。如此操作即可生成 GitHub 账户对于当前系统中的 Git 授权。

终端执行 ssh-keygen 命令按几次回车生成公私钥，公私钥存放在主目录下的隐藏目录 .ssh 中的两个文件中：

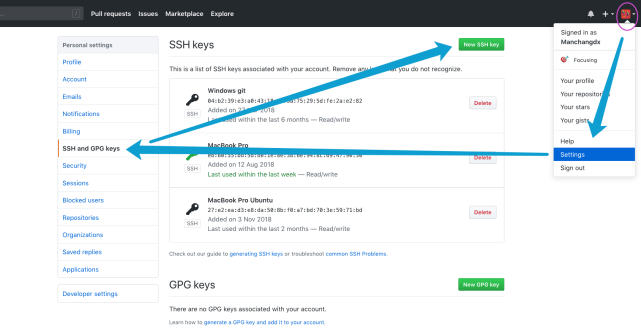


将 ~/.ssh/id\_rsa.pub 文件中的公钥内容复制出来，实验环境中可以使用右侧工具栏中的剪切板复制：

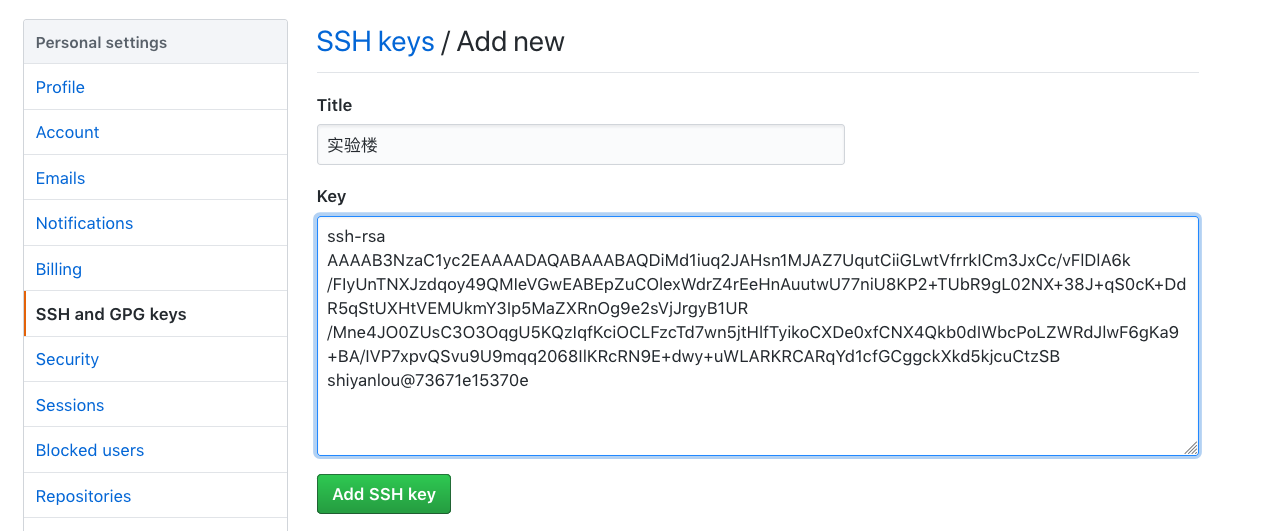




然后在 GitHub 网页上添加公钥：



Title 自定义，把剪切板中的内容粘贴到 Key 中，点击绿色按钮添加 SSH Key 即可：



使用 SSH 的好处主要有两点：

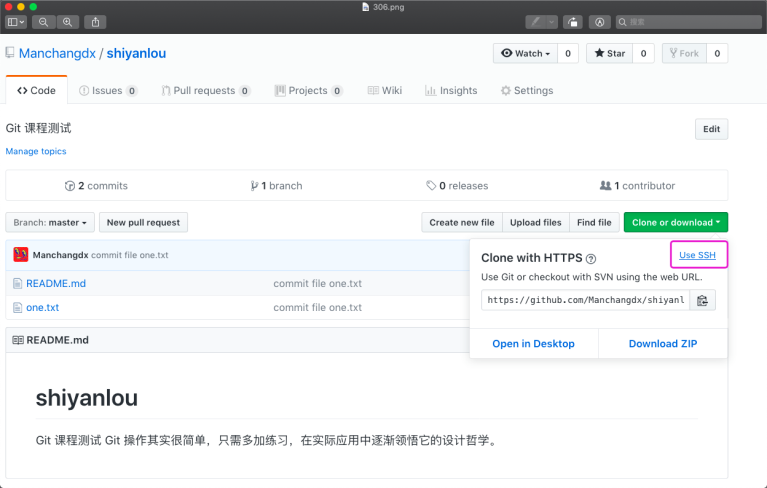
* 免密码推送，执行 git push 时不再需要输入用户名和密码了；
* 提高数据传输速度。它不是必须的，比如在课程中挑战环境是不可保存的，一次性的，这种环境就没有必要创建 SSH 了，因为相较好处来说，还是太麻烦了。

### 克隆 GitHub 上的仓库到本地

现在克隆前面我们在 GitHub 上创建的仓库，使用 git clone + [仓库地址] 命令即可，这是标准的克隆仓库命令。

回到仓库主目录，点击下图所示的绿色按钮，点击紫色框中的 “Use SSH”，然后复制这个链接。

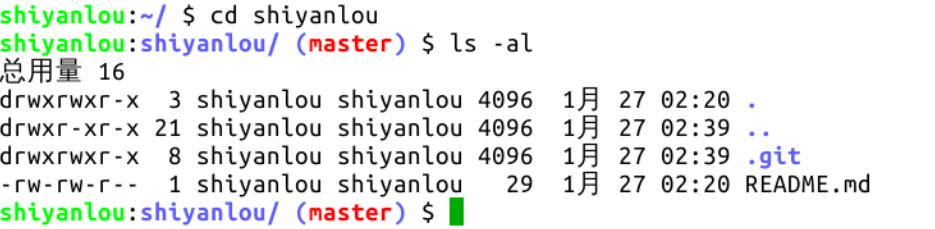
**重要的一点：只有使用这种 git 开头的地址克隆仓库，SSH 关联才会起作用。**



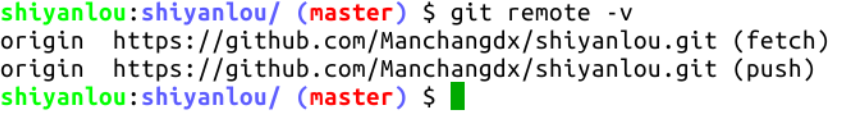
在实验环境里删除原仓库，使用此链接重新克隆仓库。克隆仓库是需要确认连接，输入 yes 即可：



进入仓库主目录，如下图所示，仓库主目录中有个 .git 隐藏目录，它里面包含了仓库的全部信息，删掉这个目录，仓库就变成普通的目录了。进入到仓库目录中，命令行前缀发生了一些变化，出现了红色的 master ，它就是当前所在的分支名：

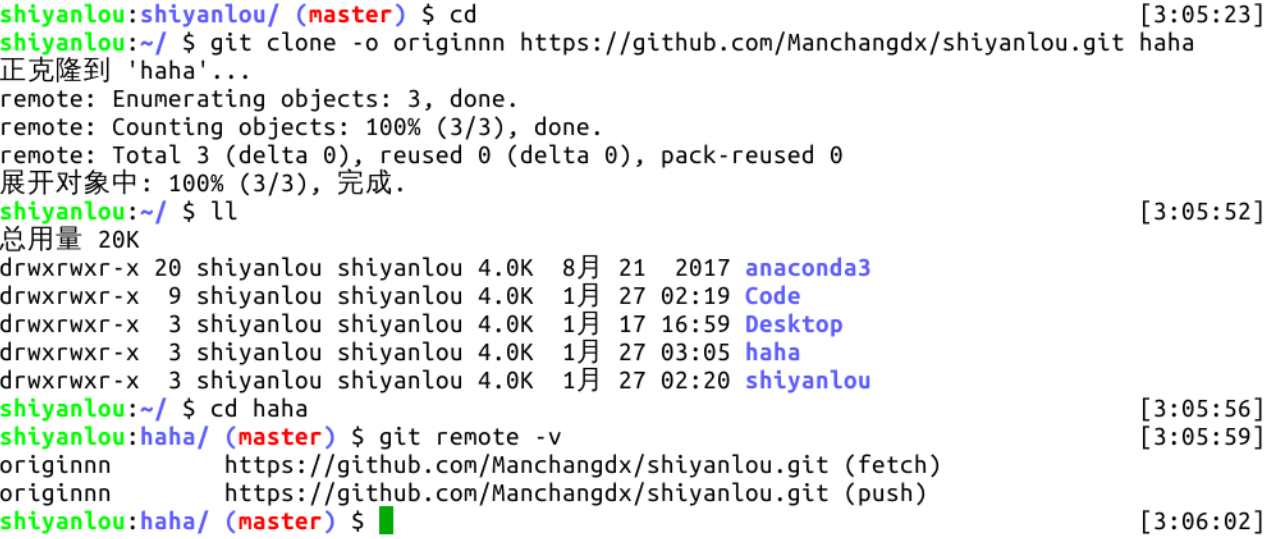


当我们在 GitHub 上创建一个仓库时，同时生成了仓库的默认主机名 origin，并创建了默认分支 master。GitHub 可以看成是免费的 Git 服务器，在 GitHub 上创建仓库，会自动生成一个仓库地址，主机就是指代这个仓库，主机名就等于这个仓库地址。克隆一个 GitHub 仓库（也叫远程仓库）到本地，本地仓库则会自动关联到这个远程仓库，执行 git remote -v 命令可以查看本地仓库所关联的远程仓库信息：



Git 要求对本地仓库关联的每个远程主机都必须指定一个主机名（默认为 origin），用于本地仓库识别自己关联的主机，git remote 命令就用于管理本地仓库所关联的主机，一个本地仓库可以关联任意多个主机（即远程仓库）。

克隆远程仓库到本地时，还可以使用 -o 选项修改主机名，在地址后面加上一个字段作为本地仓库的主目录名，举例如下：



另一个在其它 Git 教程中常见的命令 git init ，它会把当前所在目录变成一个本地仓库，因为有 GitHub 的存在，这个命令在我们的生产生活中用到的次数应该是零，除非你想费时费力自己搭建服务器。操作截图如下：

