版本控制工具GitHub的使用

[Git](http://www.oschina.net/p/git) 是一个分布式的版本控制系统，最初由Linus Torvalds编写，用作Linux内核代码的管理。在推出后，Git在其它项目中也取得了很大成功，尤其是在Ruby社区中。目前，包括[Rubinius](http://www.oschina.net/p/rubinius)和[Merb](http://www.oschina.net/p/merb)在内的很多知名项目都使用了Git。Git同样可以被诸如Capistrano和Vlad the Deployer这样的部署工具所使用。github是一个基于git的代码托管平台，付费用户可以建私人仓库，我们一般的免费用户只能使用公共仓库，也就是代码要公开。对于一般人来说公共仓库就已经足够了，而且我们也没多少代码来管理，O(∩\_∩)O~。下面是我总结的一些简单使用方法，供初学者参考。

Github的网址如下https://github.com,在该网址我们需要注册一个账户，注册用户名和密码选择你常用的邮箱。

At the heart of GitHub is an open source version control system (VCS) called [*Git*](https://help.github.com/articles/github-glossary#git). Git is responsible for everything GitHub-related that happens locally on your computer.

If you're not comfortable using the command line right now, GitHub lets you complete many Git-related actions *without* using the command line, including:

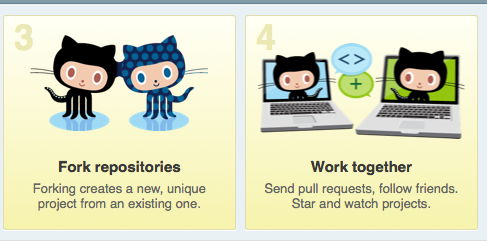
* [Creating a repository](https://help.github.com/articles/create-a-repo)
* [Forking a repository](https://help.github.com/articles/fork-a-repo)
* [Being social](https://help.github.com/articles/be-social)

However, if you find that you need to use Git, we can help you set it up!

**Tips:**

* GitHub has a [Mac](https://mac.github.com/) client! You can use it without ever touching the command line.
* To learn more about Git, see "[Getting Started - Git Basics](http://git-scm.com/book/en/Getting-Started-Git-Basics)" on the [git-scm](http://git-scm.com/) website.



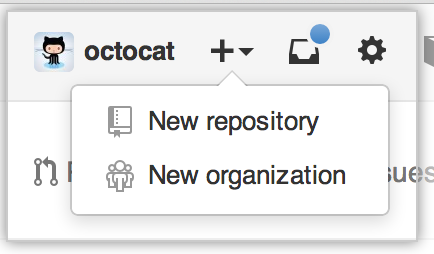
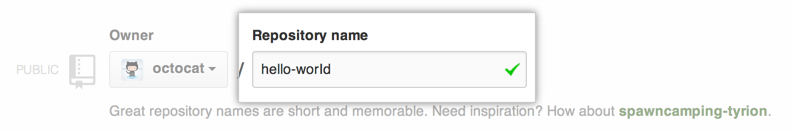
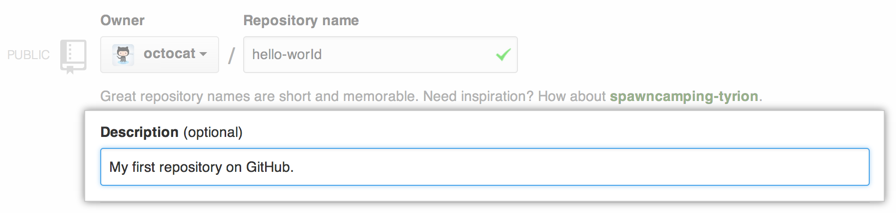
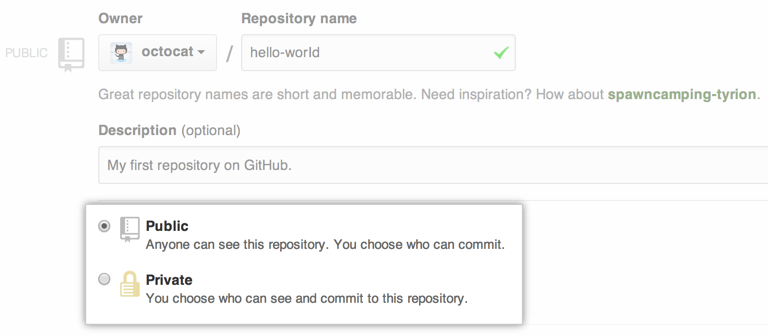
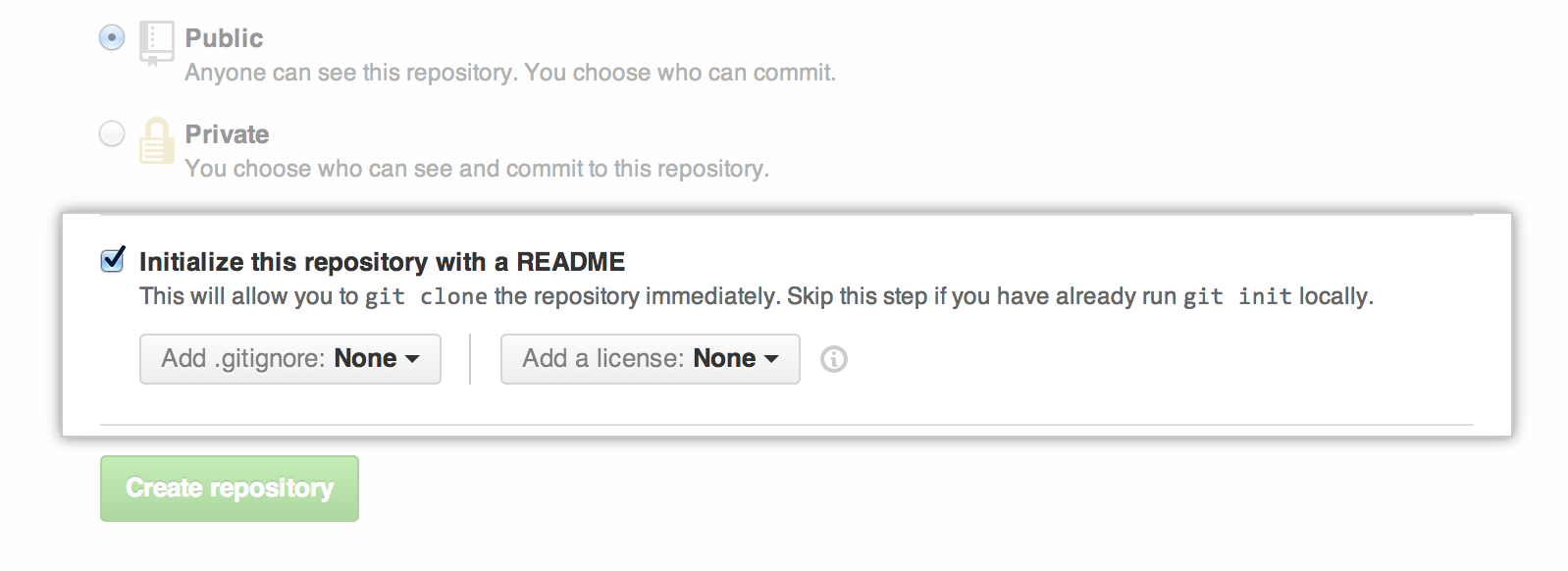
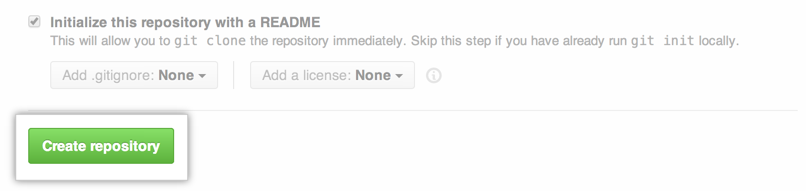


**Create A Repo**

To put your project up on GitHub, you'll need a repository for it to live in. Let's create one!

On GitHub, you can store all kinds of projects in [*repositories*](https://help.github.com/articles/github-glossary#repository). Personal repositories belong to *user accounts*, so after you've [signed up for GitHub](https://help.github.com/articles/signing-up-for-a-new-github-account), you can create your first repository!

### Create a new repository on GitHub

1. In the upper-right corner of any page, click **+**, and then click **New repository**.
2. Create a short, memorable name for your repository. For example, "hello-world".
3. Optionally, add a description of your repository. For example, "My first repository on GitHub."
4. Choose between creating a public or private repository.
   * **Public** repositories are a great choice for getting started! They're free, and visible to any user on GitHub, so you can benefit from GitHub's collaborative community.
   * **Private** repositories require a little more setup. They're paid, and only available to you, the repository owner, as well as any collaborators you choose to share with. For more information, see "[What plan should I choose?](https://help.github.com/articles/what-plan-should-i-choose)."
5. Select **Initialize this repository with a README**.
6. Click **Create repository**.

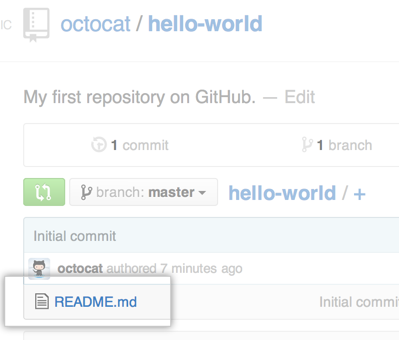
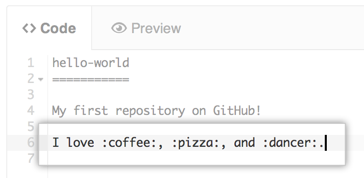
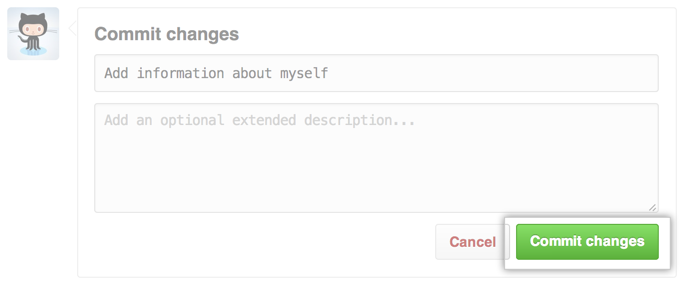
Congratulations! You've successfully created your first repository, and initialized it with a README file.

### Commit your first change

A [*commit*](https://help.github.com/articles/github-glossary#commit) is like a snapshot of all the files in your project at a particular point in time.

When you created your new repository, you initialized it with a README file. README files are a great place to describe your project in more detail, or add some documentation such as how to install or use your project. The contents of your README file are automatically shown on the front page of your repository.

Let's commit a change to the README file.

1. In your repository's list of files, click **README.md**.
2. Above the file's content, click .
3. In the Code view, type some information about yourself.
4. Above the new content, click **Preview**.
5. Review the changes you made to the file. You'll see the new content in green.
6. At the bottom of the page, under "Commit changes," type a short, meaningful commit message that describes the change you made to the file.
7. Click **Commit changes**.

以上就是在Github的网站创建一个新的代码仓库，下面我们将讲解客户端如何使用。首先客户端有了成熟的工具，我们不需要使用任何命令，直接通过界面就可以进行管理。

首先，安装该工具Github，安装完成之后，第一次需要登录你的Github账户，相当于生成一个sshkey的过程