

## Set Up Git

If you've found yourself on this page, we're assuming you're brand new to Git and GitHub. This guide will walk you through the basics and explain a little bit about how everything works along the way.

This is the guide for setting up git in Windows. There are also guides for [OSX](#) and [Linux](#).

### First: Download and Install Git

At the heart of GitHub is an open source version control system (VCS) called Git\*. Created by the same dudes that created Linux, Git is responsible for everything GitHub related that happens locally on your computer.

*\*If you don't already know what Git is, [take a crash course](#).*

1. Download and install the latest version of [Git for Windows](#).

Use the default options for each step.



Beginner
▶ Set Up Git
<a href="#">Create A Repo</a>
<a href="#">Fork A Repo</a>
<a href="#">Be Social</a>
<a href="#">Delete a repo</a>
<a href="#">Move a repo</a>
<a href="#">Leave a collaborative repo</a>
<a href="#">Send pull requests</a>
<a href="#">Set your user name, email and GitHub token</a>
Intermediate
Advanced
Troubleshooting
Third party
GitHub Resources
Git resources
Site policy

This website is [open source](#). Please help us by forking the project and adding to it.

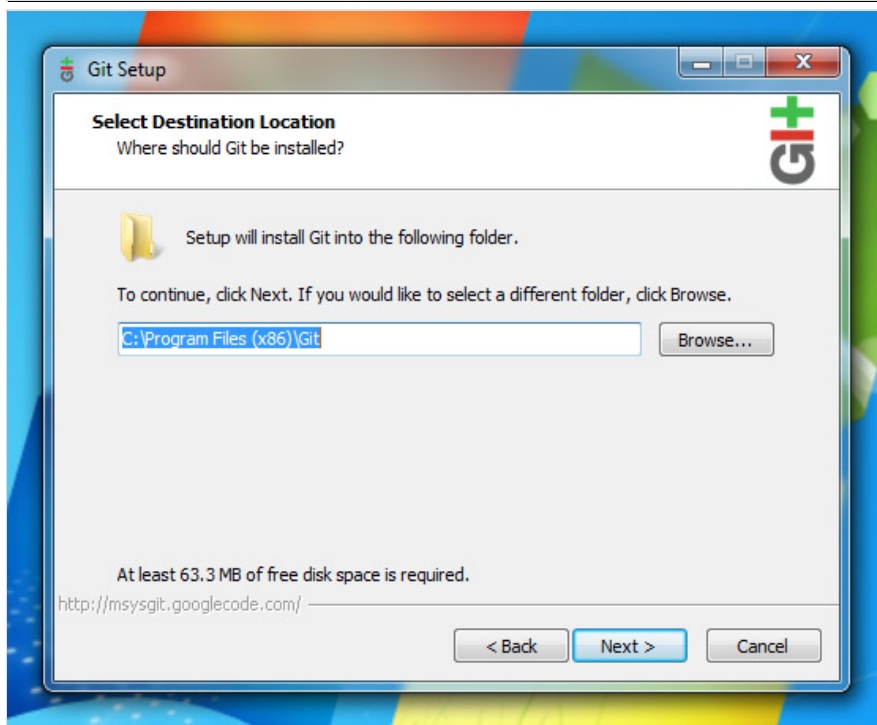
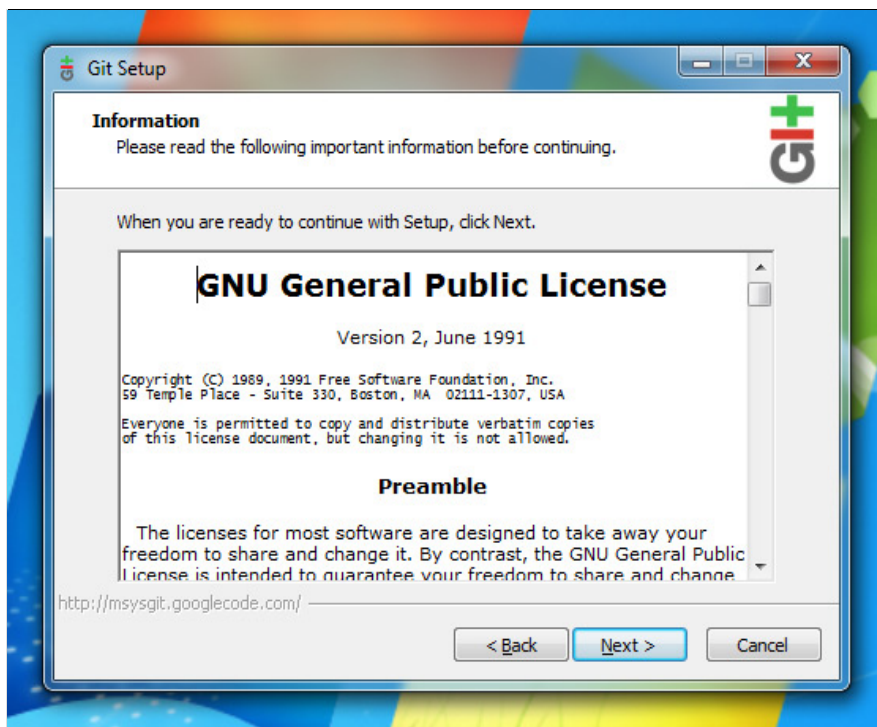
Site Status:

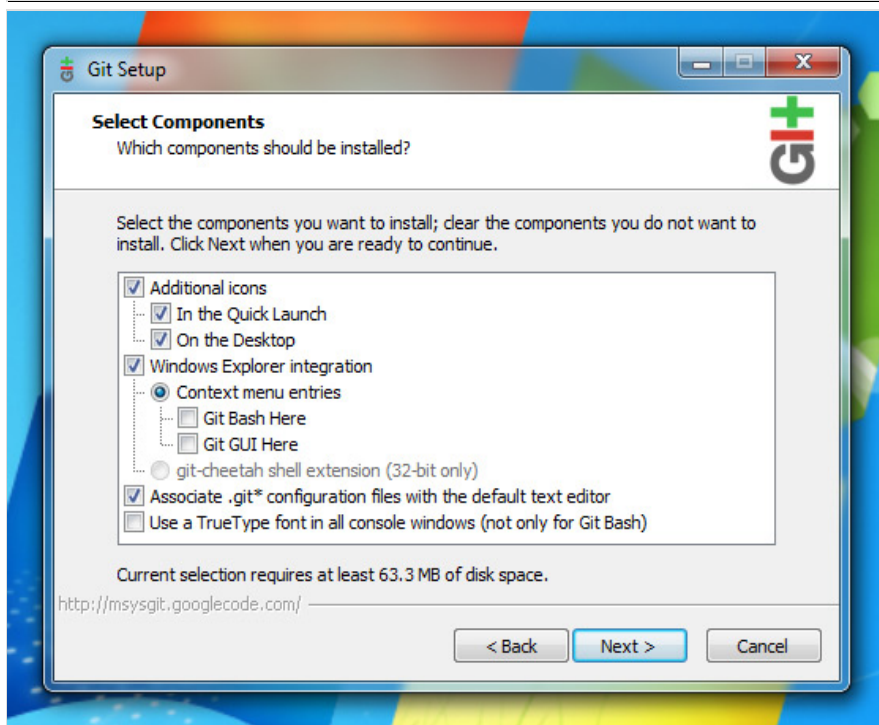
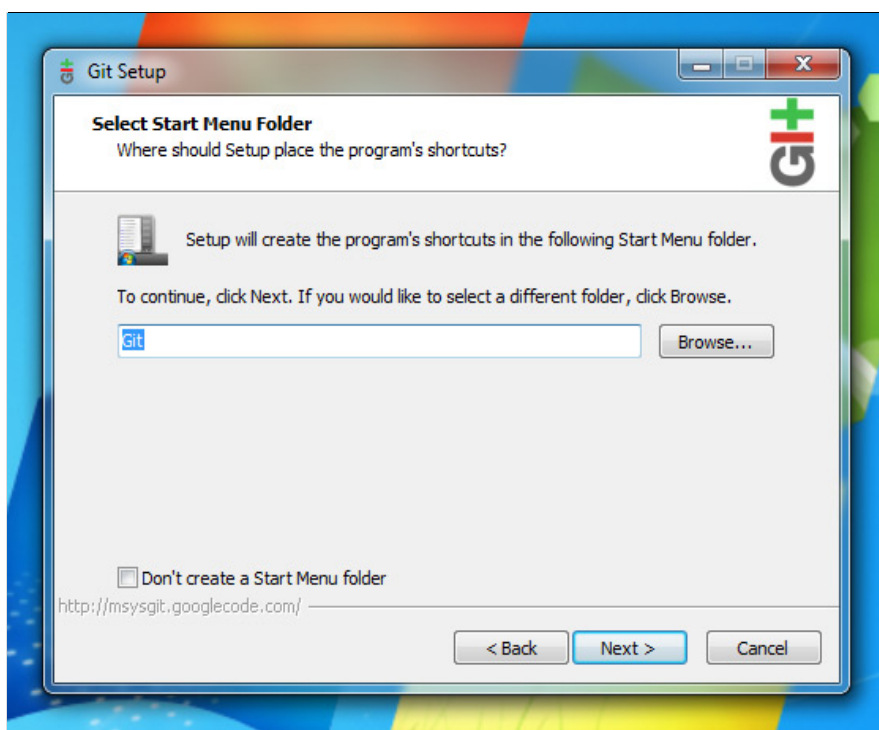
All systems operational

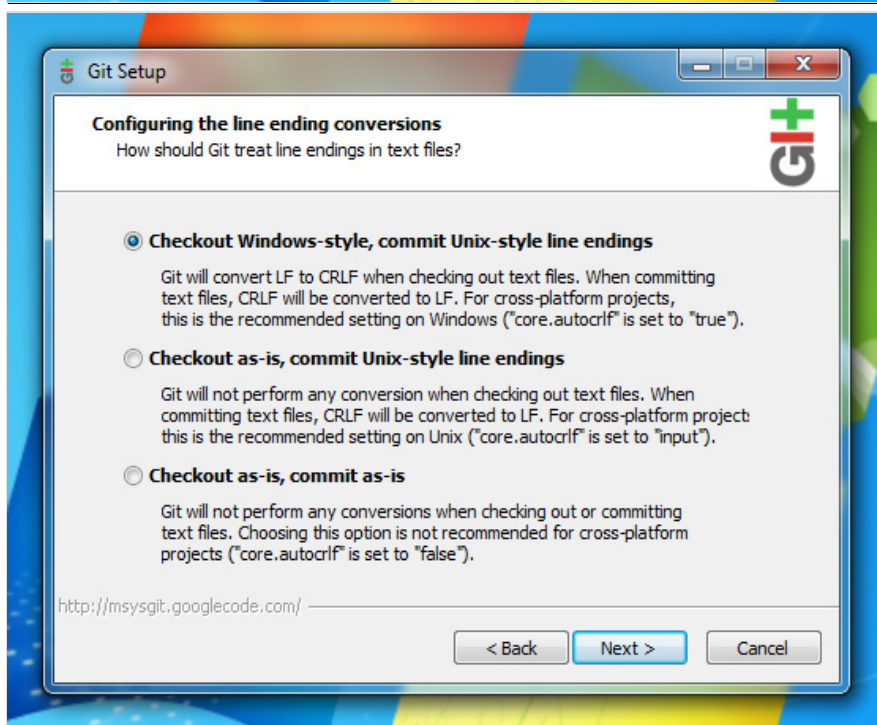
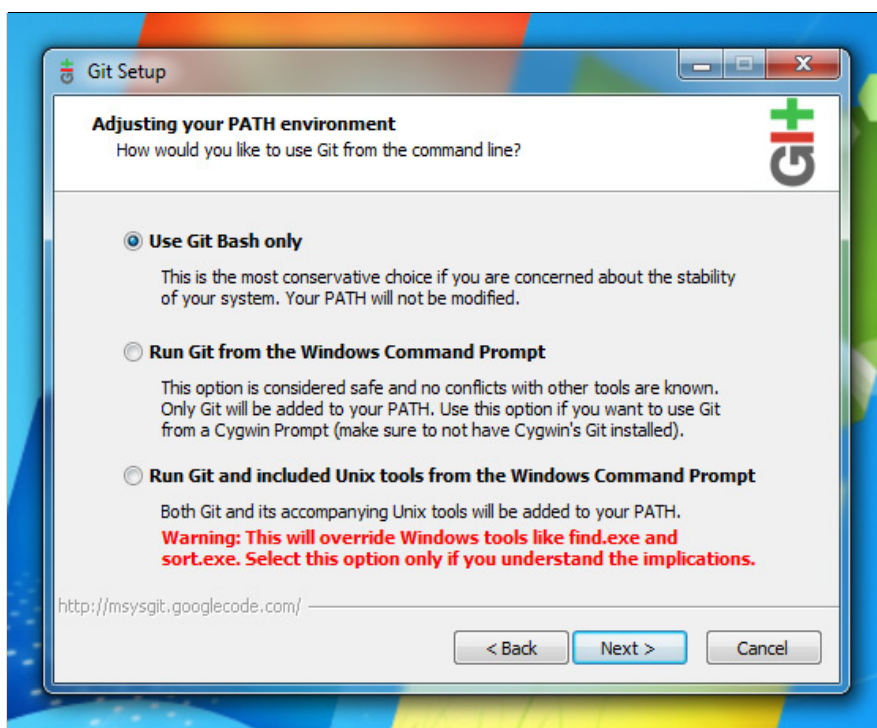
GitHub Tips

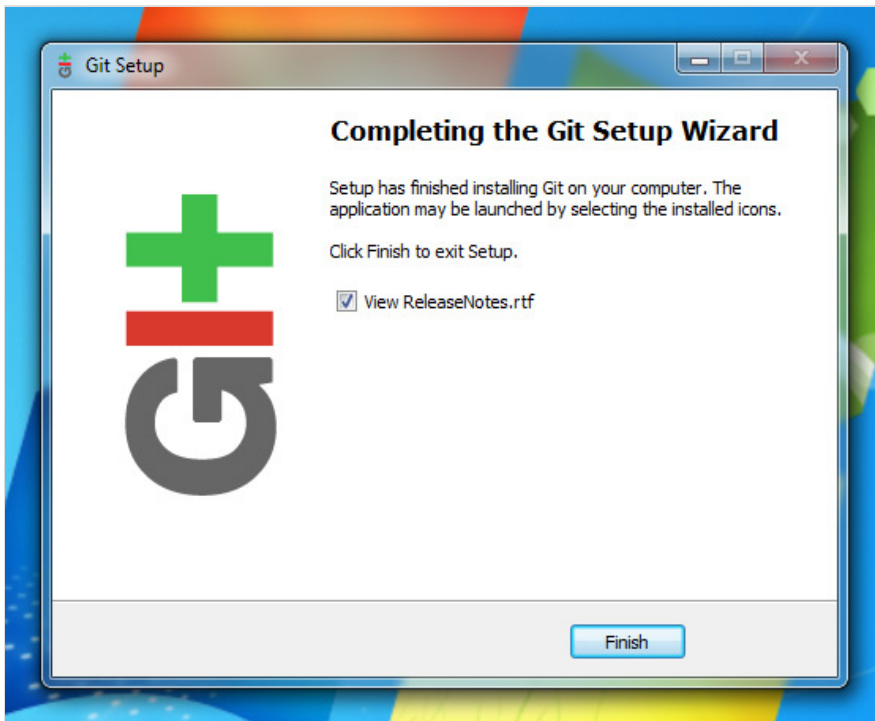
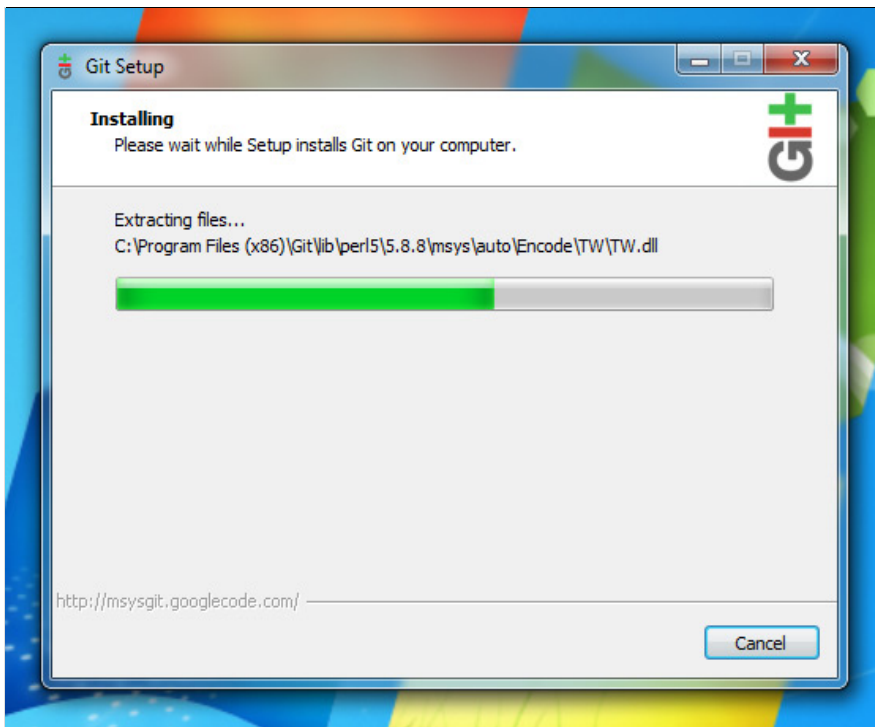
Next Tip

You can find unmerged branches locally with `git branch --no-merged`







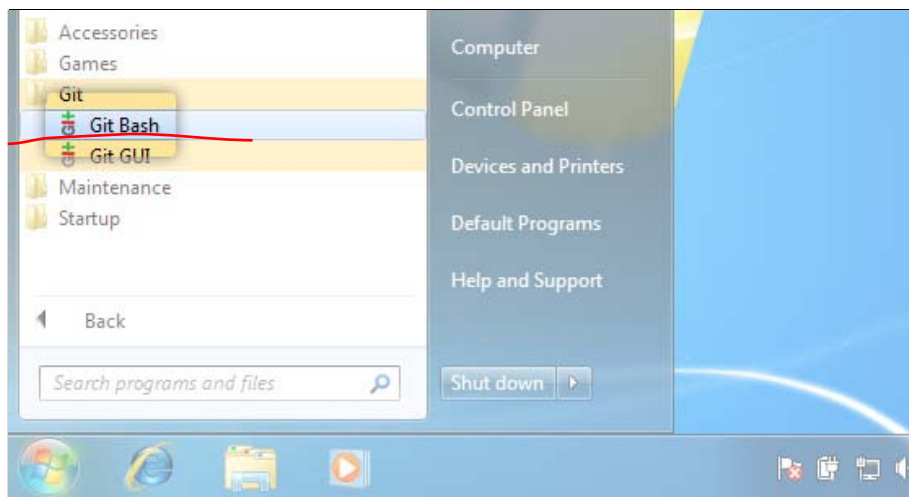


Do not use PuTTY if you are given the option. GitHub only provides support for openssh.

## Next: Set Up SSH Keys

We use SSH keys to establish a secure connection between your computer and GitHub. Setting them up is fairly easy, but does involve a number of steps.

To make sure you generate a brand new key, you need to check if one already exists. First, you need to open Git Bash (not the Windows command line), found in the start menu in the git.



[Need a quick lesson about Git Bash?](#)

1. Check for SSH keys. *Have an existing key pair? You can skip to Step 4.*

First, we need to check for existing ssh keys on your computer:

```
$ cd ~/.ssh
```

If it says "No such file or directory" skip to step 3. Otherwise continue to step 2.

2. Backup and remove existing SSH keys.

Since there is already an SSH directory you'll want to back the old one up and remove it:

```
$ ls
config id_rsa id_rsa.pub known_hosts
$ mkdir key_backup
$ cp id_rsa* key_backup
$ rm id_rsa*
```

3. Generate a new SSH key.

To generate a new SSH key, enter the code below. We want the default settings so when asked to enter a file in which to save the key, just press enter.

```
$ ssh-keygen -t rsa -C "your_email@youremail.com"
Generating public/private rsa key pair.
Enter file in which to save the key
(/Users/your_user_directory/.ssh/id_rsa):<press enter>
```

Now you need to enter a passphrase.

[Why do passphrases matter?](#)

```
Enter passphrase (empty for no passphrase):<enter a passphrase>
Enter same passphrase again:<enter passphrase again>
```

Which should give you something like this:

```
Your identification has been saved in
/Users/your_user_directory/.ssh/id_rsa.
Your public key has been saved in
/Users/your_user_directory/.ssh/id_rsa.pub.
The key fingerprint is:
01:0f:f4:3b:ca:85:d6:17:a1:7d:f0:68:9d:f0:a2:db user_name@username.com
The key's randomart image is:
+--[ RSA 2048 ]-----+
|      .+      +      |
|     = o O *   +      |
|    o = +     +      |
|   o S .      +      |
|  o o =       +      |
| o . E        +      |
+-----+-----+
+-----+-----+
```



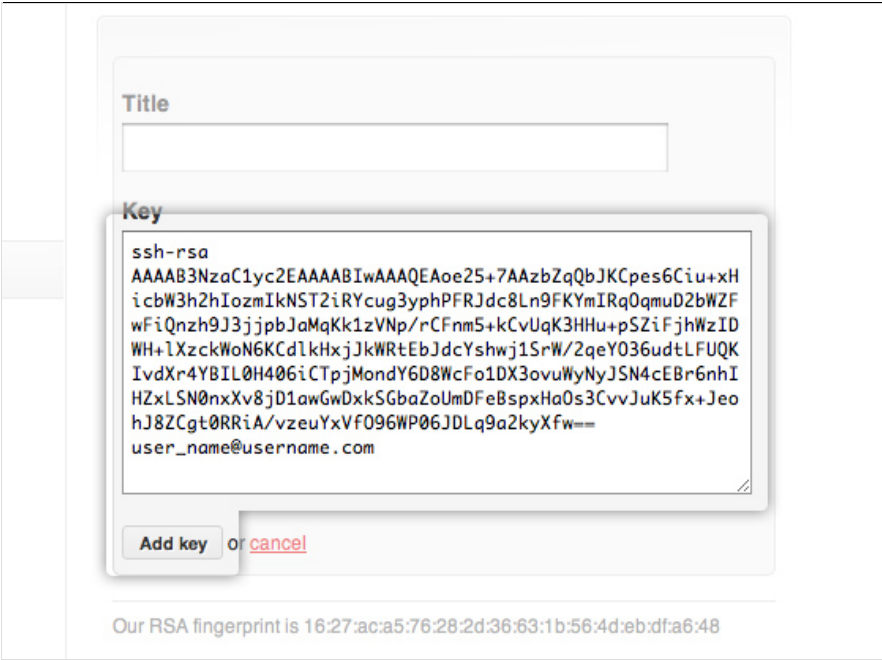
4. Add your SSH key to GitHub.

On the GitHub site Click “Account Settings” > Click “SSH Public Keys” > Click “Add another public key”

Open the id\_rsa.pub file with a text editor (Notepad, TextEdit, or gedit will do just fine). This is your public SSH key. You may need turn on “view hidden files” to find it because the .ssh directory is hidden. *It's important you copy your SSH key exactly as it is written without adding any newlines or whitespace.* Now paste it into the “Key” field.

 [Can't view hidden files? Other ways to copy:](#)

Now paste it into the “Key” field.



Hit “Add Key.”

5. Test everything out.

To make sure everything is working you'll now SSH to GitHub. *Don't change the "git@github.com" part.* That's supposed to be there.

```
$ ssh -T git@github.com
```

Which should give you this:

```
The authenticity of host 'github.com (207.97.227.239)' can't be
established.
RSA key fingerprint is 16:27:ac:a5:76:28:2d:36:63:1b:56:4d:eb:df:a6:48.
Are you sure you want to continue connecting (yes/no)?
```

Don't worry, this is supposed to happen. Type “yes”.

```
Hi username! You've successfully authenticated, but GitHub does not
provide shell access.
```

 [Having problems?](#)

Then: Set Up Your Info

Now that you have Git set up and your SSH keys entered into GitHub, it's time to configure your personal info.

1. Set your username and email.

Git tracks who makes each commit by checking the user's name and email. In addition, we use this info to associate your commits with your GitHub account. To set these, enter the code below, replacing the name and email with your own. The name should be your *actual name*, not your GitHub username.

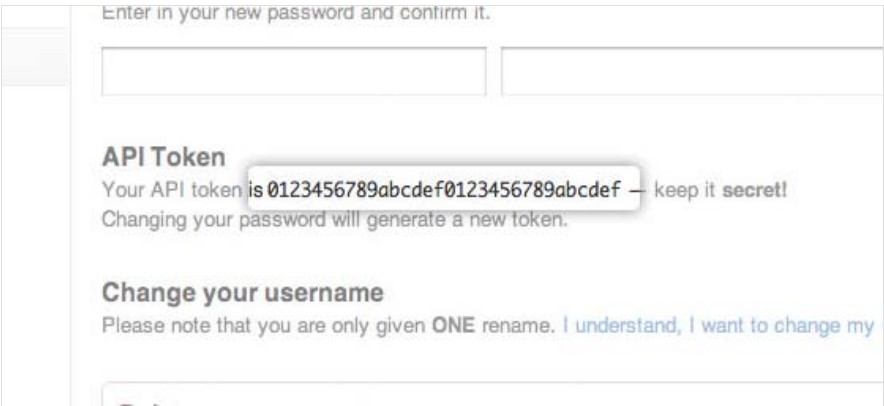
```
$ git config --global user.name "Firstname Lastname"
$ git config --global user.email "your_email@youreemail.com"
```

[More about user info](#)

2. Set your GitHub token.

Some tools connect to GitHub without SSH. To use these tools properly you need to find and configure your API Token.

On the GitHub site Click "Account Settings" > Click "Account Admin."



At the command line run the following code, using your GitHub username and token in place of the ones shown.

```
$ git config --global github.user username
$ git config --global github.token 0123456789yourf0123456789token
```


\*Note\* If you ever change your GitHub password, a new token will be created and will need to be updated.

Lastly: Celebrate

Congratulations, you now have Git and GitHub

- 1. Set Up Git
- 2. [Create A Repository](#)
- 3. [Fork A Repository](#)
- 4. [Be Social](#)

注意：这样设置之后，只能使用git bash进行命令行操作，但是TortoiseGit无法使用，需要运行puttygen.exe，选择菜单“转换”->“导入密钥”（注意导入私钥id\_rsa，不是公钥）然后菜单“文件”->“保存私钥”（比如putty.ppk），打开TortoiseGit Setting界面Git->Remote，将刚刚生成的putty.ppk文件作为Putty Key。看下面的图

	<a href="#">GitHub</a> <a href="#">About</a> <a href="#">Blog</a> <a href="#">Contact &amp; Support</a> <a href="#">Training</a> <a href="#">Site Status</a>	<a href="#">Tools</a> <a href="#">GitHub for Mac</a> <a href="#">Gist</a> <a href="#">Enterprise Install</a> <a href="#">Job Board</a>	<a href="#">Extras</a> <a href="#">Shop</a> <a href="#">The Octodex</a>	<a href="#">Documentation</a> <a href="#">GitHub Help</a> <a href="#">Developer API</a> <a href="#">GitHub Flavored Markdown</a> <a href="#">GitHub Pages</a>
--	---	--	---	---

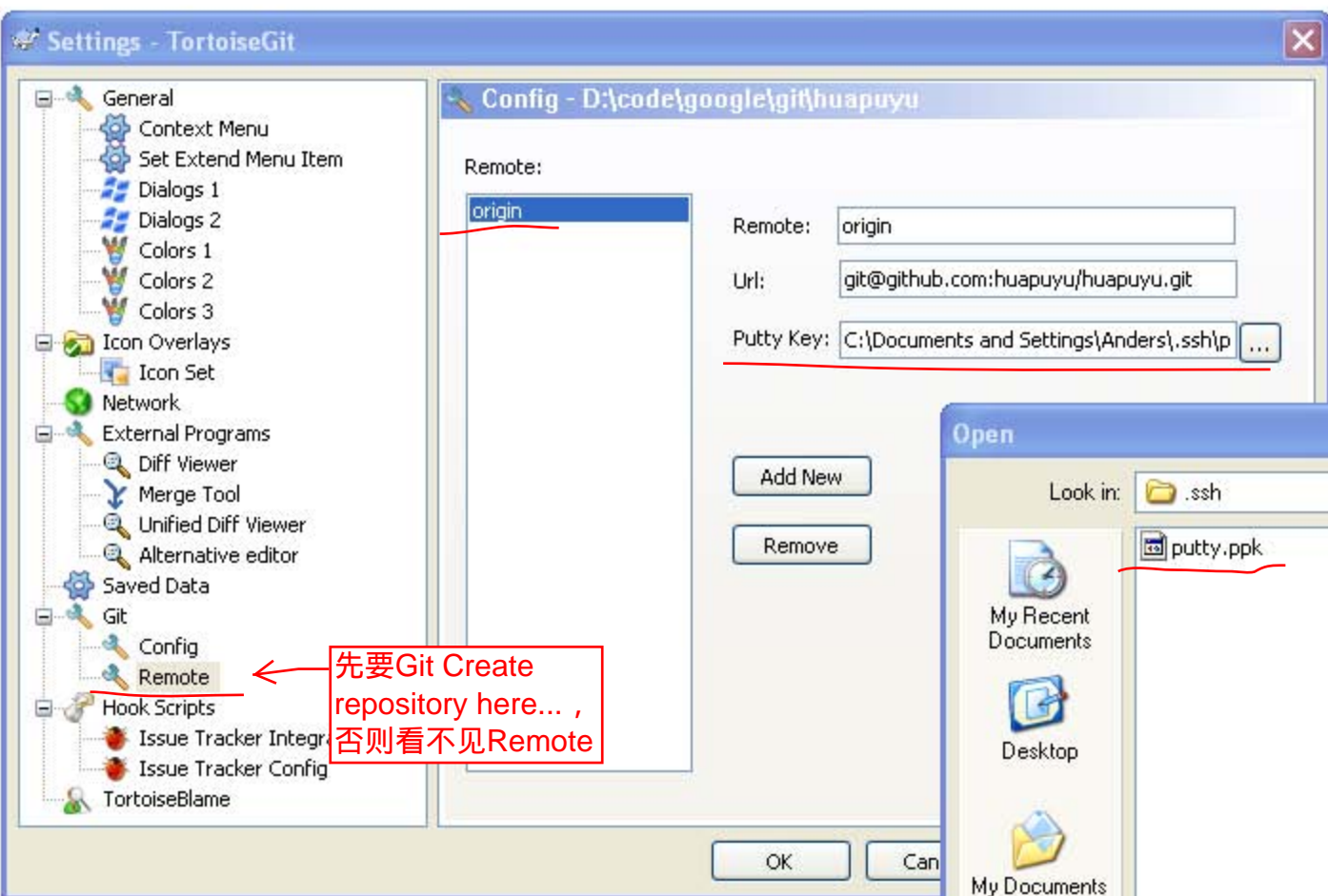




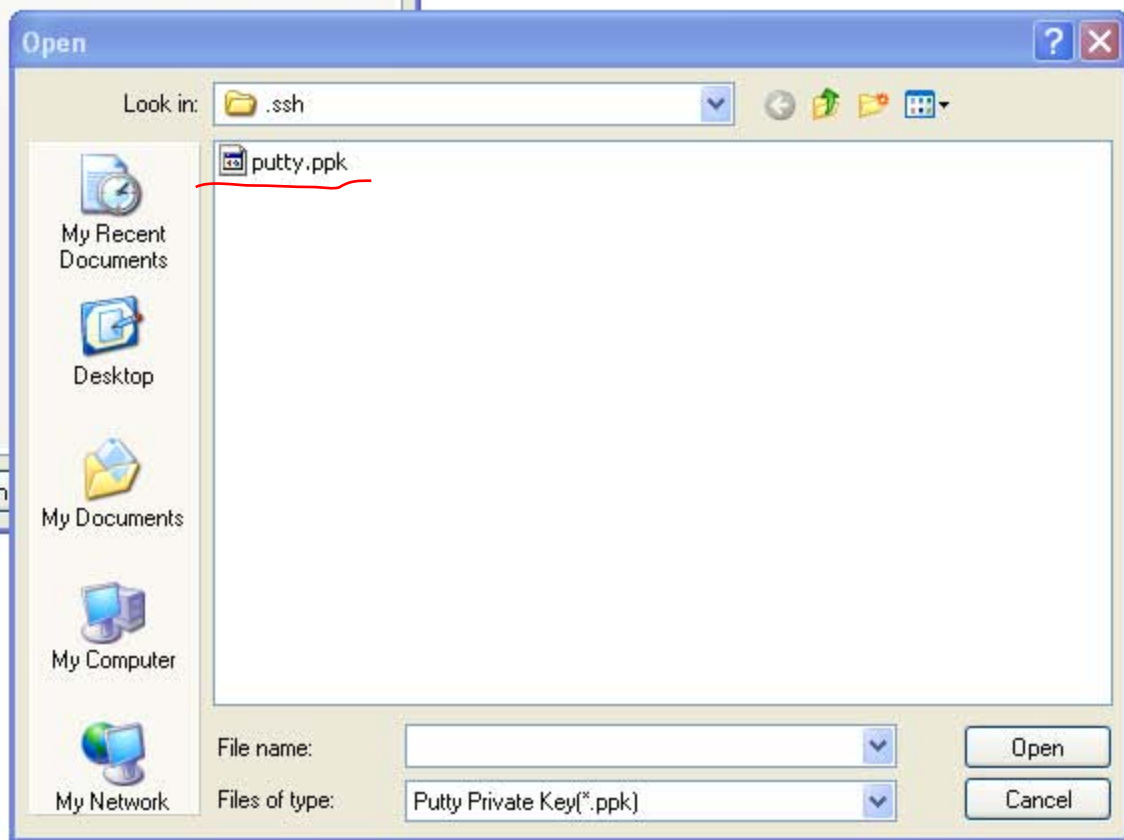
[Terms of Service](#) [Privacy](#) [Security](#)  
© 2011 GitHub Inc. All rights reserved.



Powered by the **Dedicated Servers** and  
**Cloud Computing** of Rackspace Hosting®



← 先要Git Create repository here... , 否则看不见Remote





# Preferences

type filter text

- General
- + Appearance
- Capabilities
- Compare/Patch
- Content Types
- + Editors
- Keys
- Network Connections
  - Cache
  - SSH2
  - Perspectives
  - Search
- + Security

## SSH2

Configuration options for SSH2:

General

Key Management

Known Hosts

Authentication Methods

SSH2 home: C:\Documents and Settings\Anders\.ssh

Browse...

Private keys: id\_dsa,id\_rsa

Add Private Key...

在Eclipse中使用git时一定要把这里改为.ssh目录，否则eclipse自动会生成ssh目录，其中并没有前面创建的公钥和私钥，所以访问会失败。

OK

Cancel