http://www.360doc.com/content/12/0421/09/1016783\_205350218.shtml

我写这篇文章的目的是帮助没有用过gihub和jekyll的童鞋尽快掌握使用流程。学习使用github网页的最好办法就是clone别人的代码，看懂他们的代码，并修改成自己喜欢的样子。这篇文章介绍了从最初安装软件到使用的过程。

先假设读者对ruby完全没有了解，仅对html有初步的了解（<title>等标签知道是什么意思，如果懂点儿CSS更好）。基本的linux命令（ls、cd、mkdir、rm等，不会也没关系，就用到这几个命令，自己百度一下就好）。

本文标题所说的“极简”是基于下面的原则：

* 安装尽量少的软件
* 使用尽量少的命令
* 接触尽量少的概念

如果想了解用Github Pages写博客的理念，可以看这篇文章

* [《理想的写作环境：git+github+markdown+jekyll》](http://www.yangzhiping.com/tech/writing-space.html)

下面开始一步步讲解Github Pages的使用流程：

**一、**如果想在本地预览页面，跳过这步到[二](http://www.360doc.com/content/12/0421/09/1016783_205350218.shtml)；如果不想在本地预览，则在windows下安装[msysgit](http://code.google.com/p/msysgit/downloads/list" \t "_blank)，最新版本为Git-1.7.9-preview20120201.exe，然后按照[BeiYuu的博文](http://beiyuu.com/github-pages/" \l "github" \t "_blank)里的过程配置git和github，再到[四](http://www.360doc.com/content/12/0421/09/1016783_205350218.shtml)

如果你的系统是linux的，按照[Git Hub官方帮助文件](http://help.github.com/linux-set-up-git/" \t "_blank)操作，然后跳到[四](http://www.360doc.com/content/12/0421/09/1016783_205350218.shtml)。

如果想深入了解Git，请看[10篇写给Git初学者的最佳教程](http://www.kuqin.com/managetool/20110705/92113.html" \t "_blank)。

**二、在windows下安装ruby环境**。推荐安装[RailsInstaller](http://railsinstaller.org/)，里面包含了[Ruby](http://ruby-lang.org/" \t "_blank)、[Rails](http://rubyonrails.org/" \t "_blank)、[Bundler](http://gembundler.com/" \t "_blank)、[Git](http://git-scm.com/" \t "_blank)、[Sqlite](http://sqlite.org/" \t "_blank)、[TinyTDS](https://github.com/rails-sqlserver/tiny_tds" \t "_blank)、[SQL Server support](https://github.com/rails-sqlserver/activerecord-sqlserver-adapter" \t "_blank)和[DevKit](https://github.com/oneclick/rubyinstaller/wiki/Development-Kit" \t "_blank)。

不过最近的RailsInstaller里包含的ruby版本升到了1.9.3，如果以后要使用[Octopress](http://www.octopress.org" \t "_blank)的话必须使用ruby1.9.2，建议使用以前的版本，我把以前的版本放到了[这里](http://yanping.me/files/upload/railsinstaller-2.0.1.exe)。

**三、紧接着上一步，配置git和github**

在RailsInstaller安装结束时安装程序会提示是否配置Git环境（这样的话给配置git和github带来极大的方便，又减少了几条命令）。选择”是”，然后出现下面的提示

填写github注册时的用户名和邮箱，就完成了公钥和密钥的生成，在C:\Documents and Settings\用户名下，有个隐藏目录名为.ssh，id\_rsa.pub文件就是公钥，id\_rsa就是密钥。

在Github网站找到 “Account Settings” > Click “SSH Keys” > Click “Add SSH key”

用文本编辑器打开id\_rsa.pub文件，并把里面的内容（包括空格和新行）复制出来，填到Github的账户设置的SSH设置里。

在开始菜单里找到RailsInstaller –> Git Bash，执行它，就打开了下面的命令窗口，以后的操作都是在这个窗口下进行的

**注意**: 在窗口里我们可以看到当前路径显示为/c/Sites，其实它表示的是C:\Sites，这个目录是RailsInstaller在安装的时候建的。

用下面的命令测试一下git是否连接正常

ssh -T git@github.com

想了解普通配置方法的，可进一步阅读：

* Github上的[帮助](http://help.github.com/set-up-git-redirect" \t "_blank)
* [BeiYuu的博文](http://beiyuu.com/github-pages/#github)
* [window下建立github连接](http://www.pizn.me/2011/10/02/start-github-on-window.html)

**四、安装jekyll和相关的包**

稍微对配置做一下修改，把淘宝的镜像加到gem的镜像列表里

gem sources --remove http://rubygems.org/

gem sources -a http://ruby.taobao.org/

然后用gem sources -l看看现在源列表

\*\*\* CURRENT SOURCES \*\*\*

http://ruby.taobao.org

如果是上面这样就可以安装jekyll了

gem install jekyll

Jekyll需要用到directory\_watcher、liquid、open4、maruku和classifier这几个包，用上面的命令 可以自动安装。Jekyll默认用maruku来解析markdown语言，你也可以用别的程序来解析，比如rdiscount或kramdown，都给 装上吧：

gem install rdiscount kramdown

以上命令涉及到gem install的时候，如果你用的是linux系统，就要用sudo gem install代替。

**五、建立github pages**。

这一步是本文的重点，也是本文异于网络上其他文章的地方，我在这里用到了Github提供的Github pages generator的功能，减少了使用的命令数量，也绕开了远程代码库这个概念（省略了与git remote相关的操作，不过随着github使用的加深，这些概念也是不能避免的）

* 在github.com上创建代码库，比如新建一个名为example的代码库：登录到自己的Github账户，选择New repository
* 在线生成pages: 点上图中的Admin

接下来的页面可以不用填，直接点Create Page，马上会转到一个404页面，不要慌，要过一会系统才会帮你把网页生成好。如下图：

* 克隆自己的代码库
* git clone git@github.com:yanping/example.git

这样git会把存放在github上的代码库文件下载到本地的，生成名为example的目录。删除该目录下的index.html，这是系统生成 的，不是我想要的页面，注意不要把.git目录删除，这是个隐藏目录，里面包含这个代码库的配置信息，以上的步骤都是为了得到这些配置信息且避免了使用命 令。

要注意的几点

* 我的github用户名为yanping，如果我建立名为yanping.github.com的代码库，必须在master分支下的才 可以看到页面。在本例中我建的是普通的项目代码库，系统自动生成的github pages是在gh-pages这个分支下的。在以上的操作中，我并没有对分支进行任何操作，都是系统自动完成的。
* 创建个人主页面username.github.com的步骤请看[GotGithub教程](http://www.worldhello.net/gotgithub/03-project-hosting/050-homepage.html" \l "user-homepage" \t "_blank)
* git@github.com:yanping/example.git 是example的路径，它的规则是 **git@github.com:用户名/代码库名.git**
* 克隆别人的代码库。在本地另一位置，克隆别人的代码库，比如
* git clone git@github.com:mojombo/mojombo.github.com.git

删掉.git目录，然后把文件都复制到自己的本地代码库example下

* 删除\_post下的文件（可保留一两篇作为模板），修改example里的文件，尤其是配置信息，比如\_config.yml、disqus的配置，CNAME文件等，更进一步，按照自己的喜好修改网页的布局和样式，这些都可在后期慢慢摸索。然后执行下面的操作
  + git add .表示添加当前目录下的所有文件
  + git commit -am "message" 表示提交所有更改，这是提交到本地，”message” 换成自己的注释信息
  + git push 把在本地的更改提交到远程服务器

要写博客的时候，在\_post里新建一个markdown文件，文件名和文件里面的头部信息（学名叫YAML front matter）按照模板的格式改，编辑好内容后，再依次执行上面三条命令。

如果你不熟悉markdown语法，请看[这里](http://wowubuntu.com/markdown/)。

进一步阅读:

* Github Pages的[官方说明文件](http://pages.github.com/" \t "_blank)
* [jekyll主页上提供的示例网站](https://github.com/mojombo/jekyll/wiki/sites)，可以clone他们的网站折腾一翻
* [在github上建立pages的过程](http://www.pizn.me/2011/09/22/create-github-page.html)
* 关于jekyll静态网站的介绍，请看我翻译的[文章](http://chen.yanping.me/cn/blog/2011/12/15/building-static-sites-with-jekyll/" \t "_blank)
* [像黑客一样写博客——Jekyll入门](http://www.soimort.org/tech-blog/2011/11/19/introduction-to-jekyll_zh.html)

其他技巧：

* [优化Jekyll站点的SEO技巧](http://www.pizn.me/2012/01/16/the-seo-for-jekyll-blog.html)
* [为Jekyll博客添加category 分类](http://www.pizn.me/2012/02/23/use-category-plugin-for-jekyll-blog.html)
* [搭建Jekyll博客的一些小技巧](http://www.pizn.me/2012/03/01/some-tips-for-jekyll-blog.html)

**六、关于代码高亮**（如果你不贴代码，请跳过这步）

* 用js插件：[DlHightLight](http://mihai.bazon.net/projects/javascript-syntax-highlighting-engine" \t "_blank)或[Google Code Prettify](http://code.google.com/p/google-code-prettify/" \t "_blank)
* 用[gist](https://gist.github.com/" \t "_blank)：强烈推荐菜鸟使用，省心省事，支持语言多
* 用[pygment](http://pygments.org)：要安装python以及python的包管理软件，又是个大坑，不建议菜鸟使用，尤其是使用windows的

**七、关于公式**（如果你不贴公式，请跳过）

* 使用maruku来解析markdown文件，可以把LaTeX解析成图片，优点是网页加载速度快。但是在windows下安装复杂，且需要安装有LaTeX
* [Mathjax](http://www.mathjax.org/)，请看我博文的[介绍](http://chen.yanping.me/cn/blog/2012/03/10/octopress-with-latex/" \t "_blank)，缺点是动态加载，速度慢。

**八、评论**

国外的[Disqus](http://disqus.com)和国内的[友言](http://uyan.cc/" \t "_blank)

其他社会化服务

* 分享：国内的[jiathis](http://jiathis.com)和国外的[addthis](http://addthis.com" \t "_blank)
* 图片：国内的[yupoo](http://www.yupoo.com/" \t "_blank) 、[poco](http://www.poco.cn/" \t "_blank)，国外的[Flickr](http://www.flickr.com/" \t "_blank)、[imgur](http://imgur.com" \t "_blank)

**九、关于域名**

在本地代码库里新建名为CNAME的文本文件，把域名地址放进去。假设你的域名是domain.com，可以用命令

echo 'domain.com' > CNAME

然后

git add CNAME

git commit -am "CNAME file added"

git push

接着在自己的域名注册商那里改一下指向就行了。如果想对github域名绑定的机制有更多的了解，请看我的三篇博文:[文1](http://chen.yanping.me/cn/blog/2011/12/04/github-pages-domain/)，[文2](http://chen.yanping.me/cn/blog/2011/12/26/github-pages-domain-2/" \t "_blank)和[文3](http://chen.yanping.me/cn/blog/2012/02/06/github-pages-domain-3/" \t "_blank)

**十、其他可供选择的模板**，推荐两款比较好用的

* [Octopress](http://www.octopress.org)：windows下的[教程1](http://sinosmond.github.com/blog/2012/03/12/install-and-deploy-octopress-to-github-on-windows7-from-scratch/" \t "_blank)、[教程2](http://tonytonyjan.heroku.com/2012/03/01/install-octopress-on-windows/" \t "_blank)，[ubuntu下的教程](http://www.yangzhiping.com/tech/octopress.html" \t "_blank)
* [Jekyll Bootstrap](http://jekyllbootstrap.com/)

**十一、常犯的错误**

* 明明要给是要做项目主页，却在master分支下上传页面。只有名为username.github.com的是个人主代码库，username是你的github用户名，向这个代码库推送的网页默认的是master分支，直接就可以浏览。其他代码库都是项目代码库
* clone别人的代码库到本地后，没有把它的.git目录删除
* 没有把别人页面里的配置部分彻底改掉，比如disqus的配置，CNAME文件，<title>等