

Table of Contents

1. [Introduction](#)
2. 基本安装
 - i. [Node.js](#) 安装
 - ii. [Gitbook](#) 安装
 - iii. [Gitbook](#) 命令行速览
3. 图书编辑
 - i. [gitbook](#) 命令行&markdown 编辑
 - ii. [gitbook editor](#) 编辑
4. 图书输出
 - i. 输出为静态网站
 - ii. 输出PDF
5. 发布
 - i. 发布到[gitbook.com](#)
 - ii. 发布到[github](#) 仓库
 - iii. 发布到[Github Pages](#)
6. 结束

Gitbook 入门教程

本书参考如下两篇文章整理而成，并且有所修改：

1. 在线阅读:<http://wanqingwong.com/gitbook-zh/>
2. 在线阅读:<http://www.chengweiyang.cn/gitbook>

GitBook 是一个基于 Node.js 的命令行工具，可使用 Github/Git 和 Markdown 来制作精美的电子书。

本书将简单介绍如何安装、编写、生成、发布一本在线图书,且示例全部在windows下展示(其他系统差不多一致):

支持格式

GitBook支持输出多种文档格式，如：

- 静态站点：GitBook默认输出该种格式，生成的静态站点可直接托管搭载Github Pages服务上；
- PDF：需要安装[ebook-concert](#)依赖；
- eBook：需要安装[ebook-concert](#)；
- 单HTML网页：支持将内容输出为单页的HTML，不过一般用在将电子书格式转换为PDF或eBook的中间过程；
- JSON：一般用于电子书的调试或元数据提取。

Gitbook项目地址

- GitBook项目官网：<http://www.gitbook.io>
- GitBook Github地址：<https://github.com/GitbookIO/gitbook>

本项目地址

- 仓库：<https://github.com/yuzeshan/gitbook-studying>
- 在线阅读：<http://yuzeshan.gitbooks.io/gitbook-studying/content/>

基本安装

本章内容讲述Gitbook的安装及命令行的功能快速预览说明。

Node.js安装



Node.js 是一个基于Chrome JavaScript 运行时建立的一个平台，用来方便地搭建快速的，易于扩展的网络应用· Node.js 借助事件驱动，非阻塞 I/O 模型变得轻量 and 高效，非常适合 run across distributed devices 的 data-intensive 的实时应用·

Node.js的安装，请移步[这里](#)：

在 [Windows](#)、[Mac OS X](#) 與 [Linux](#) 中安裝 [Node.js](#) 網頁應用程式開發環境

安装完成这后，你可以在终端模式(win下打开cmd即可)下检验一下：

```
C:\Users\lenovo> node -v  
v0.10.28
```

看到些提示，就表示你已成功安装上了 Node.js 。

安装gitbook前必须先安装 **node** 。

Ps:本文中，运行命令几乎都在**Windows cmd**下，但是也有个比较好用的工具，[git bash](#),[点击下载](#)

Gitbook安装

Gitbook是从NPM安装的，命令行(打开cmd，路径每个电脑不一):

```
C:\Users\lenovo> npm install gitbook-cli -g
```

安装完之后，你可以检验下是否安装成功:

```
C:\Users\lenovo> gitbook -V  
0.3.3
```

如果你看到了与上面类似的版本信息，则表示你已成功完装上了Gitbook。

需要注意的是：用户首先需要安装 **nodejs**，以便能够使用 **npm** 来安装 **gitbook**。

Gitbook 命令行速览

Gitbook 是一个命令行工具，使用方法(这里简单介绍):

本地预览

```
C:\Users\lenovo> gitbook serve ./图书名称
```

输出一个静态网站

```
C:\Users\lenovo> gitbook build 图书目录 输出目录
```

注意:上面`win`下命令运行所处所在的目录必须是写好的`gitbook`书籍目录前一个， 可以使用`cd..`切换， 否则建立时会出错,后面会详细讲解

查看帮助

```
C:\Users\lenovo> gitbook help
```

```
build [book] [output]      build a book
--format      Format to build to (Default is website; Values are website, json, ebook)
--log         Minimum log level to display (Default is info; Values are debug, info, warn, error, disabled)

pdf [book] [output]      build a book to pdf
--log         Minimum log level to display (Default is info; Values are debug, info, warn, error, disabled)

epub [book] [output]      build a book to epub
--log         Minimum log level to display (Default is info; Values are debug, info, warn, error, disabled)

mobi [book] [output]      build a book to mobi
--log         Minimum log level to display (Default is info; Values are debug, info, warn, error, disabled)

serve [book]      Build then serve a gitbook from a directory
--port           Port for server to listen on (Default is 4000)
--lrport         Port for livereload server to listen on (Default is 35729)
--watch          Enable/disable file watcher (Default is true)
--format         Format to build to (Default is website; Values are website, json, ebook)
--log            Minimum log level to display (Default is info; Values are debug, info, warn, error, disabled)

install [book]      install plugins dependencies

init [directory]      create files and folders based on contents of SUMMARY.md
```

书籍编辑

gitbook书籍编辑有两种方法：

- gitbook命令行&markdown编辑： `README.md` 和 `SUMMARY.md` 是Gitbook项目必备的两个文件，也就是一本最简单的gitbook也必须含有这两个文件，它们在一本Gitbook中具有不同的用处；
- 使用gitbook官方编辑器编辑，[下载](#)到本地即可。

gitbook命令行&markdown编辑

1. 首先介绍下一本书的主要文件：

README.md

这个文件相当于一本Gitbook的简介，最上层(和SUMMARY.md同级)的是本书的Introduction。

SUMMARY.md()

这个文件是一本书的目录结构，使用Markdown语法，这个文件在使用gitbook命令行之前要先写好，以便之后生成书籍目录,如我们这本书的 SUMMARY.md：

```
* [基本安装](howtouse/README.md)
- [Node.js安装](howtouse/Nodejsinstall.md)
- [Gitbook安装](howtouse/gitbookinstall.md)
- [Gitbook命令行速览](howtouse/gitbookcli.md)
* [图书编辑](book/README.md)
- [gitbook命令行&markdown编辑](book/gitbook-cli.md)
- [gitbook editor编辑](book/editor.md)
* [图书输出](output/README.md)
- [输出为静态网站](output/outfile.md)
- [输出PDF](output/pdfandebook.md)
* [发布](publish/README.md)
- [发布到gitbook.com](publish/gitbook.md)
- [Github集成](publish/github.md)
- [发布到Github Pages](publish/gitpages.md)
* [结束](end/README.md)
```

列表加链接，链接中可以使用目录，也可以不必使用。

2. 书籍目录初始化&内容编写

当这个目录文件创建好之后，我们可以使用Gitbook的命令行工具将这个目录结构生成相应的目录及文件,步骤如下：

- 打开cmd,cd命令进入创建的存书籍的目录，如:E:\gitbook\gitbook-studying;
- 在这个目录下我们创建了上述的SUMMARY.md文件;
- 然后使用命令gitbook init,则生成一系列目录文件，见下方；

```
$ gitbook init
$ ls
LICENSE      SUMMARY.md  book        output
README.md    howtouse    publish
$ tree .
.
├── LICENSE
├── README.md
├── SUMMARY.md
├── book
│   ├── README.md
│   ├── gitbook-cli.md
│   └── editor.md
├── howtouse
│   ├── Nodejsinstall.md
│   └── README.md
```



```
| | | gitbookcli.md
| | | gitbookinstall.md
| | | output
| | | | README.md
| | | | outfile.md
| | | | pdfandebook.md
| | | publish
| | | | README.md
| | | | gitbook.md
| | | | github.md
| | | | gitpages.md
```

我们可以看到，**gitbook**给我们生成了与 **SUMMARY.md** 所对应的目录及文件。

每个目录中，都有一个 **README.md** 文件，相当于一章的说明。

最后，我们在对应目录中用**markdown**编辑器(这里推荐使用**markdownpad**)编辑自己想要的内容即可。

gitbook editor编辑

[gitbook editor](#) 实际上就是一个本地应用版的在线编辑器，使用方式和 [gitbook](#)在线编辑器类似，而这里的目录及章节是使用编辑器添加生成，同时也会存放到本地之前建立的目录。

图书输出

本章内容讲述如何将编写好的图书输出为我们所需要的格式。目前为止，**Gitbook**支持如下格式输出：

- 静态HTML，可以看作一个静态网站
- PDF格式
- eBook格式
- 单个HTML文件
- JSON格式

我们这里着重说下如何输出静态的HTML和PDF文件。

输出为静态网站

你有两种方式输出一个静态网站：

本地预览时自动生成

当你在自己的电脑上编辑好图书之后，你可以使用Gitbook的命令行进行本地预览：

```
E:\gitbook\gitbook-studying>gitbook serve ./图书目录
```

这里会启动一个端口为 4000 用于预览的服务器：

```
E:\gitbook\gitbook-studying> gitbook serve .
Press CTRL+C to quit ...

Starting build ...
Successfully built !

Starting server ...
Serving book on http://localhost:4000
```

你可以你的浏览器中打开这个网址：<http://localhost:4000>：



这里你会发现，你在你的图书项目的目录中多了一个名为 `_book` 的文件目录，而这个目录中的文件，即是生成的静态网站内容。

使用 **build** 参数生成到指定目录

与直接预览生成的静态网站文件不一样的是，使用这个命令，你可以将内容输入到你想要的目录中去,步骤如下：

1. `mkdir outbook`
2. `cd..`,退到上一层目录，即E:\gitbook (如果不，会出错)
3. 然后,E:\gitbook>`gitbook build gitbook-studying gitbook-studying/outbook`
4. 则在 E:\gitbook\gitbook-studying\outbook下生成了同样的静态html文件

无论哪种方式，你都可以将这个文件打包，然后把你的书发给你的朋友们了！

输出PDF

输出pdf文件步骤如下：

1. 由于生成PDF文件依赖于 `ebook-convert`，故首先在该处[ebook-convert下载链接](#)点击下载所需要的版本，安装即可；
2. 打开cmd，进入E:\gitbook目录；
3. E:\gitbook>gitbook pdf gitbook-studying gitbook-studying/gitbook入门教程.pdf
4. 则在目录E:\gitbook\gitbook-studying下生成了该pdf文件

然后，你会发现你的目录里多了一个名为 `gitbook入门教程.pdf` 的文件，就是它了！

但是，笔者在测试的时候发现，生成的PDF文件在排版上不是合理，如页面的边距明显过小，如果是图文混排的话，整个版面感觉有些零乱。

发布

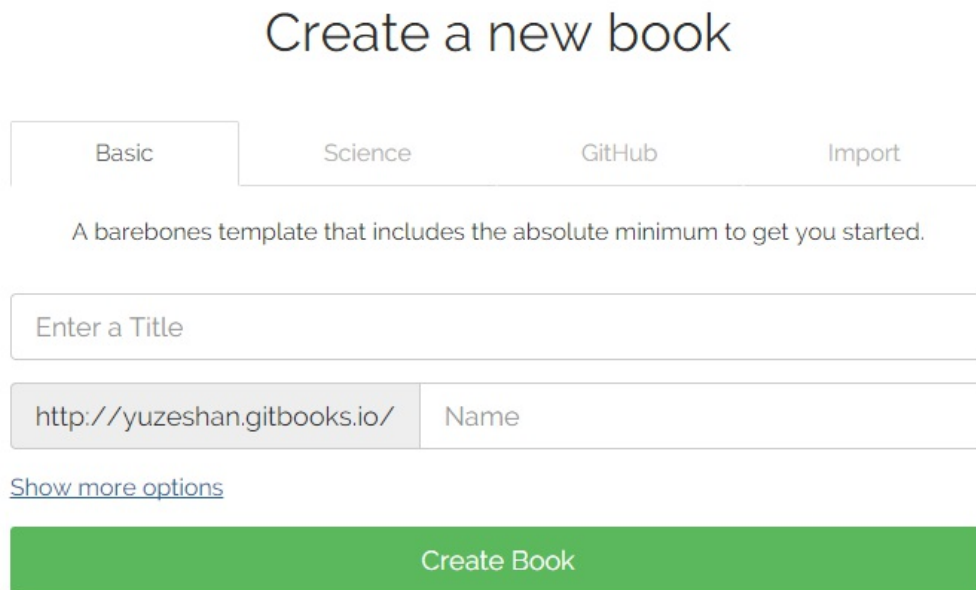
本章内容讲述将之前编辑的gitbook书籍发布到线上，主要是：

- 将其发布到[GitBook.com](https://gitbook.com)线上个人仓库
- 发布到github仓库，使用git管理，同时编辑时也可以同步到[GitBook.com](https://gitbook.com)上
- 发布到github pages上面

发布到Gitbook.com

GitBook.com 是一个围绕 gitbook 发行书籍的社区，于 2014 年初创，GitBook.com 提供免费和付费的服务，而且免费账户就可以享受诸多服务，这里，我们将编写的gitbook书籍发布到Gitbook个人账户里。

登陆 GitBook.com 后，在用户页面，点击create a new book后，如下图所示：



创建的书籍，title: gitbook入门教程 ,git url为 `https://git.gitbook.com/yuzeshan/gitbook-studying.git`

这样，gitbook.com线上的仓库就建好了，由于GitBook.com 上的每本书都使用 Git 项目来管理，所以，事先写好的书籍就可以通过git提交上去，具体步骤如下：

1. 首先打开cmd，进入书籍目录，`E:\gitbook\gitbook-studying`;
2. 运行`E:\gitbook\gitbook-studying> git init`，这里如果git提示错误，则是没配置git的环境变量的path;
3. 将目录的章节文件夹一个个加进，git里，使用 `git add` 文件夹名；
4. 接下来，`git commit -m "提交信息(必填，否则出错)"`；
5. 然后，将本地的git 仓库与gitbook.com上远程仓库连接，`git remote add gitbook https://git.gitbook.com/yuzeshan/gitbook-studying.git` (注意:不是github上的`git remote add origin git@....`);
6. 最后，将本地仓库全部push到远程，`git push -u gitbook master`，以后每次更改，然后add再commit，而第二次push时，直接 `git push gitbook master` 即可

提交到 GitBook.com 后，书籍就自动发布了，用户就可以通过书籍的地址访问了，例如：<http://yuzeshan.gitbooks.io/gitbook-studying/content/>

ps:如果本地没有git初始化过得仓库，则可以将在在线建立的仓库，克隆到本地，进行修改，提交，**push**，简洁步骤如下：

- 首先进入想要建立书籍的目录，这里是`E:\gitbook\gitbook-studying`;
- 运行命令 `git clone https://git.gitbook.com/yuzeshan/gitbook-studying.git`;

- 然后，通过之前介绍的书籍编辑:`gitbook init`,`gitbook serve` 等方法，编辑好书籍后，再用`git`命令`add commit`添加提交书籍目录文件；
- 最后， `git push gitbook master` 即可。

发布到github仓库

[GitBook.com](#) 为每本书籍都创建了一个 Git 项目，并且使用这个 Git 项目来管理书籍源码（注意：这里的源码是指所有用户提交的内容）。正如在 [编辑书籍](#) 中介绍的那样，我们可以通过向书籍的 Git 项目提交内容来更新书籍。

另外，[GitBook.com](#) 还可以集成 [GitHub](#)，所以用户可以将书籍的源码通过 [GitHub](#) 上的项目来管理(而且在github上编辑时也可以同步到GitBook.com上对应的仓库)，这样可以使用 [GitHub](#) 带来的各种优点，例如：

- 其它用户可以 fork
- 用户可以点赞，获得更新提醒
- 用户可以贡献自己的内容


等等

这里，接着之前的实例，将使用 [GitHub](#) 中的项目来替代 [GitBook.com](#) 上的项目：<https://git.gitbook.com/yuzeshan/gitbook-studying.git>分为以下三大步骤：

在Github上创建仓库，并设置

- 登录Github账户，创建仓库 `gitbook-studying`；
- 显示如下：

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


 Set up in Desktop

 or

HTTPS

SSH


git@github.com:yuzeshan/gitbook-studying.git



We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).


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

```
echo # gitbook-studying >> README.md
git init
git add README.md
git commit -m "first commit"
git remote add origin git@github.com:yuzeshan/gitbook-studying.git
git push -u origin master
```



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

```
git remote add origin git@github.com:yuzeshan/gitbook-studying.git
git push -u origin master
```



...or import code from another repository

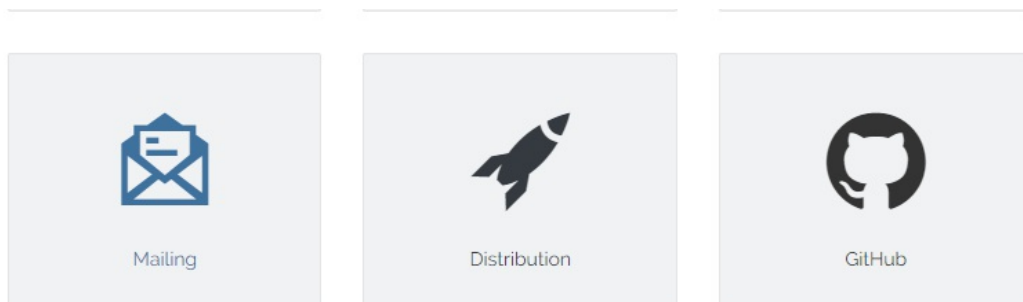
You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

- 将建好的仓库克隆到本地，但是使用cmd的时候，发现clone被拒绝，未找到原因，于是换作[git bash](#)同样使用cd进入书籍目录(E:\gitbook\gitbook-studying-github，此为新的空目录)
- 运行命令：`$git clone git@github.com:yuzeshan/gitbook-studying.git`，在此目录生成了 `gitbook-studying/.git` 文件
- ``$cd gitbook-studying`，此时git bash目录结构为`/e/gitbook/gitbook-studying-github/gitbook-studying`

在Gitbook.com上为该项目设置Github集成

首先，到Gitbook.com页面，将书籍的 Git 项目设置为 GitHub 上的项目，进入书籍属性页面，找到 "GitHub" 图标，如下图：



输入将要使用的 GitHub 上的项目，注意是公开的项目，如下图：

✓ Settings updated successfully. x

🔗 Need some help? Learn more about [the GitHub integration](#).

GitHub Repository

yuzeshan/gitbook-studying

Link your book to a GitHub repository to edit this repository directly from the editor.

Integration

This book is linked to the GitHub repository: [yuzeshan/gitbook-studying](#).

It will automatically updates & builds your book whenever you push to this repository (only public repos).

Add a deployment webhook

保存后，可以看到之前不可点击的 "Add a deployment webhook" 按钮已经可以点击了，这个按钮表示：每当用户配置的 GitHub 上的项目更新时，自动更新Gitbook.com书籍！

将本地书籍导入Github上对应仓库

现在，将书籍原来的 Git 项目内容导入到新建的 GitHub 中的项目中，步骤如下：

1. 将之前编辑的书籍目录为 `E:\gitbook\gitbook-studying` 中除.git文件，全部拷贝到克隆的目录中，即 `E:\gitbook\gitbook-studying-github\gitbook-studying` 中；
2. 将已经编辑好的书籍文件，用 `git add/commit` 命令依次添加，提交后；

3. 接下来将书籍仓库与远程github连接到一起，运行命令：`git remote add github https://github.com/yuzeshan/gitbook-studying.git` (注意:并不是`git remote add origin https://github.com:yuzeshan/gitbook-studying.git`,也许不运行此命令也可，因为克隆的文件本身就关联了，未尝试。。。);
4. 最后，将书籍仓库push到github仓库上，运行命令：`git push -u github master`;
5. `git push` 命令中的 `-u` 表示将本地 `master` 分支的上游分支设置为 `github/master`，所以以后修改了本地 `master` 分支后，`git push` 将推送到 `github` 上，而非原来的 `git remote add gitbook https://git.gitbook.com/yuzeshan/gitbook-studying.git`。

经过以上步骤后，就可以在本地修改文件，直接`add`，`commit`，然后`push`上github仓库上即可，修改github仓库的同时，`gitbook.com`上对应的书籍也会同步更新。

发布到github pages

将生编写好的格式为.md的文件通过Gitbook处理，然后再发布到Github Pages上去。我个人比较喜欢将源码，即 .md 文件与Github Pages静态文件存放在一个仓库中。 .md 文件为 master 分支，而html文件为 gh-pages 分支。具体流程是这样的：

创建仓库与分支

- 登录到Github，创建一个新的仓库，名称我们就命令为 book ，这样我就得到了一个 book 的空仓库。
- 克隆仓库到本地： `git clone git@github.com:USER_NAME/book.git` 。
- 创建一个新分支： `git checkout -b gh-pages` ，注意，分支名必须为 gh-pages 。
- 将分支push到仓库： `git push -u origin gh-pages` 。
- 切换到主分支： `git checkout master` 。

经过这一步处理，我们已经创建好 gh-pages 分支了，有了这个分支，Github会自动为你分配一个访问网址：

<http://USERNAME.github.io/book>

你可以在项目页面右下 settings 中看到：

GitHub Pages

Changes may take up to **ten minutes** to be visible.

Your site is published at <http://vole2014.github.io/book>

Update your site

To update your site, push your HTML or **jenkins** updates to your gh-pages branch. [More info.](#)

Overwrite your site by using our automatic page generator.

Automatic page generator

Author your content in our markdown editor, select a theme, then publish.

当然，由于我们内容还没有上传所以你点开链接，也只是一个404页面。

同步静态网站代码到分支

下面我们就可以将build好的静态网站代码同步到 gh-pages 分支中去了：

- 切换出master分支目录。我们需要将 gh-pages 分支内容存放到另一个目录中去。
- 克隆 gh-pages 分支： `git clone -b gh-pages git@github.com:USERNAME/book.git book-end` 。这步我们只克隆了 gh-pages 分支，并存放在一个新的目录 book-end 里面。
- Copy静态多站代码到 book-end 目录中。
- Push到仓库。

然后，等十来分钟的样子，你就可以访问到你的在线图书了。而后，每次修改之后，都可以将生成的代码Copy到 book-end 目录，再Push一下就OK了。

当然，对于 `gh-pages` 存放问题，你也可以直接在 `master` 分支目录中直接 `git clone gh-pages` 分支，假如名称为 `book-end`，然后修改一下 `.gitignore` 文件，将 `book-end/` 添加进去，这样主分支就不会理会 `book-end` 内容的修改了。

笔者曾试过直接使用 `gitbook build --output=/PATH/book-end` 这个方式输出到 `gh-pages` 分支目录中，但发现 `gitbook` 在 `build` 的时候，相当于删除存在的这个目录，然后再新建目录，再写入内容，原来的 `git` 信息会完全删除掉，显示这不是我们想要看到的，所以只能 `Copy` 才行。

结束

以上这些是笔者在参考网上几个相关的教程，然后自己动手实践一步步写完了，**gitbook**使用简明的**markdown**语法，简洁漂亮。

gitbook将这些独立的文件组合起来，做成一个静态站(可以作为网站教程之用)或是生成**PDF**，非常之便捷。

本书记录的内容是简单的，且某些细节没有指出，如果有出错的地方，请见谅，同时也欢迎**Fork**这个项目,改进该教程。

Ps:

Github项目地址:<https://github.com/yuzeshan/gitbook-studying>

在线阅读地址:<http://yuzeshan.gitbooks.io/gitbook-studying/content/>