

Assignment 0

“

工欲善其事，必先利其器。

本次任务较为简单，并且请注意，完成本次任务并不需要任何前置知识，请认真完成，推荐完成时间：5 - 10个小时。

完成本次实验，你将学会：

- 认识Github
- 创建一个自己的博客
- Markdown的使用
- 拥有一个学生邮箱
- 虚拟机的基本操作
- 配置C++开发环境
- 安装一个库
- 用命令行来运行一个程序

在Github上创建自己的博客

“

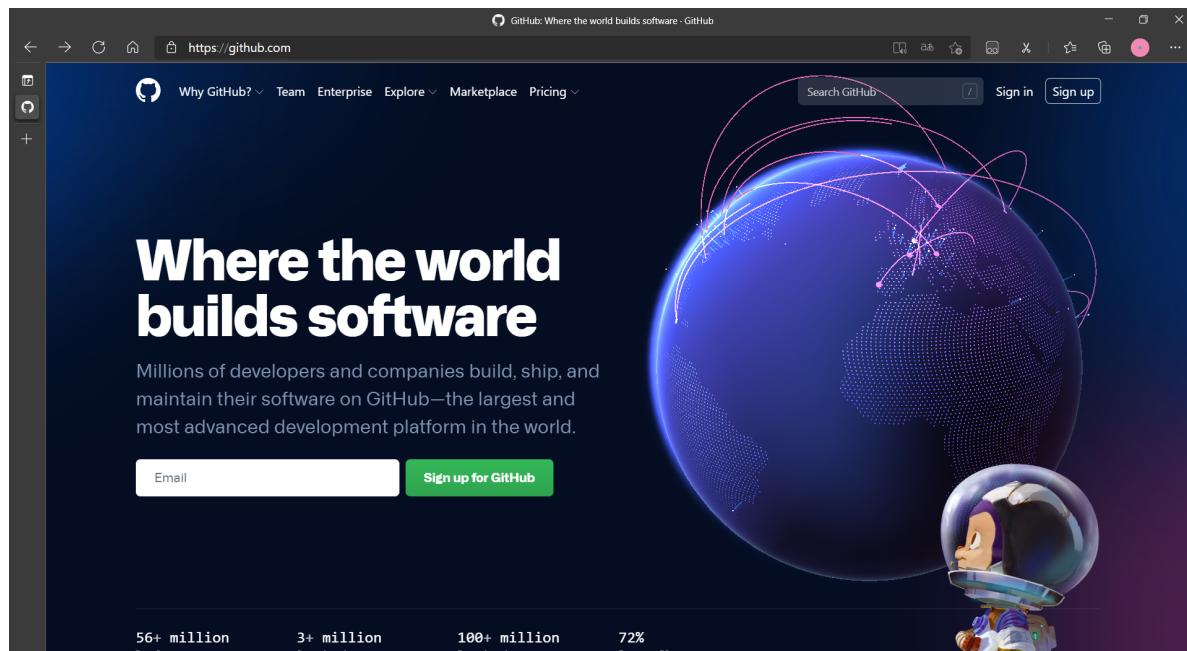
前人已经写了详细的教程，并经过了验证，故本节部分转载了[GitHub+Hexo 搭建个人网站详细教程](#)-[知乎\(zhihu.com\)](#)。由于文章是19年所写，我对教程的正确性进行了验证，对现在不适用的部分做出修改。

初识Github

GitHub是通过Git进行版本控制的软件源代码托管服务平台，由GitHub公司（曾称Logical Awesome）的开发者Chris Wanstrath、P. J. Hyett和汤姆·普雷斯顿·沃纳使用Ruby on Rails编写而成。

其上托管着大量的代码，任何人都可以上传到自己的代码到Github上。

网址：[GitHub: Where the world builds software · GitHub](#)



这是Github的主页，你可以下来详细了解Github，很酷的是右侧的数据流都是真是存在的。

你现在要做的是注册一个Github账号，点击右上角 **Sign up** 即可。

Join GitHub

Create your account

Username *
 ✓

Email address *
 ✓

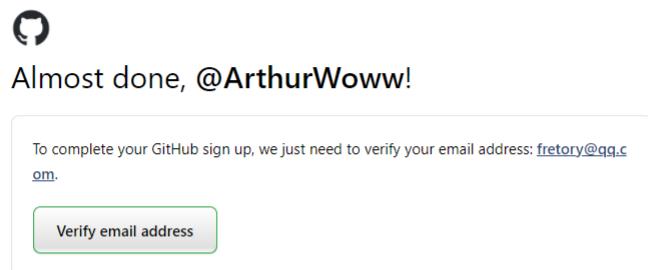
Password *

Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter.
[Learn more.](#)

Email preferences
 Send me occasional product updates, announcements, and offers.

Verify your account

在此界面完成信息填写，提交即可。



Once verified, you can start using all of GitHub's features to explore, build, and share projects.

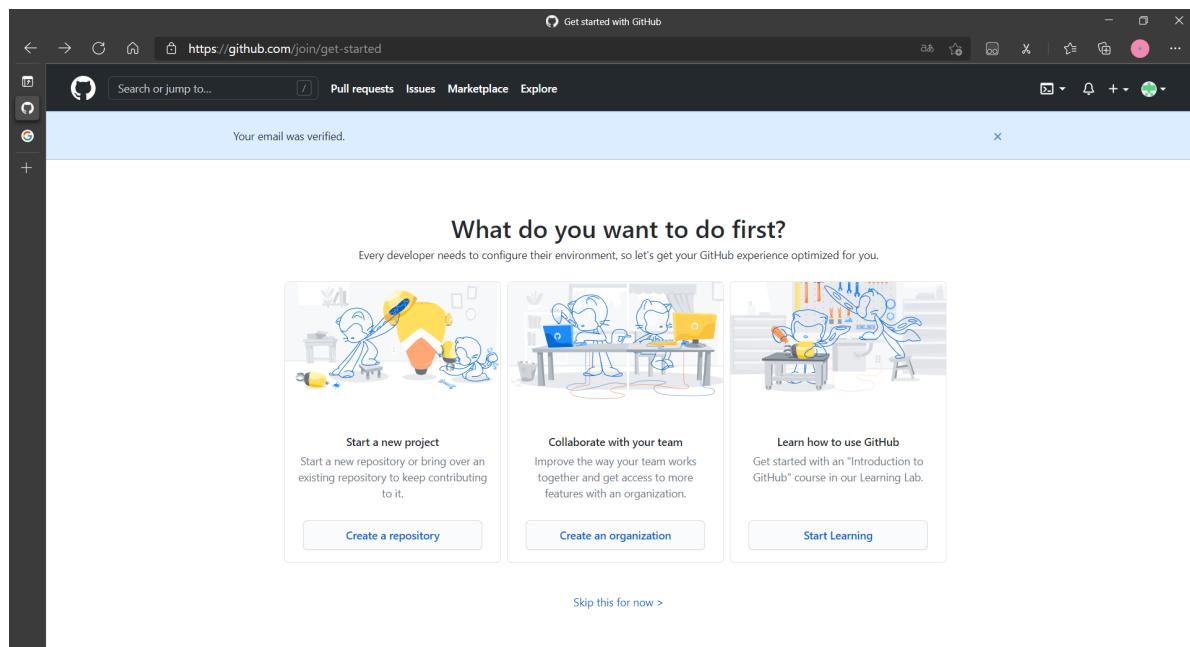
Button not working? Paste the following link into your browser:
https://github.com/users/ArthurWoww/emails/153661239/confirm_verification/4bee664ed3c12b1415de18331c04ed850460df10

[Email preferences](#) · [Terms](#) · [Privacy](#) · [Sign in to GitHub](#)

You're receiving this email because you recently created a new GitHub account or added a new email address. If this wasn't you, please ignore this email.

你将收到如上图的一封邮件，点击 Verify email address 即可验证你的GitHub账号。

完成验证后，你将看到如下的界面：



你可以选择跟随这些提示，来进行Github的探索✿

创建你的第一个仓库

点击 **Create a new project**，我们创建我们的第一个仓库。为了后续我们的博客的部署，请将仓库命名为 **username.github.io**

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere?
[Import a repository.](#)

Owner * Repository name *

 ArthurWoww / ArthurWoww.github.io ✓

Great repository names are short and memorable. Need inspiration? How about [jubilant-octo-succotash?](#)

Description (optional)

 **Public**
Anyone on the internet can see this repository. You choose who can commit.

 **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

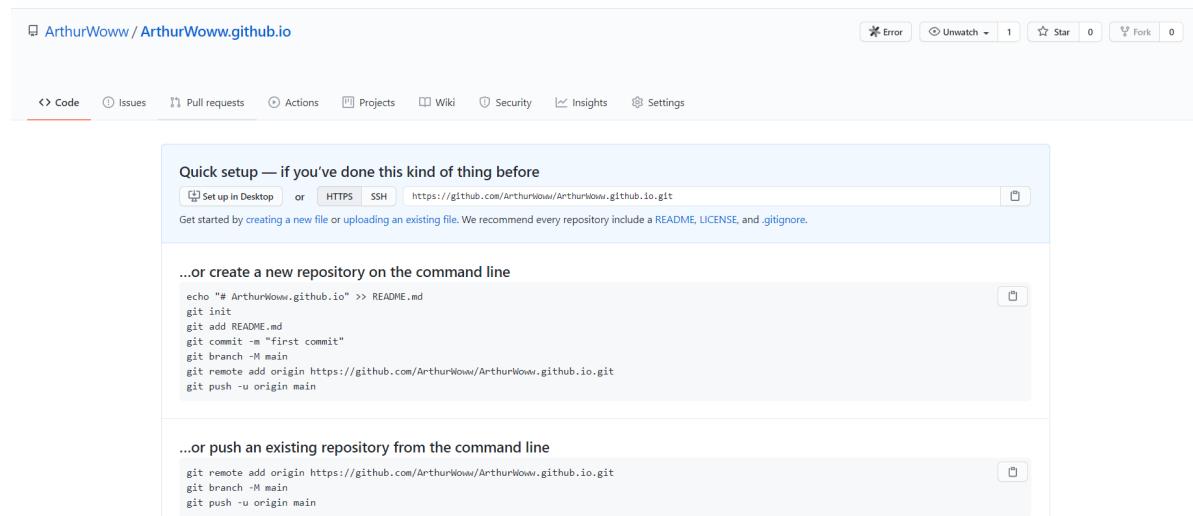
Add a README file
This is where you can write a long description for your project. [Learn more.](#)

Add .gitignore
Choose which files not to track from a list of templates. [Learn more.](#)

Choose a license
A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

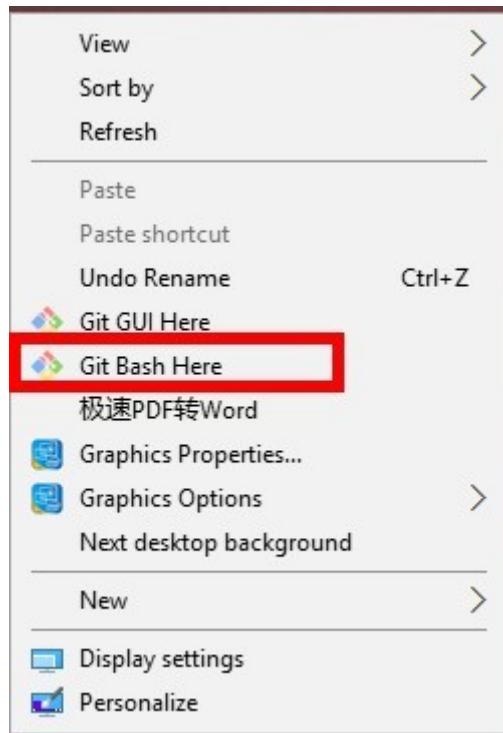
创建完成后你将跳转到你的仓库页面（下图），这里列举此项目的详情。



The screenshot shows a GitHub repository page. At the top, it displays the repository name 'ArthurWoww / ArthurWoww.github.io'. To the right, there are buttons for 'Error', 'Unwatch', 'Star', and 'Fork'. Below the header, there's a 'Quick setup' section with instructions for setting up the repository on desktop or via HTTPS/SSH. It also provides commands for creating a new repository on the command line and pushing an existing one. The main content area is currently empty, showing a message: 'This repository is empty.'

安装Git

什么是Git？简单来说Git是开源的分布式版本控制系统，用于敏捷高效地处理项目。我们网站在本地搭建好了，需要使用Git同步到GitHub上。如果想要了解Git的细节，参看[廖雪峰](#)老师的Git教程：[Git教程](#) 从Git官网下载：[Git - Downloading Package](#) 现在的机子基本都是64位的，选择64位的安装包，下载后安装，在命令行里输入git测试是否安装成功，若安装失败，参看其他详细的Git安装教程。安装成功后，将你的Git与GitHub帐号绑定，鼠标右击打开Git Bash



或者在菜单里搜索Git Bash，设置user.name和user.email配置信息：

```
1 git config --global user.name "你的GitHub用户名"
2 git config --global user.email "你的GitHub注册邮箱"
```

生成ssh密钥文件：

```
1 ssh-keygen -t rsa -C "你的GitHub注册邮箱"
```

然后直接三个回车即可，默认不需要设置密码

然后找到生成的.ssh的文件夹中的id_rsa.pub密钥，将内容全部复制

This PC > Local Disk (C:) > Users > Rescue > .ssh >				
	Name	Date modified	Type	Size
ss	node_modules	4/24/2017 11:21 AM	File folder	
	id_rsa	4/23/2017 7:22 PM	File	2 KB
	id_rsa.pub	4/23/2017 7:22 PM	PUB File	1 KB
	known_hosts	5/12/2017 2:17 PM	File	1 KB

打开[GitHub_Settings_keys](#) 页面，新建new SSH Key

Title

|

Key

Begins with 'ssh-rsa', 'ssh-dss', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp521'

Add SSH key

Title为标题，任意填即可，将刚刚复制的id_rsa.pub内容粘贴进去，最后点击Add SSH key。

在Git Bash中检测GitHub公钥设置是否成功，输入 ssh <git@github.com>：



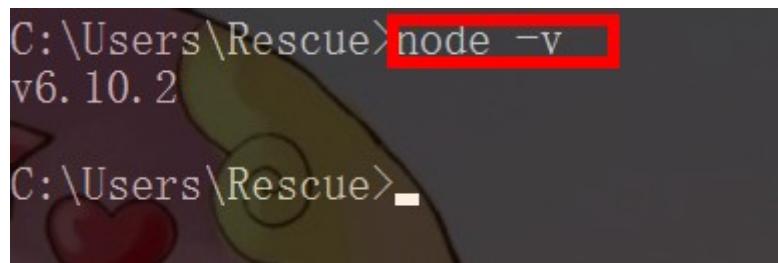
```
MINGW64 /c/Users/Rescue/.ssh
Rescue@DESKTOP-SOH7ITK MINGW64 ~/ssh
$ ssh git@github.com
PTY allocation request failed on channel 0
Hi RunDouble! You've successfully authenticated, but GitHub does not provide shell access.
Connection to github.com closed.

Rescue@DESKTOP-SOH7ITK MINGW64 ~/ssh
$ |
```

如上则说明成功。这里之所以设置GitHub密钥原因是，通过非对称加密的公钥与私钥来完成加密，公钥放置在GitHub上，私钥放置在自己的电脑里。GitHub要求每次推送代码都是合法用户，所以每次推送都需要输入账号密码验证推送用户是否是合法用户，为了省去每次输入密码的步骤，采用了ssh，当你推送的时候，git就会匹配你的私钥跟GitHub上面的公钥是否是配对的，若是匹配就认为你是合法用户，则允许推送。这样可以保证每次的推送都是正确合法的。

安装Node.js

Hexo基于Node.js，Node.js下载地址：[Download | Node.js](#) 下载安装包，注意安装Node.js会包含环境变量及npm的安装，安装后，检测Node.js是否安装成功，在命令行中输入 node -v：



```
C:\Users\Rescue>node -v
v6.10.2
C:\Users\Rescue>
```

检测npm是否安装成功，在命令行中输入npm -v：



到这了，安装Hexo的环境已经全部搭建完成。

安装Hexo

Hexo的安装十分容易，官方也提供了文档，为方便大家理解，我还是在此为大家详细介绍一下。

“

官方指南：[文档 | Hexo](#)

按下 **win + R** 键，输入 **cmd** 回车，打开cmd。

“

在cmd中切换不同的盘符直接输入盘符名即可，例如：**D:** 就是切换到D盘

进入某一文件输入 **cd Folder** 即可。

输入 **E:** 进入E盘(当然，你可以选择任何你喜欢的位置)

继续输入命令，完成hexo的安装。

```
1 npm install -g hexo-cli
```

安装 Hexo 完成后，请执行下列命令，Hexo 将会在指定文件夹中新建所需要的文件。

```
1 hexo init <folder>
2 cd <folder>
3 npm install
```

注意：**<folder>** 请替换为你想要放置的目录名，例如blog。

详细日志，前方有 **>** 标识的为输入的命令。

```
1 WangTao@WANT-TAO C:\Windows\System32
```

```
2 > E:
3 WangTao@WANT-TAO E:

4 > hexo init blog
5 INFO Cloning hexo-starter https://github.com/hexojs/hexo-starter.git
6 INFO Install dependencies
7 added 190 packages from 445 contributors in 38.337s
8
9 15 packages are looking for funding
10    run `npm fund` for details
11
12
13
14

15 |
16 |     New major version of npm available! 6.14.5 -> 7.6.3
17 |     Changelog: https://github.com/npm/cli/releases/tag/v7.6.3
18 |             Run npm install -g npm to update!
19 |
20

21
22 INFO Start blogging with Hexo!
23 WangTao@WANT-TAO E:

24 > cd blog
25 WangTao@WANT-TAO E:\blog

26 > npm install
27 npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@2.3.2
  (node_modules\fsevents):
28 npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for
  fsevents@2.3.2: wanted {"os":"darwin","arch":"any"} (current:
  {"os":"win32","arch":"x64"})
29
30 added 5 packages from 1 contributor in 1.034s
31
32 15 packages are looking for funding
33    run `npm fund` for details
```

认识Hexo

新建完成后，指定文件夹的目录如下：

```
1 .
2 └── _config.yml
3 └── package.json
4 └── scaffolds
5 └── source
6   └── _drafts
7   └── _posts
8 └── themes
```

- **_config.yml**

网站的 [配置](#) 信息，您可以在此配置大部分的参数。

- **scaffolds**

[模板](#) 文件夹。当您新建文章时，Hexo 会根据 scaffold 来建立文件。

Hexo的模板是指在新建的文章文件中默认填充的内容。例如，如果您修改scaffold/post.md中的Front-matter内容，那么每次新建一篇文章时都会包含这个修改。

- **source**

资源文件夹是存放用户资源的地方。除 [_posts](#) 文件夹之外，开头命名为 [_](#) (下划线)的文件 / 文件夹和隐藏的文件将会被忽略。Markdown 和 HTML 文件会被解析并放到 [public](#) 文件夹，而其他文件会被拷贝过去。

- **themes**

[主题](#) 文件夹。Hexo 会根据主题来生成静态页面。

Hexo配置

您可以在 [_config.yml](#) 中修改大部分的配置。

- **网站**

参数	描述
<code>title</code>	网站标题
<code>subtitle</code>	网站副标题
<code>description</code>	网站描述
<code>keywords</code>	网站的关键词。支援多个关键词。
<code>author</code>	您的名字
<code>language</code>	网站使用的语言。对于简体中文用户来说，使用不同的主题可能需要设置成不同的值，请参考你的主题的文档自行设置，常见的有 <code>zh-Hans</code> 和 <code>zh-CN</code> 。
<code>timezone</code>	网站时区。Hexo 默认使用您电脑的时区。请参考 时区列表 进行设置，如 <code>America/New_York</code> , <code>Japan</code> , 和 <code>UTC</code> 。一般的，对于中国大陆地区可以使用 <code>Asia/Shanghai</code> 。

其中，`description` 主要用于SEO，告诉搜索引擎一个关于您站点的简单描述，通常建议在其中包含您网站的关键词。`author` 参数用于主题显示文章的作者。

- **网址**

参数	描述	默认值
<code>url</code>	网址, must starts with <code>http://</code> or <code>https://</code>	
<code>root</code>	网站根目录	<code>url's pathname</code>
<code>permalink</code>	文章的 <u>永久链接</u> 格式	<code>:year/:month/:day/:title/</code>
<code>permalink_defaults</code>	永久链接中各部分的默认值	
<code>pretty_urls</code>	改写 <code>permalink</code> 的值来美化 URL	
<code>pretty_urls.trailing_index</code>	是否在永久链接中保留尾部的 <code>index.html</code> , 设置为 <code>false</code> 时去除	<code>true</code>
<code>pretty_urls.trailing_html</code>	是否在永久链接中保留尾部的 <code>.html</code> , 设置为 <code>false</code> 时去除(对尾部的 <code>index.html</code> 无效)	<code>true</code>

这里建议只用修改网站标题即可。更多详情[配置 | Hexo](#)

Hexo指令

• 预览网站

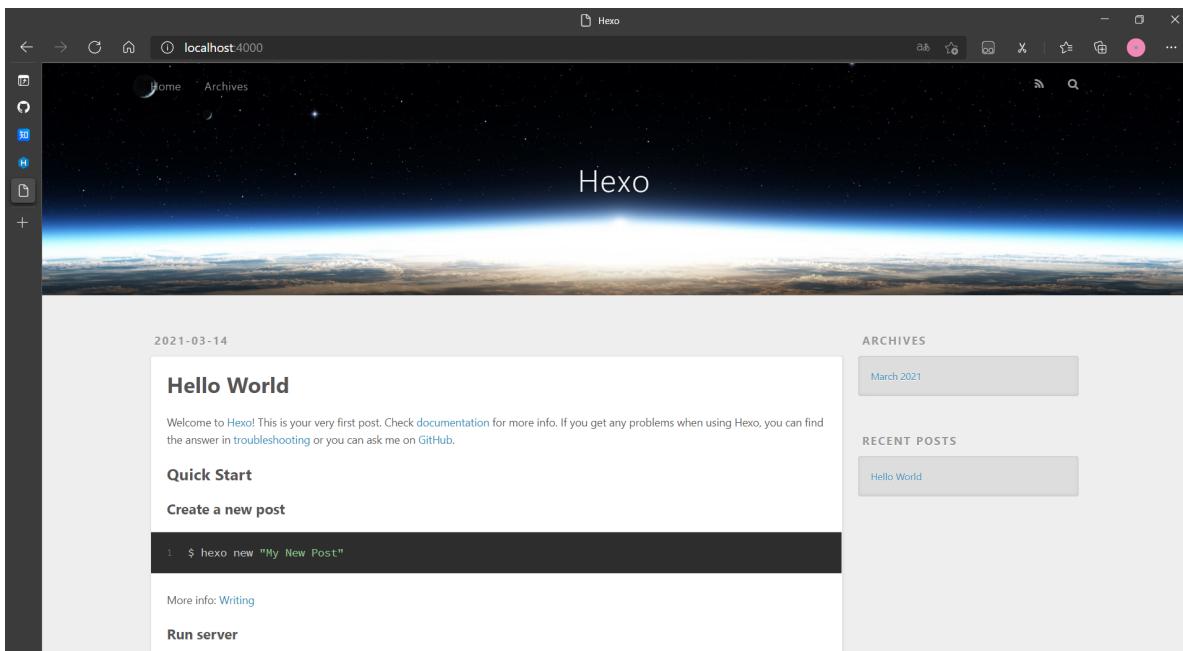
在cmd中输入 `hexo server` , 等待出现以下提示后

```

1  > hexo server
2  INFO  Validating config
3  INFO  Start processing
4  INFO  Hexo is running at http://localhost:4000 . Press Ctrl+C to stop.

```

即可在<http://localhost:4000> (本地) 预览网站。



• 生成静态文件

```
1 hexo generate
```

• 创建一篇新文章

```
1 hexo new [layout] <title>
```

新建一篇文章。如果没有设置 `layout` 的话，默认使用 `_config.yml` 中的 `default_layout` 参数代替。如果标题包含空格的话，请使用引号括起来。

```
1 $ hexo new "post title with whitespace"
```

参数	描述
<code>-p</code> , <code>--path</code>	自定义新文章的路径
<code>-r</code> , <code>--replace</code>	如果存在同名文章，将其替换
<code>-s</code> , <code>--slug</code>	文章的 Slug，作为新文章的文件名和发布后的 URL

默认情况下，Hexo 会使用文章的标题来决定文章文件的路径。对于独立页面来说，Hexo 会创建一个以标题为名字的目录，并在目录中放置一个 `index.md` 文件。你可以使用 `--path` 参数来覆盖上述行为、自行决定文件的目录：

```
1 hexo new page --path about/me "About me"
```

以上命令会创建一个 `source/about/me.md` 文件，同时 Front Matter 中的 title 为 "About me"

注意！title 是必须指定的！如果你这么做并不能达到你的目的：

```
1 hexo new page --path about/me
```

此时 Hexo 会创建 `source/_posts/about/me.md`，同时 `me.md` 的 Front Matter 中的 title 为 "page"。这是因为在上述命令中，hexo-cli 将 `page` 视为指定文章的标题、并采用默认的 `layout`。

部署到Github

下面的指示基于 [一键部署](#) 编写。

1. 安装 [hexo-deployer-git](#).

“

插件安装命令 `npm install hexo-deployer-git --save`

2. 在 `_config.yml`（如果有已存在的请删除）添加如下配置：

```
1 deploy:  
2   type: git  
3   repo: https://github.com/<username>/<project>  
4   # example, https://github.com/hexojs/hexojs.github.io  
5   branch: gh-pages
```

3. 运行 `hexo clean && hexo deploy`。

4. 查看 [username.github.io](#) 上的网页是否部署成功。（username 替换为你的用户名）

“

整合完这部分教程，我花了大概两个小时左右，但从初学者角度讲，我写的并不是很详细，甚至还有一些故意留下的问题，我希望你能自己发现，并寻求解决方案，并总结下来。

任务

最后我希望你能搭建成功你自己的博客，所以并没有太多要求，留下你的博客地址即可。

Markwdon写作

“

上面我们发现，我们的文章都是先写出md类型的文件，然后再被渲染成网页，提交到GitHub，本节主要介绍Markdown的协作工具Typora，包括你现在正在阅读的内容都是Typora编写的。

下载

在Typory官网即可下载[Typora — a markdown editor, markdown reader.](#)

创建一篇新的文章

```
hexo new hexo new "my first passage"
```

通过以上的命令，我们创建一篇文章，系统会提示文章的信息

```
1 > hexo new hexo new "my first passage"
2 INFO Validating config
3 INFO Created: E:\blog\source\_posts\my-first-passage.md
```

可以看到，文章被创建到了 `E:\blog\source_posts\my-first-passage.md`，我们打开这个文件。

撰写

“

注意：我使用的Typora是一体化的布局，并使用了主题，可能和大家的不太一样，但大同小异。



以上就是Typora的主界面,你可以尽情在此撰写. 因为Typora是所见即所得,你所写的文章均能实时的被显示出来.

如果你想更详细的了解其语法,请参考[younghz/Markdown: Markdown 基本语法。 \(github.com\)](#)

我们撰写了以下内容

```
1 ---  
layout: hexo  
title: my first passage  
date: 2021-03-14 19:21:15  
tags:  
---
```

你好,世界

10 这是我的第一篇文章,它是用`Markdown`进行编写的.

二级标题

它语法简单,非常适合撰写文字.

这是它的**代码块**

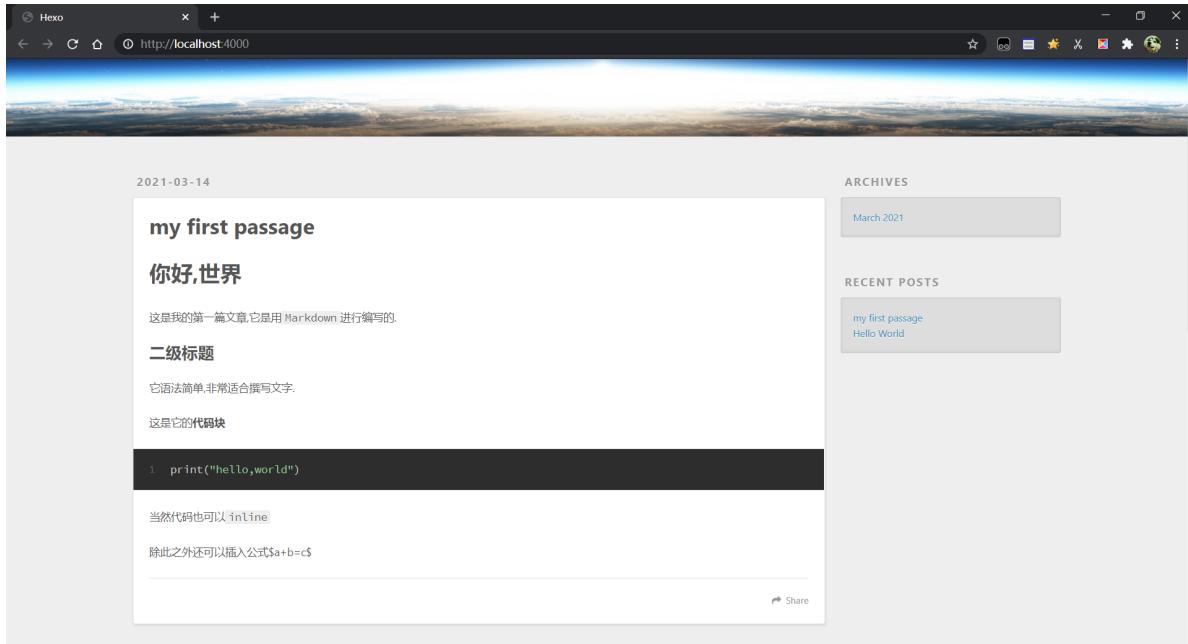
```
20    ```python  
     print("hello,world")  
   ```
```

当然代码也可以`inline`

24 除此之外还可以插入公式\$ $a+b=c$$

| 预览博客

输入 `hexo server` 以在本地预览,博客截图如下



## 申请学生邮箱

“

很多我们使用的工具都需要付费,显然价格对学生而言昂贵的,但是他们提供了教育优惠,大部分工具学生可以免费使用

### 邮箱申请地址

在科技楼背后一楼信息中心,找老师申请即可,注意礼仪,记得一定要一个**别名邮箱**,因为部分工具,类似于  
`181xxxx@xxx.edu.cn` 会被判断为欺诈.

“

tips:如果老师问为什么申请,请不要说是谁谁谁让你来申请的,如果你抱有此思想大可不必完成此项.  
自行破解以后我们使用到的付费软件或者自行寻找开源版本.

### 学生权益

- [Kite - Free AI Coding Assistant and Code Auto-Complete Plugin](#) 学生免费专业版代码自动补全
- [Request a discount - GitHub Education](#) Github学生包申请,包含大量权益
- [Request a discount - GitHub Education](#) jet brains全家桶专业版免费
- ....

## 注意

学生优惠是各个工具提供给**学生的**优惠,请勿滥用,请遵守道德和法律,包括但不限于售卖,转让学生权益

## vmware安装

### 下载

登录[Download VMware Workstation Pro - My VMware](#)网站即可进行下载.

(可能安装后提示注册,可自行搜索一个key完成激活)

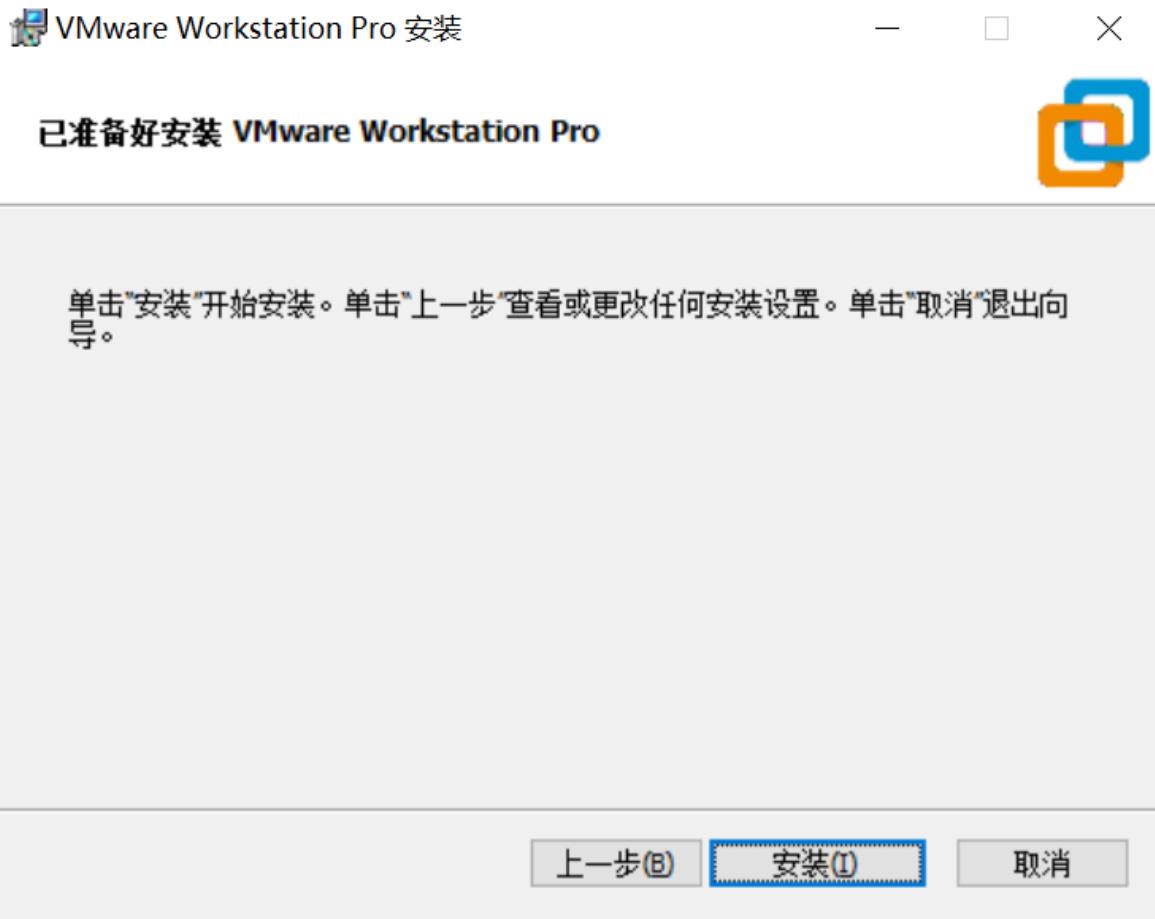
## 使用VMware安装Ubuntu16.04

### 安装过程

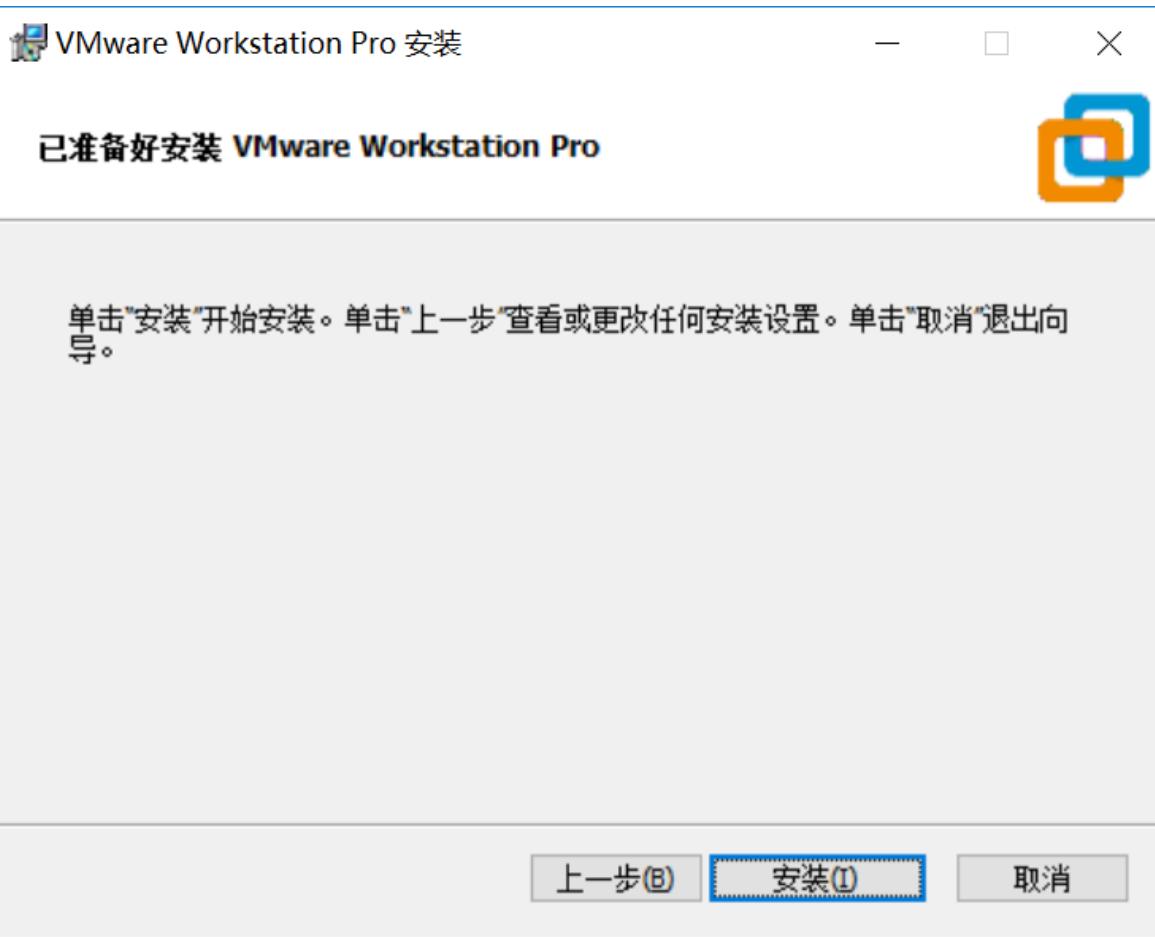
“

本文转载自[Windows10用VMware安装Linux（Ubuntu16.04）虚拟机 - 小白的学习笔记](#)  
[\(corina.cc\)](#).UBUNTU16.04的镜像请自己想办法在网上下载,如果找不到可以私信我获取.

安装向导, 【下一步】



许可协议, 接受许可协议条款, 下一步



自定义安装, 更换安装位置, 下一步



VMware Workstation Pro 安装



## 已准备好安装 VMware Workstation Pro

单击“安装”开始安装。单击“上一步”查看或更改任何安装设置。单击“取消”退出向导。

上一步(B)

安装(I)

取消

用户体验设置，下一步



VMware Workstation Pro 安装



## 已准备好安装 VMware Workstation Pro

单击“安装”开始安装。单击“上一步”查看或更改任何安装设置。单击“取消”退出向导。

上一步(B)

安装(I)

取消

快捷方式，下一步



VMware Workstation Pro 安装



## 已准备好安装 VMware Workstation Pro

单击“安装”开始安装。单击“上一步”查看或更改任何安装设置。单击“取消”退出向导。

上一步(B)

安装(I)

取消

安装



VMware Workstation Pro 安装



## 已准备好安装 VMware Workstation Pro

单击“安装”开始安装。单击“上一步”查看或更改任何安装设置。单击“取消”退出向导。

上一步(B)

安装(I)

取消



## VMware Workstation Pro 安装



### 自定义安装

选择安装目标及任何其他功能。



#### 安装位置:

D:\Program Files (x86)\VMware\VMware Workstation\

[更改...](#)

增强型键盘驱动程序(需要重新引导以使用此功能)  
此功能要求主机驱动器上具有 10MB 空间。

[上一步\(B\)](#)

[下一步\(N\)](#)

[取消](#)

安装导向已完成，许可证，输入密钥



## VMware Workstation Pro 安装



### 自定义安装

选择安装目标及任何其他功能。



#### 安装位置:

D:\Program Files (x86)\VMware\VMware Workstation\

[更改...](#)

增强型键盘驱动程序(需要重新引导以使用此功能)  
此功能要求主机驱动器上具有 10MB 空间。

[上一步\(B\)](#)

[下一步\(N\)](#)

[取消](#)

粘贴密钥，输入



## VMware Workstation Pro 安装



### 自定义安装

选择安装目标及任何其他功能。



#### 安装位置:

D:\Program Files (x86)\VMware\VMware Workstation\

[更改...](#)

增强型键盘驱动程序(需要重新引导以使用此功能)  
此功能要求主机驱动器上具有 10MB 空间。

[上一步\(B\)](#)

[下一步\(N\)](#)

[取消](#)

完成



## VMware Workstation Pro 安装



### 自定义安装

选择安装目标及任何其他功能。



#### 安装位置:

D:\Program Files (x86)\VMware\VMware Workstation\

[更改...](#)

增强型键盘驱动程序(需要重新引导以使用此功能)  
此功能要求主机驱动器上具有 10MB 空间。

[上一步\(B\)](#)

[下一步\(N\)](#)

[取消](#)

成功安装



VMware Workstation Pro 安装

-

□

×

## 自定义安装

选择安装目标及任何其他功能。



### 安装位置:

D:\Program Files (x86)\VMware\VMware Workstation\

更改...

增强型键盘驱动程序(需要重新引导以使用此功能(E))  
此功能要求主机驱动器上具有 10MB 空间。

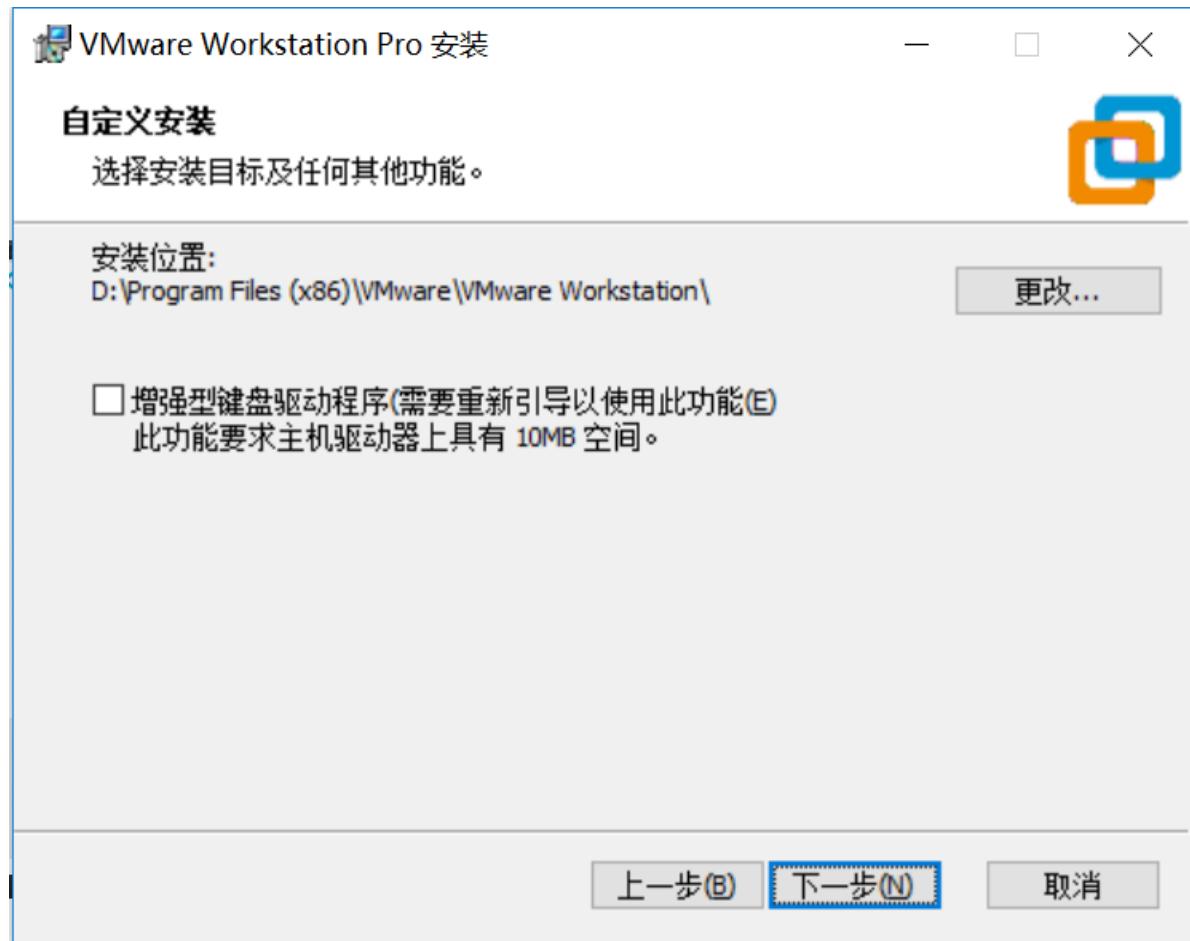
上一步(B)

下一步(N)

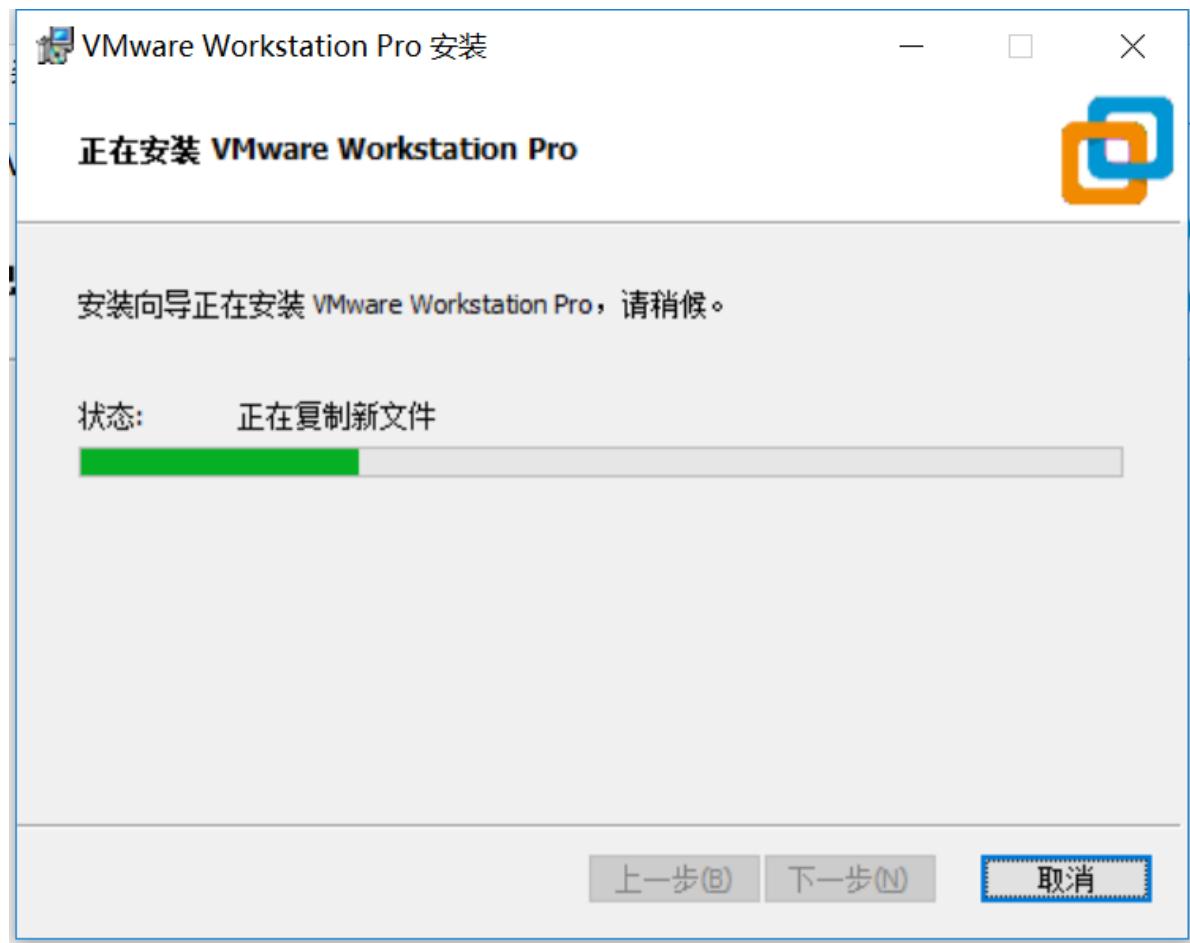
取消

- 在VMware中安装虚拟机

打开VMware Workstation Pro，创建新的虚拟机



新建虚拟机向导，选择自定义，下一步



硬件兼容性，下一步



VMware Workstation Pro 安装



## 正在安装 VMware Workstation Pro

安装向导正在安装 VMware Workstation Pro，请稍候。

状态: 正在复制新文件



上一步(B)

下一步(N)

取消

安装客户机操作系统，稍后安装操作系统，下一步



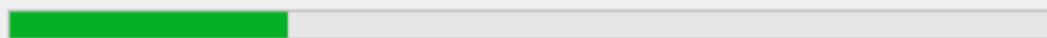
VMware Workstation Pro 安装



## 正在安装 VMware Workstation Pro

安装向导正在安装 VMware Workstation Pro，请稍候。

状态: 正在复制新文件

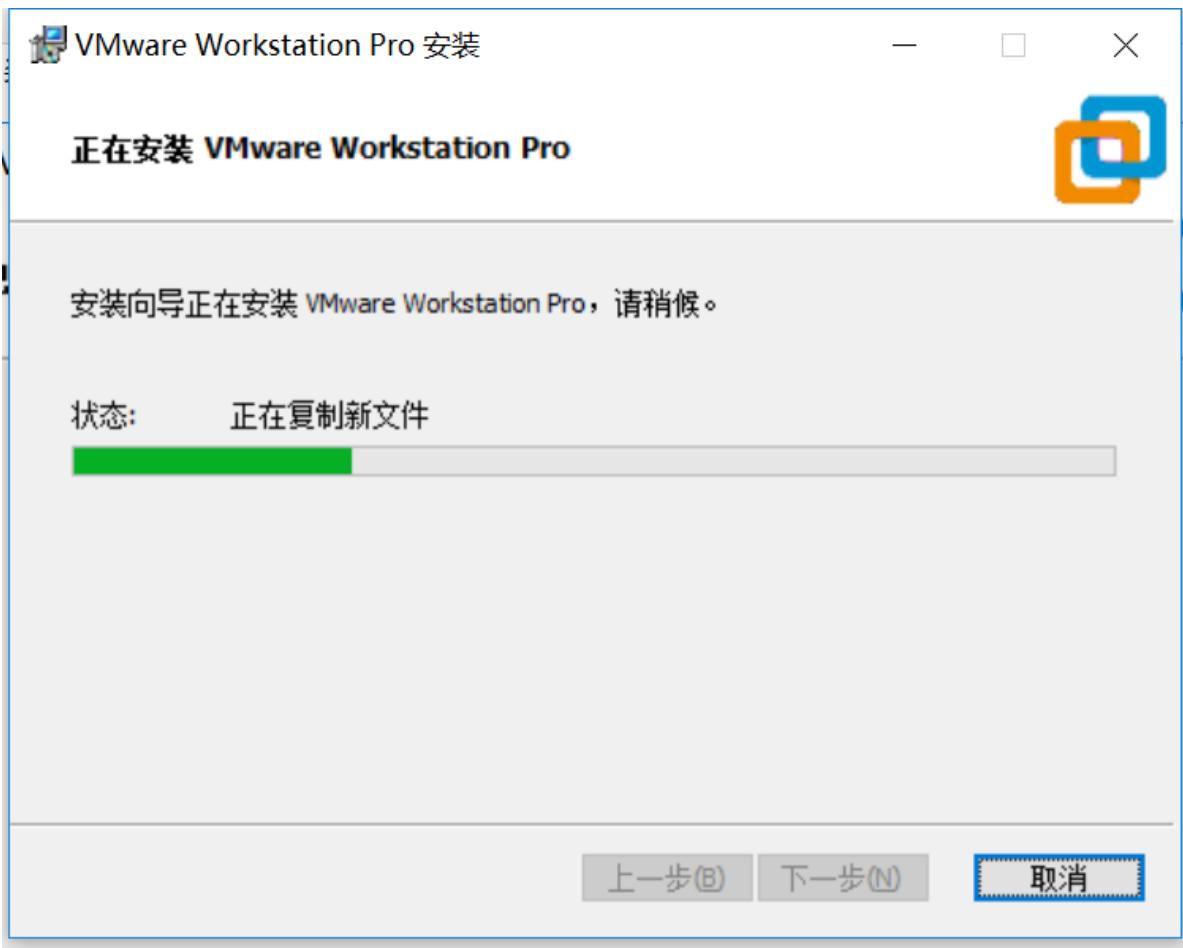


上一步(B)

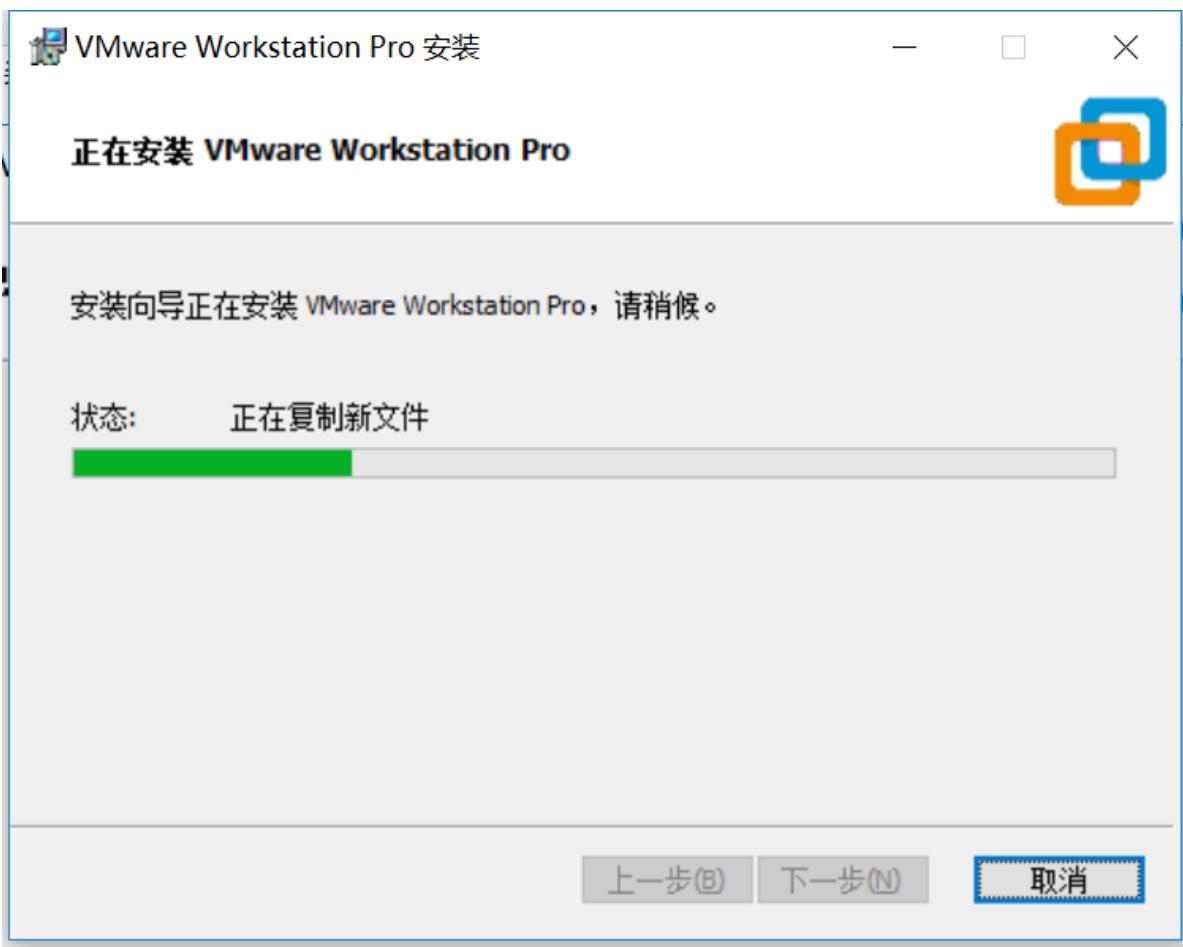
下一步(N)

取消

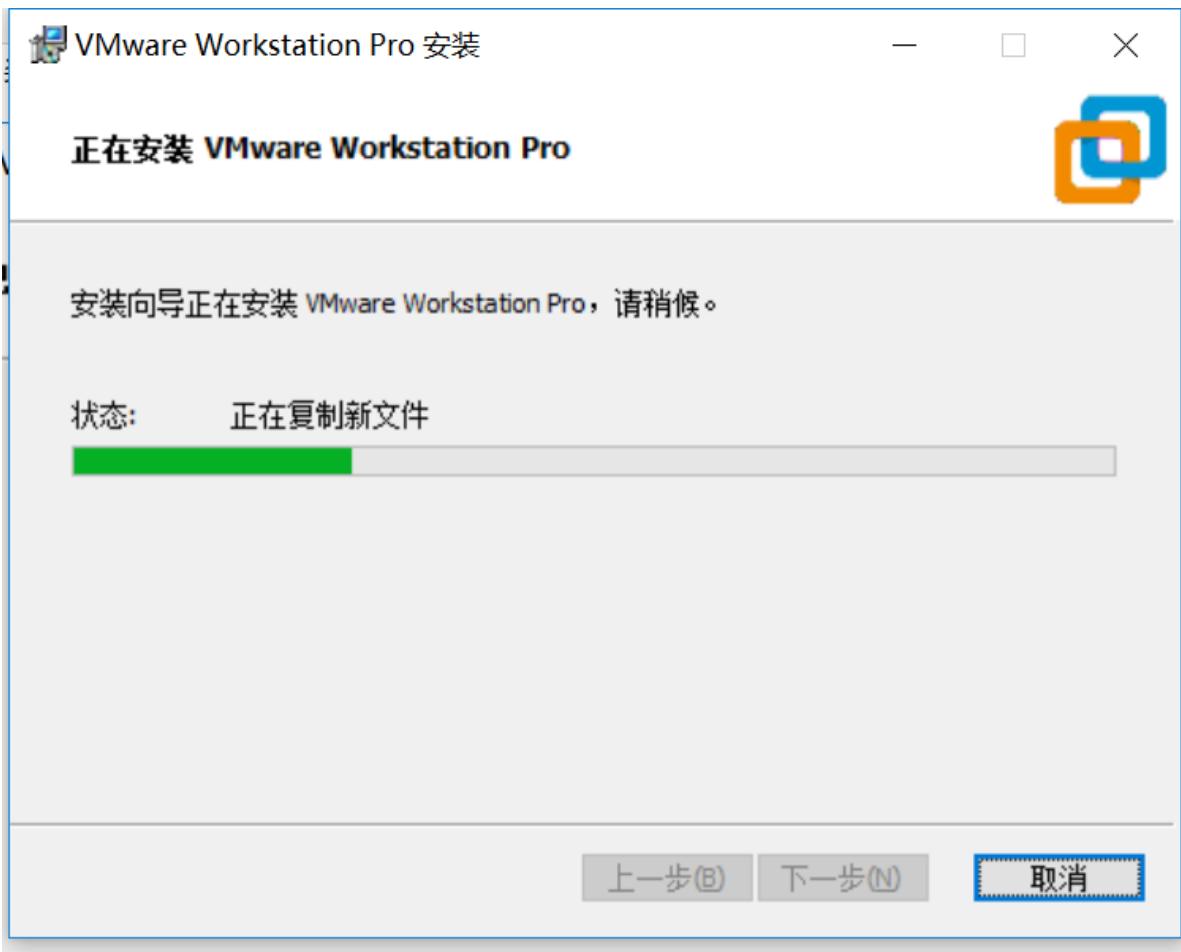
选择客户机操作系统，Linux，版本选择Ubuntu64位，下一步



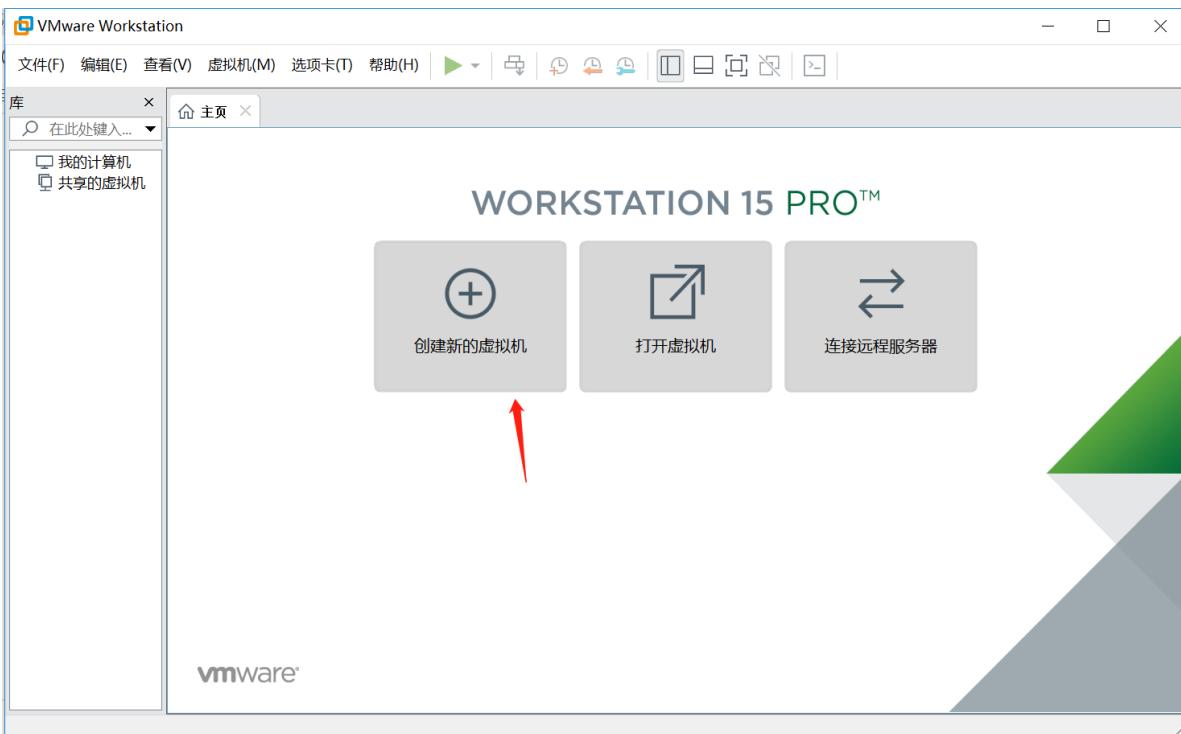
命名虚拟机并设置位置，选择预留的E盘，下一步



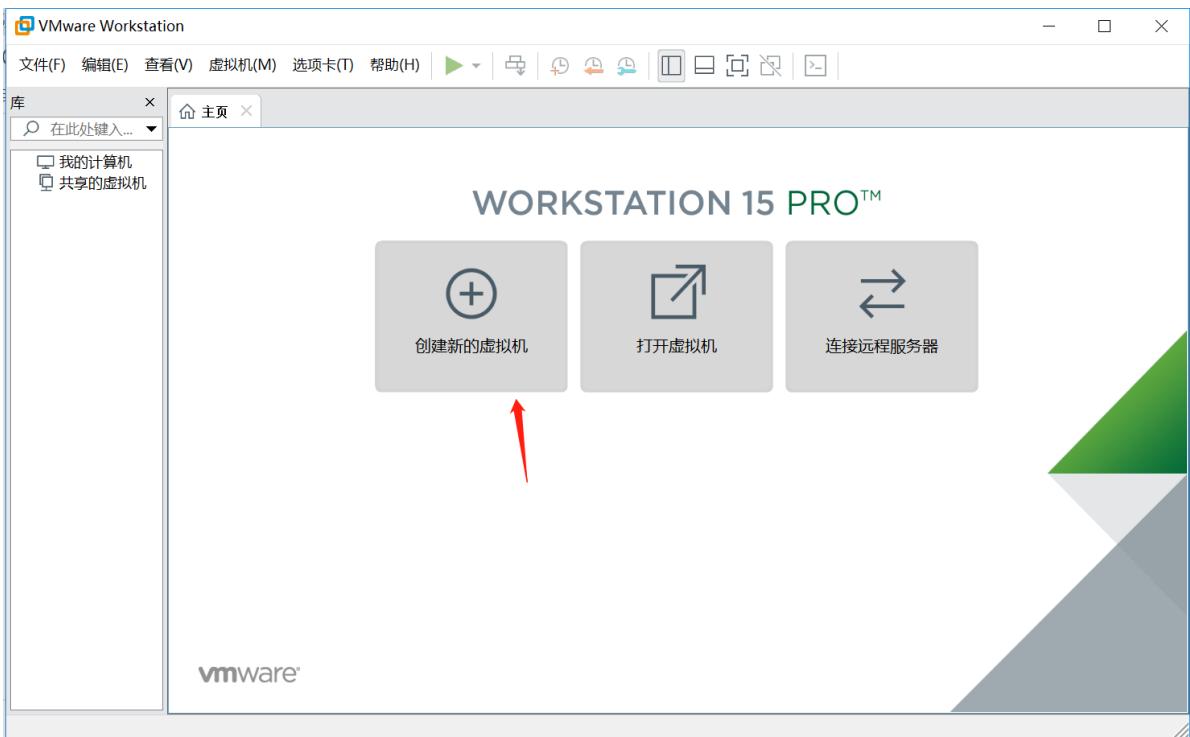
处理器配置，根据电脑的情况选择分配到虚拟机的处理器数量，下一步



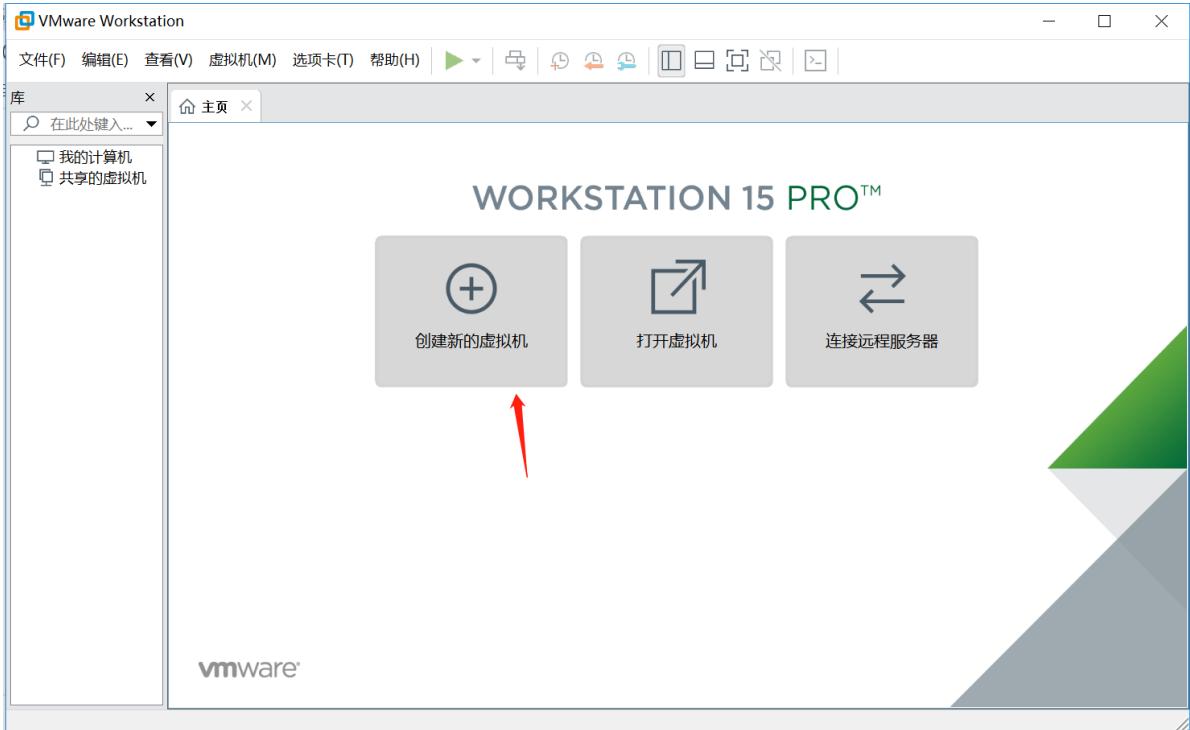
内存设置，根据电脑的情况设置虚拟机的内存量，本例设置8G，即8192M，下一步



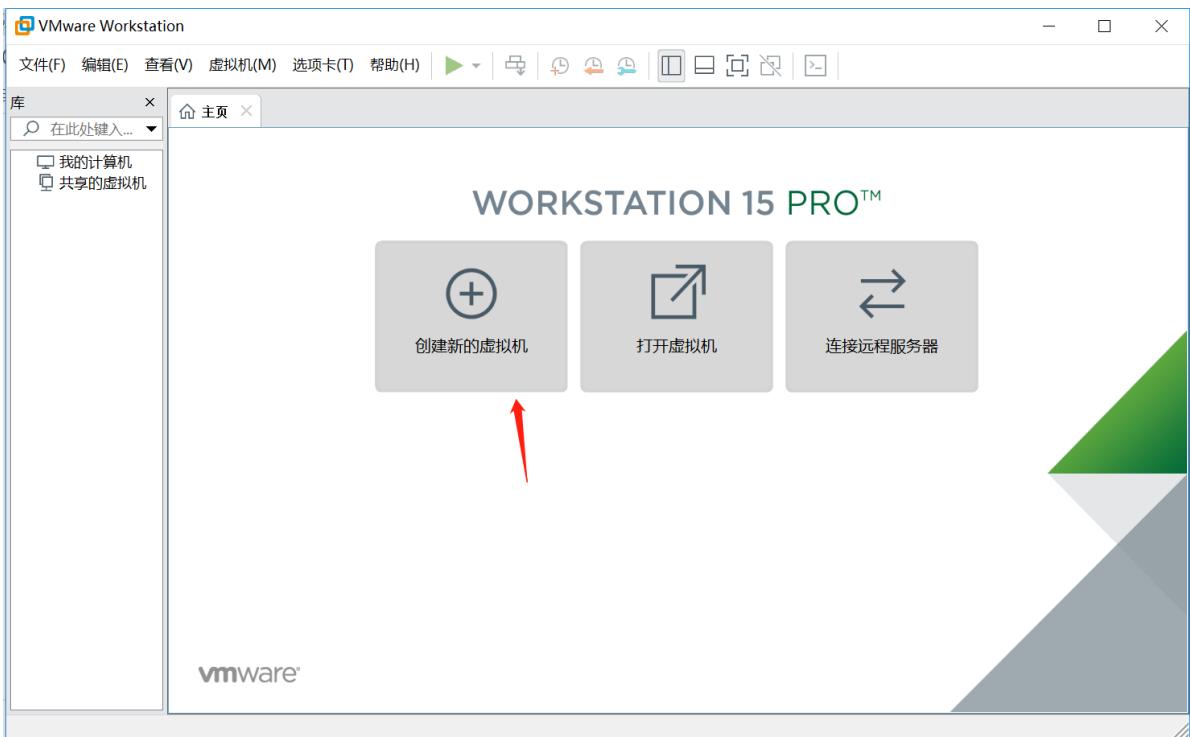
选择网络，使用桥接网络，下一步



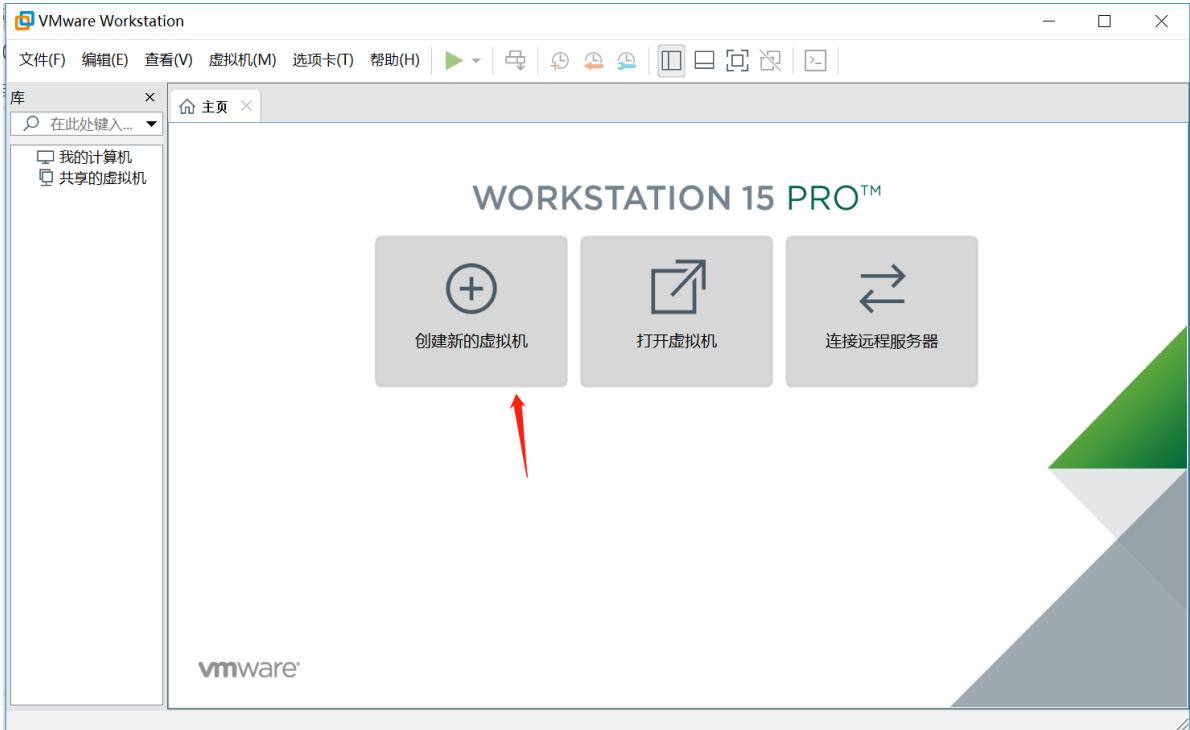
控制器类型，按推荐的选择，下一步



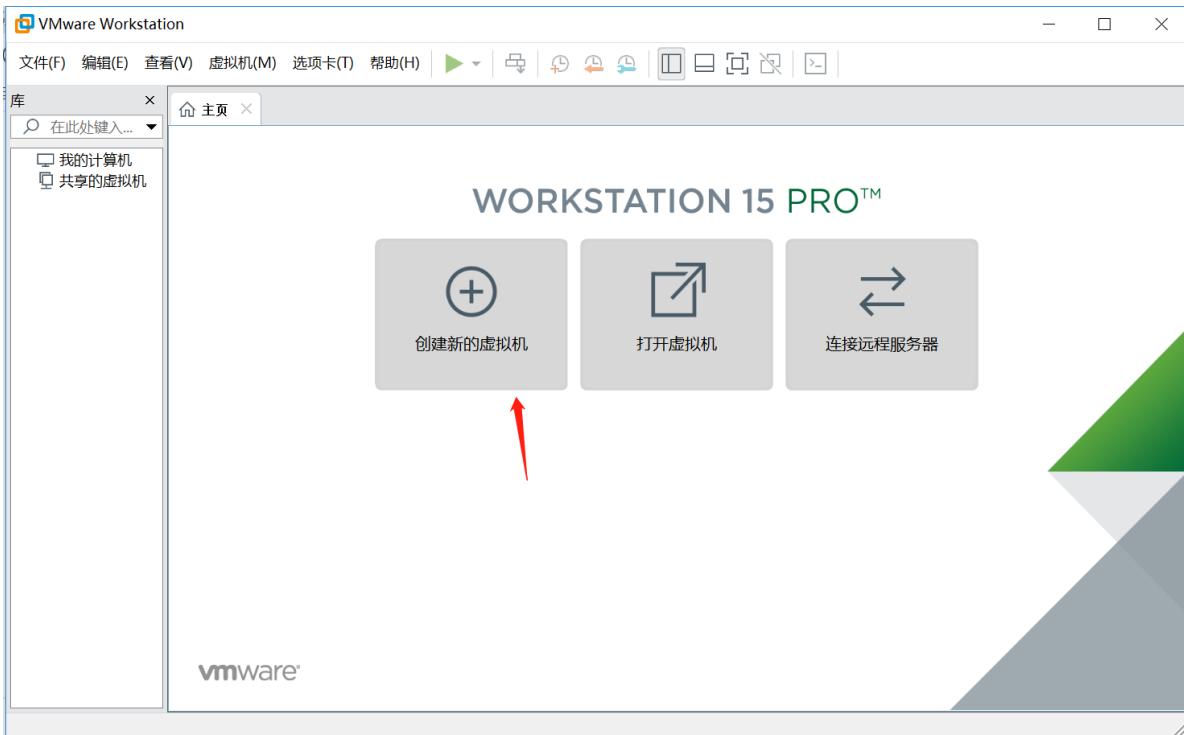
磁盘类型，按推荐的选择，下一步



选择磁盘，创建新虚拟磁盘，下一步



磁盘容量，根据电脑的情况设置磁盘大小，本例设置40G，选择储存为单个文件，下一步



指定磁盘文件，浏览选择文件位置并命名，下一步



准备好创建虚拟机，自定义硬件



VMware Workstation Pro 安装



## VMware Workstation Pro 安装向导已完成

VMWARE  
WORKSTATION  
PRO™  
**15**

单击“完成”按钮退出安装向导。

如果要立即输入许可证密钥，请按下面的“许可证”按钮。

许可证(L)

完成(F)

在CD/DVD中，找到之前下载好的Ubuntu iso镜像文件，关闭



VMware Workstation Pro 安装



## VMware Workstation Pro 安装向导已完成

VMWARE  
WORKSTATION  
PRO™  
**15**

单击“完成”按钮退出安装向导。

如果要立即输入许可证密钥，请按下面的“许可证”按钮。

许可证(L)

完成(F)

返回准备好创建虚拟机界面，完成

创建成功

- 启动Linux虚拟机，配置Ubuntu

开机此虚拟机



安装





Preparing to install, 选择Download updates while installing Ubuntu, continue

## 新建虚拟机向导



### 命名虚拟机

您希望该虚拟机使用什么名称?

虚拟机名称(V):

Ubuntu 64 位

位置(L):

E:\

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

Installation tape, 选择Erase disk and install Ubuntu, Install now

## 新建虚拟机向导



### 命名虚拟机

您希望该虚拟机使用什么名称?

虚拟机名称(V):

Ubuntu 64 位

位置(L):

E:\

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

在弹出的Write the changes to disk中选择continue

**命名虚拟机**

您希望该虚拟机使用什么名称?

虚拟机名称(V):

Ubuntu 64 位

位置(L):

E:\

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

Where are you, 选择中国地区, 它默认给定Shanghai, continue

**命名虚拟机**

您希望该虚拟机使用什么名称?

虚拟机名称(V):

Ubuntu 64 位

位置(L):

E:\

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

语言, 本例中选择English(US), Continue (可能是虚拟机上安装的原因, 这里显示不全, 可用tab键选择Continue)



## 命名虚拟机

您希望该虚拟机使用什么名称?

虚拟机名称(V):

Ubuntu 64 位

位置(L):

E:\

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

Who are you, 填写个人信息与密码, continue (可用tab键选择Continue)

## 新建虚拟机向导



### 命名虚拟机

您希望该虚拟机使用什么名称?

虚拟机名称(V):

Ubuntu 64 位

位置(L):

E:\

浏览(R)...

在“编辑”>“首选项”中可更改默认位置。

< 上一步(B)

下一步(N) >

取消

等待安装



Install (as superuser)

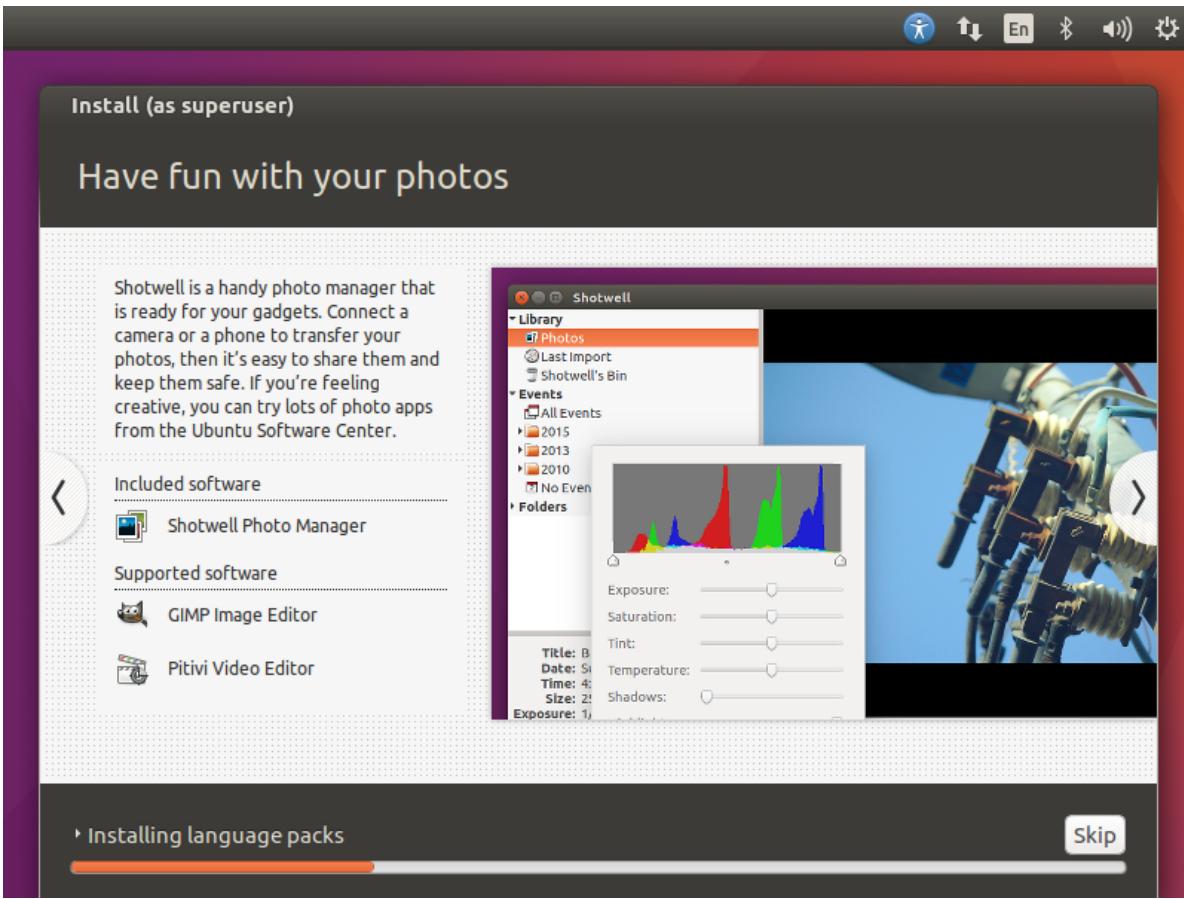
Welcome to Ubuntu

Fast and full of new features, the latest version of Ubuntu makes computing easier than ever. Here are just a few cool new things to look out for...

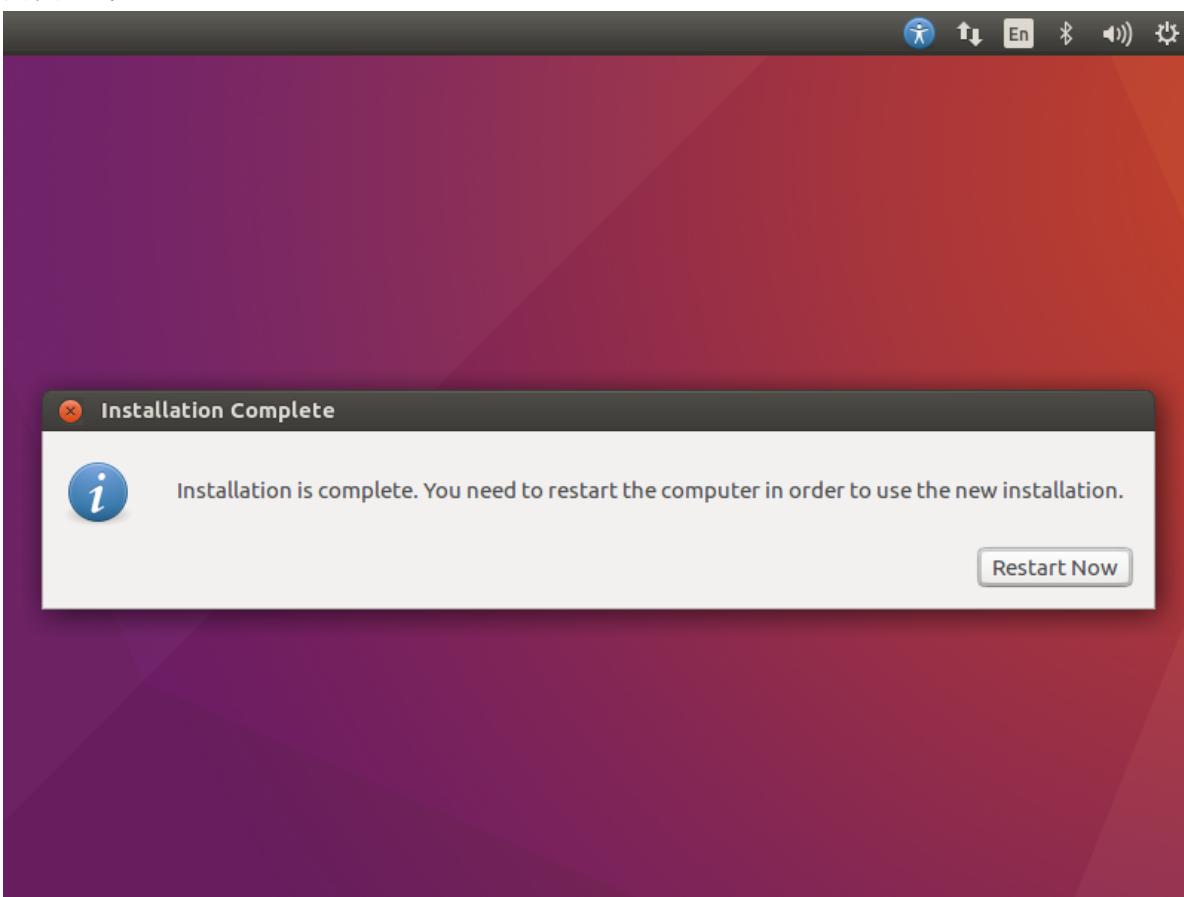


Installing system

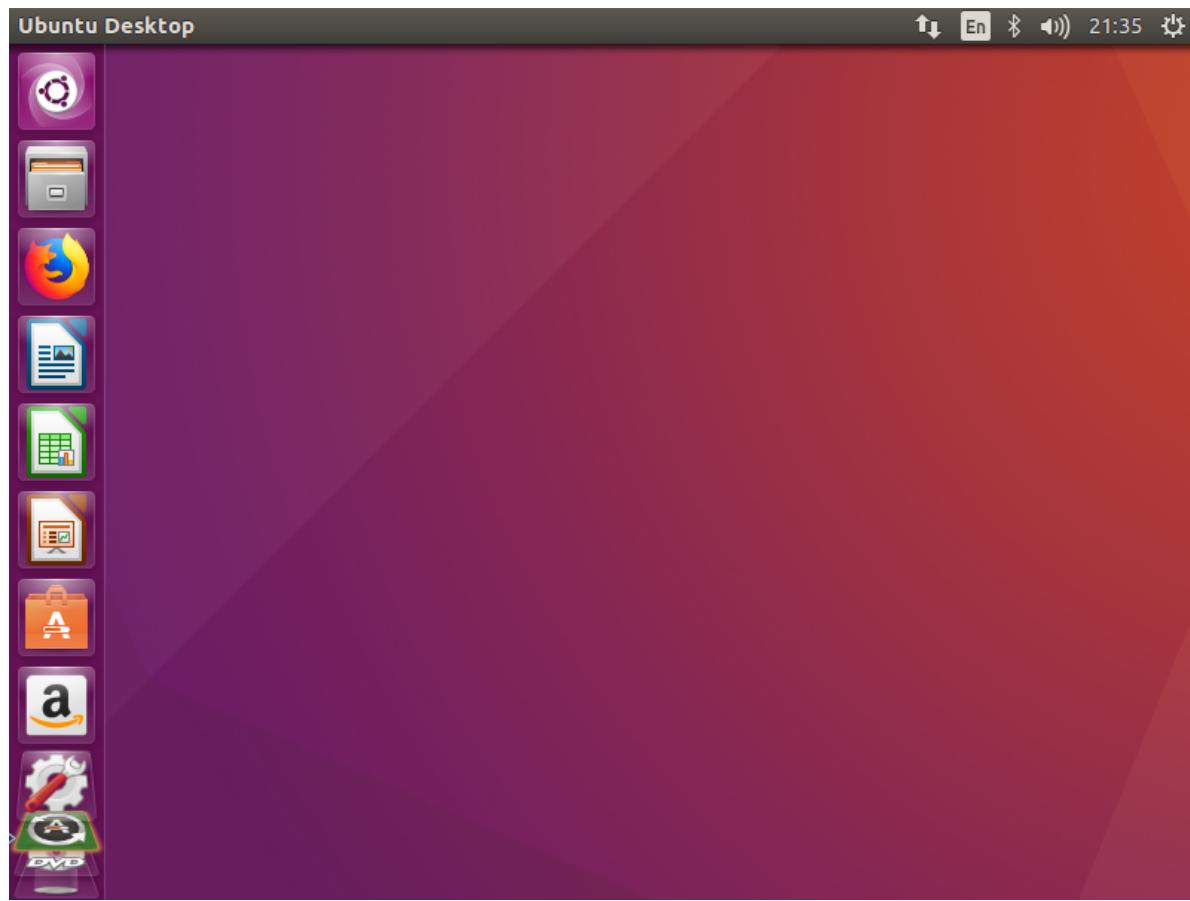
Skip



安装完成，Restart now



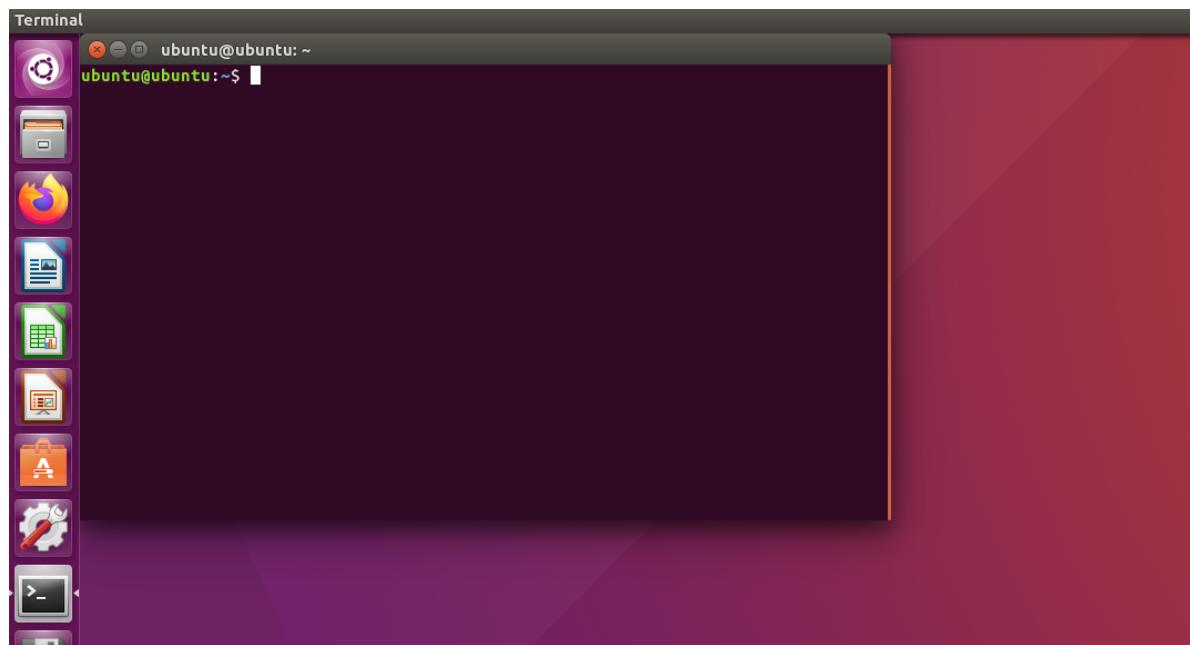
重启后输入密码登录账号即可使用



## UBUNTU初体验

### 安装c/c++环境

通过 右键->OPEN TERMINAL 打开终端



输入 sudo apt-get install build-essential

输入密码确认,如果还需确认输入 `y` 即可.

## 检查是否安装成功

输入 `gcc -v` 和 `g++ -v` 测试

```
ubuntu@ubuntu:~$ gcc -v
Using built-in specs.
COLLECT_GCC=gcc
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/5/lto-wrapper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1~16.04.12' --with-bugurl=file:///usr/share/doc/gcc-5/README.Bugs --enable-languages=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix=/usr --program-suffix=-5 --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --enable-libmpx --enable-plugin --with-system-zlib --disable-browser-plugin --enable-java-awt=gtk --enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-5-amd64/jre --enable-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-5-amd64 --with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-5-amd64 --with-arch-directory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --enable-multiarch --disable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.12)
ubuntu@ubuntu:~$ g++ -v
Using built-in specs.
COLLECT_GCC=g++
COLLECT_LTO_WRAPPER=/usr/lib/gcc/x86_64-linux-gnu/5/lto-wrapper
Target: x86_64-linux-gnu
Configured with: ../src/configure -v --with-pkgversion='Ubuntu 5.4.0-6ubuntu1~16.04.12' --with-bugurl=file:///usr/share/doc/gcc-5/README.Bugs --enable-languages=c,ada,c++,java,go,d,fortran,objc,obj-c++ --prefix=/usr --program-suffix=-5 --enable-shared --enable-linker-build-id --libexecdir=/usr/lib --without-included-gettext --enable-threads=posix --libdir=/usr/lib --enable-nls --with-sysroot=/ --enable-clocale=gnu --enable-libstdcxx-debug --enable-libstdcxx-time=yes --with-default-libstdcxx-abi=new --enable-gnu-unique-object --disable-vtable-verify --enable-libmpx --enable-plugin --with-system-zlib --disable-browser-plugin --enable-java-awt=gtk --enable-gtk-cairo --with-java-home=/usr/lib/jvm/java-1.5.0-gcj-5-amd64/jre --enable-java-home --with-jvm-root-dir=/usr/lib/jvm/java-1.5.0-gcj-5-amd64 --with-jvm-jar-dir=/usr/lib/jvm-exports/java-1.5.0-gcj-5-amd64 --with-arch-directory=amd64 --with-ecj-jar=/usr/share/java/eclipse-ecj.jar --enable-objc-gc --enable-multiarch --disable-werror --with-arch-32=i686 --with-abi=m64 --with-multilib-list=m32,m64,mx32 --enable-multilib --with-tune=generic --enable-checking=release --build=x86_64-linux-gnu --host=x86_64-linux-gnu --target=x86_64-linux-gnu
Thread model: posix
gcc version 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.12)
```

成功则会返回上述的版本信息

## 测试

我们可以编写一个程序来进行测试

```
1 ubuntu@ubuntu:~$ gedit test.cpp #编辑一个test.cpp文件,如果没有就创建一个
2 ubuntu@ubuntu:~$ g++ test.cpp -o test.o # 编译test.cpp ,编译后的文件名为test.o
3 ubuntu@ubuntu:~$./test.o # 执行test.o
4 Hello,World!
```

其中,我们向 `test.cpp` 写入了以下内容

```
1 #include<iostream>
2 using namespace std;
3
4 int main(){
5 cout<<"Hello,World!"<<endl;
6 return 0;
7 }
```

输出为 `Hello,World!`

## 任务

你可能一定觉得上面几步很容易就完成了,所以我布置了以下的任务.

“

在Ubuntu上安装GMP,NTL库,并尝试编写测试程序

参考链接(官网)

[The GNU MP Bignum Library \(gmplib.org\)](http://gmplib.org)

[NTL: A Library for doing Number Theory \(libntl.org\)](http://libntl.org)

## 任务总结

本次的任务很少,主要有以下内容:

- 使用hexo搭建自己的博客
- 使用markdown写一篇文章,内容不限,并发布到你的博客上
- 安装ubuntu
- 在Ubuntu上安装GMP,NTL库
- 经量测试上述的两个库

由于学生邮箱申请看个人想法,所以不列入必须.写这么多繁琐的内容主要是让大家体验和习惯配置环境.

## 考核

完成度80%以上

## 建议

不要满足于我给的教程,不要认为我给的教程一定是对的,不要认为我给出的教程一定的最优的解决方案,多去探索和发现.

如果你觉得我的行文令你难受,欢迎提出意见.

如果你觉得完成我提出的任务是一种折磨,请立即停止,去做你自己觉得更有意义的事情,不要折磨自己:)

如果在完成过程中,有任何困难,请先自行通过各种渠道解决,我仅提供**有限**的帮助

## 如何提交

不需要提交,若近期有交流,骄傲的展示你的成果即可.

## 我希望收到的

你在任务完成过程的任何吐槽,你的宝贵经验

## Deadline

两周