

给信息组学弟学妹的 Linux 入门手把手教程

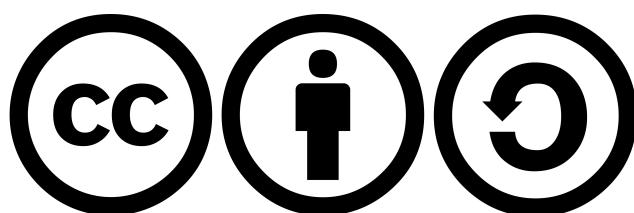
iotang

2021 年 7 月 18 日



本文的 4 个要素

本作品所有内容采用[知识共享署名-相同方式共享 4.0 国际许可协议](#)进行许可，
以兼容 GNU LESSER GENERAL PUBLIC LICENSE Version 3 协议。



目录

1 欢迎使用 Linux!	5
1.1 Linux 是开源软件	5
1.2 为什么是 Ubuntu?	6
2 Ubuntu 的安装	7
2.1 下载镜像文件	7
2.1.1 我该选哪一个?	7
2.1.2 下载的速度太慢了?	7
2.2 制作启动盘	9
2.2.1 在 Linux 下	9
2.2.2 在 Windows 下	10
2.3 安装 Ubuntu	10
2.3.1 BIOS 设置	10
2.3.2 开始安装	10
3 初始设置	15
3.1 终端命令的使用	15
3.1.1 基本快捷键	15
3.1.2 基本命令	15
3.1.3 对于文件和目录的命令	15
3.1.4 对于系统状态的命令	16
3.2 第一次更新软件	16
3.2.1 更换软件源	16
3.2.2 apt 是什么?	17
3.2.3 更新软件列表	17
3.2.4 更新软件	18
3.2.5 查找软件	18
3.2.6 安装软件	18
3.3 安装中文输入法	19
3.3.1 添加中文语言支持	19
3.3.2 切换到 fcitx	21
3.3.3 安装搜狗输入法	23
4 完善工作环境	24
4.1 在右键菜单中添加“新建文件”	24
4.2 使用 Code::Blocks 作为 C++ IDE	24
4.2.1 安装	24
4.2.2 新建一个 C++ 文件	24
4.2.3 配置 Code::Blocks	26
4.2.4 编译和运行程序	29
4.3 以 Emacs 作为 C++ 代码编辑器	31
4.3.1 安装	31
4.3.2 配置 Emacs	31

4.3.3 考场上在没有 Emacs 配置的时候配置 Emacs	32
4.3.4 Emacs 初步	37
4.3.5 在 Emacs 中进行文件操作	39
4.3.6 分屏和操作 Buffer	39
4.3.7 使用 Eshell 操作终端	39
4.4 用 CrossOver 运行 Windows 程序	41
4.4.1 安装	41
4.4.2 在 CrossOver 中安装 QQ	42
4.4.3 * 闸总行为：无限试用 CrossOver	45
4.5 用 Typora 编写 Markdown 文档	46
4.5.1 安装 Typora	46
4.5.2 Markdown 语法	47
4.6 用 WPS Office 编辑 Word、Excel 和 PowerPoint	50
4.6.1 安装	50
4.7 以 LemonLime 作为评测软件	51
4.7.1 使用 .deb 安装包安装 LemonLime	51
4.7.2 使用源代码编译最新的 LemonLime	51
4.7.3 LemonLime 的使用	51
5 练习：完善 Ubuntu 及个性化	52
5.1 校准系统时间	52
5.2 关闭 SSH 功能和端口	52
5.3 更改壁纸	52
5.4 更改系统颜色主题	52
5.5 更改终端颜色主题	52
5.6 不使用 Firefox 作为浏览器	52
5.7 自定义默认程序	52
5.8 在 Firefox 或其它浏览器中安装扩展程序	52
5.9 使用 GeoGebra 作为数学作图软件	53
5.10 使用 GIMP 作为图像编辑器	53
5.11 使用 Krita 作为绘图程序	53
5.12 使用 TeXstudio 与 XeLaTeX 编辑与编译 TeX 文档	53
5.13 在外部介质中备份你的 Ubuntu	53
5.14 使用 Ubuntu 在洛谷上切一道新题	53
5.15 使用 Ubuntu 完成一场模拟赛	53
6 附录	54
6.1 NOI Linux 2.0 系统情况简表	54

1 欢迎使用 Linux!

学弟学妹们好！感谢你们参加信息学竞赛！
比克提尼 iotang 在这里献上最诚挚的祝福！



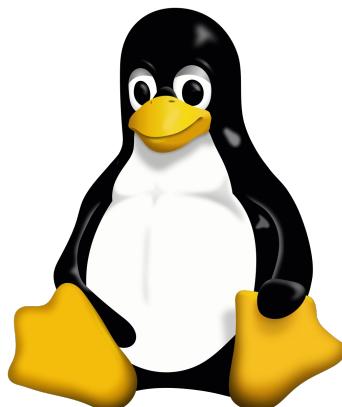
iotang 与 *Konqi* (左上、右上)、*Kiki* (左下) 和 *Kate* (右下) 向你发出问候

大家应该都对 Linux 有所耳闻，不过我猜有许多人都以为 Linux 很难用、不好用而不敢迈出第一步。现在，就让我来手把手带你们入门 Linux！

1.1 Linux 是开源软件

Linux 遵循 GNU 通用公共许可证，任何人都可以自由使用它的源代码。

(注意，这里并没有说 Linux 是“不要钱”地使用的，不过既然你都可以搞到源代码了，那么收费基本也就没必要了。不过还是有服务以付费的方式给出。)



Linux 的吉祥物 *Tux*

1.2 为什么是 Ubuntu ?

Ubuntu 是 Linux 操作系统中的一种。接下来给大家带来的 Linux 教程主要是以 Ubuntu 为平台来实现的。

首先很明显的是，NOI Linux 就是一个换皮的 Ubuntu。至于为什么是 Ubuntu，可能与 Ubuntu 在中国的强大的用户数量有关。

The screenshot shows a web browser window with the URL <https://www.noicn.org/gynoi/jsgz/2021-07-16/732450.shtml>. The page title is "NOI 全国青少年信息学奥林匹克竞赛". The main content is an article titled "NOI Linux 2.0发布，将于9月1日起正式启用！" (NOI Linux 2.0 released, will be officially used from September 1st!). It includes a timestamp "2021-07-16 12:59:40" and a view count "阅读量: 6155". The text states: "经过多轮开发和内部测试，NOI Linux 2.0版 (Ubuntu-NOI 2.0版) 已经基于Ubuntu 20.04.1版定制完成，现正式对外发布。根据NOI科学委员会决议，该系统将自2021年9月1日起作为NOI系列比赛和CSP-J/S等活动的标准环境使用。在此日期前，NOI相关活动标准环境仍为旧版NOI Linux。" Below the text is a graphic featuring a penguin wearing a red scarf with the text "NOI LINUX 2.0" and "NOI LINUX 2.0". A note below the graphic says: "该系统将自2021年9月1日起作为NOI系列比赛和CSP-J/S认证等活动的标准环境使用。" At the bottom of the page, there is a note: "系统下载链接: NOI Linux 2.0版 (注意: 安装系统时请断开网络)" and "欢迎各位老师和选手试用新系统，并提出改进建议和意见。意见反馈邮箱: NOI竞赛办公室 (noi@ccf.org.cn) 。"

NOI Linux 2.0 也基于 *Ubuntu 20.04*

用户多教程就多，问题解决也方便，不像笔者硬是要搞个 Arch Linux 然后折腾。

所以说，从竞赛与使用方面，这边还是建议大家用 Ubuntu。



Ubuntu 的标志

2 Ubuntu 的安装

2.1 下载镜像文件

非常简单，你只要先百度 ubuntu，进入官网（注意：有中文官网），然后进入下载栏目下载就可以了。

2.1.1 我该选哪一个？

在下载界面你可以发现一些不同版本：



两种选择

其中，带“LTS”的版本意为长期支持版本，有 5 年的免费安全和维护更新时间；而不带 LTS 的一般只维护 9 个月，因为带 LTS 的版本每 2 年发布一个，而不带 LTS 的每半年就发布一个，你需要及时更新。

这里为了稳定性，我们下载那个带 LTS 的版本。

2.1.2 下载的速度太慢了？

你可以去其它的镜像网站。这里以网易开源镜像站为例子：

首先随便搜到它的主页。

镜像名	上次更新时间	使用帮助
archlinux/	2021-07-09 11:37	archlinux使用帮助
archlinux-cn/	2021-07-09 11:46	archlinux-cn使用帮助
archlinuxarm/	2021-07-09 12:08	-

http://mirrors.163.com/

然后找到 `ubuntu-releases`。

ubuntu-releases

找到 *ubuntu-releases*

然后选择正确的版本。

Index of /ubuntu-releases/

..		
14.04/	18-Aug-2020 16:05	-
14.04.6/	18-Aug-2020 16:05	-
16.04/	19-Aug-2020 01:01	-
16.04.6/	19-Aug-2020 01:01	-
16.04.7/	19-Aug-2020 01:01	-
18.04/	13-Aug-2020 23:39	-
18.04.4/	13-Aug-2020 23:39	-
18.04.5/	13-Aug-2020 23:39	-
20.04/	15-Feb-2021 16:47	-
20.04.2/	15-Feb-2021 16:47	-
20.04.2.0/	15-Feb-2021 16:47	-
20.10/	23-Oct-2020 01:11	-
21.04/	23-Apr-2021 03:34	-
bionic/	13-Aug-2020 23:39	-

/ubuntu-releases/

下载桌面版，即名字里面有 `desktop` 的那个。

Index of /ubuntu-releases/20.04.2.0/

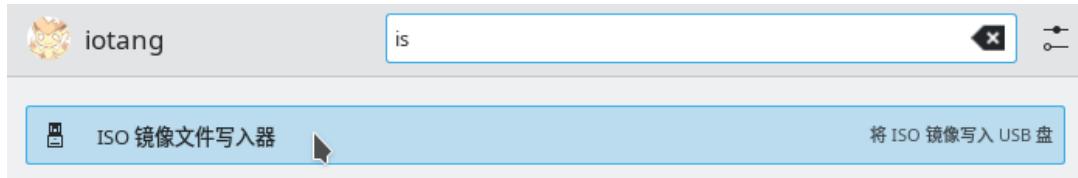
..		
FOOTER.html	12-Feb-2021 03:02	810
HEADER.html	15-Feb-2021 16:47	4007
SHA256SUMS	12-Feb-2021 03:07	204
SHA256SUMS.gpg	12-Feb-2021 03:07	833
ubuntu-20.04.2-live-server-amd64.iso	02-Feb-2021 01:58	1G
ubuntu-20.04.2-live-server-amd64.iso.torrent	05-Feb-2021 01:46	91K
ubuntu-20.04.2-live-server-amd64.iso.zsync	05-Feb-2021 01:46	2M
ubuntu-20.04.2-live-server-amd64.list	02-Feb-2021 01:58	9680
ubuntu-20.04.2-live-server-amd64.manifest	02-Feb-2021 01:49	16K
ubuntu-20.04.2.0-desktop-amd64.iso ←	10-Feb-2021 03:07	3G
ubuntu-20.04.2.0-desktop-amd64.iso.torrent	12-Feb-2021 03:02	215K
ubuntu-20.04.2.0-desktop-amd64.iso.zsync	12-Feb-2021 03:02	5M

选择桌面版

2.2 制作启动盘

2.2.1 在 Linux 下

系统但凡是有点良心都会自带一个启动盘创建器。比如笔者的：



KDE 下的启动盘创建器

此时，你需要准备一个 U 盘（最好至少 8 GB，并且笔者建议这个 U 盘应该是空的，以确保**没有重要文件在里面被抹去**，因为制作启动盘时 U 盘里面的所有内容都会丢失。）

启动盘创建器的用法基本都一样。注意不要选错 U 盘。



KDE 启动盘创建器

2.2.2 在 Windows 下

去下载 Rufus 启动盘创建器。



Rufus

2.3 安装 Ubuntu

我们马上会在目标电脑（机房里面的一台）上安装 Ubuntu。

2.3.1 BIOS 设置

首先打开目标电脑，从屏幕亮起来（甚至从电源键按下前）开始狂按 BIOS 键（比如 F12、ESC、DEL 等等）。在 BIOS 设置中打开 U 盘启动，打开 UEFI 模式优先（原先一般是 Legacy 优先）。

2.3.2 开始安装

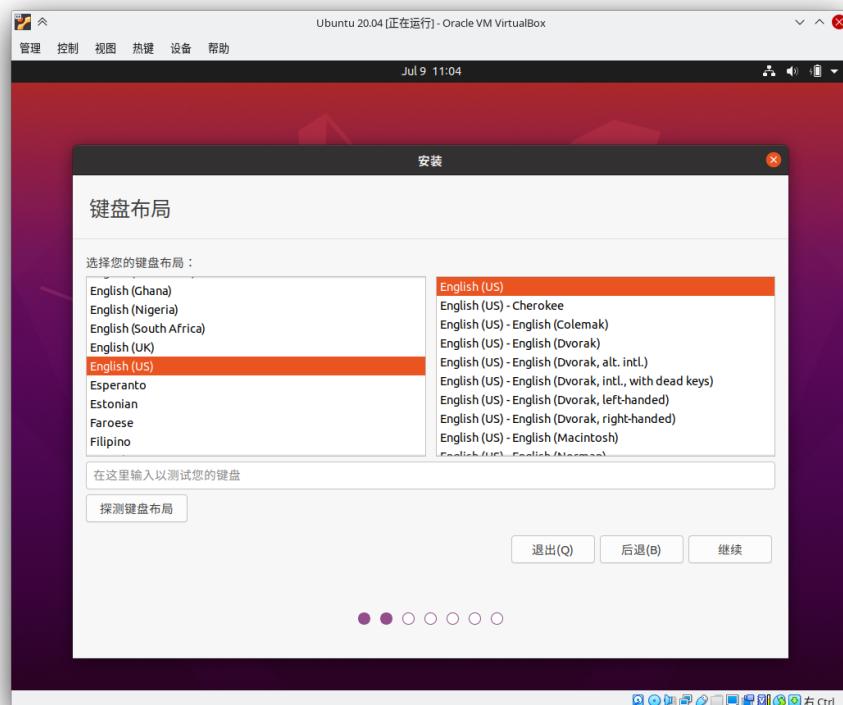
关机，插上启动盘，开机时仍然狂按 BIOS 键，之后会出来一个界面让你选择启动位置。选择你的 U 盘（一般叫 USB-HDD 什么的）。

之后在试用 Ubuntu 和安装 Ubuntu 中选择安装 Ubuntu。



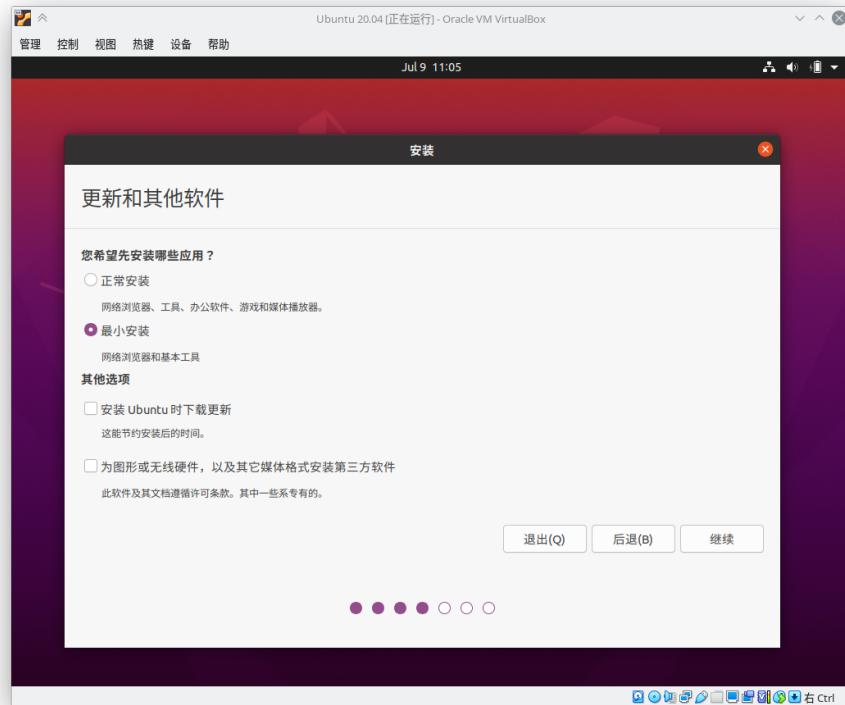
安装 *Ubuntu*

选择键盘布局为 English (US)。



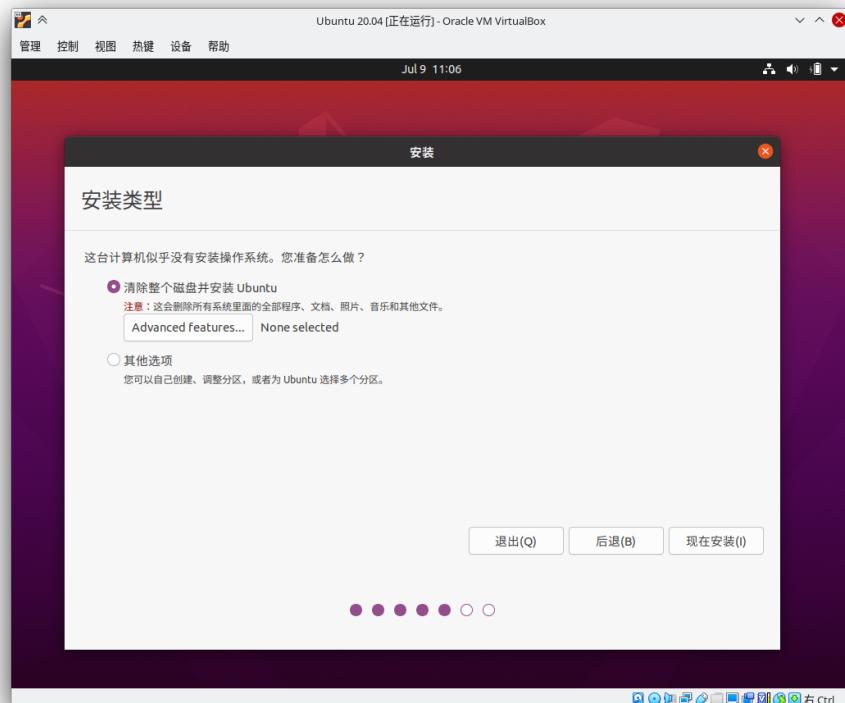
选择键盘布局

建议先选择“最小安装”，并且不选“安装 Ubuntu 时下载更新”。



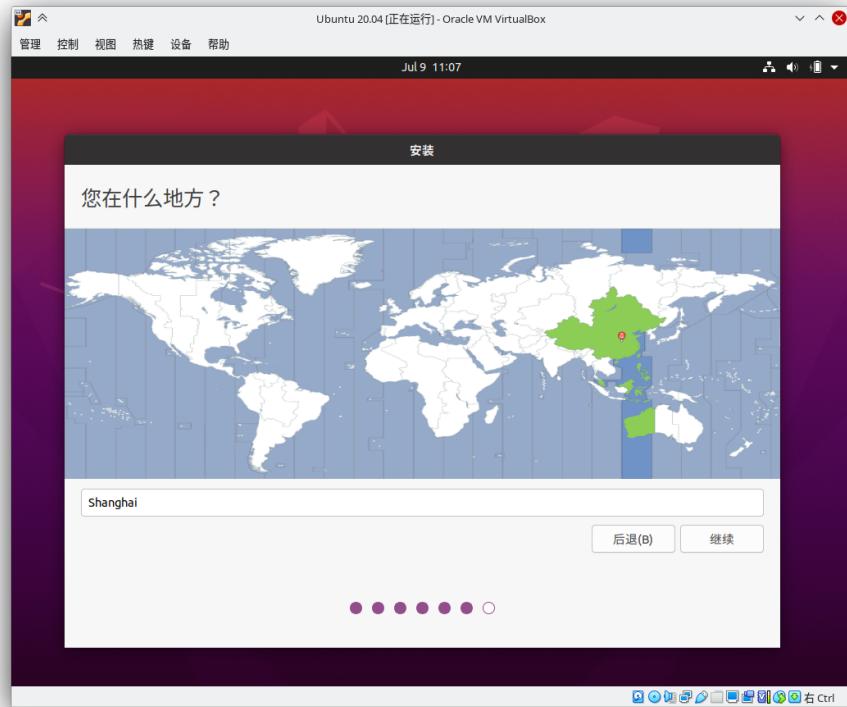
更新和其他软件

根据需要选择安装类型。



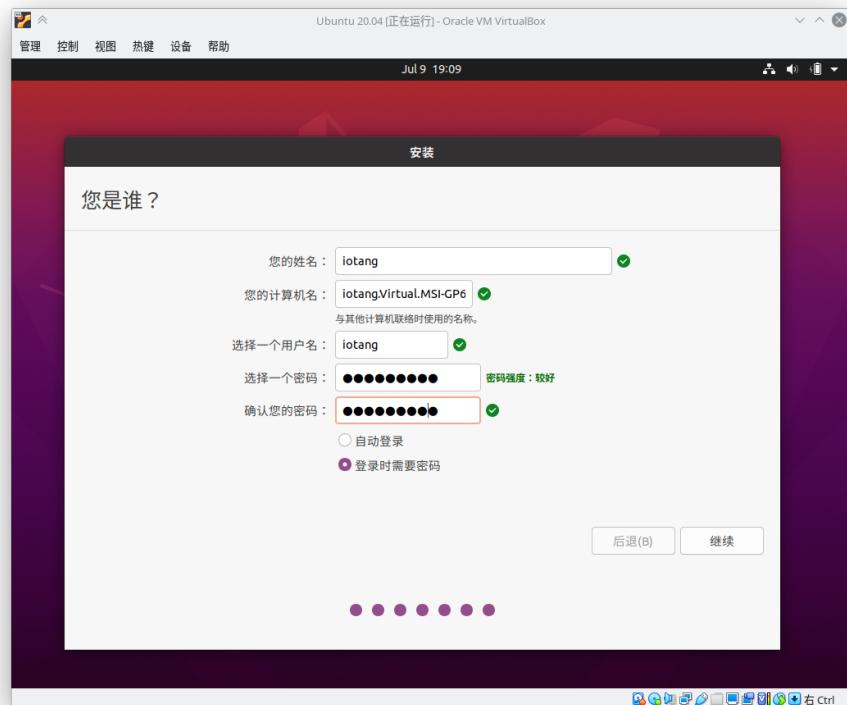
安装类型

选择时区。



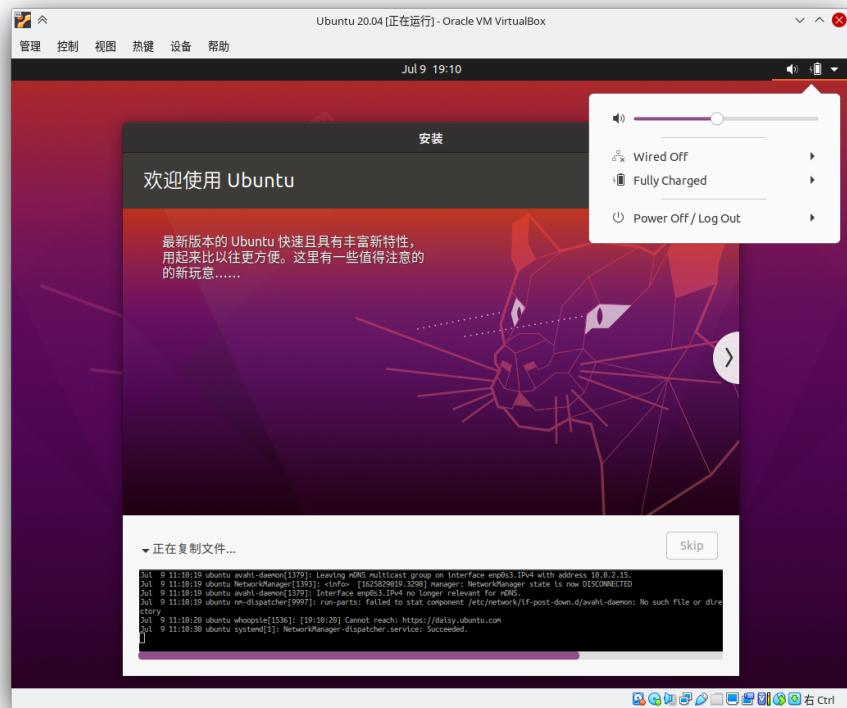
我是中国弗兰常杀人

设置第一个用户。



第一个用户一定是管理员

建议把网络关掉，以防安装程序用巨慢的速度下载一些东西，把安装时间拖得很长。



关闭网络

之后关机，然后拔掉启动盘后开机。

准备迎接 Ubuntu！第一次启动可能非常慢。

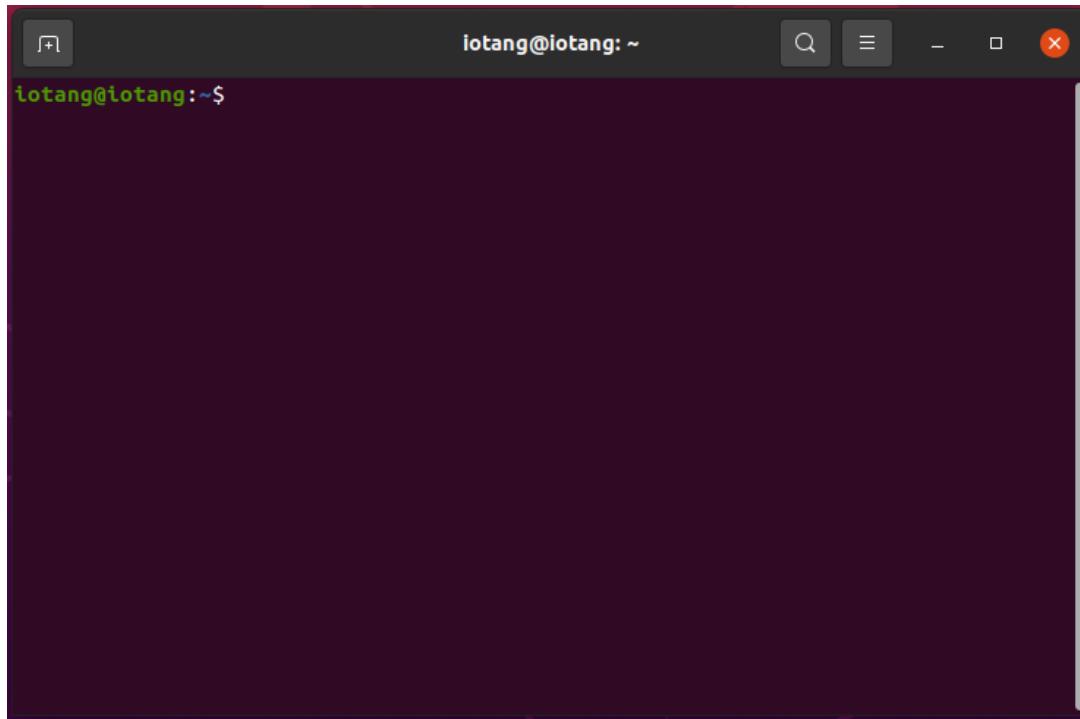
在上图中你可以看到一只哺乳动物。这是 Focal Fossa，Ubuntu 20.04 的吉祥物。“Fossa”是马达加斯加长尾灵猫。

3 初始设置

3.1 终端命令的使用

无论如何，你必须得学会终端的使用方式。

用快捷键 **Ctrl + Alt + T** 打开一个终端。



一个终端

在“\$”前面的内容是这样：

用户名@主机名：当前目录\$
iotang@iotang:~\$

3.1.1 基本快捷键

Tab 键 自动补全。

Ctrl + c 中断当前正在运行的程序。

Ctrl + Shift + c 复制。

Ctrl + Shift + v 粘贴。

3.1.2 基本命令

sudo 以管理员权限运行之后的命令，即 **Super User Do**。

man 查询手册。比如查询命令 **sudo**：输入 **man sudo**。请通过自己的能力找到以下命令的详细用法，并自己练习。

3.1.3 对于文件和目录的命令

~ 家目录，即 **/home/** 你的用户名。默认打开终端进入的就是你的家目录。

. 现在的目录。

.. 上一级目录。

`pwd` 显示当前位置，即 print working directory。

`ls` 列出当前目录下的文件与目录，即 list。

`ls -l` 列出当前目录下的文件与目录，显示文件相关属性。

`ls -a` 列出当前目录下的文件与目录，包括隐藏文件。

`ls -la` 列出当前目录下的文件与目录（包括隐藏文件），并显示文件相关属性。

`cd` 改变目录，即 change directory。

`cd ..` 进入上一级目录。

`cp` 复制，即 copy。

`mv` 移动，即 move。

`rm` 删除，即 remove。不可撤销！

`mkdir` 创建目录，即 make directory。

`rmdir` 删除目录。

3.1.4 对于系统状态的命令

`df` 显示文件系统中还有多少剩余空间。

`df -h` 显示文件系统中还有多少剩余空间，用兆字节和吉字节为单位来显示设备空间使用量。`-h` 的 `h` 是 human-readable 的意思，因为默认是用千字节为单位来表示使用量的。

`free` 显示内存使用情况。

`free -m` 以兆字节为单位显示内存使用情况。

`uname -a` 显示所有的系统信息。

`lsb_release -a` 显示当前 Ubuntu 版本。

3.2 第一次更新软件

3.2.1 更换软件源

更换 Ubuntu 的软件源到国内某一个，否则软件安装和更新的速度将非常慢，因为默认源地址不在国内。这边以清华大学开源软件镜像站为例子：

百度到清华大学开源软件镜像站 Ubuntu 镜像使用帮助。

The screenshot shows a web browser displaying the 'Tsinghua University Open Source Software Mirror Station' website at <https://mirror.tuna.tsinghua.edu.cn/help/ubuntu/>. The page title is 'Ubuntu 镜像使用帮助'. It features several links: AOSP, AUR, AdoptOpenJDK, CPAN, CRAN, and CTAN. A dropdown menu for selecting the Ubuntu version is set to '20.04 LTS'. A text box contains a snippet of a configuration file:

```
# 默认注释了源码镜像以提高 apt update 速度，如有需要可自行取消注释
deb https://mirrors.tuna.tsinghua.edu.cn/ubuntu/ focal main restricted universe multiverse
```

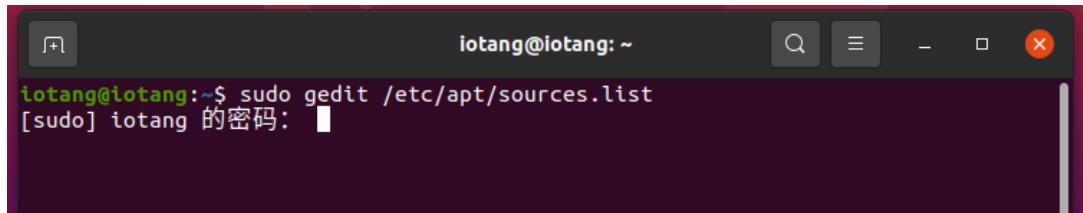
贵校镜像站 Ubuntu 镜像使用帮助

选择正确的 Ubuntu 版本，复制镜像内容，然后编辑 `/etc/apt/sources.list`:

```
$ sudo gedit /etc/apt/sources.list
```

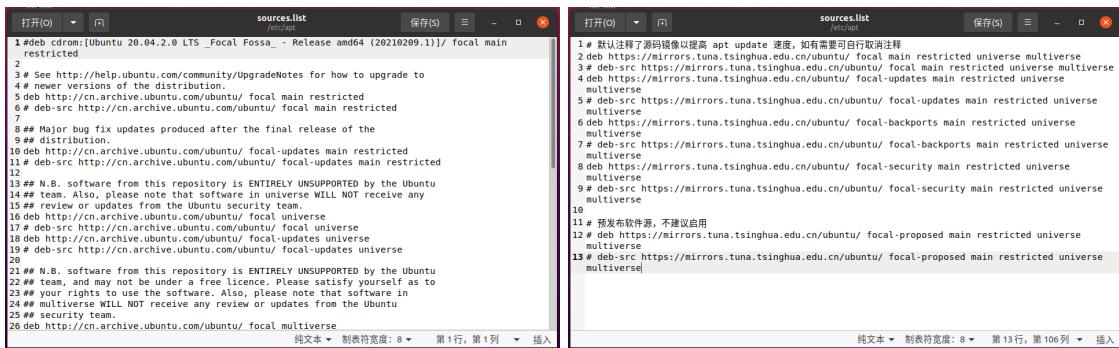
gedit 是 GNOME 桌面下的一个文本编辑器，即 **Gnome Edit**。

从这里开始，以“\$”开头的东西代表你要在终端中执行这个语句（执行的语句里面没有“\$”）。比如对于上面那个命令，你可以先用快捷键 **Ctrl + Alt + T** 打开一个终端，然后输入 **sudo gedit /etc/apt/sources.list**（没有“\$”）：



打开 */etc/apt/sources.list*

用你找到的镜像内容替换文件里原来的所有内容。



替换前

替换后

3.2.2 apt 是什么？

注意到了上文的 */etc/apt/sources.list* 中的 apt 了吗？

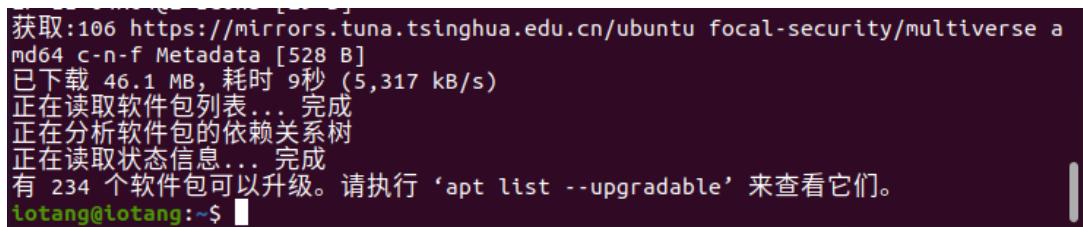
apt 是 Linux 下的安装包管理工具之一。Ubuntu 使用 apt。

3.2.3 更新软件列表

```
$ sudo apt update
```

更新软件列表可以让 apt 知道现在有哪些软件，以及那些软件的版本。

apt 会将软件列表和目前的状况比对，然后就可以得出哪些软件可以更新。

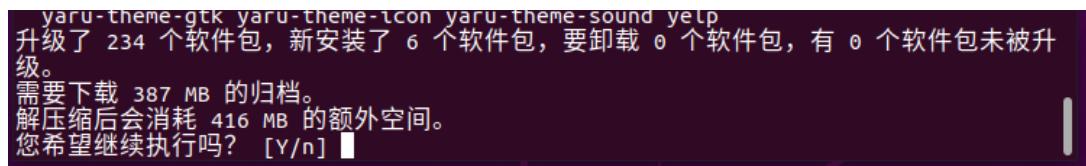


apt 找到了好多可以更新的软件

3.2.4 更新软件

```
$ sudo apt upgrade
```

这可以让 apt 更新目前所有的软件。



确定界面

在这里，[Y/n] 的 Y 是大写，意味着默认是“是”。也就是说，如果你在这里直接按下回车，那么就按“是”处理。

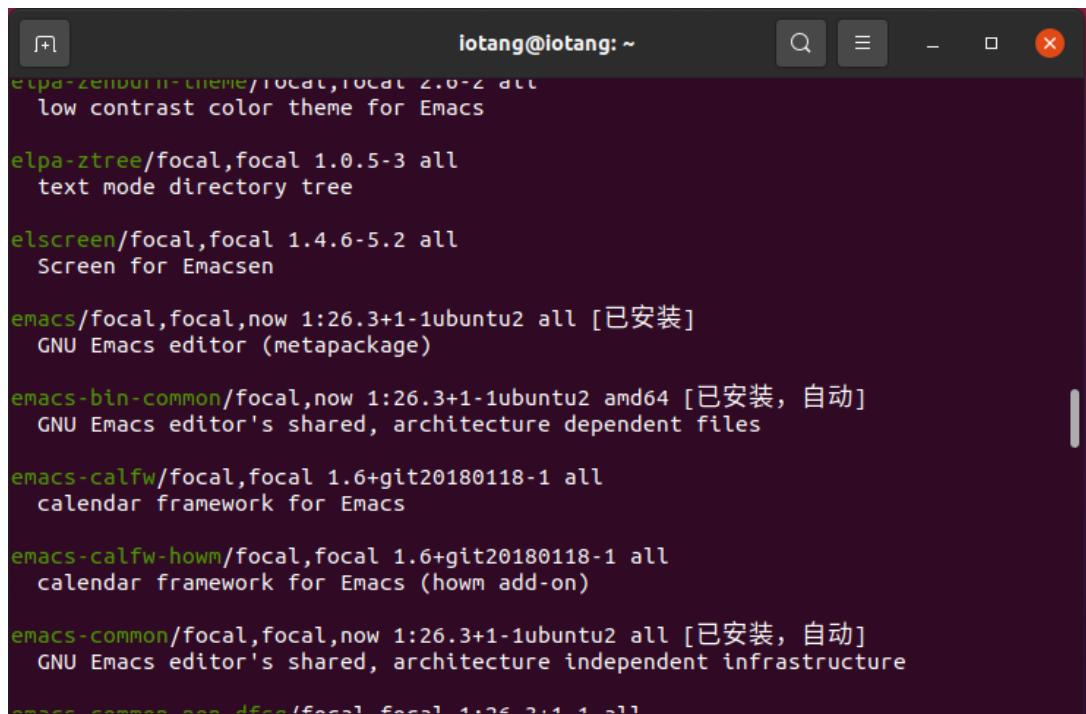
3.2.5 查找软件

```
$ apt search xxx
```

让 apt 在软件列表里查找相应字段。比如我想安装一个文本编辑软件 Emacs：

```
$ apt search emacs
```

可以发现有一个软件包叫 emacs。



apt 找到了 Emacs

3.2.6 安装软件

```
$ sudo apt install xxx
```

比如我们刚才找到了 Emacs 的软件名就叫 emacs，所以我们可以这样安装 emacs：

```
$ sudo apt install emacs
```

3.3 安装中文输入法

Linux 下想输入中文的话，一个方法是使用输入法。而搜狗输入法在 Linux 下仍然可用。

百度到给 Linux 的搜狗输入法的主页：



搜狗输入法已经支持到了 *Ubuntu 20.04*

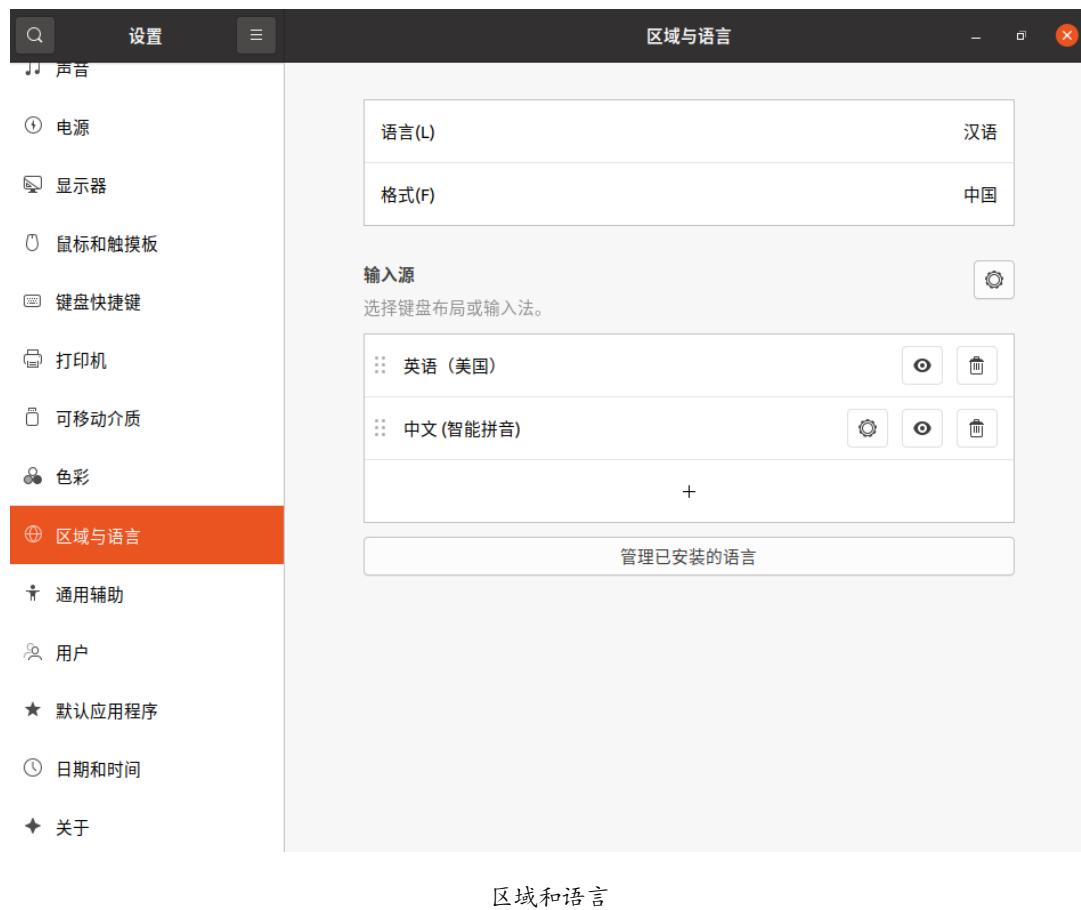
下载会得到一个以 .deb 结尾的安装包：sogoupinyin_ 版本号 _amd64.deb。



搜狗拼音安装包

3.3.1 添加中文语言支持

打开系统设置，找到“区域和语言”，点击“管理已安装的语言”。



如果有语言支持没有安装完整，可以选择安装。



在“语言”栏下点击“添加或删除语言”。



添加或删除语言

在弹出来的窗口里勾上“中文（简体）”，应用。



添加中文（简体）

3.3.2 切换到 fcitx

回到“语言支持”窗口，在键盘输入法系统中，选择“fcitx”。



选择 *fcitx*

如果没有 *fcitx*, 那么把 *fcitx* 装上:

```
$ sudo apt install fcitx
```

选择“应用到整个系统”。



应用到整个系统

3.3.3 安装搜狗输入法

直接双击打开这个安装包，在弹出的应用商店里安装。



重启系统，发现搜狗输入法已经在列表中。



搜狗拼音已经可以使用

用 `Ctrl + Space` 切换输入法，就可以使用搜狗输入法了。



大功告成

4 完善工作环境

4.1 在右键菜单中添加“新建文件”

```
$ cd ~/Templates  
$ touch "空白文档.txt"
```

想一想：上面的两行命令分别有什么作用？

练习：让右键菜单“新建文件”中出现一个“新建 C++ 文件”。

4.2 使用 Code::Blocks 作为 C++ IDE

以下内容适合确实很新手的新手使用。

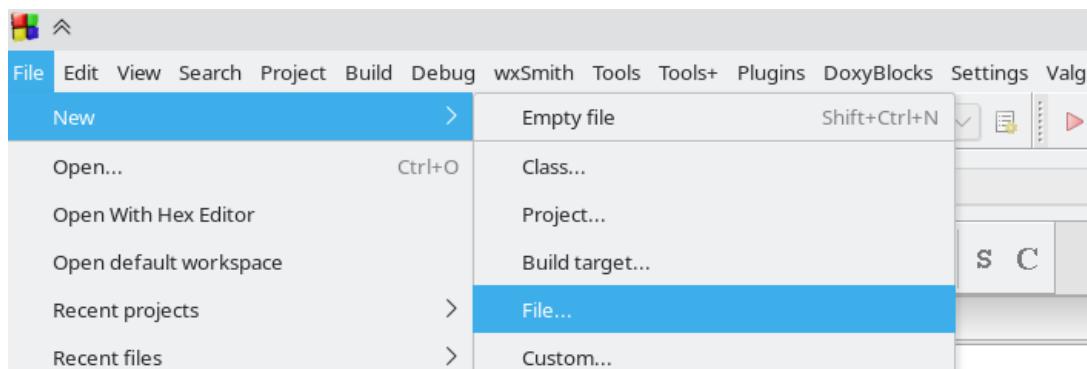
据 NOI 官网，NOI Linux 2.0 已经可以使用 Code::Blocks 作为 C++ 的 IDE。

4.2.1 安装

```
$ sudo apt install codeblocks
```

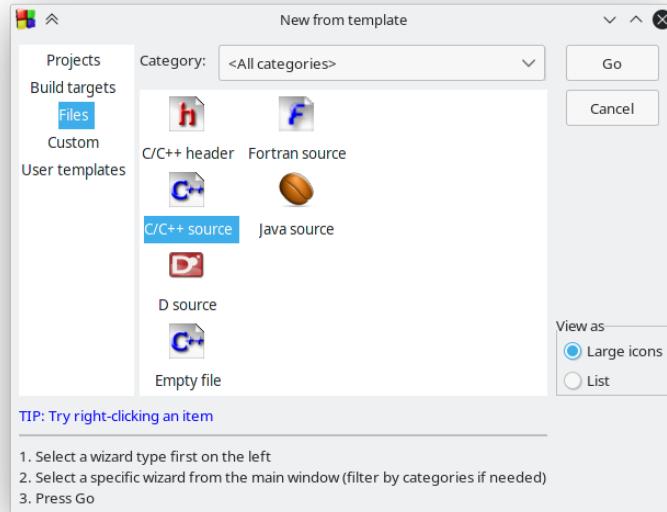
4.2.2 新建一个 C++ 文件

在上面的菜单中选择新建一个文件。



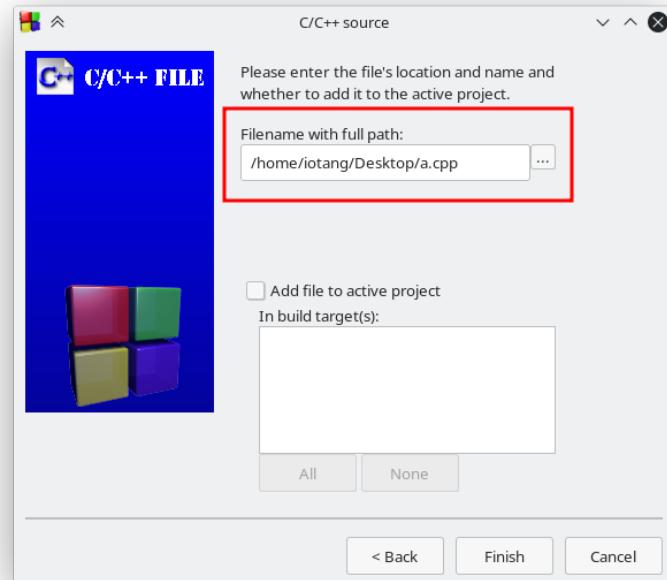
新建文件

选择“C++ 源代码”。



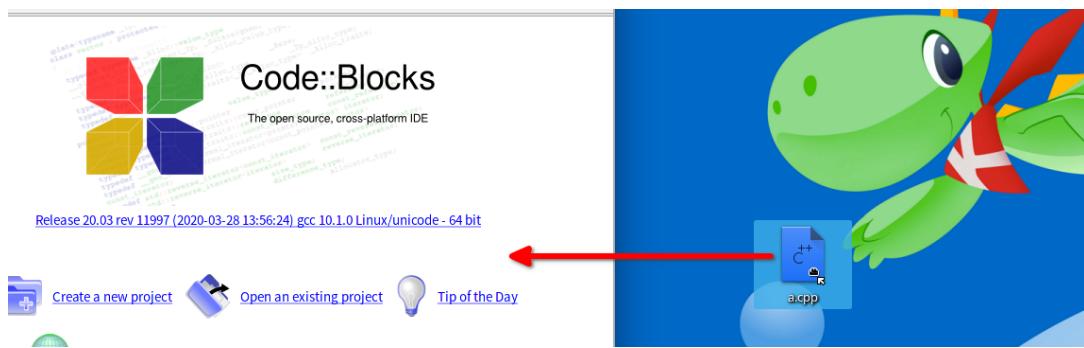
选择模板

在之后填写这个文件的路径。



新建文件的位置

当然还有一种办法，就是在文件管理器中新建一个文件，然后把它拖到 Code::Blocks 里面就好了。

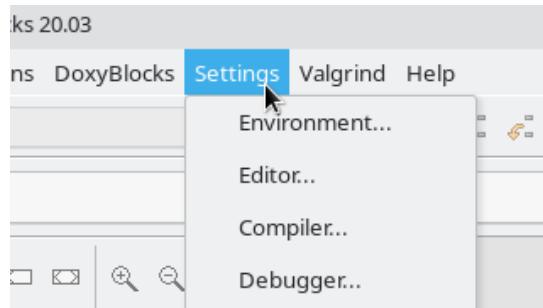


把 *a.cpp* 拖到窗口里面来

4.2.3 配置 Code::Blocks

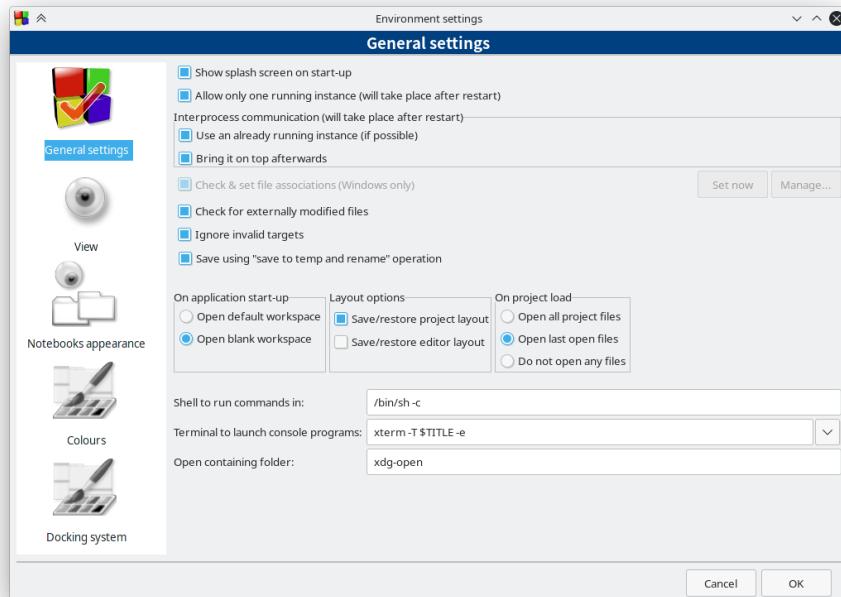
Code::Blocks 提供了有 Dev-Cpp 感觉的设置。

在上面的选项卡中可以进入设置。



设置的位置

第一个是 Code::Blocks 环境的设置。



环境的设置

在这里你可以设置各种东西，比如自动保存。
你可能要留意一下这儿。

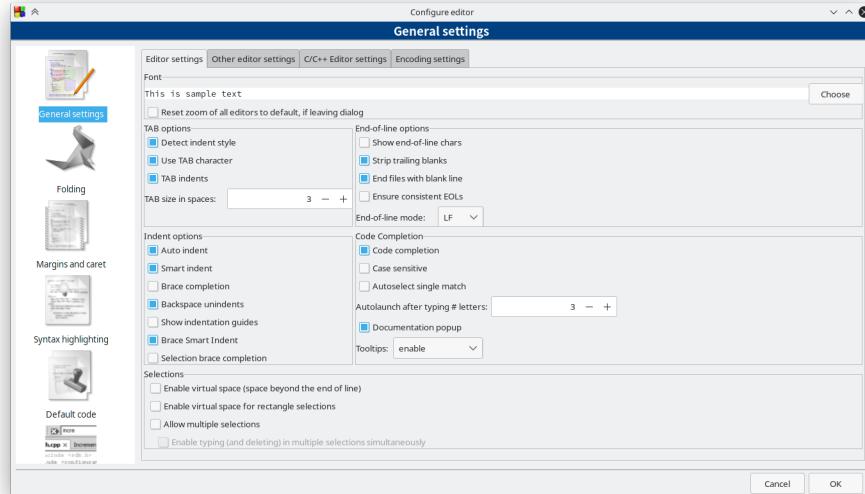


终端的设置

这里是设置 Code::Blocks 跑程序时启动的终端，就像你用 Dev-Cpp 运行时跳出来的那个黑框框一样。问题来了，你可能没有 `xterm`。你可以安装它，也可以把它换成你用 `Ctrl + Alt + T` 打开的那个终端。如果你喜欢后者，把这一行的内容换成下面：

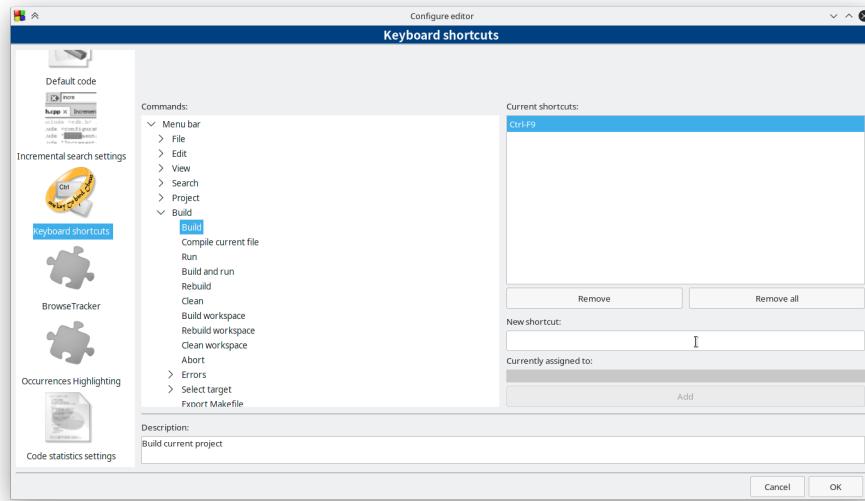
```
gnome-terminal -T $TITLE --
```

第二个是编辑器的设置。



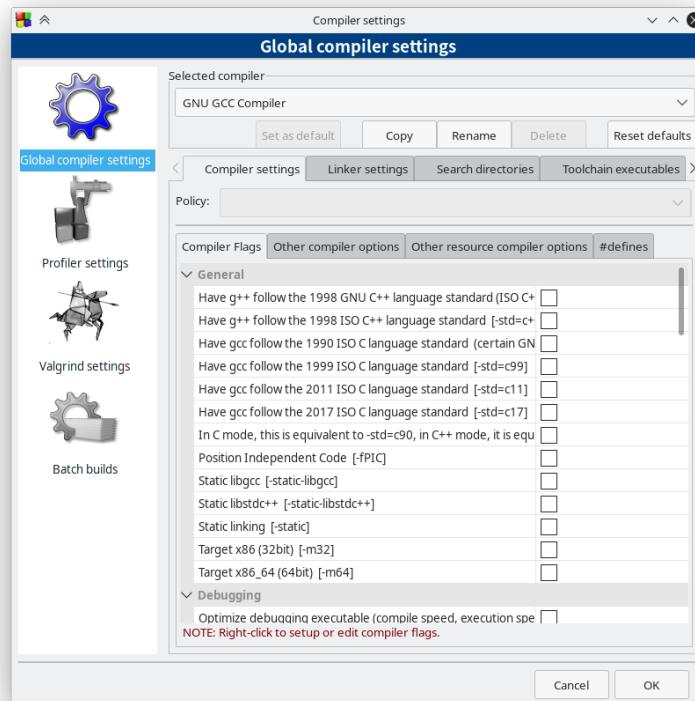
编辑器的设置

在这里可以设置字体、缩进、快捷键等编辑器元素。
比如在这里设置快捷键：



快捷键的设置

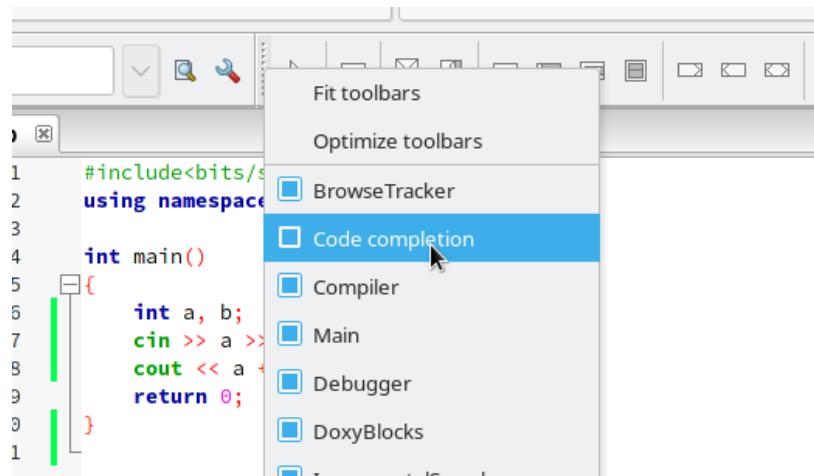
第三个是编译器的设置。



编译器的设置

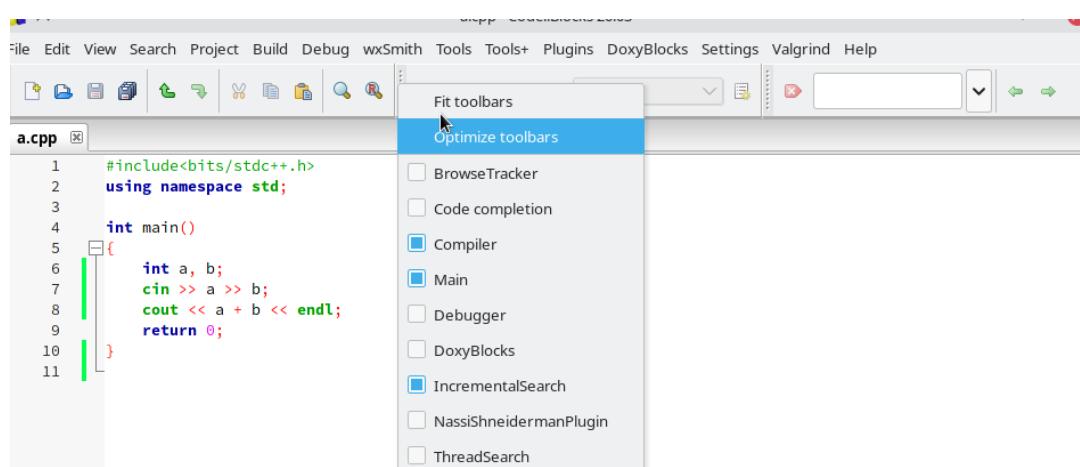
在这里可以设置编译选项等。

而在编辑界面，如果你不喜欢上方太多的工具栏，你可以右键打开设置隐藏它们：



工具栏的设置

比如：



iotang 的工具栏设置

4.2.4 编译和运行程序

你可以用这个工具栏编译和运行程序：



编译器工具栏，前三个从左到右：编译、运行、编译且运行

也可以使用快捷键。

默认的快捷键是这样的：

Ctrl + F9 编译这个项目（不过你的项目就是这一个文件）。

Ctrl + Shift + F9 编译当前文件。

Ctrl + F10 运行。

F9 编译且运行。

F8 开始调试。

运行时，按照你的设置会弹出一个终端。

The screenshot shows the Code::Blocks IDE interface. On the left, the code editor window displays the file 'a.cpp' with the following content:

```
1 #include<bits/stdc++.h>
2 using namespace std;
3
4 int main()
5 {
6     int a, b;
7     cin >> a >> b;
8     cout << a + b << endl;
9     return 0;
10 }
```

To the right of the code editor is a terminal window titled 'Desktop : cb_console_runn — Konsole'. It shows the command-line interface with the following output:

```
1 2
3

Process returned 0 (0x0)   execution time : 2.536 s
Press ENTER to continue.
```

编译且运行

4.3 以 Emacs 作为 C++ 代码编辑器

以下内容适合不是很新手的新手使用。

这里我假设大家都是 C++ 选手。

Emacs 是著名的集成开发环境和文本编辑器，被公认为是最受专业程序员喜爱的代码编辑器之一。

你可以在任何地方看见 Emacs 教徒和 Vim 教徒之间的争端。

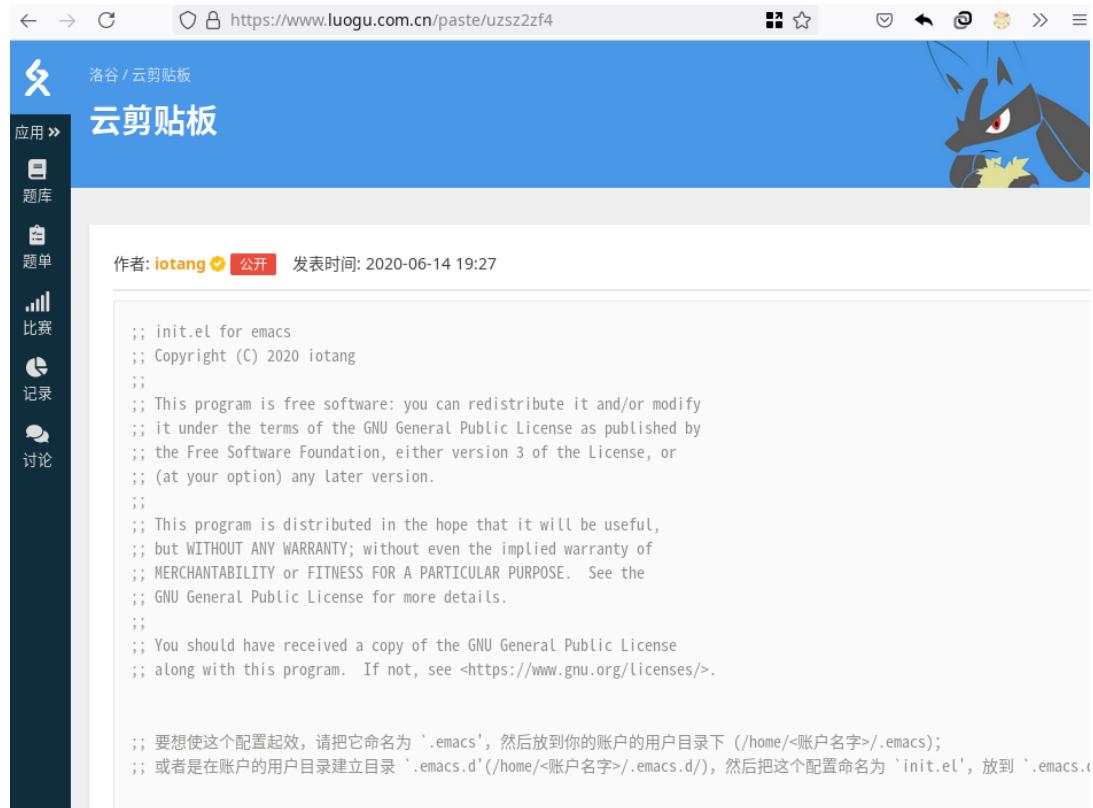
而 Emacs 比较符合正常人的操作逻辑，所以我们暂且把 Vim 放到一边，来使用 Emacs。

4.3.1 安装

```
$ sudo apt install emacs
```

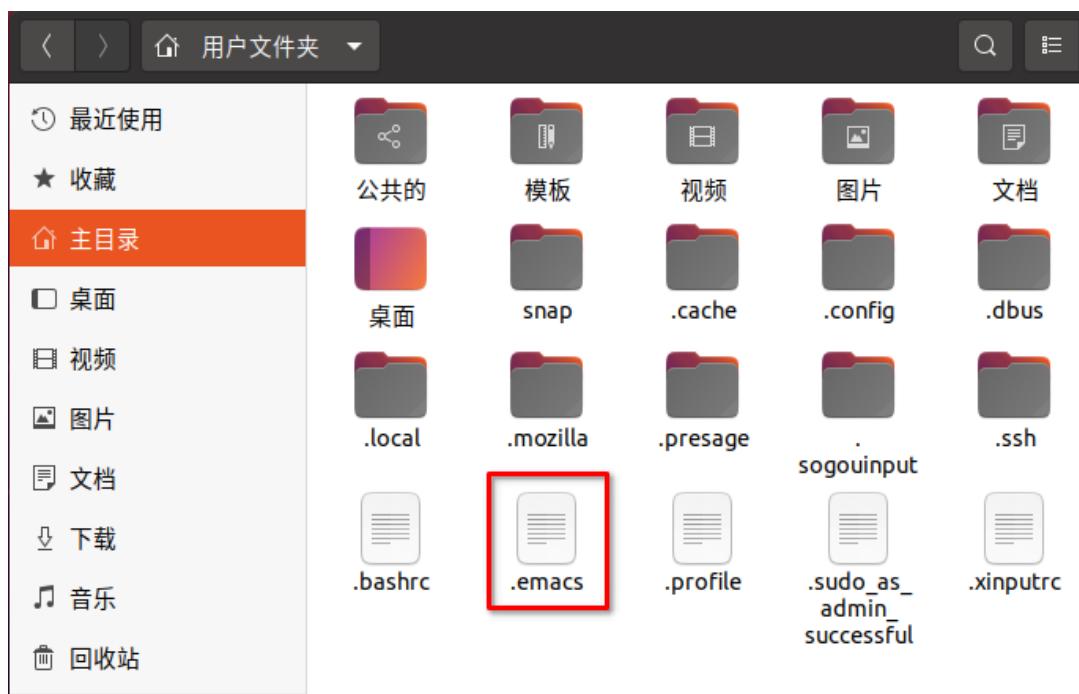
4.3.2 配置 Emacs

首先搞到豪华配置：[网址](#)。

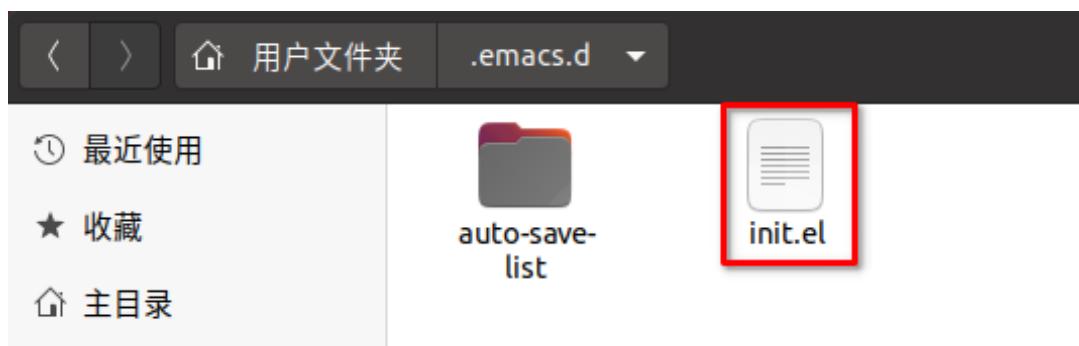


当年 HNOI 省选赛场上的配置

要想使这个配置起效, 请把它命名为 `.emacs`, 然后放到你的账户的用户目录下 (`/home/<账户名字>/.``.emacs`)；或者是在账户的用户目录建立目录 `.emacs.d` (`/home/<账户名字>/.emacs.d/`), 然后把这个配置命名为 `init.el`, 放到 `.emacs.d` 下。



.emacs



.emacs.d/init.el

(以“.”开头的文件是隐藏文件。在文件管理器中按 $Ctrl + h$ 显示它们。)

4.3.3 考场上在没有 Emacs 配置的时候配置 Emacs

之前提到的“配置”就是 Emacs 在启动之前会调用的一堆命令。

Emacs 的命令的语言是 Emacs Lisp，语法基本都长这样：

(函数名字 参数 参数 参数 ...)

半角分号 “;” 之后的内容都是注释：

(函数名字 参数 参数 参数 ...) ; 注释

比如：

```
(+ a b) ;; 加法
(- a b) ;; 减法
(defun 函数名 (参数表) 函数体) ;; 一个函数，效果是定义一个函数
```

Lisp 尝试计算一切，包括函数的参数。单引号可以防止 Lisp 瞎计算东西：

```
(write (* 114 514)) ;; 输出 58596
(write '(* 114 514)) ;; 输出 (* 114 514)
```

然后你可以在上面的配置中看到这个。

```
(defun compile-file ()(interactive)(compile
(format "g++ %s -o %s -g -lm -Wall -std=c++98 -fsanitize=address"
(buffer-name)(file-name-sans-extension (buffer-name)))))
```

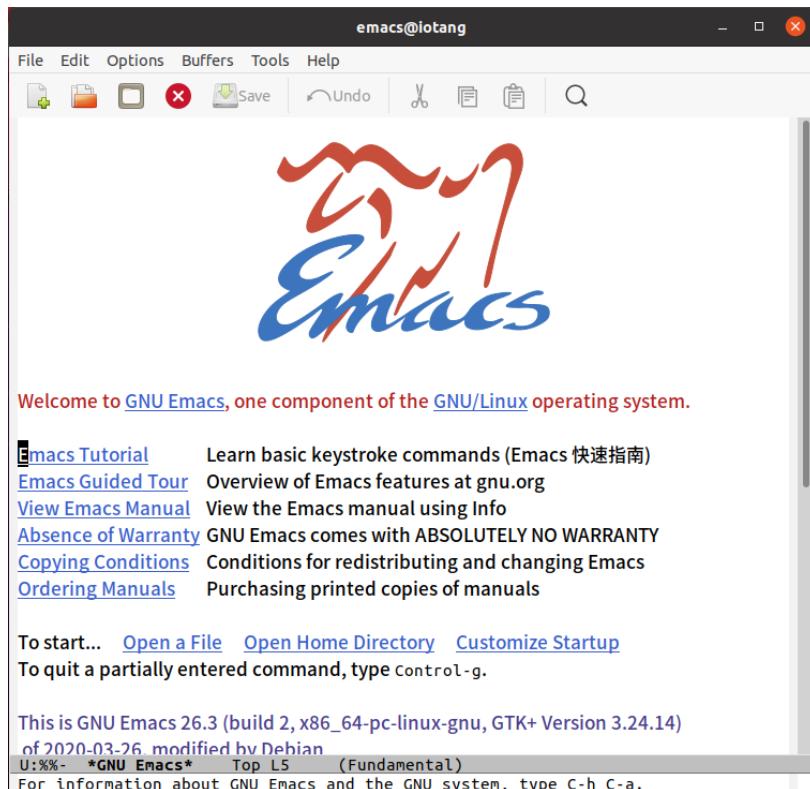
我们再加几个换行。

```
(defun compile-file ()(interactive)
  (compile
    (format
      "g++ %s -o %s -g -lm -Wall -std=c++98 -fsanitize=address"
      (buffer-name)(file-name-sans-extension (buffer-name))
    )
  )
)
```

(interactive 表示它可以通过 M-x 调用。)

呃，就是这样。

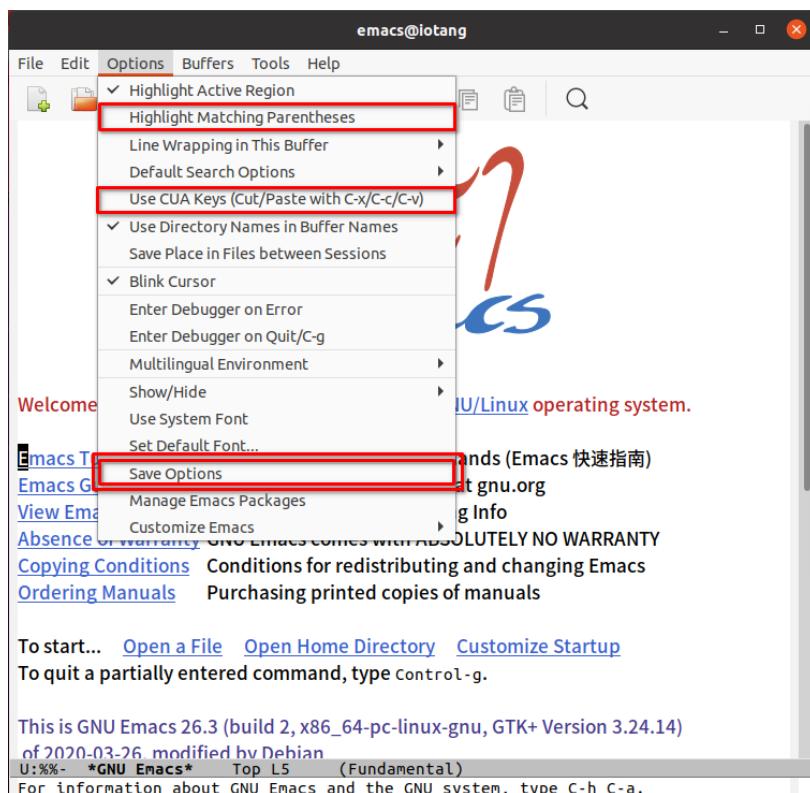
然后没有配置的 Emacs 是这样的：



Emacs 原本的模样

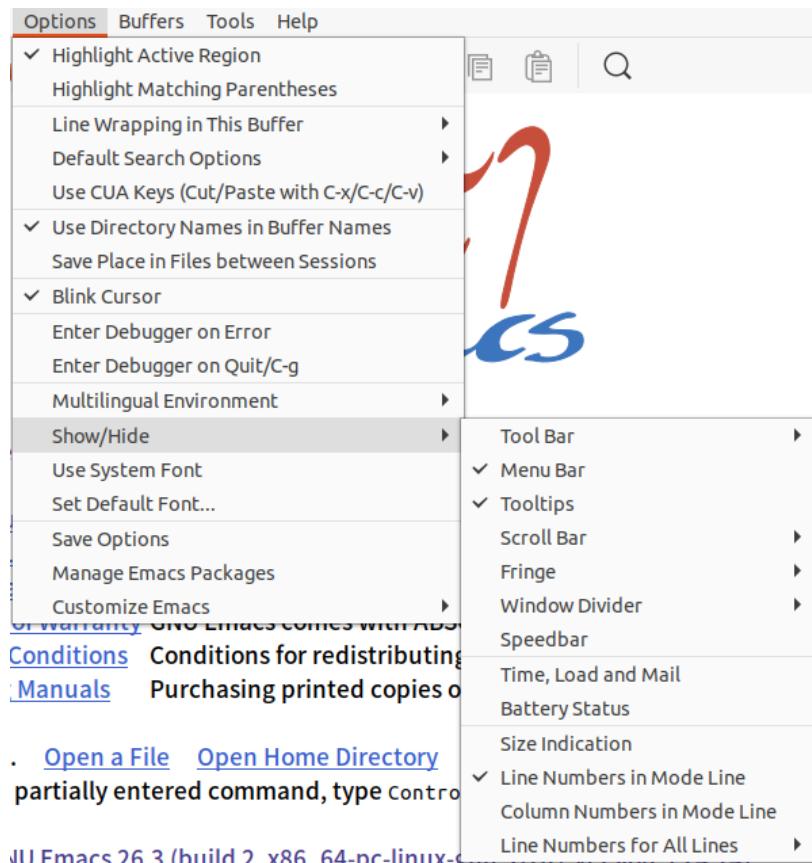
首先通过非命令方法设置：

在这里可以设置括号匹配和设置 CUA 键（剪切、复制、粘贴分别是 C-x, C-c, C-v（大写的 C 代表 Ctrl, 大写的 M 代表 Alt）），以及保存设置。



Emacs 设置

在这里可以设置 Emacs 的各种东西是否显示。



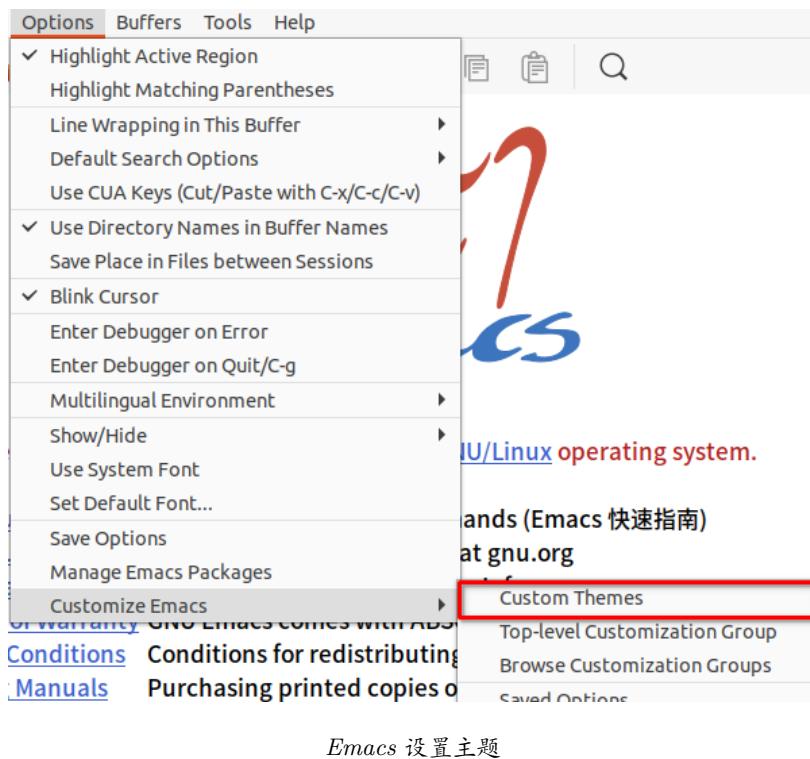
Emacs 设置显示 / 隐藏

比如这个就是 Tool Bar。



Emacs 的工具条

在这里可以设置 Emacs 的主题。



Emacs 设置主题

一定要记得**保存设置**。

然后有些东西是必须要把命令背下来的：

```
;; emacs 和系统的剪贴板共用。
(setq-default x-select-enable-clipboard t)

;; 显示行号。
(global-linum-mode t)

;; 透明度。
(set-frame-parameter (selected-frame) 'alpha (list 90 70))
(add-to-list 'default-frame-alist (cons 'alpha (list 90 70)))

;; Ctrl-z 撤销。
(global-set-key (kbd "C-z") 'undo)

;; Ctrl-a 全选。
(global-set-key (kbd "C-a") 'mark-whole-buffer)

;; Ctrl-h 替换。
(define-key key-translation-map (kbd "C-h") (kbd "M-%"))

;; Ctrl-s 保存。
(global-set-key (kbd "C-s") 'save-buffer)

;; 撤销记录扩大。
```

```
(setq-default kill-ring-max 65535)

;; C++ 代码风格。
;; "bsd" = 所有大括号换行。这是真理
;; "java" = 所有大括号不换行。else 接在右大括号后面。
;; "k&r" = "awk" = 只有命名空间旁、定义类、定义函数时的大括号换行。else 接在右大括号后面。
;; "stroustrup" = 只有命名空间旁、定义类、定义函数时的大括号换行。else 换行。
;; "whitesmith" = 所有大括号换行。大括号有一次额外缩进。
;; "gnu" = 所有大括号换行。每次左括号开始，会有一层额外缩进。这是 emacs 默认
;; "linux" = 只有命名空间旁、定义类、定义函数时的大括号换行。else 接在右大括号后面。
;; 一般来说，这个风格应该有 8 格的空格缩进。
(setq-default c-default-style "bsd")

;; C++ 代码缩进单位长度。
(setq-default c-basic-offset 3)

;; 使用 tab 缩进。
(setq-default indent-tabs-mode t)

;; tab 的长度。务必和缩进长度一致。
(setq-default default-tab-width 3)
(setq-default tab-width 3)

;; 换行的时候自动缩进。
(global-set-key (kbd "RET") 'newline-and-indent)

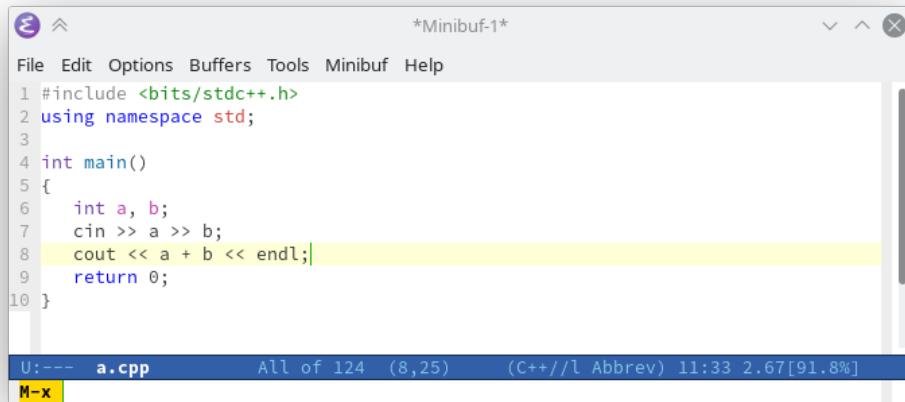
;; 一键编译。
(defun compile-file ()(interactive)(compile
(format "g++ %s -o %s -g -lm -Wall -std=c++98 -fsanitize=address"
(buffer-name)(file-name-sans-extension (buffer-name))))))
(global-set-key [f9] 'compile-file)
(global-set-key [f10] 'compile)

;; 一键 GDB。
(global-set-key [f2] 'gud-gdb)
```

根据需要调整，比如笔者就是 3 格 Tab 缩进的毒瘤。

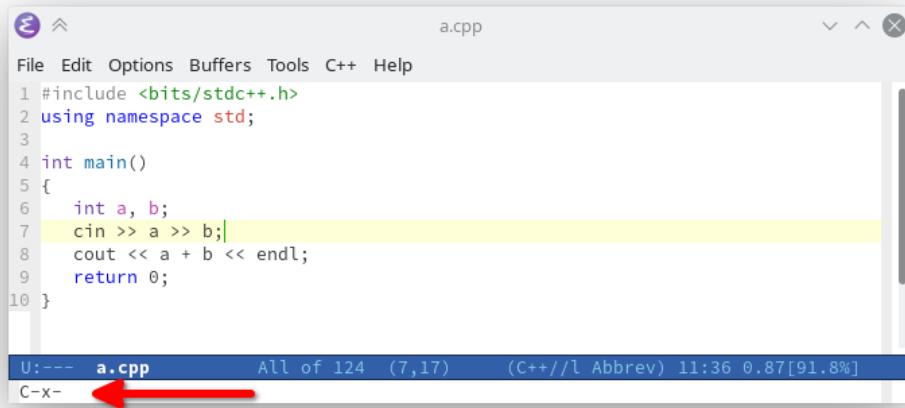
4.3.4 Emacs 初步

这里参考了：<https://www.cnblogs.com/holbrook/archive/2012/02/15/2357335.html>。



Emacs 的一个 frame

整个窗口在 Emacs 中叫做 frame，图形界面下的 Emacs 可以打开多个 frame。
 每个 frame 从上到下分成 3 部分，分别是缓冲区、状态栏和回显区。
 缓冲区是编辑的主区域，但是在这里操作的还不是真正的文件，而是文件的一个缓存 (buffer)。只有执行写入操作时，才会将 buffer 的内容写入到文件。
 缓冲区可以分成多个区域，缓冲不同的内容。
 状态栏显示当前的一些状态信息。
 最下面是回显区，提示当前正在进行的操作。
 如果一个命令没有输入完，这里会显示已经输入的指令。



没有输入完的命令

这里笔者已经按了 **C-x**，但它不对应任何命令，所以 Emacs 把它显示出来提醒笔者输入接下来的命令。
 按 **C-g** 取消命令。
 连按 3 下 **ESC** 也可以取消命令。

4.3.5 在 Emacs 中进行文件操作

C-x C-f (先按 C-x 再按 C-f) 寻找文件。之后输入文件的名字，如果没有这个文件就会新建它。但如果那个文件有目录没有被创建，它会提示你执行 M-x make-directory RET RET (先按 M-x 再输入 make-directory 再按 2 次回车 (RET 就是 Return)) 来自动创建目录。

C-x C-s 保存文件。

C-x s 保存所有文件。

C-x C-c 保存所有文件并退出 Emacs。

C-x C-w 另存为文件。

4.3.6 分屏和操作 Buffer

C-x 1 (先按 C-x 再按 1) 保留当前光标所在的区域，去掉其它所有区域。

C-x 2 上下分屏。

C-x 3 左右分屏。

C-x 0 去掉当前光标所在的区域。

C-x 左方向键 切换到上一个 buffer。

C-x 右方向键 切换到下一个 buffer。

C-x k RET 删除一个 buffer。蔡徐坤

```

File Edit Options Buffers Tools C++ Help
1 #include <bits/stdc++.h>
2 using namespace std;
3
4 int main()
5 {
6     int a, b;
7     cin >> a >> b;
8     cout << a + b << endl;
9     return 0;
10 }

U:--- a.cpp      All of 124 (11,0)      (C++/l Abbrev) 11:43 1.60[91.8%]
1 --- mode: compilation; default-director
2 Compilation started at Sat Jul 10 11:43
3
4 g++ -o a a.cpp -g -lm -Wall -std=c++2a
5
6 Compilation finished at Sat Jul 10 11:43
|


U:--- *compilation*  All of 187 (7,0) | U:--- *eshell*      All of 62 (6,12)
1 Welcome to the Emacs shell
2
3 ~/Desktop $ ./a
4 1 2
5 3
6 ~/Desktop $ |

```

分屏可以组合

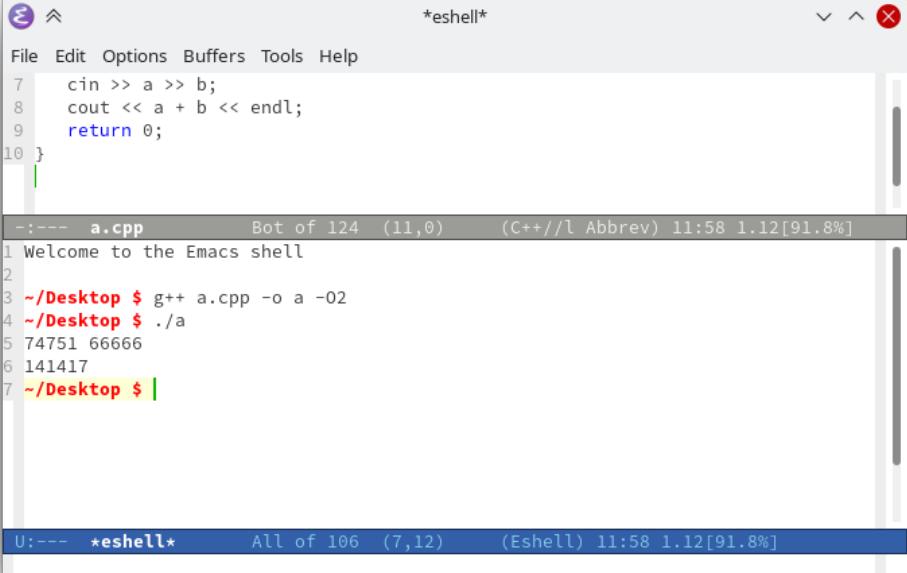
4.3.7 使用 Eshell 操作终端

M-x eshell RET 打开 Eshell。

通过 M-x 执行命令时，可以使用 Tab 补全。

所以你可以只输入 M-x es TAB RET。

你甚至可以只输入 M-x es RET。



The screenshot shows the Emacs interface with the Eshell buffer (*eshell*) active. The top part of the screen displays a C++ code editor with a snippet of code:

```
7 cin >> a >> b;
8 cout << a + b << endl;
9
10 }
```

Below the code editor is the Eshell buffer, which contains the following session:

```
-:--- a.cpp      Bot of 124 (11,0)      (C++/l Abbrev) 11:58 1.12[91.8%]
1 Welcome to the Emacs shell
2
3 ~/Desktop $ g++ a.cpp -o a -O2
4 ~/Desktop $ ./a
5 74751 66666
6 141417
7 ~/Desktop $ |
```

The bottom status bar indicates "U:--- *eshell* All of 106 (7,12) (Eshell) 11:58 1.12[91.8%]".

在 *Eshell* 中执行 *Shell* 命令

4.4 用 CrossOver 运行 Windows 程序

CrossOver 是一款系统兼容软件，可以让你在 Linux 系统上运行 Windows 应用，比如 QQ。

4.4.1 安装

百度到 CrossOver 的下载页面。

The screenshot shows the 'CrossOver 20 下载专区' page. It features two main download sections:

- CrossOver Mac 版本20**: Includes a large 'X' icon, a 'Try Download' button, and a 'Download Software' link.
- CrossOver Linux 版本20**: Includes a Linux penguin icon, a 'Try Download' button, and a 'Download Software' link.

Both sections have a 'More' link and a list of features or benefits.

CrossOver 20 下载专区

下载 .deb 包并安装。

The screenshot shows a '软件下载' (Software Download) window with three download options:

文件名	大小	操作
CrossOver 20 Linux deb	135 MB	下载
CrossOver 20 Linux rpm	212 MB	下载
CrossOver 20 Linux bin	211 MB	下载

下载 CrossOver 安装包

4.4.2 在 CrossOver 中安装 QQ

打开 CrossOver，选择“安装 Windows 软件”。



CrossOver 20

等更新完后搜索 QQ，尽量选择新版本。



CrossOver 中安装 QQ 9.0

继续安装。

接下来 CrossOver 会让你安装一些依赖。这个当然选择“是”。



安装依赖

接下来会出现 QQ 的安装流程。走完流程后关掉 QQ, 然后 CrossOver 会探测到安装完成:



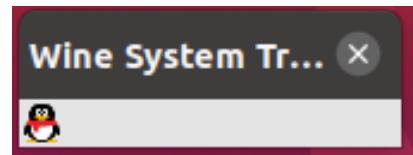
安装完成

之后就可以在 CrossOver 中启动 QQ 了。



CrossOver 安装了 QQ

注意在 QQ 启动后别把这个 Wine System Tray 给关了，因为你要用这个栏退出 QQ。

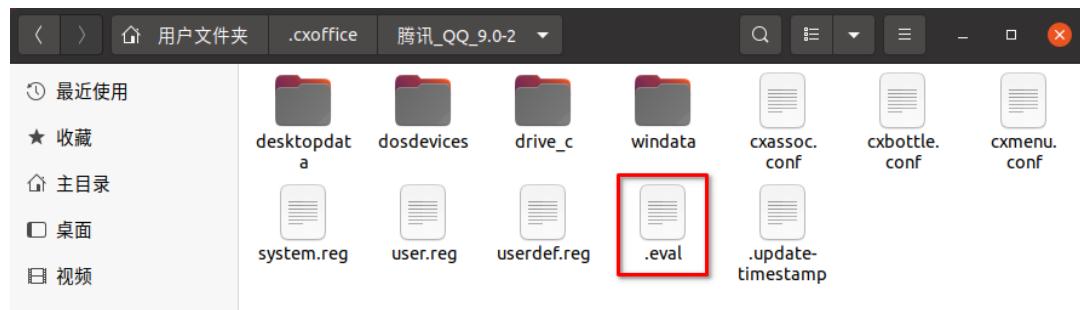


Wine System Tray

4.4.3 * 防总行为：无限试用 CrossOver

请支持开发者！

删掉这个文件重置使用时间：



作弊行为

4.5 用 Typora 编写 Markdown 文档

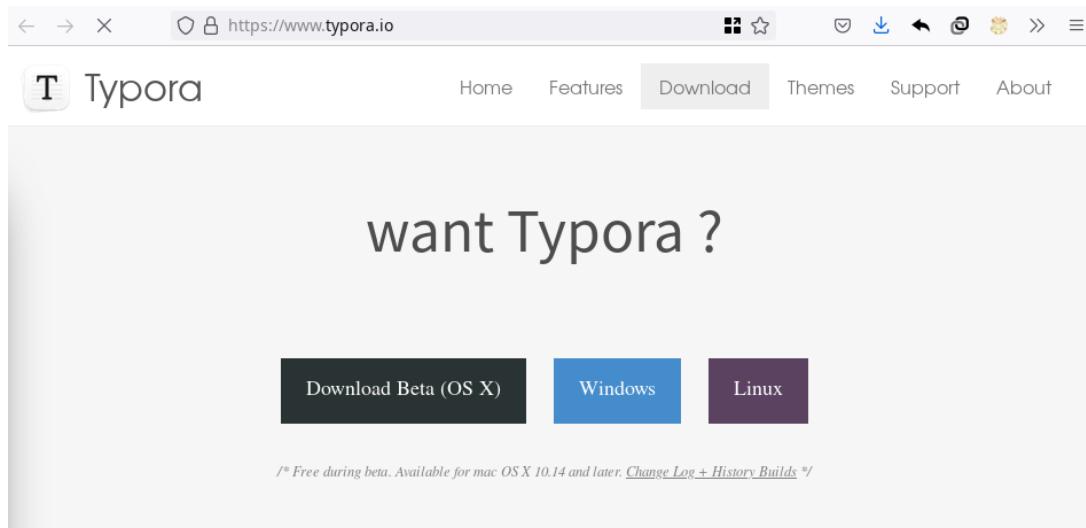
以下参考了 <https://www.runoob.com/markdown/md-tutorial.html>。

Markdown 是一种轻量级标记语言，它允许人们使用易读易写的纯文本格式编写文档。

Typora 是一个编辑器，可以在编辑时以 Markdown 的语法实时渲染出效果。

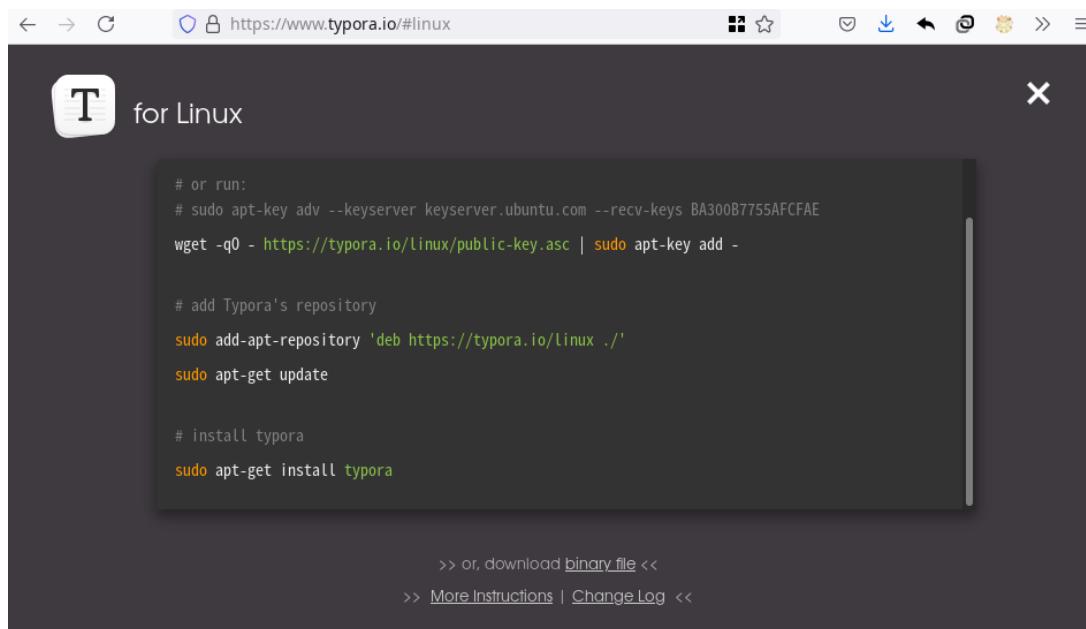
4.5.1 安装 Typora

百度到 Typora 的下载页面。



下载 Typora

直接被官方命令胡脸。



安装 Typora

(apt 可以替代 apt-get)

按它所说依次执行：

```
$ wget -qO - https://typora.io/linux/public-key.asc | sudo apt-key add -
$ sudo add-apt-repository 'deb https://typora.io/linux ./'
$ sudo apt update
$ sudo apt install typora
```

4.5.2 Markdown 语法

用 # 表示标题。

```
# 一级标题
## 二级标题
### 三级标题
#### 四级标题
##### 五级标题
###### 六级标题
```



Markdown 标题

段落的换行是使用两个以上空格加上回车，或者直接插入一个空行。(Typora 在这里并不是非常严格)

```
(这一行的末尾有俩空格) 上一行
下一行

再下一行
```

粗体和斜体。

```
* 斜体文本 *
_ 斜体文本 _
** 粗体文本 **
```

```
-- 粗体文本 --
*** 粗斜体文本 ***
--- 粗斜体文本 ---
```



Markdown 字体

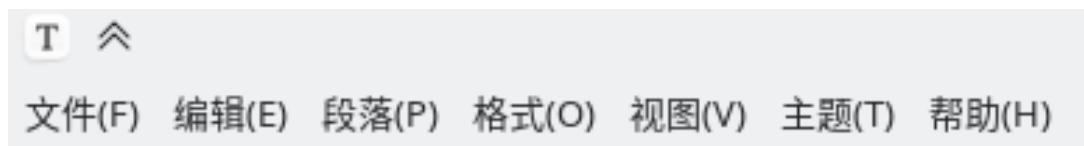
在左上角“文件”>“偏好设置”中打开 Markdown 内联公式。



Markdown 内联公式

用一对美元符号“\$”框住 LaTeX 数学公式。

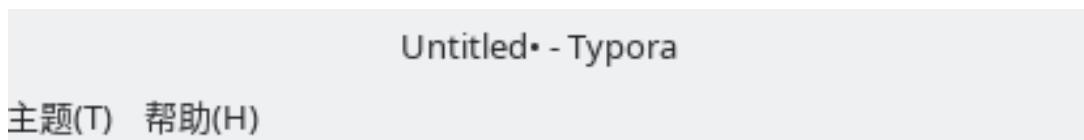
```
$x_1, x_2 = \frac{-b \pm \sqrt{b^2 - 4a \times c}}{2a}$
```



在 *Markdown* 中使用 *LaTeX* 数学公式

用一对两个美元符号 “**\$\$**” 框住一大段 *LaTeX* 数学公式。这会使公式居中。

```
$$
\mathbf{V}_1 \times \mathbf{V}_2 = \begin{vmatrix}
\mathbf{i} & \mathbf{j} & \mathbf{k} \\
\frac{\partial X}{\partial u} & \frac{\partial Y}{\partial u} & 0 \\
\frac{\partial X}{\partial v} & \frac{\partial Y}{\partial v} & 0
\end{vmatrix}
$$
```



在 *Markdown* 中使用大型 *LaTeX* 数学公式

更多语法请去 <https://www.runoob.com/markdown/md-tutorial.html> 看。

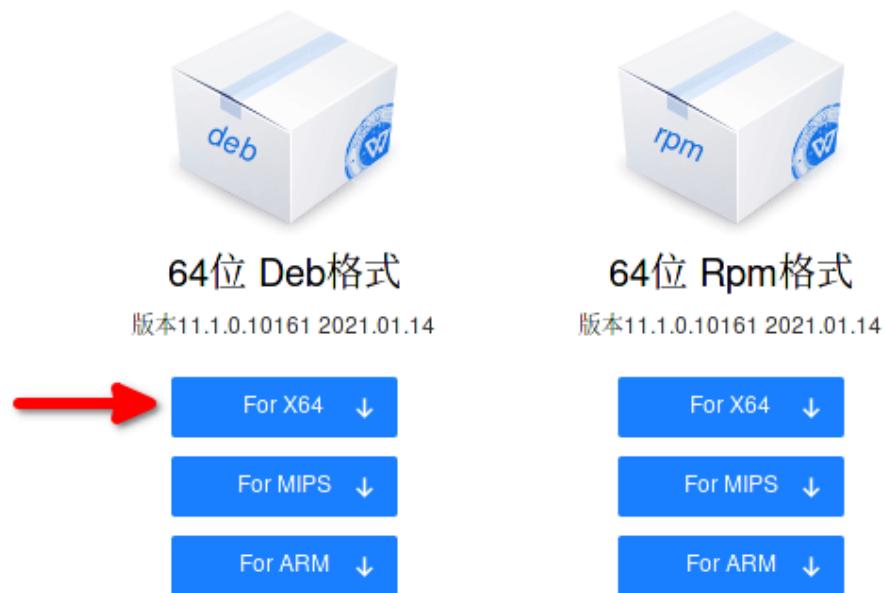
4.6 用 WPS Office 编辑 Word、Excel 和 PowerPoint

4.6.1 安装

百度到 WPS Office for Linux 的下载页面。



仍然下载 x64 版本的 deb 安装包。



下载 WPS Office

以你的经验安装这个安装包。

4.7 以 LemonLime 作为评测软件

4.7.1 使用 .deb 安装包安装 LemonLime

去 [GitHub 上的下载区](#) 下载 .deb 安装包安装。

4.7.2 使用源代码编译最新的 LemonLime

作为这个软件的开发者之一，我当然希望大家都用上船新版本的 LemonLime！

请阅读[构建指南](#)。

4.7.3 LemonLime 的使用

请阅读 LemonLime 自带的用户手册。



LemonLime 用户手册的位置

5 练习：完善 Ubuntu 及个性化

通过完善你的 Ubuntu 来熟悉它的操作，以及锻炼自己通过搜索引擎独立解决问题的能力。
请自己完成下列问题。

5.1 校准系统时间

系统时间不正确会给某些应用程序带来困扰。

5.2 关闭 SSH 功能和端口

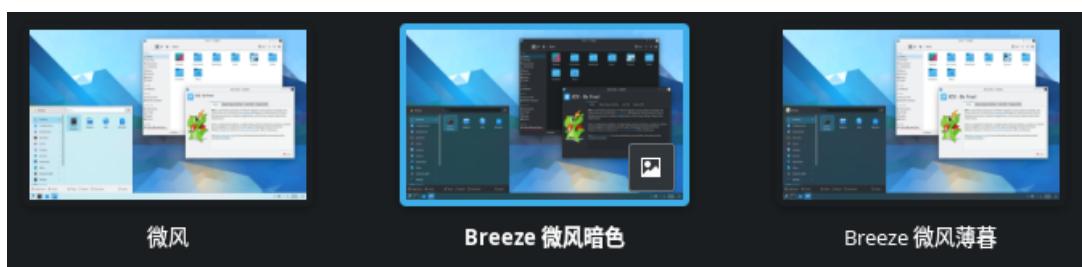
嗯……

5.3 更改壁纸

清一色的 Focal Fossa 并不是那么有创造力……？

5.4 更改系统颜色主题

你喜欢暗色主题还是亮色主题？



KDE 的默认主题中的 3 个

5.5 更改终端颜色主题

想换一下终端默认的基佬紫？

5.6 不使用 Firefox 作为浏览器

……虽然笔者并不赞同这样的做法，但是笔者也没办法，因为这是你的 Ubuntu。

5.7 自定义默认程序

也许你想让 Emacs 也默认打开其它类型的文件？或者……

5.8 在 Firefox 或其它浏览器中安装扩展程序

以下功能值得考虑：

- 关闭刚刚打开的标签页？
- 屏蔽广告？
- 翻译网站内容？

- 深色主题？

5.9 使用 GeoGebra 作为数学作图软件

GeoGebra 可以实现和几何画板类似的功能。

5.10 使用 GIMP 作为图像编辑器

GIMP 可以实现和 PhotoShop 类似的效果。

你会在 GIMP 中碰见 Wilber：那只棕色的小家伙。

5.11 使用 Krita 作为绘图程序

Krita 可以实现和 SAI 类似的效果，或者以用 Windows 画图的方式使用它。

你会在 Krita 中碰见 Kiki：一只电子松鼠。在 Krita 的启动界面可以看到她。

5.12 使用 TeXstudio 与 XeLaTeX 编辑与编译 TeX 文档

用 L^AT_EX 排版系统编辑非常规范的文档，比如论文，或者你自己出题的题面，或者你现在看到的这篇文章。

你会在 L^AT_EX 中碰见 The TeX Lion：一只活跃的狮子。

5.13 在外部介质中备份你的 Ubuntu

如果由于某些原因 Ubuntu 坏掉了，通过外部介质来补救。

5.14 使用 Ubuntu 在洛谷上切一道新题

一切的开始。

5.15 使用 Ubuntu 完成一场模拟赛

祝你好运。

6 附录

6.1 NOI Linux 2.0 系统情况简表

类别	软件/模块	版本	备注说明
系统	Kernel	5.4.0-42-generic	64位
语言环境	GCC	9.3.0	C编译器
	G++	9.3.0	C++编译器
	FPC	3.0.4	Pascal编译器
	Python	2.7	非竞赛语言
		3.8	非竞赛语言
调试工具	GDB	9.1	
	DDD	3.3.12	
集成开发环境	Code::Blocks	20.03	C/C++集成开发环境
	Lazarus	2.0.6	Pascal集成开发环境
	Geany	1.36	C/C++/Pascal（轻量级）集成开发环境
文本编辑工具	VS Code	1.54.3	
	Emacs	26.3	
	Gedit	3.36.2	
	Vim	8.1	
	Joe	4.6	
	nano	4.8	
	sublime text	3.2.2	
其他软件	Firefox	79.0	网页浏览器
	Midnight Commander (mc)	4.8.24	终端
	XTerm (UXTerm)	3.5.3	终端
	Arbiter-local	1.02	程序评测工具单机版

NOI Linux 2.0 系统情况简表