1. VMware 12不支持Ubuntu 18.04, 使用VMware 14.1.1安装成功
2. 安装source code pro字体，下载好预编译的字体

在用户目录下新建.fonts文件夹，拷贝TTF,OTF文件夹至该目录下，执行命令

fc-cache –fv

1. Ubuntu 16 Terminal 字体外观中有 Allow bold text选项，可以使得终端中没有粗体字。但是在Ubuntu 18的Terminal中没有发现此选项。

* 命令ls下显示的粗体字（比如文件夹）

$ dircolors -p > ~/.dircolors

修改DIR 01；34 为 00；34 // 01 bold 00 normal

* 命令提示符（PS1）粗体

修改.bashrc文件PS1的定义

01的地方改为00 // 01 bold 00 normal

1. 修改gnome主题

$ sudo apt install gnome-tweak-tool

$ gnome-tweaks

默认gnome-shell加锁的

$ sudo apt install gnome-shell-extensions

$ reboot

打开shell theme开关，就可以切换shell主题了

1. Ubuntu Dock自动隐藏

Settings -> Dock -> Auto-hide the Dock (enable it)

1. Install Clang

<http://clang.llvm.org/get_started.html>

Note: cmake -G “Unix Makefiles” -DCMAKE\_BUILD\_TYPE=Release ../llvm

默认是debug模式

* Check out LLVM

$ cd ~/Downloads

$ svn co <http://llvm.org/svn/llvm-project/llvm/trunk> llvm

* Check out Clang

$ cd llvm/tools

$ svn co <http://llvm.org/svn/llvm-project/cfe/trunk> clang

$ cd ../..

* Check out extra Clang tools

$ cd llvm/tools/clang/tools

$ svn co <http://llvm.org/svn/llvm-project/clang-tools-extra/trunk> extra

$ cd ../../..

* Check out Compiler-RT

$ cd llvm/projects

$ svn co <http://llvm.org/svn/llvm-project/compiler-rt/trunk> compiler-rt

$ cd ../..

* Check out libcxx (libc++ is a new implementation of the C++ standard library)

$ cd llvm/projects

$ svn co <http://llvm.org/svn/llvm-project/libcxx/trunk> libcxx

$ cd ../..

* Check out libcxxabi

$ cd llvm/projcts

$ svn co <http://llvm.org/svn/llvm-project/libcxxabi/trunk> libcxxabi

$ cd ../..

* Check out LLD

$ cd llvm/tools

$ svn co <http://llvm.org/svn/llvm-project/lld/trunk> lld

$ cd ../..

* Check out LLDB

$ cd llvm/tools

$ svn co <http://llvm.org/svn/llvm-project/lldb/trunk> lldb

$ cd ../..

Note:

See <http://lldb.llvm.org/build.html>

Preliminaries: Swig, libedit, Python

$ sudo apt install swig libedit-dev libncurses5-dev python-dev

* Build LLVM and Clang

$ mkdir build

$ cd build

$ cmake -G “Unix Makefiles” -DCMAKE\_BUILD\_TYPE=Release ../llvm

$ make

$ sudo make install

Note:

默认安装目录/usr/local

1. Using Clang

* Using libc++

See：<https://libcxx.llvm.org/docs/UsingLibcxx.html>

$ clang++ -stdlib=libc++ test.cpp

Clang寻找libc++的头文件路径/usr/include/c++/v1

Clang寻找libc++的库路径/usr/lib

* 当安装的libc++库不在标准路径中时

$ clang++ -stdlib=libc++ -nostdinc++ \

-I /usr/local/include/c++/v1 \

-L /usr/local/lib \

-Wl,-rpath, /usr/local/lib

Test.cpp

Note:

Libc++安装路径是/usr/local

-Wl,-rpath,/usr/local/lib 指定运行库查找路径

实例：

在/usr/lib中建立软链接libc++.so.1, libc++abi.so.1至/usr/local/lib中对应的文件

$ clang++ -stdlib=libc++ -nostdinc++ \

-I /usr/local/include/c++/v1 \

Test.cpp

这时可以不使用-Wl,-rpath,<libcxx-install-prefix>

或者

$ clang++ -stdlib=libc++ Test.cpp

-nostdinc++ 表示不在标准路径中找头文件，需要在-I指定目录中找

而标准路径有/usr/include, /usr/local/include, 所以我安装的头文件是在标准路径中

1. Install Chrome

$ sudo wget <http://www.linuxidc.com/files/repo/google-chrome.list> -P /etc/apt/sources.list.d/

$ wget -q -O – <https://dl.google.com/linux/linux_signing_key.pub> | sudo apt-key add –

$ sudo apt upate

$ sudo apt install google-chrome-stable

1. Install Chinese input method

* $ sudo apt install ibus-pinyin
* Settings - Region&Language - Input Sources

Click ‘Manage Installed Language’, 添加中文字体，重启

Click ‘+’ , 选择‘Chinese Intelligent Pinyin’

1. Install vim-plug

* Download

$ curl –fLo ~/.vim/autoload/plug.vim –create-dirs \

https://raw.githubusercontent.com/junegunn/vim-plug/master/plug.vim

* Configure

$ mkdir ~/.vim/plugged

$ vim ~/.vimrc

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

call plug#begin(‘~/.vim/plugged’)

Plug ‘fatih/vim-go’

call plug#end()

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. Install boost

$sudo apt install libboost-all-dev

1. Install libstdc++ man

$ sudo apt install libstdc++-7-doc

1. Install Rust

<https://www.rust-lang.org/tools/install>

$ curl <https://sh.rustup.rs> –sSf | sh

1. Install GCC

<https://gcc.gnu.org/>

* Download the source

Download form one of the available mirrors at <https://gcc.gnu.org/mirrors.html>

***gcc-9.2.0.tar.xz***

* Decompress

$ cd ~/Downloads

$ xz –d gcc-9.2.0.tar.xz

$ tar –xvf gcc-9.2.0.tar

* Download GCC’s prerequisites

$ cd gcc-9.2.0

$ ./contrib/download\_prerequisites

--- snip ---

gmp-6.1.0.tar.bz2: OK

mpfr-3.1.4.tar.bz2: OK

mpc-1.0.3.tar.gz: OK

isl-0.18.tar.bz2: OK

All prerequisites downloaded successfully.

* Configure GCC

$ cd ../

$ mkdir gcc-9.2.0-build

$ cd gcc-9.2.0-build

$ ../gcc-9.2.0/configure –disable-multilib

* Build GCC

$ make

* Test whether GCC binaries built correctly

$ make –k check

[

Optional:

$ sudo apt install autogen

]

* Install GCC

$ sudo make install

$ sudo vi /etc/ld.so.conf.d/x86\_64-linux-gnu.conf

--- snip ---

/usr/local/lib64

…

$ sudo ldconfig

1. Reserved