

Introduction of Linux

Tao Ji

| oslab2017_class1@163.com

Yuhan Liu

| oslab2017_class2@163.com

PART I

- Brief Introduction
- Basic Conceptions & Environment
- Install & Configure a Virtual Machine
- Basic Commands

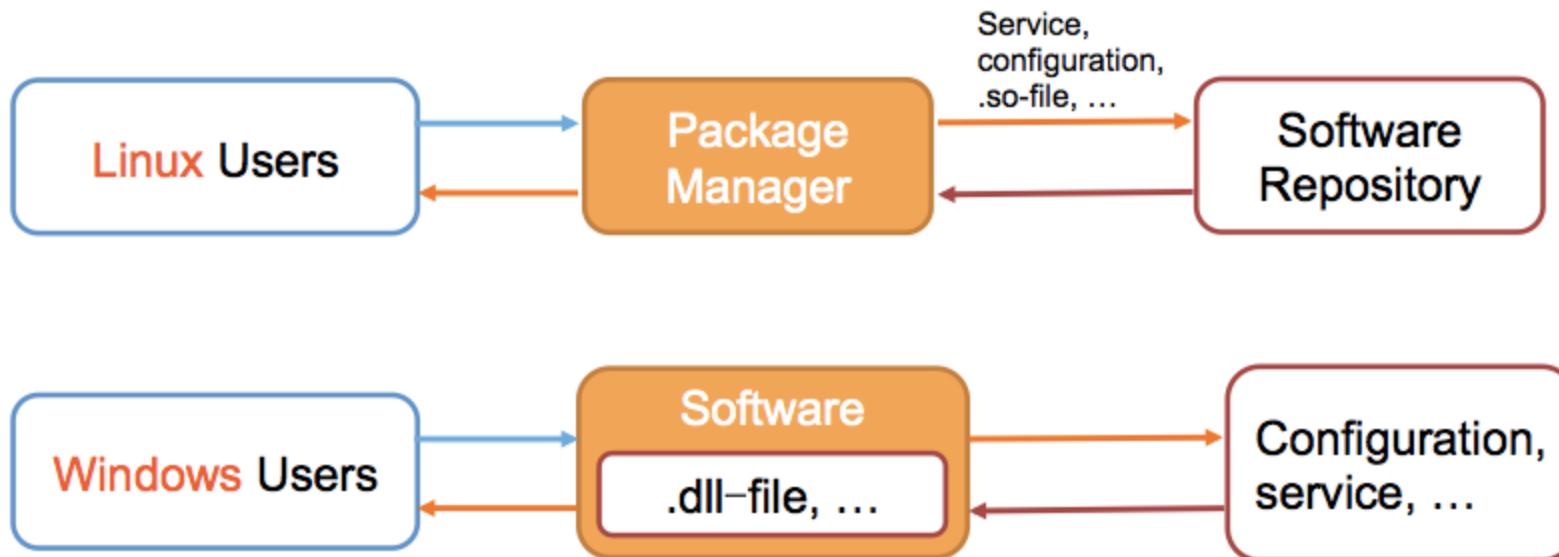
PART II

- Shell Script
- Compile & Debug (for C)
- Text Editor (Vim, Sublime text, Atom)

PART I

- Brief Introduction
- Basic Conceptions & Environment
- Install & Configure a Virtual Machine
- Basic Commands

Linux vs Windows Software



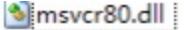
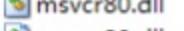
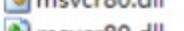
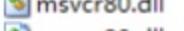
Linux install software

Package Manager: `apt-get` (Advanced Package Tool)

```
zheng@kernel:~$ sudo apt-get autoremove
Reading package lists... Done
Building dependency tree
Reading state information... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
zheng@kernel:~$ sudo apt-get install gcc
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  manpages-dev libc-dev-bin linux-libc-dev
Use 'apt-get autoremove' to remove them.
The following extra packages will be installed:
  binutils gcc-4.4 libc-dev-bin libgomp1 linux-libc-dev manpages-dev
Suggested packages:
  binutils-doc gcc-multilib autoconf automake1.9 libtool flex bison gdb
  gcc-doc gcc-4.4-multilib libmudflap0-4.4-dev gcc-4.4-doc gcc-4.4-locales
  libgcc1-dbg libgomp1-dbg libmudflap0-dbg libcloog-pp10 libpp1-c2 libpp17
Recommended packages:
  libc6-dev libc-dev
The following NEW packages will be installed:
  binutils gcc gcc-4.4 libc-dev-bin libgomp1 linux-libc-dev manpages-dev
0 upgraded, 7 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,147kB of archives.
After this operation, 22.8MB of additional disk space will be used.
Do you want to continue [Y/n]?
```

Windows install software

msvcr80.dll

 msvcr80.dll	C:\Program Files\AliWangWang	612 KB
 msvcr80.dll	C:\Program Files\AliWangWang\7.21.18C	612 KB
 msvcr80.dll	C:\Program Files\AliWangWang\8.00.06C	612 KB
 msvcr80.dll	C:\Program Files\AliWangWang\8.00.08C	612 KB
 msvcr80.dll	C:\Program Files\AliWangWang\new	612 KB
 msvcr80.dll	C:\Program Files\Baidu\BaiduYun	618 KB
 msvcr80.dll	C:\Program Files\Baidu\BaiduYunGuanjia	618 KB
 msvcr80.dll	C:\Program Files\Tencent\Qzone	612 KB
 msvcr80.dll	C:\Program Files\Microsoft SQL Server\90\Setup Bootstrap	612 KB
 msvcr80.dll	C:\Program Files\Tencent\QQMusic\QzoneMusic	618 KB
 msvcr80.dll	C:\Program Files\Tencent\Qzone\Ver_247.311	612 KB
 msvcr80.dll	C:\Program Files\Tencent\QQMusic\QzoneMusic\QQMusicAd...	618 KB
 msvcr80.dll	C:\Program Files\Common Files\Tencent\QQMiniDL\41\BT	618 KB
 msvcr80.dll	C:\Program Files\Common Files\Tencent\QQMiniDL\41\eMule	618 KB

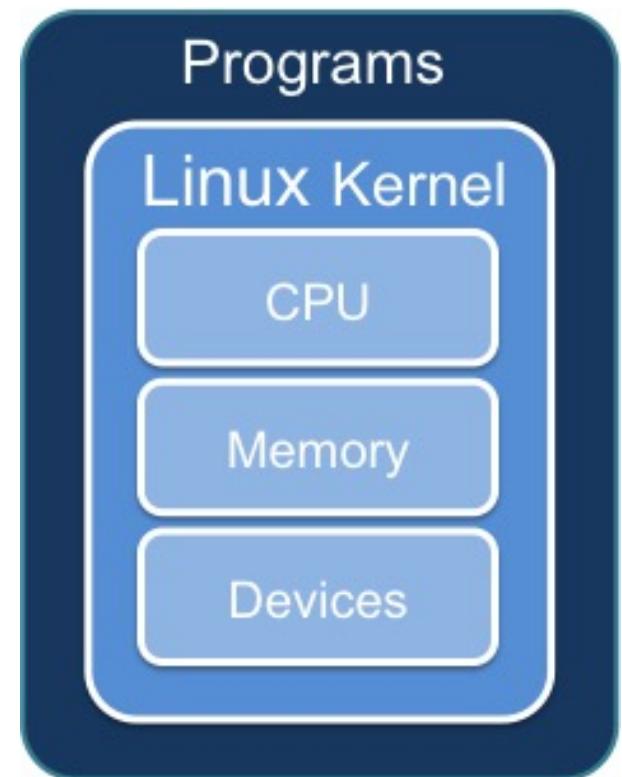
PART I

- Brief Introduction
- Basic Conceptions & Environment
- Install & Configure a Virtual Machine
- Basic Commands

Linux Kernel

The most important component of Linux OS, containing all the operating system's **core functions** and the **device drivers**.

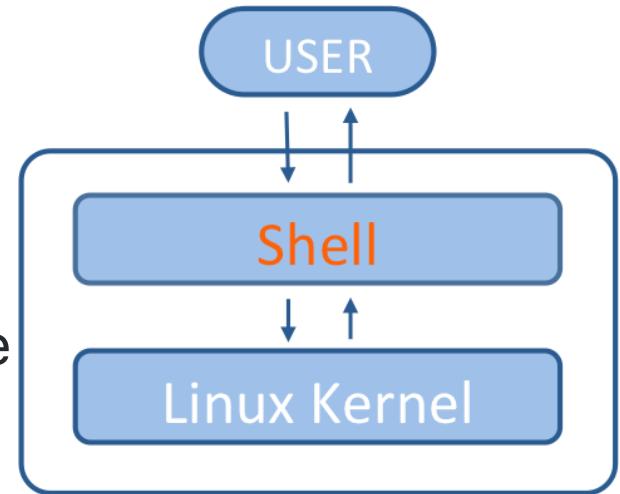
- memory management
- process scheduling
- file system
- ...



Shell (CLI shell)

Command Line Interface

A **program** which accepts commands as text input and **converts commands** to appropriate operating system functions.



Terminal ↔ Shell

The terminal send information to the shell, receive and display the information from the shell.

cd (change directory)

```
cd  
cd ~  
cd -  
cd ..
```

pwd (print working directory)

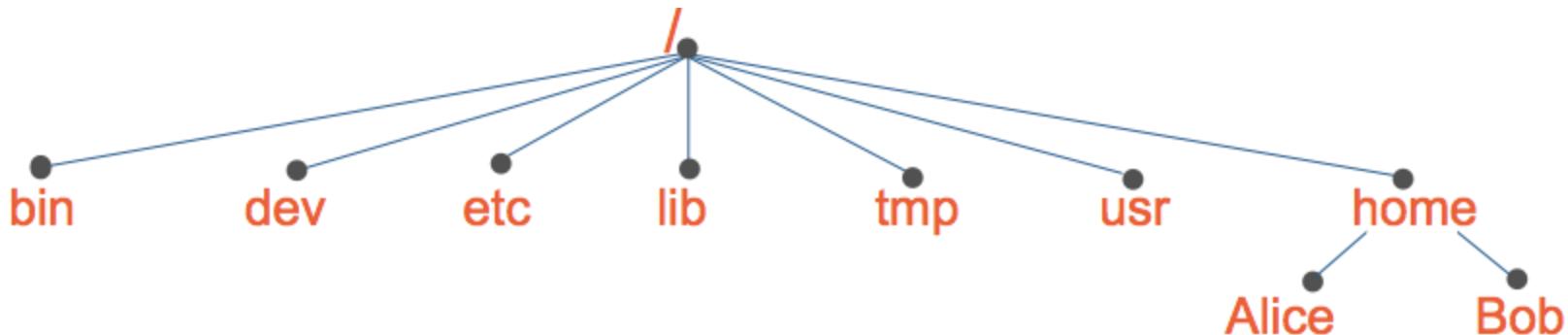
```
pwd
```

ls (list segment)

- l long - Displaying long format
- a all - Lists all files in the given directory
- R recursive - Recursively lists subdirectories.
- d directory - Shows information about a directory

```
ls  
ls -l  
ls -a  
ls -R  
ls -d  
ls -la  
ls -ld  
...  
...
```

File System



Tree structure, with the root directory " / "

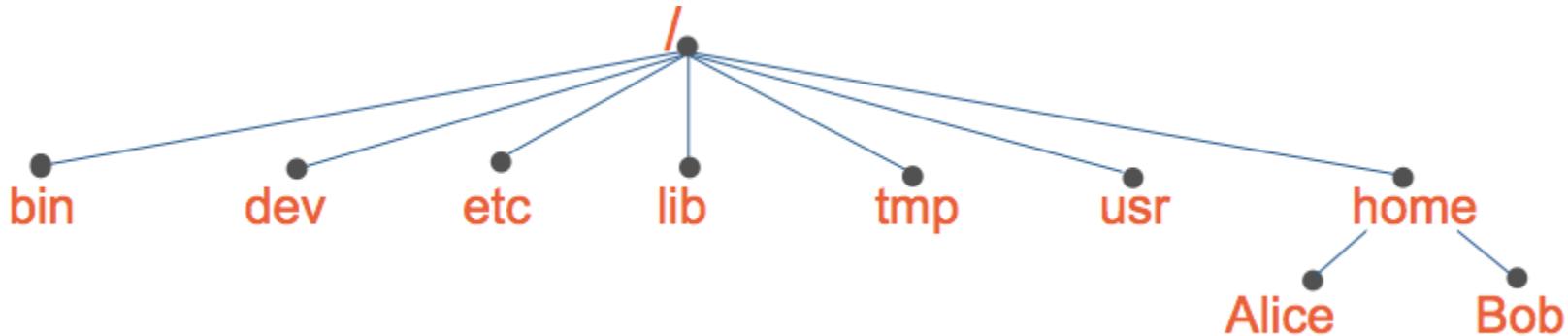
/home/Alice/...

~ = /home/Alice

.

..

File System



- /bin : essential tools and other programs
- /dev : files representing the system's hardware devices
- /etc : system configuration files
- /home : the home directory for all system's users
- /lib : essential system library files
- /proc : files that give information about current system
- /usr : files related to user tools and applications

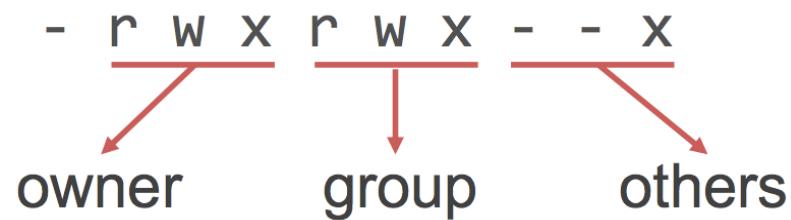
User & Group

The system determines whether or not a **user** or **group** can access a file or directory.

There is a special user called **Super User** or the **root** which has permission to access any file and directory.

Three **Permissions**:

- **r** = read
- **w** = write
- **x** = execute



sudo (superuser do)

groupadd

```
sudo groupadd TA  
sudo groupadd boys  
sudo groupadd girls
```

useradd

```
sudo useradd jt -m -g TA -G boys -s /bin/bash  
sudo useradd lyh -m -g TA -G girls -s /bin/bash
```

passwd

```
sudo passwd jt  
sudo passwd lyh
```

su (switch user)

```
su jt
```

chmod (change mode)

```
chmod 660 class1.txt  
chmod g-w class1.txt
```

cat (concatenate)

```
cat class1.txt  
cat jt.txt
```

Environment Variables

Environment variables are a **set of values** that can affect the way running processes will behave on a computer.

- **PATH** -- Contains a colon-separated list of directories that the shell searches for commands that do not contain a slash in their name.
- **HOME** -- Contains the location of the user's home directory.
- ...

Set The Environment Variables:

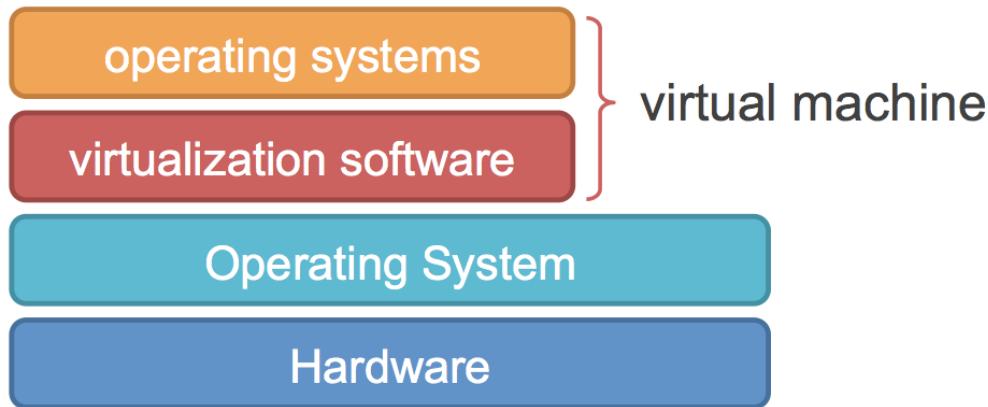
```
export VARIABLE = value      # temporary  
/etc/profile                 # permanent, all users  
  
~/.profile                   # permanent, one user  
~/.bashrc
```

PART I

- Brief Introduction
- Basic Conceptions & Environment
- Install & Configure a Virtual Machine
- Basic Commands

Virtual Machine

A virtual machine is an emulation of a particular computer system.



Virtualization Software provide (hardware) resources virtually to the new OS.

- VMware
- Virtual Box
- Virtual PC

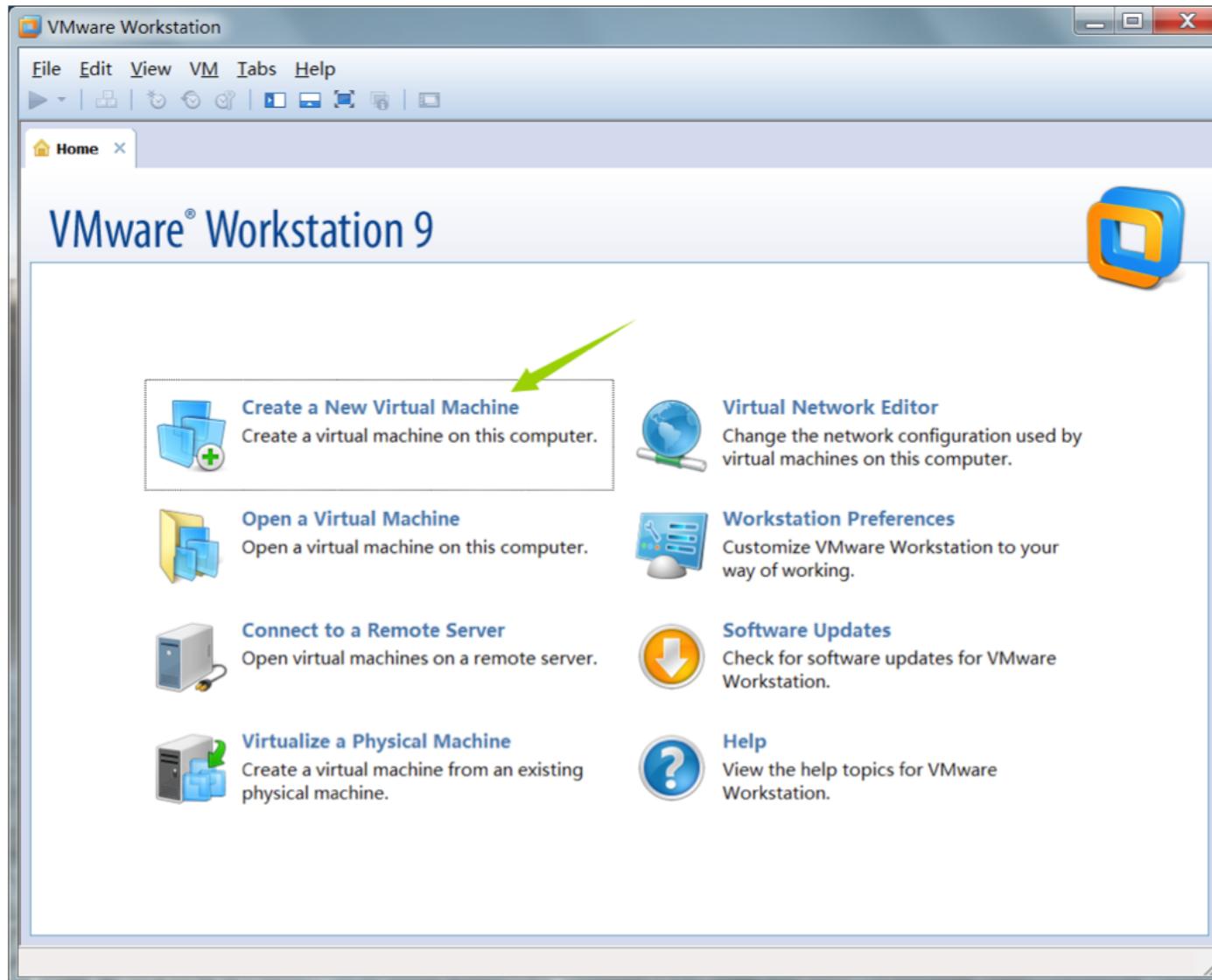
Install the Virtual Machine

VMware Workstation 9.0 + Ubuntu 14.04 LTS (kernel 3.19)



1. Download the Setup File of Vmware 9.0
2. Download the Ubuntu Ubuntu 14.04 LTS from the official website www.ubuntu.com/download/desktop
3. Install VMWare 9.0
4. Create a Virtual Machine in the VMware

Create a Virtual Machine



Create a Virtual Machine

The image shows two windows of the "New Virtual Machine Wizard" for VMware Workstation 9.

Welcome to the New Virtual Machine Wizard

What type of configuration do you want?

- Typical (recommended)
Create a Workstation 9.0 virtual machine in a few easy steps.
- Custom (advanced)
Create a virtual machine with advanced options, such as a SCSI controller type, virtual disk type and compatibility with older VMware products.

Choose the Virtual Machine Hardware Compatibility

Which hardware features are needed for this virtual machine?

Virtual machine hardware compatibility:

- Hardware: Workstation 9.0 (selected)
- Compatible: ESX Server (checkbox checked)

Compatible products:

- Fusion 5.0
- Workstation 9.0

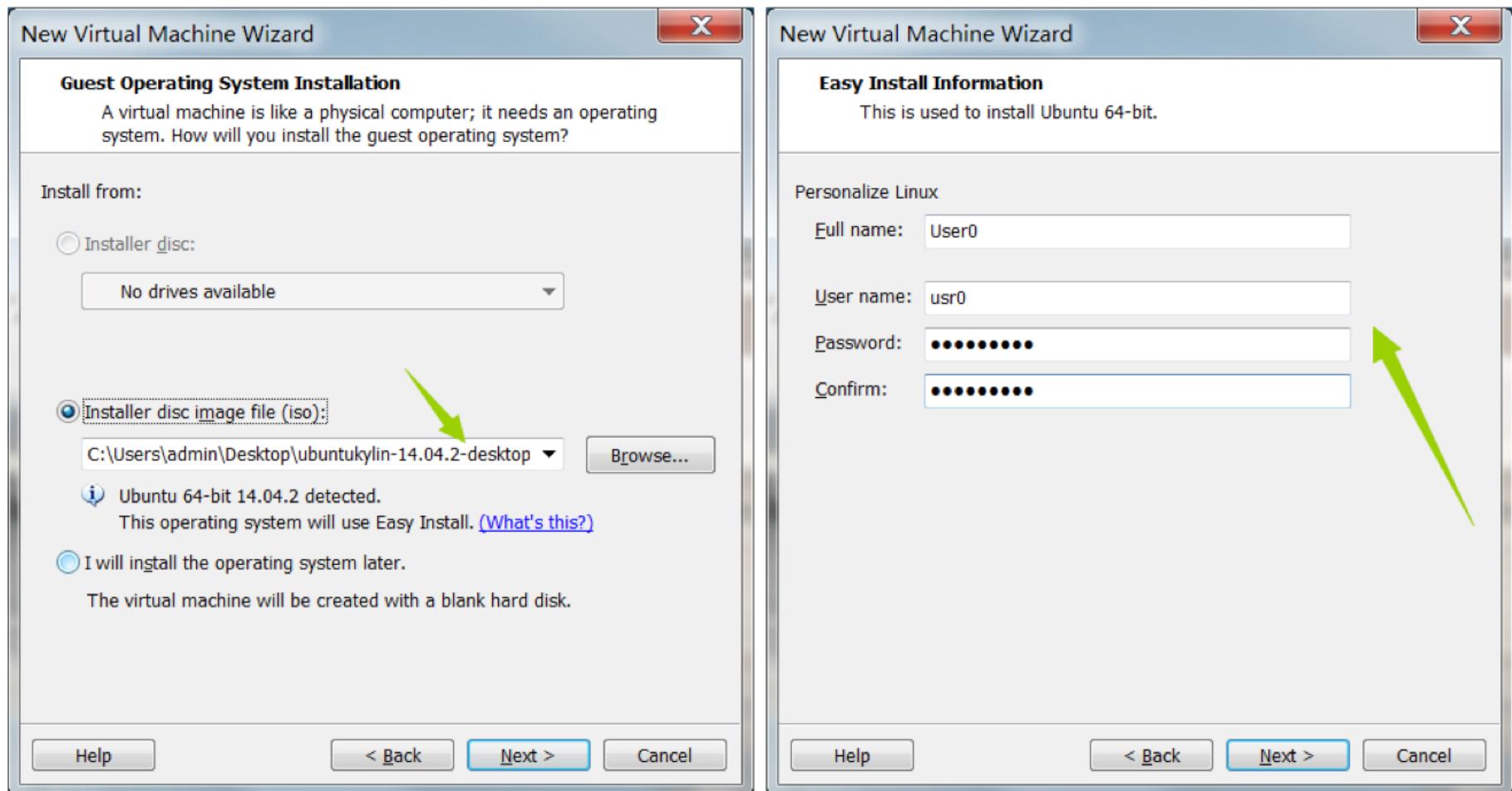
Limitations:

- 64 GB memory limit
- 8 processor limit
- 10 network adapter limit
- 2 TB disk size limit

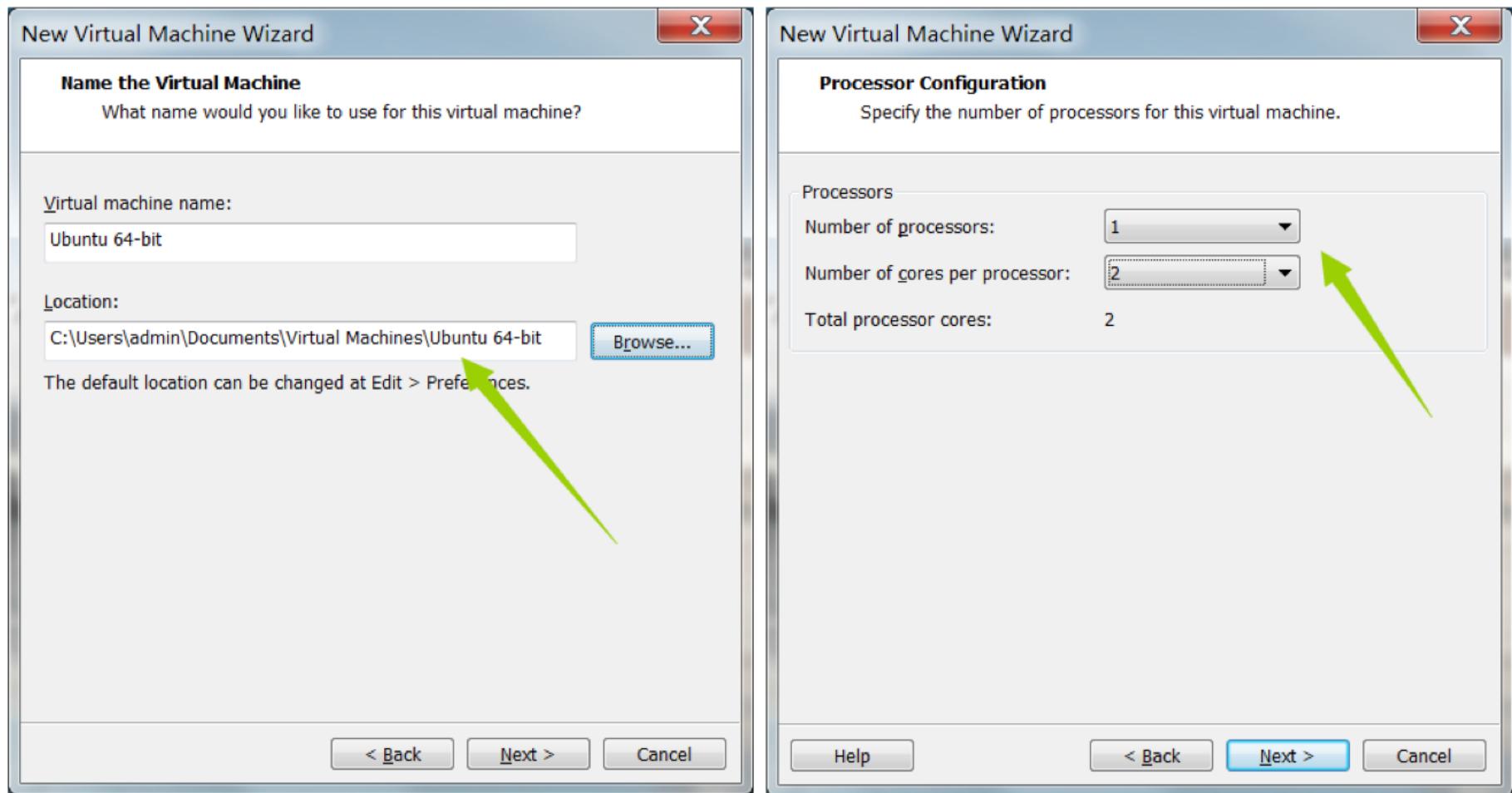
Buttons at the bottom of both windows:

- Help
- < Back
- Next >
- Cancel

Create a Virtual Machine



Create a Virtual Machine



Create a Virtual Machine

New Virtual Machine Wizard

Memory for the Virtual Machine

How much memory would you like to use for this virtual machine?

Specify the amount of memory allocated to this virtual machine. The memory size must be a multiple of 4 MB.

Memory for this virtual machine: 4096 MB

Maximum recommended memory: 13644 MB

Recommended memory: 1024 MB

Guest OS recommended minimum: 512 MB

Help < Back Next > Cancel

New Virtual Machine Wizard

Network Type

What type of network do you want to add?

Network connection

Use bridged networking
Give the guest operating system direct access to an external Ethernet network. The guest must have its own IP address on the external network.

Use network address translation (NAT)
Give the guest operating system access to the host computer's dial-up or external Ethernet network connection using the host's IP address.

Use host-only networking
Connect the guest operating system to a private virtual network on the host computer.

Do not use a network connection

Help < Back Next > Cancel

Create a Virtual Machine

The image displays two windows from the "New Virtual Machine Wizard".

Left Window: Select I/O Controller Types

This window asks, "Which SCSI controller type would you like to use?". It lists three options:

- IDE Controller: ATAPI
- SCSI Controller:
 - BusLogic (Not available for 64-bit guests)
 - LSI Logic (Recommended)
 - LSI Logic SAS

A green arrow points to the "LSI Logic SAS" option.

Right Window: Select a Disk

This window asks, "Which disk do you want to use?". It shows a list of disk options:

- Create a new virtual disk

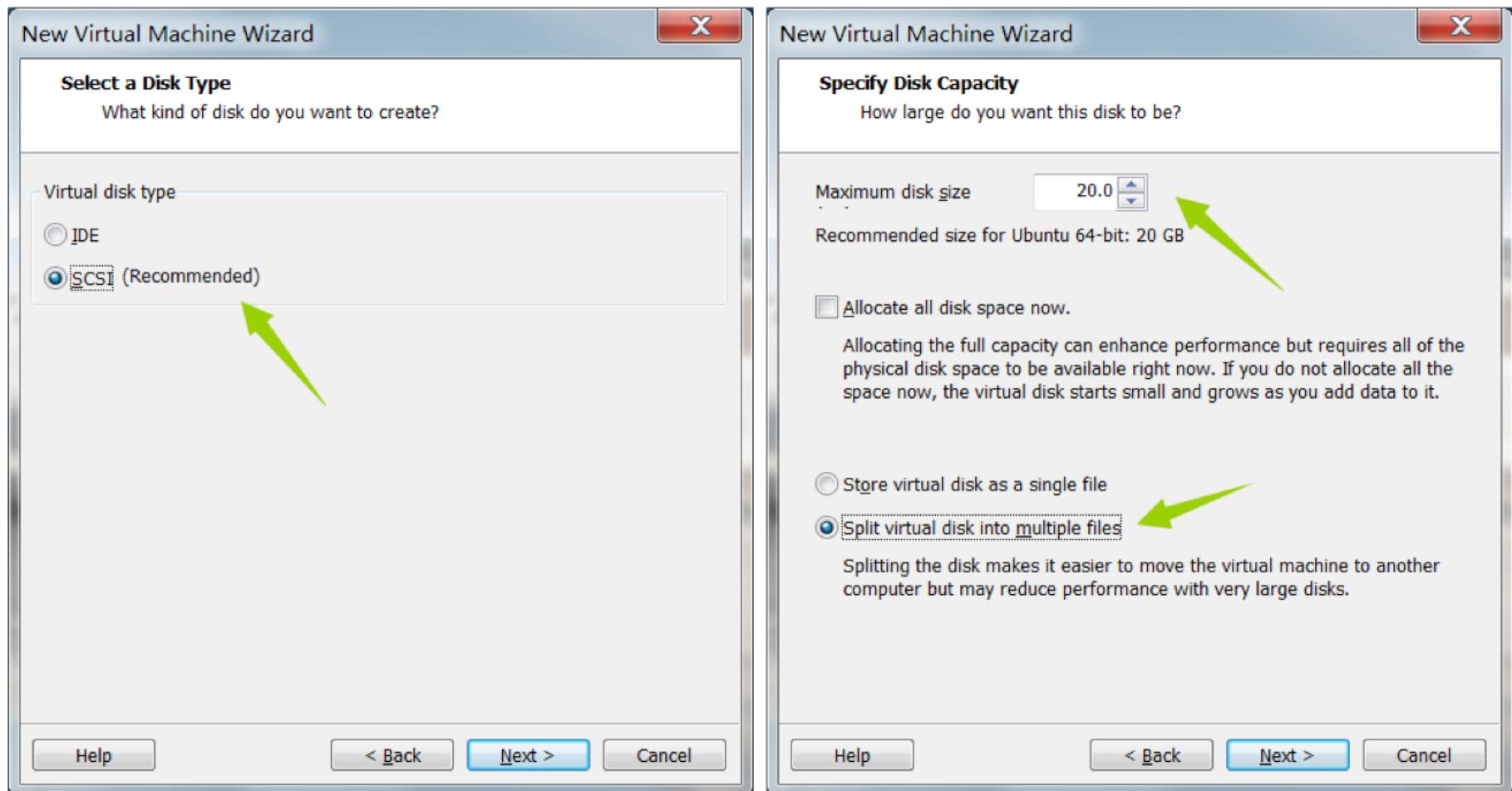
A green arrow points to this option. A detailed description follows: "A virtual disk is composed of one or more files on the host file system, which will appear as a single hard disk to the guest operating system. Virtual disks can easily be copied or moved on the same host or between hosts."
- Use an existing virtual disk

Description: Choose this option to reuse a previously configured disk.
- Use a physical disk (for advanced users)

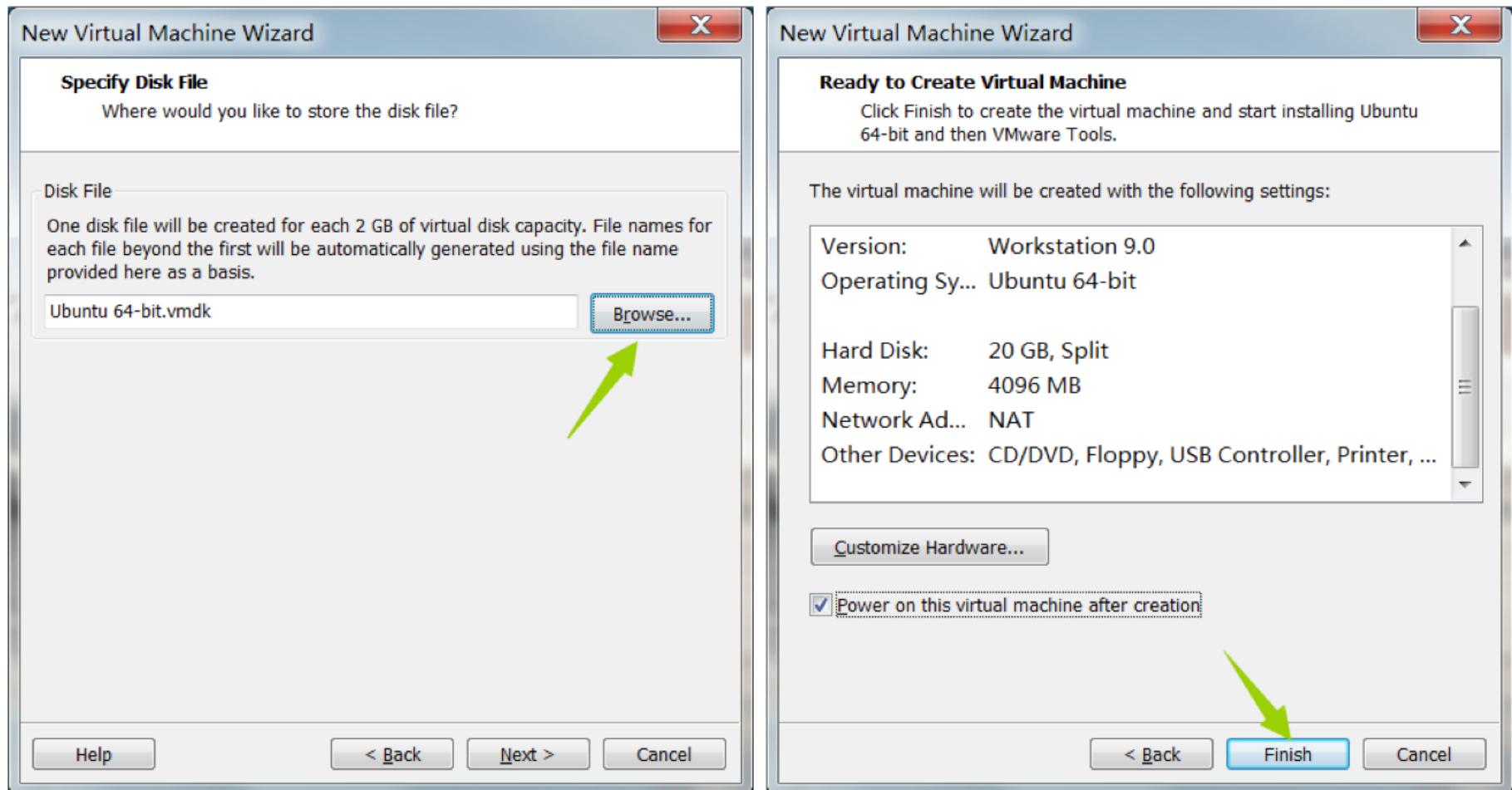
Description: Choose this option to give the virtual machine direct access to a local hard disk.

Both windows have standard navigation buttons at the bottom: Help, < Back, Next >, and Cancel.

Create a Virtual Machine



Create a Virtual Machine



Create a Virtual Machine (Mac)

Mac Virtual Machine -- Parallels desktop

Parallels desktop

magnet:?

xt=urn:btih:B219AFA0B62595C6E303DCB2BCA7D23EC7B0CE35

ubuntu-14.04.5-desktop-i386.iso

magnet:?

xt=urn:btih:5EE7E1DC3E01F362B0E53BFEE9E4D6DCDEDAD61B

Create a Virtual Machine (Mac)



Create a Virtual Machine (Mac)



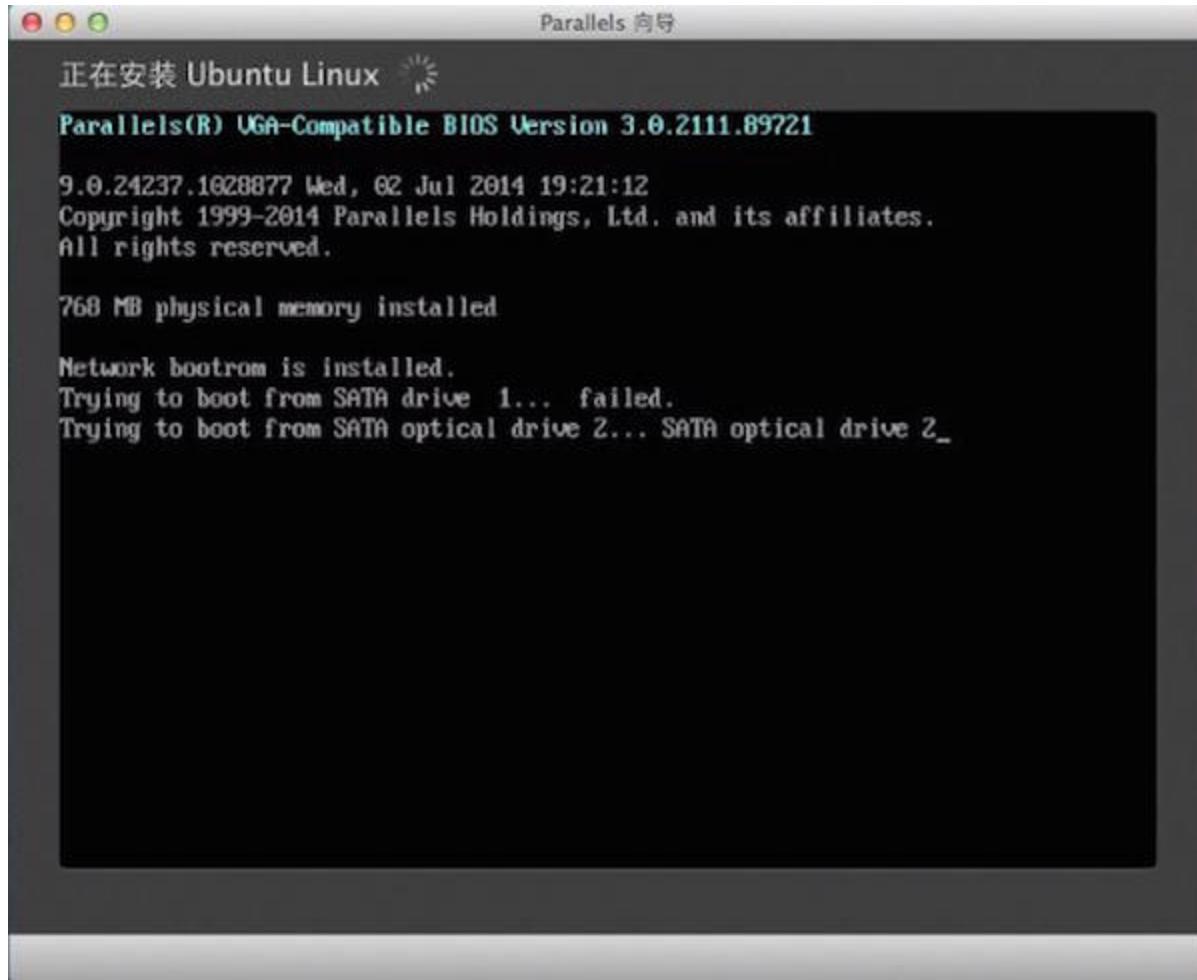
Create a Virtual Machine (Mac)



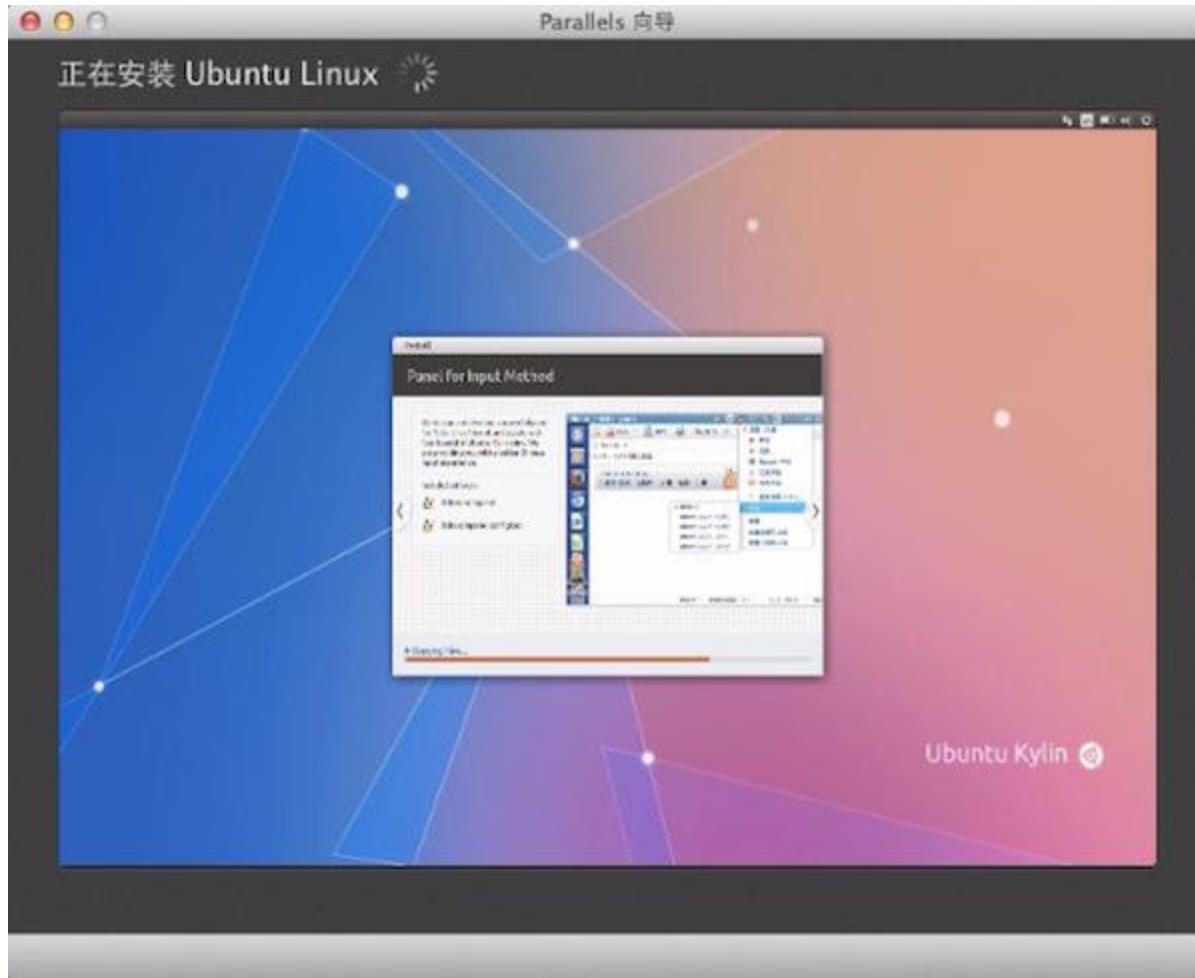
Create a Virtual Machine (Mac)



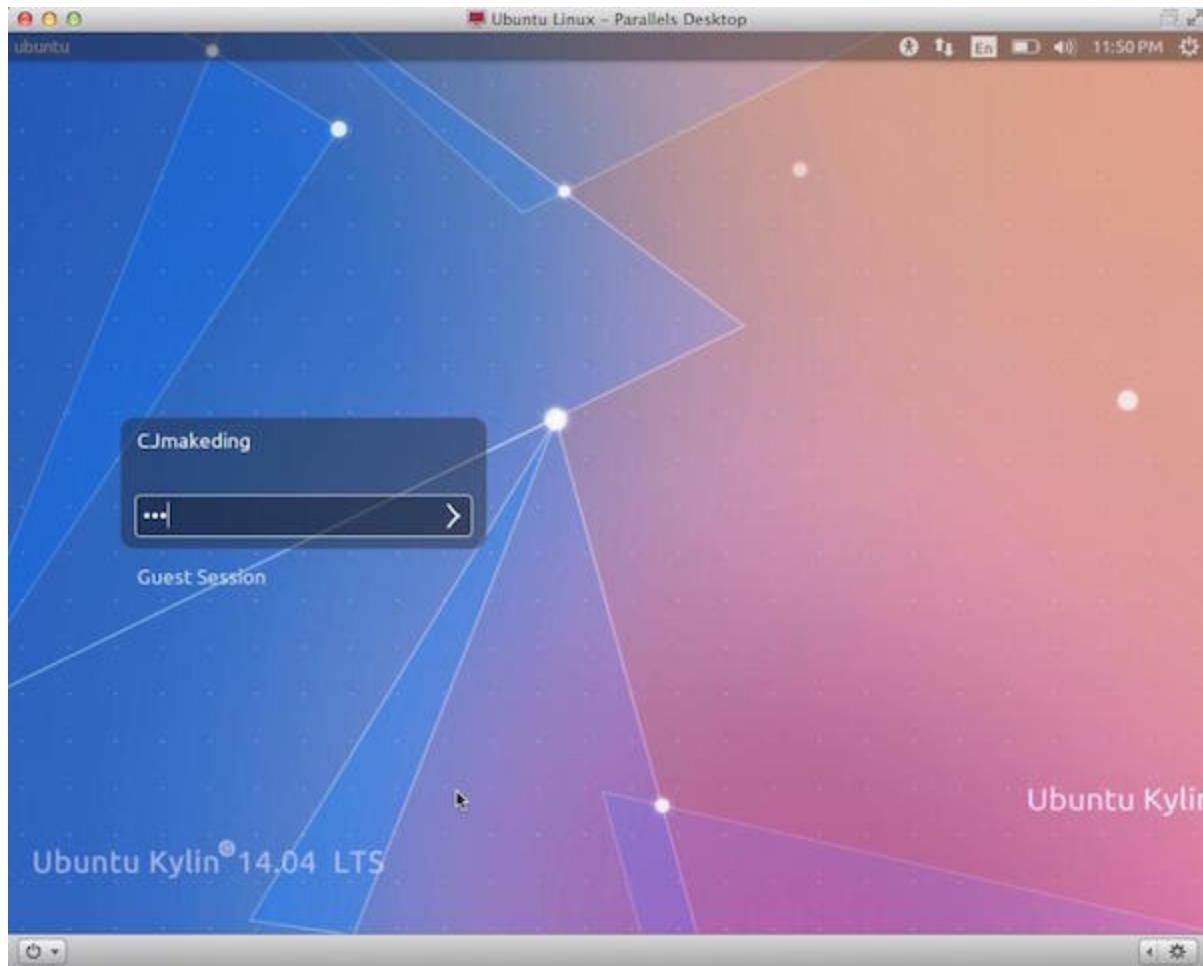
Create a Virtual Machine (Mac)



Create a Virtual Machine (Mac)



Create a Virtual Machine (Mac)



PART I

- Brief Introduction
- Basic Conceptions & Environment
- Install & Configure a Virtual Machine
- **Basic Commands**

Basic Commands

command [-options] [arguments]

- touch rename mv cp
- mkdir rmdir rm
- find grep
- > >> | xargs
- awk
- man help --help

touch

```
touch class1.txt
```

rename

```
rename 's/oslab/oslab0/' o*b?.txt
```

mv (move)

```
mv oslab.txt oslab1.txt  
mv oslab01.txt oslab02.txt /home/oslab
```

cp (copy)

```
cp oslab03.txt /home/oslab
```

mkdir (make directory)

```
mkdir Lesson1/ rename
```

rmdir (remove **empty** directory)

```
rmdir empty_directory
```

rm (remove)

-r recursive

-i interactive

-f force

```
rm -rf ~/Lesson1/*  
rm -i oslab04.txt
```

find

```
find ~ -name "*.txt"
```

grep

| globally search a regular expression and print

```
grep match_pattern file_name  
grep apple oslab05.txt  
grep -i apple oslab05.txt
```

> & >> (redirection)

```
cat oslab06.txt oslab07.txt > oslab08.txt  
cat oslab06.txt oslab07.txt >> oslab08.txt
```

| (pipeline)

```
command1 | command2  
cat oslab09.txt | grep jt
```

xargs

```
cat oslab09.txt | ls -l  
cat oslab09.txt | xargs ls -l
```

awk (Aho, Weinberg & Kernighan)

AWK is a programming language designed for text processing and typically used as a data extraction and reporting tool.

pattern { action }

BEGIN、regular expression、END

{ function calls, variable assignments, calculations }

```
awk 'BEGIN { print "Hello, world!" }'
```

man (manual)

```
man ls
```

help

```
help cd
```

--help

```
ls --help
```

Wikipedia

<https://en.wikipedia.org/wiki/AWK>

PART II

- Shell Script
- Compile & Debug (for C)
- Text Editor (Vim, Sublime text, Atom)

Variable

Define, Assignment & Read

```
VariableName=value  
read VariableName
```

- no space between VarName and the equality sign
- first letter: a-z A-Z
- no keywords of shell

Use a variable

```
$VariableName  
${VariableName}
```

Special Variables

```
$0 # filename of the script  
$n # the n-th argument  
$# # the number of the arguments  
$HOME # user directory  
$$ # PID
```

Examples:

test1.sh

```
#!/bin/bash  
read a  
read b  
c=$((a+b)**a)  
echo $c
```

with arguments

```
#!/bin/bash  
echo $[$1+$2]**$1
```

String

single quotes

```
str='no variables or escape character'
```

double quotes

```
v='variables'  
str="$v or \"escape character\""
```

connecting

```
str1="connecting strings"  
str2="simple"  
str3=$str1" is "$str2
```

string length

```
 ${#string}
```

substring

```
 ${string:begin:len}
```

Example:

```
#!/bin/bash
str="alibaba is a great company"
echo ${#str}
echo ${str:1:4}
echo ${#str:1:4}
```

printf

differences from “printf” in C

- no ()
- using space between two arguments

if the number of arguments is **greater** than the number of **%** in format, the format-string will be **reused** repeatedly

```
printf "%s %s\n" 1 2 3 4
```

output:

```
1 2  
3 4
```

Branches

```
if [condition]
then
...
else
...
fi
```

or

```
if [condition1]; then
...
elif [condition2]; then
...
else
...
fi
```

Operator

Numerical Comparison Operators

Operator	Remark
-eq	==
-ne	!=
-gt	>
-lt	<
-ge	>=
-le	<=

Other Operators

Operator	Remark
=	== for string
!=	!= for string
-z	If the string is empty
-f / -d	is file / is dir.
-r / -w / -x	check permission
-e	if a file/dir. exists

Example:

```
#!/bin/bash
YACCESS=`date -d yesterday +%Y%m%d`
FILE="access_${YACCESS}.log.tgz"
if [ -f "$FILE" ];then
    echo "OK"
else
    echo "error $FILE"
fi
```

Loop

```
for variable in list  
    do  
        ...  
done
```

```
while [ condition ]  
    do  
        ...  
done
```

```
break  
continue
```

Example:

```
for FILE in $HOME/*
do
    echo $FILE
done

count=0
while [ $count -lt 5 ]
do
    count=$[$count+1]
    echo $count
done
```

PART II

- Shell Script
- Compile & Debug (for C)
- Text Editor (Vim, Sublime text, Atom)

Compilation & Execution

GCC (GNU C Compiler → GNU Compiler Collection)

```
gcc test.c      # compile the C source file
```

produce an executable file named (by default) `a.out`

```
./a.out      # run the program a.out
```

Useful Options

```
gcc -o test test.c
gcc -g -o test test.c
gcc test.c -g -o test
```

Separate Compilation

compile a program with several separate files

```
gcc -c test1.c  
gcc -c test2.c  
gcc test1.o test2.o -o test
```

-c : compile to produce an object file, which is not executables just machine-level representations of the source code.

Linking with Libraries

Library

lib+name.a (-static)

lib+name.so (default)

-l+name Link with libraries manually

-L+lib's dir Give the directory manually

```
gcc hello.c -shared -o libhello.so
gcc test.c -lhello -L. -o test
export LD_LIBRARY_PATH=.:${LD_LIBRARY_PATH}
```

```
gcc hello.c -c hello.o
ar -r libhello.a hello.o
gcc test.c -lhello -L. -static -o test
```

make↔Makefile

Build the program **automately** according to the **makefile**.

Makefiles are based on rules as:

```
target [target ...]: [component ...]
Tab ↵ [command 1]
.
.
.
Tab ↵ [command n]
```

```
hello.o: hello.c hello.h
Tab ↵ gcc hello.c -c -g
```

Debugging with GDB (GNU debugger)

gdb

Enter the gdb environment.

Command	Remark
file [file name]	load a executable file
r	run
c	continue
b [line number] b [function name]	set Breakpoint
s, n	excute a line of source code
p [variable name]	print the value of a variable
q	quit
help [command]	

PART II

- Shell Script
- Compile & Debug (for C)
- Text Editor (Vim, Sublime text, Atom)

Recommended Editors

Sublime



Atom



Vim(CLI)



Superorities

Cross-platform

Extensible

Lightweight

Sublime



A sophisticated text editor for code, markup and prose

source: <http://www.sublimetext.com/>

Installation for Linux

via Package Manager(apt-get)

Install the GPG key:

```
wget -qO - https://download.sublimetext.com/  
sublimehq-pub.gpg | sudo apt-key add -
```

Select the channel to use:

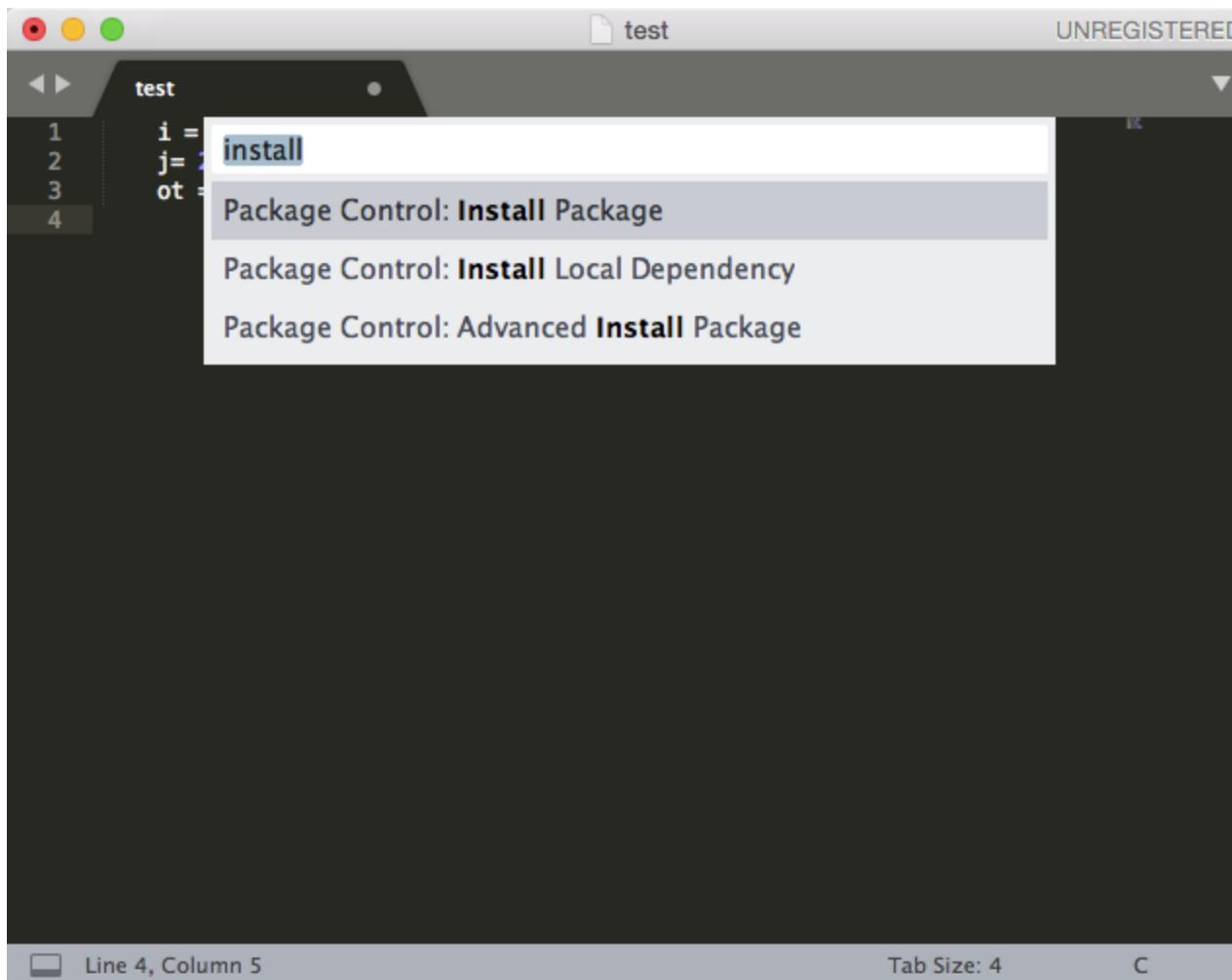
```
echo "deb https://download.sublimetext.com/ apt/stable/"  
| sudo tee /etc/apt/sources.list.d/sublime-text.list
```

Update apt sources and install Sublime Text:

```
sudo apt-get update  
sudo apt-get install sublime-text
```

Package Control

- go to Command Palette (**ctrl+shift+p**)
- type **install**
- you will see a list of plugins



Plugins

To see the list of plugins(**Preferences=>Package Settings**)

Alignment

For code alignment(**ctrl+alt+a**)

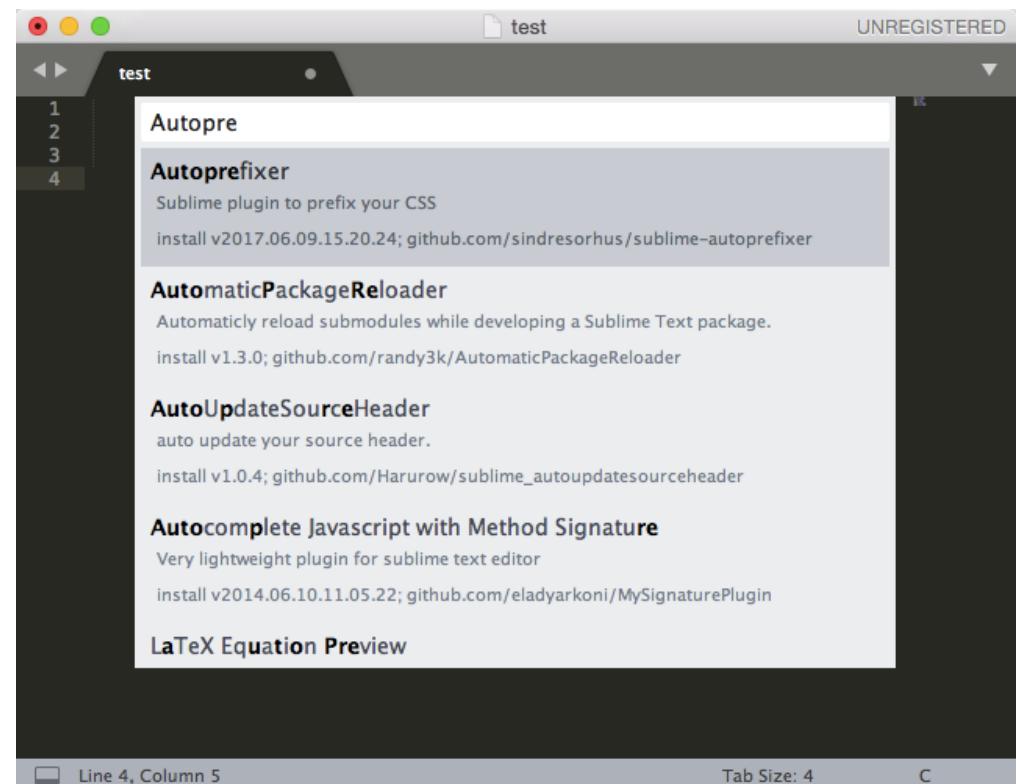
BracketHighlighter

For code highlighting

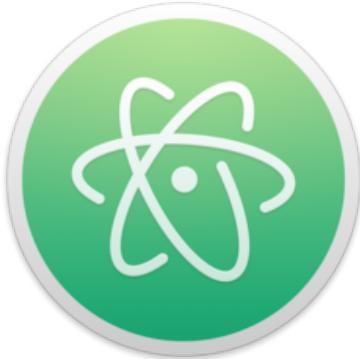
DictionaryAutoComplete

For dictionary completing

...



Atom



A hackable text editor for the 21st Century

source: <https://atom.io/>

Similar to Sublime

Installation for Linux

via Package Manager(apt-get)

```
sudo add-apt-repository ppa:webupd8team/atom  
sudo apt-get update  
sudo apt-get install atom
```

Vim



Vim is a highly configurable text editor built to make creating and changing any kind of text very efficient.

Installation for Linux

via Package Manager(apt-get)

```
sudo apt-get install vim  
vimtutor # obtain a vim tutorial
```

Create a file

```
vim filename
```

Three Modes

Command Mode

all the keys are bound to commands (typing "j" -- it will move the cursor down one line)

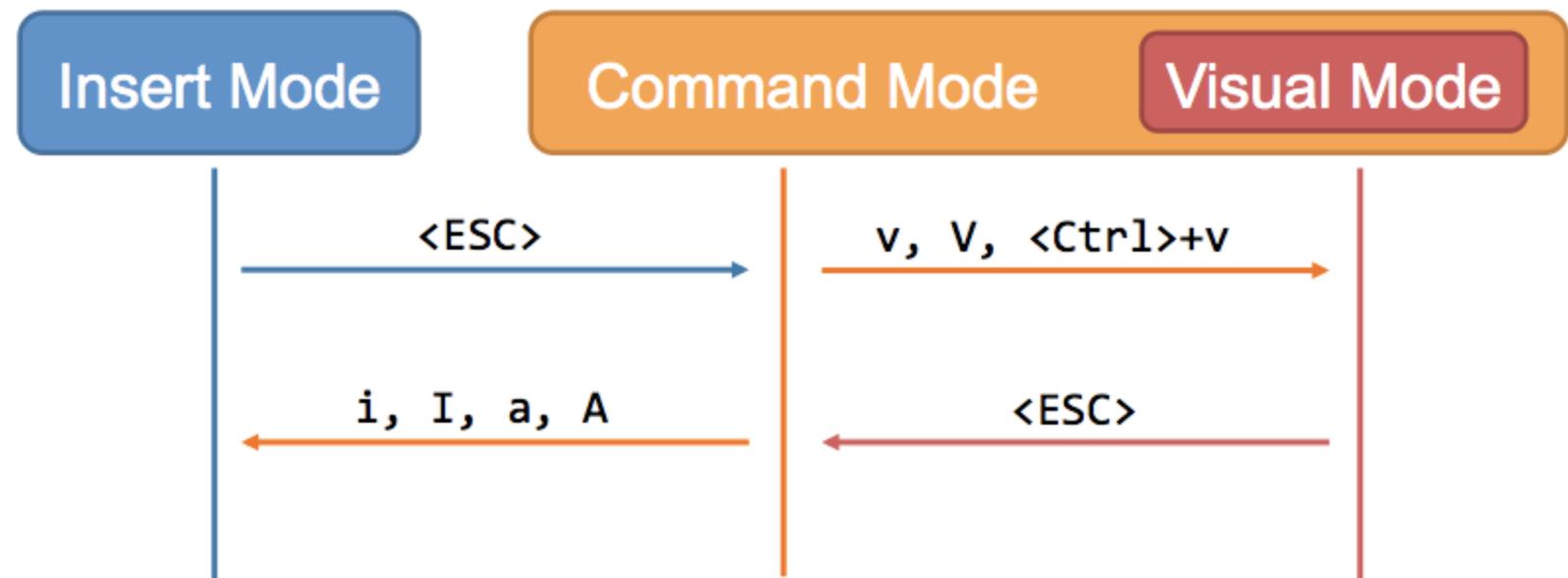
Insert Mode

all the keys are exactly keys (typing "j" -- inserting "j")

Visual Mode

helps to visually select some text, may be seen as a submode of the command mode

Three Modes



Keys in command mode

Quit and Save

- `w` write the current buffer to disk (save)
- `q` close the current window
- `x` or `wq` save and close
- `q!` close without save

Scroll the Screen

downwards

- **ctrl+f** 1 page
- **ctrl+d** 1/2 page
- **ctrl+e** 1 line

upwards

- **ctrl+y** 1 page
- **ctrl+u** 1/2 page
- **ctrl+b** 1 line

Movement of the Cursor

- `h` moves the cursor one character to the left.
- `j` moves the cursor down one line.
- `k` moves the cursor up one line.
- `l` moves the cursor one character to the right.
- `0` moves the cursor to the beginning of the line.
- `$` moves the cursor to the end of the line.
- `w` moves forward one word.
- `b` moves backward one word.
- `G` moves to the end of the file.
- `gg` moves to the beginning of the file.
- ``.` moves to the last edit.