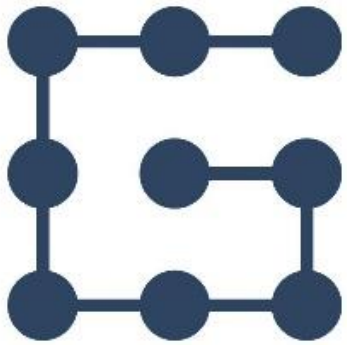


ROS机械臂开发

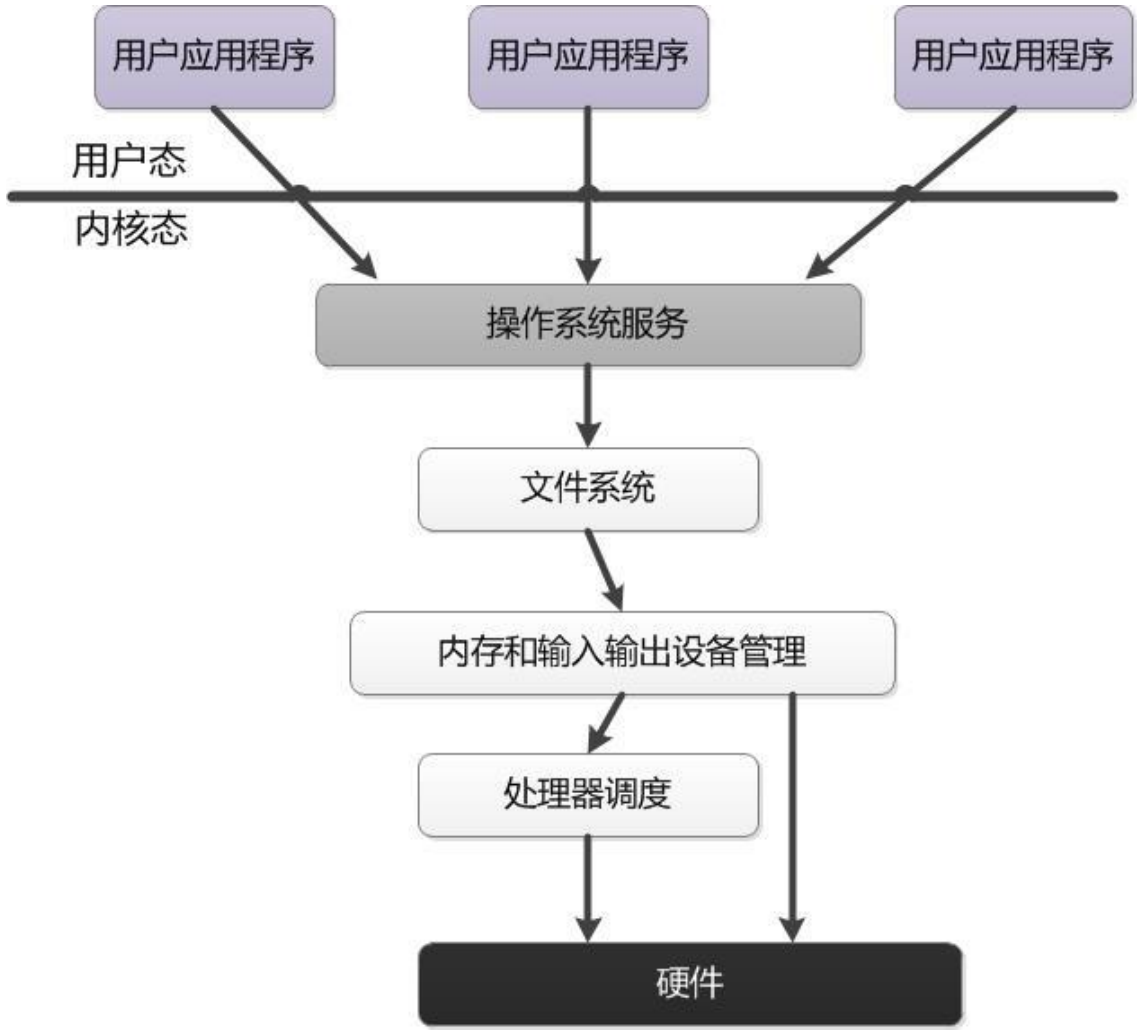
—— 1. Linux基础简介

目录

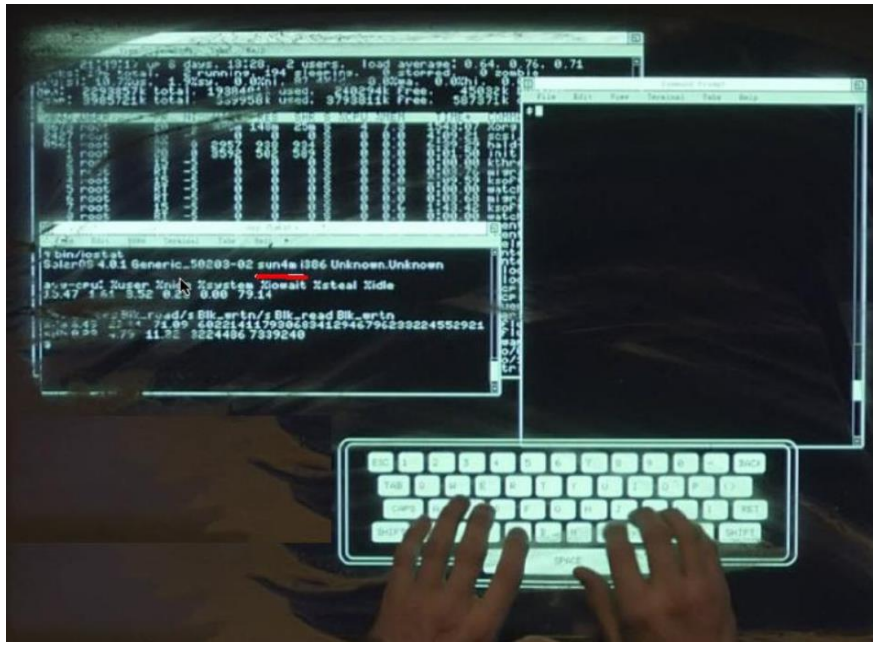
目录

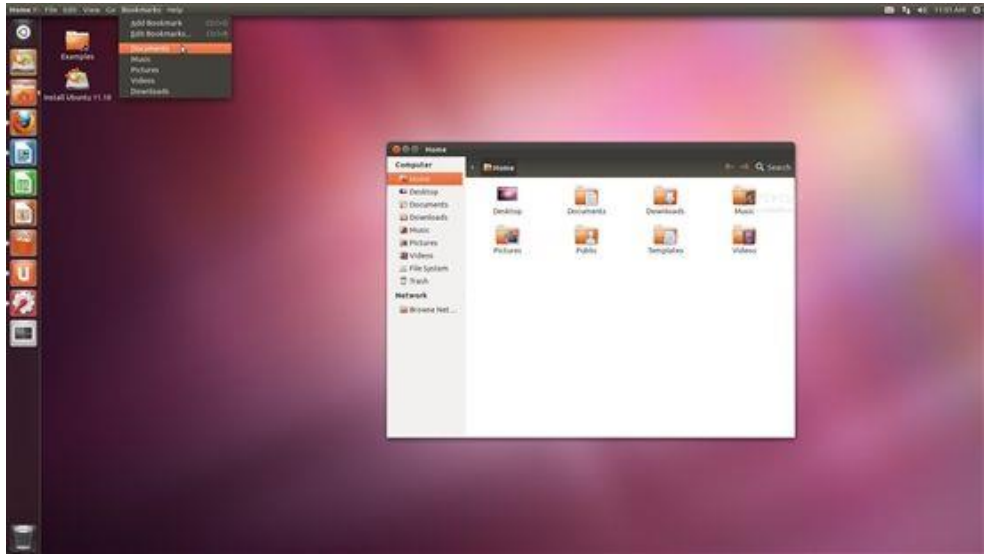
1. Linux简介 —— 操作系统是什么

- 操作系统（Operating System，OS）
  - 管理和控制计算机硬件与软件资源的计算机程序；
  - 直接运行在“裸机”上的最基本的系统软件；
  - 任何其他软件都必须在操作系统的支持下才能运行。



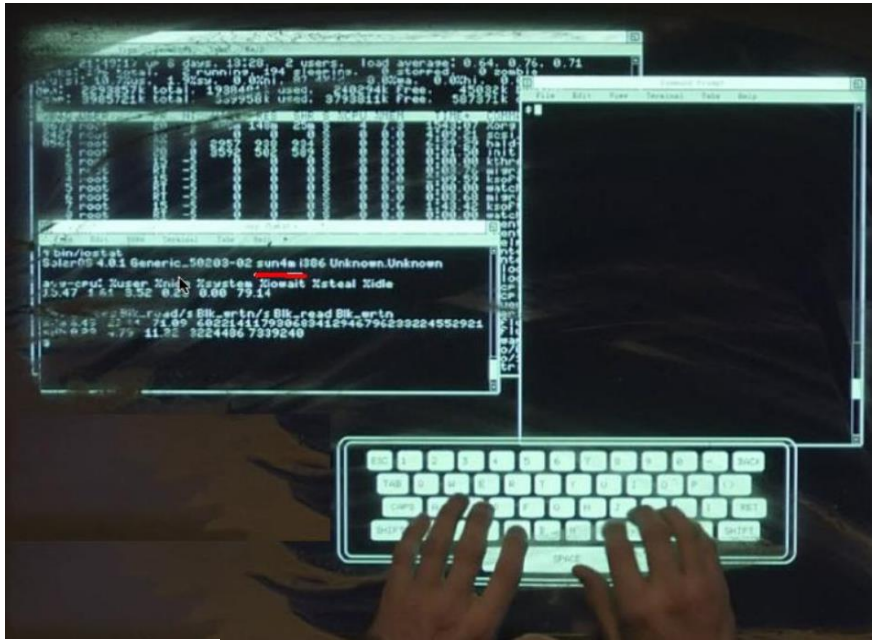
1. Linux简介 —— 操作系统是什么





1. Linux简介 —— Unix是什么

- 最早的多用户、多任务操作系统
- 应用于从巨型计算机到普通PC机等多种不同的平台上
- 是目前应用面最广、影响力最大、稳定性最好的操作系统



1. Linux简介 —— Unix是什么

- Unix在发展过程中产生了许多版本或分支：

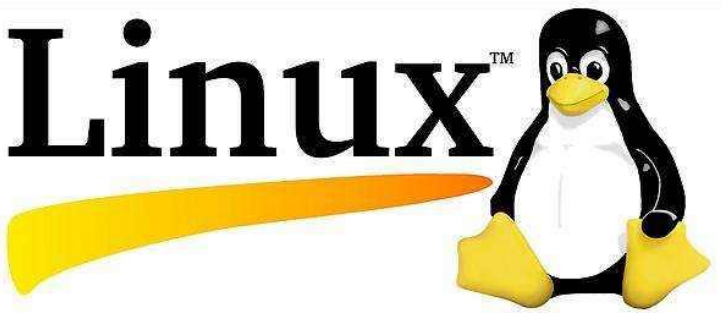
  - BSD，美国加州大学伯克利分校推出的“Berkeley Software Distribution”，简称BSD。
  - AIX，IBM公司主持研究的Unix版本，主要是针对IBM计算机硬件环境进行了优化和增强。
  - HP-UX，HP公司的Unix系统版本，主要运行在HP的计算机和工作站上。
  - Solaris，原来称为Sun OS，是Sun公司开发的Unix版本，包含有Sun公司开发的许多图形用户界面系统工具和应用程序，主要用于Sun公司的计算机和工作站上。

1. Linux简介 —— Linux是什么

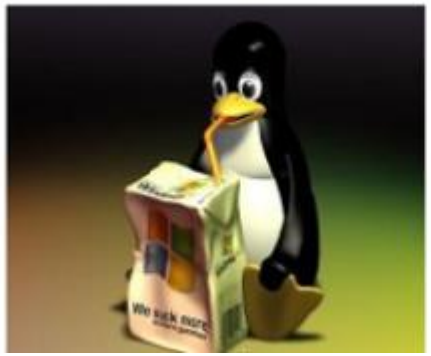
- 关于Linux，首先要注意以下两个**要点**：
  - 类Unix系统：Linux是一种自由、开放源码的类似Unix的操作系统
  - Linux内核：严格来说，Linux这个词本身只表示Linux内核

1. Linux简介 —— Linux是什么

- 关于Linux诞生历史，关注以下几个**要点**：



  - 1991年，芬兰的业余计算机爱好者Linus Torvalds
  - 编写了一款类似Minix的系统（基于微内核架构的类Unix操作系统）
  - 被ftp管理员命名为Linux
  - 加入到自由软件基金的GNU计划中
  - Linux以一只可爱的**企鹅**作为标志，象征着敢作敢为、热爱生活。



1. Linux简介 —— Linux发行版是什么

ubuntu 友帮拓

redhat

openSUSE

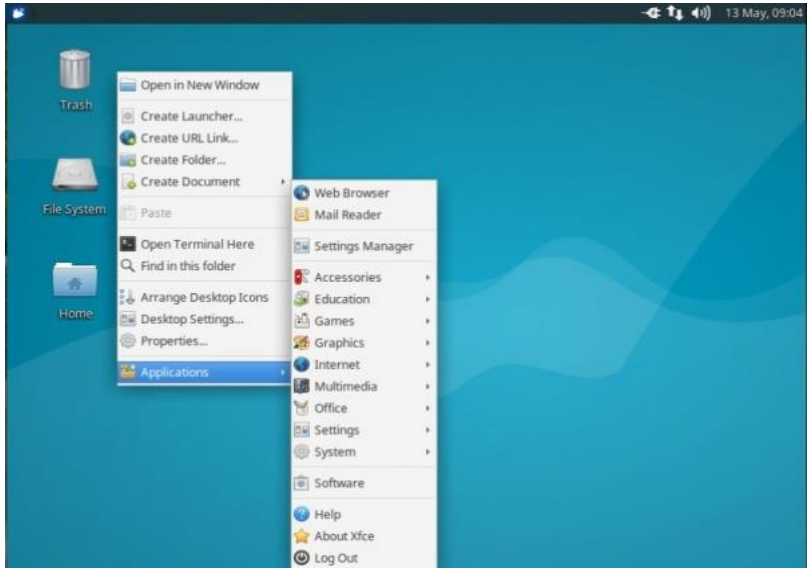
1. Linux简介 —— Linux VS Windows

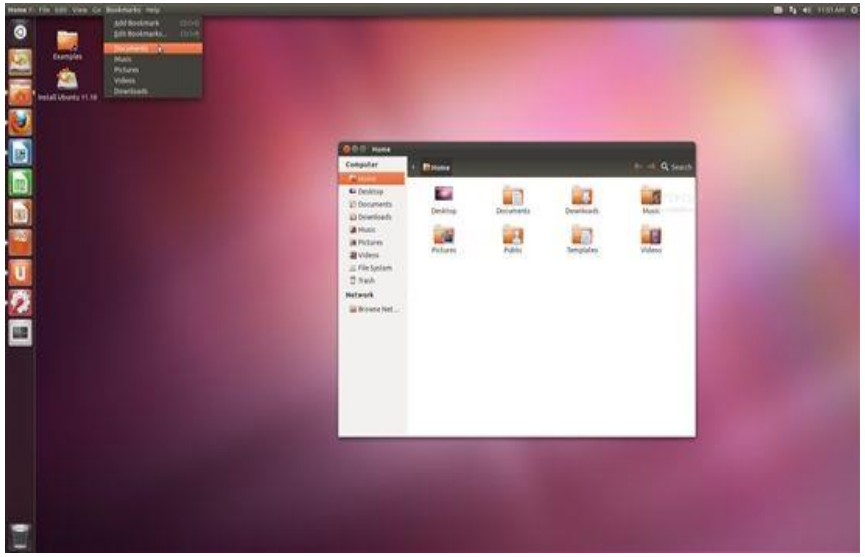
➢Linux和Windows的不同之处如下 ：

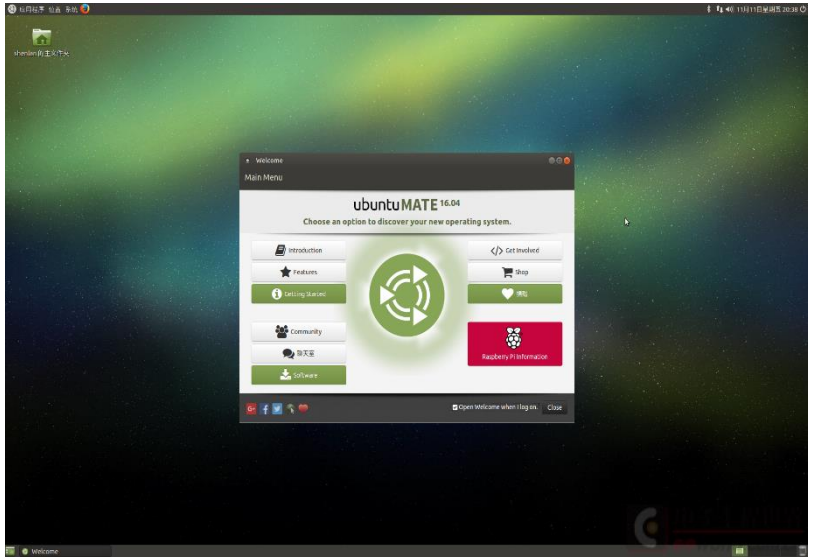
比较项	Linux	Windows
定位	Linux的设计定位是网络，设计灵感来自于网络操作系统Unix，因此它的命令的设计比较简单、简洁。由于纯文本可以非常好地跨网络工作，所以Linux配置文件和数据都以文本为基础	Windows最初的目标是家庭和办公应用，例如打印、图形化服务
图形用户界面	图形环境并没有集成到Linux内核中，而是运行于系统之上的单独一层，这意味着可以在需要时再运行GUI	Windows是把GUI直接集成到操作系统内的
文件扩展名	Linux不使用文件扩展名来识别文件的类型，而是根据文件头的内容来识别其类型	使用文件扩展名来识别文件的类型
文件执行	Linux通过文件访问权限来判断是否为可执行文件。程序和脚本（其实是文本文件）的创建者或管理员可以将需要执行的文件赋予可执行权限，这样做有利于安全。保存到系统上的可执行文件不能自动执行，因此可以防止许多脚本病毒	对于Windows来说用户双击.exe为扩展名的文件系统都尝试加载执行
系统重启问题	Linux的设计思想之一是，遵循“牛顿运动定律”，一旦开始运行，它将保持运行状态，直到受到外界因素的干扰，比如硬件故障。除了内核之外，其他软件的安装、卸载都不需要重新引导系统	Windows在安装软件，特比是安装驱动程序后，经常需要重启系统
远程管理	可以远程地完成Linux中的很多工作。只要系统的基本网路服务在运行，就可以远程登录并管理系统。如果系统中一个特定的服务出现了问题，可以在进行故障诊断的同时让其他服务继续运行；当在一个系统上同时运行多个服务的时候（例如同时运行FTP、DNS、WWW服务），这种管理方式非常重要	Windows的远程管理功能较弱

1. Linux简介 —— Ubuntu

- Ubuntu 每六个月发布一个新版本，用户可以免费升级到最新版本 。
- Ubuntu版本的命名遵从“Y.MM（开发代号）”格式，Y代表年份，MM代表月份 。
- 由于受到来自官方的和非官方的社区支持，Ubuntu还有不少衍生版本 。

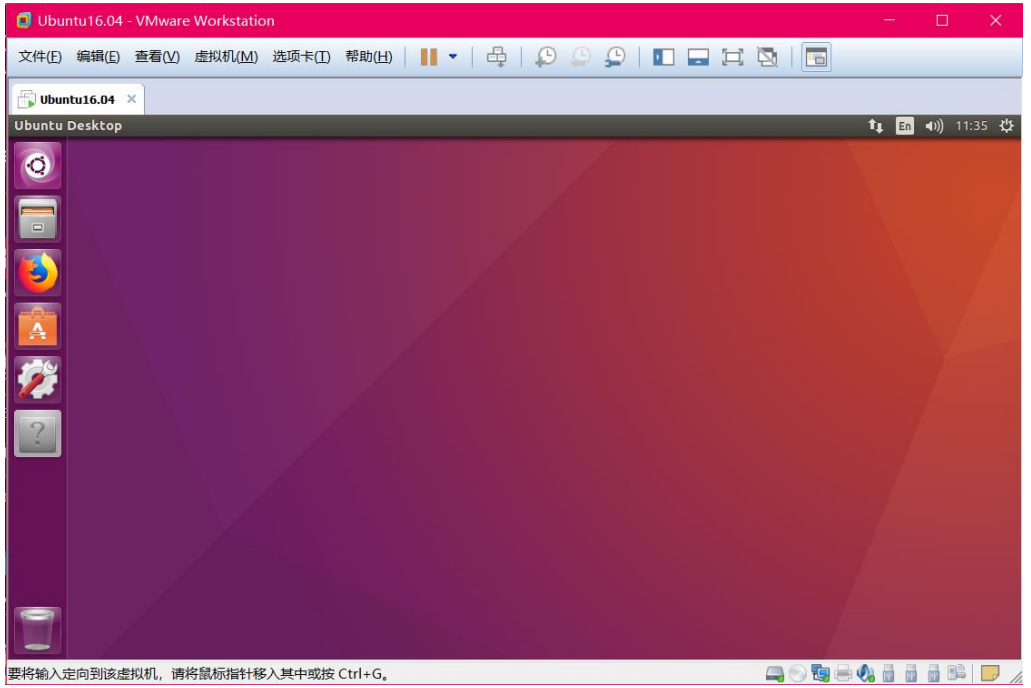






1. Linux简介 —— Ubuntu

- Ubuntu支持三种安装方式：
  - 光盘安装
  - Wubi安装
  - 虚拟机安装



目录

➢2. 命令行使用基础

2. 命令行使用基础 —— 为什么要学习命令行

- 命令对于熟练使用Linux/Unix系统而言是必不可少的 ；
- 命令行应用的可扩展性、灵活性更好；
- 打破了使用Windows时一个鼠标“一点到底”的简单与乏味，它提供给用户更大的灵活性与想象空间；
- 命令已成为Linux/Unix的典型标志，也已成为Linux/Unix的魅力所在。

2. 命令行使用基础 —— Shell，通向Linux圣殿的桥梁

- Shell既是一种命令语言，又是一种程序设计语言
  - 命令语言：它可以互动式地解释和执行用户输入的命令；
  - 程序设计语言：它定义了各种变量和参数，并提供了许多在高级语言中才具有的控制结构，包括循环和分支。
- Shell不是Linux系统内核的一部分
  - 但可以调用系统内核的大部分功能来执行程序、创建文档并以并行的方式协调各个程序的运行。
- Shell的种类繁多
  - Ubuntu操作系统默认使用的BASH Shell

2. 命令行使用基础 —— Shell，通向Linux圣殿的桥梁

```
probot@probot-vpc: ~
probot@probot-vpc:~$  help
GNU bash, version 4.3.48(1)-release (x86_64-pc-linux-gnu)
These shell commands are defined internally.  Type `help' to see this list.
Type `help name' to find out more about the function `name'.
Use `info bash' to find out more about the shell in general.
Use `man -k' or `info' to find out more about commands not in this list.

A star (*) next to a name means that the command is disabled.

 job_spec [&]                            history [-c] [-d offset] [n] or hist>
 (( expression ))                        if COMMANDS; then COMMANDS; [ elif C>
 . filename [arguments]                  jobs [-lnprs] [jobspec ...] or jobs >
 :                                       kill [-s sigspec | -n signum | -sigs>
 [ arg... ]                              let arg [arg ...]
 [[ expression ]]                        local [option] name[=value] ...
 alias [-p] [name[=value] ... ]          logout [n]
 bg [job_spec ...]                       mapfile [-n count] [-O origin] [-s c>
 bind [-lpsvPSVX] [-m keymap] [-f file>  popd [-n] [+N | -N]
 break [n]                               printf [-v var] format [arguments]
 builtin [shell-builtin [arg ...]]       pushd [-n] [+N | -N | dir]
 caller [expr]                           pwd [-LP]
 case WORD in [PATTERN [| PATTERN]...)>  read [-ers] [-a array] [-d delim] [->
 cd [-L|[-P [-e]] [-@]] [dir]            readarray [-n count] [-O origin] [-s>
 command [-pVv] command [arg ...]        readonly [-aAf] [name[=value] ...] o>
 compgen [-abcdefgjksuv] [-o option]  >  return [n]
 complete [-abcdefgjksuv] [-pr] [-DE] >  select NAME [in WORDS ... ;] do COMM>
 compopt [-o|+o option] [-DE] [name ..>  set [-abefhkmnptuvxBCHP] [-o option->
 continue [n]                            shift [n]
 coproc [NAME] command [redirections]    shopt [-pqsu] [-o] [optname ...]
 declare [-aAfFgilnrtux] [-p] [name[=v>  source filename [arguments]
 dirs [-clpv] [+N] [-N]                  suspend [-f]
 disown [-h] [-ar] [jobspec ...]         test [expr]
 echo [-neE] [arg ...]                   time [-p] pipeline
 enable [-a] [-dnps] [-f filename] [na>  times
 eval [arg ...]                          trap [-lp] [[arg] signal_spec ...]
 exec [-cl] [-a name] [command [argume>  true
 exit [n]                                type [-afptP] name [name ...]
 export [-fn] [name[=value] ...] or ex>  typeset [-aAfFgilrtux] [-p] name[=va>
 false                                   ulimit [-SHabcdefilmnpqrstuvxT] [lim>
 fc [-e ename] [-lnr] [first] [last] o>  umask [-p] [-S] [mode]
 fg [job_spec]                           unalias [-a] name [name ...]
 for NAME [in WORDS ... ] ; do COMMAND>  unset [-f] [-v] [-n] [name ...]
```

2. 命令行使用基础 —— 常用命令

cd 命令

- 语法：cd <目录路径>
- 功能：改变工作目录。若没有指定“目录路径”，则回到用户的主目录。

pwd 命令

- 语法：pwd
- 功能：此命令显示出当前工作目录的绝对路径。

```
probot@probot-vpc:~$ pwd
/home/probot
probot@probot-vpc:~$ cd ..
probot@probot-vpc:/home$ pwd
/home
probot@probot-vpc:/home$ cd probot/
probot@probot-vpc:~$ pwd
/home/probot
probot@probot-vpc:~$ cd /home/probot/
probot@probot-vpc:~$ pwd
/home/probot
```

2. 命令行使用基础 —— 常用命令

➢ mkdir命令

- 语法：mkdir [选项] <目录名称>
- 功能：创建一个目录。

➢ ls 命令

- 语法：ls [选项] [目录名称...]
- 功能：列出目录的内容。

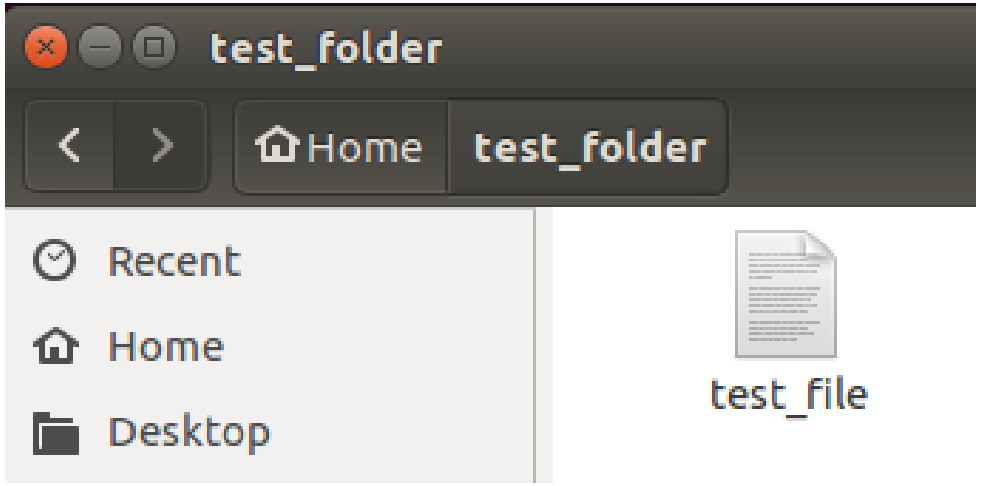
```
probot@probot-vpc:~$ mkdir test_folder
probot@probot-vpc:~$ ls
catkin_ws  Desktop  Downloads  test_folder
```

2. 命令行使用基础 —— 常用命令

touch命令

- 语法：touch [选项] [文件名称...]
- 功能：改变文件或目录时间。

```
probot@probot-vpc:~$ cd  test_folder/
probot@probot-vpc:~/test_folder$ ls
probot@probot-vpc:~/test_folder$ touch test_file
probot@probot-vpc:~/test_folder$ ls
test_file
probot@probot-vpc:~/test_folder$
```



2. 命令行使用基础 —— 常用命令

mv命令

- 语法：mv [选项] <源文件或目录> <目地文件或目录>
- 功能：为文件或目录改名或将文件由一个目录移入另一个目录中。

cp命令

- 语法：cp [选项] <源文件名称或目录名称> <目的文件名称或目录名称>
- 功能：把给出的一个文件或目录拷贝到另一文件或目录中，或者把多个源文件复制到目标目录中。

```
probot@probot-vpc:~/test_folder$ mv test_file ../test_mv_file
probot@probot-vpc:~/test_folder$ cd ..
probot@probot-vpc:~$ ls
catkin_ws  Desktop  Downloads  test_folder  test_mv_file
probot@probot-vpc:~$ cp test_mv_file test_folder/test_cp_file
```

2. 命令行使用基础 —— 常用命令

rm命令

- 语法：rm [选项] <文件名称或目录名称...>
- 功能：该命令的功能为删除一个目录中的一个或多个文件或目录，它也可以将某个目录及其下的所有文件及子目录均删除。对于链接文件，只是删除了链接，原有文件均保持不变。

```
probot@probot-vpc:~$ cd test_folder/
probot@probot-vpc:~/test_folder$ rm test_cp_file
probot@probot-vpc:~/test_folder$ cd ..
probot@probot-vpc:~$ rm test_folder/
rm: cannot remove 'test_folder/': Is a directory
probot@probot-vpc:~$ rm -r test_folder/
```

2. 命令行使用基础 —— 常用命令

sudo命令

- 语法：sudo [选项>][指令]
- 功能：以其他身份来执行指令。

```
probot@probot-vpc:~$ sudo apt-get update
Hit:1 http://mirrors.aliyun.com/ubuntu xenial InRelease
Hit:2 http://mirrors.aliyun.com/ubuntu xenial-updates InRelease
Hit:3 http://mirrors.aliyun.com/ubuntu xenial-backports InRelease
Hit:4 http://mirrors.aliyun.com/ubuntu xenial-security InRelease
Get:5 http://packages.ros.org/ros/ubuntu xenial InRelease [4,040 B]
```

```
probot@probot-vpc:~$ sudo apt-get install tree
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  tree
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 40.6 kB of archives.
```

目录

3. C++/Python编程基础

VS

3. C++/Python编程基础

C++

```cpp
#include <iostream>

using namespace std;

int main()
{
    int a = 5;

    for(a; a<10; a++)
    {
        cout << "a = " << a << endl;
    }

    return 0;
}
```

Python

```python
for a in range(5, 10):
    if a < 10:
        print 'a = ', a
        a += 1
    else:
        break
```

```
probot@probot-vpc:~/test$ g++ c++_for.cpp -o c++_for
probot@probot-vpc:~/test$ ./c++_for
a = 5
a = 6
a = 7
a = 8
a = 9
probot@probot-vpc:~/test$ python python_for.py
a =  5
a =  6
a =  7
a =  8
a =  9
```

3. C++/Python编程基础

C++

```cpp
#include <iostream>
using namespace std;

int main()
{
    int a = 5;

    while(a < 10)
    {
        cout << "a = "  << a << endl;
        a++;
    }

    return 0;
}
```

Python

```python
a = 5

while a < 10:
    print 'a = ', a
    a += 1
```

```
probot@probot-vpc:~/test$ g++ c++_while.cpp -o c++_while
probot@probot-vpc:~/test$ ./c++_while
a = 5
a = 6
a = 7
a = 8
a = 9
probot@probot-vpc:~/test$ python python_while.py
a =  5
a =  6
a =  7
a =  8
a =  9
```

3. C++/Python编程基础

C++

```cpp
#include <iostream>

class A
{
public:
    int i;
    void test()
    {
        std::cout << i << std::endl;
    }
};

int main()
{
    A a;

    a.i = 10;
    a.test();

    return 0;
}
```

Python

```python
class A:
    i = 10
    def test(self):
        print self.i

a = A()
a.test()
```

```
probot@probot-vpc:~/test$ g++ c++_class.cpp -o c++_class
probot@probot-vpc:~/test$ ./c++_class
10
probot@probot-vpc:~/test$ python python_class.py
10
```

Thank you!