

# SOME TIPS ABOUT THE LINUX SHELL

Speaker: Carlos Beltrán-González

October 15, 2007

# INTRODUCTION

As you have noticed this year we will use Linux Ubuntu for the Operating Systems exercise. The main goals are:

- 1 Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- 2 Provide the essentials for the use of a Linux box
- 3 Practice how to program in a Linux box in C++
- 4 Practice some C++ programing techniques (classes, multithread, communications)
- 5 Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library

# INTRODUCTION

As you have noticed this year we will use Linux Ubuntu for the Operating Systems exercise. The main goals are:

- 1 Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- 2 Provide the essentials for the use of a Linux box
- 3 Practice how to program in a Linux box in C++
- 4 Practice some C++ programing techniques (classes, multithread, communications)
- 5 Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library

# INTRODUCTION

As you have noticed this year we will use Linux Ubuntu for the Operating Systems exercise. The main goals are:

- 1 Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- 2 Provide the essentials for the use of a Linux box
- 3 Practice how to program in a Linux box in C++
- 4 Practice some C++ programing techniques (classes, multithread, communications)
- 5 Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library

# INTRODUCTION

As you have noticed this year we will use Linux Ubuntu for the Operating Systems exercise. The main goals are:

- 1 Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- 2 Provide the essentials for the use of a Linux box
- 3 Practice how to program in a Linux box in C++
- 4 Practice some C++ programing techniques (classes, multithread, communications)
- 5 Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library

# INTRODUCTION

As you have noticed this year we will use Linux Ubuntu for the Operating Systems exercise. The main goals are:

- 1 Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- 2 Provide the essentials for the use of a Linux box
- 3 Practice how to program in a Linux box in C++
- 4 Practice some C++ programing techniques (classes, multithread, communications)
- 5 Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library

# MAIN GOALS

## GNU LINUX ENVIRONMENT

- Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- Provide the essentials for the use of a Linux box
- Practice how to program in a Linux box in C++

## ESERCITAZIONI

- Practice some C++ programming techniques (classes, multithread, communications)
- Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library

# MAIN GOALS

## GNU LINUX ENVIRONMENT

- Provide a very accessible way to use the computer (LiveCD). This allow you to follow the exercises everywhere you may have access to a computer.
- Provide the essentials for the use of a Linux box
- Practice how to program in a Linux box in C++

## ESERCITAZIONI

- Practice some C++ programing techniques (classes, multithread, communications)
- Learn how to use some basic functions of the YARP(Yet Another Robotic Platform) library



## A Typical Linux File System

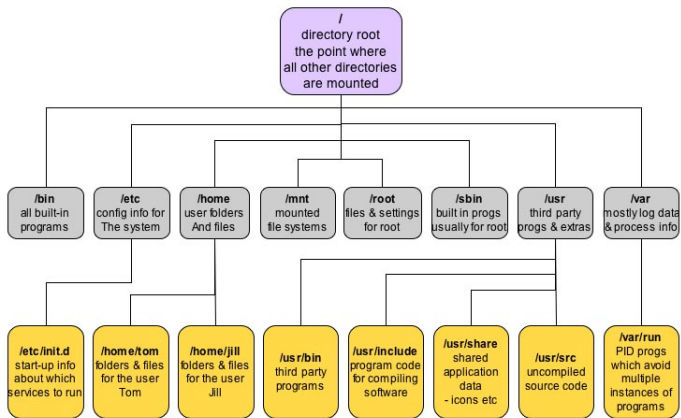


FIGURE: The Linux's filesystem structure

### COMMAND: *man*

- Gives you a "manual page" about a particular command
- Use: `man ls`
- Give you the man page of the `ls` command

### ADVANCE USE

- *-al Creates a list of "hidden" files*
- *Use: `ls -al`*

### COMMAND: *man*

- Gives you a "manual page" about a particular command
- Use: `man ls`
- Give you the man page of the `ls` command

### ADVANCE USE

- *-al* Creates a list of "hidden" files
- Use: `ls -al`

### COMMAND: *ls*

- Creates a list of the available files in the directory
- Use: *ls*

### ADVANCE USE

- *-a/* Creates a list of "hidden" files
- Use: *ls -a/*

### COMMAND: *ls*

- Creates a list of the available files in the directory
- Use: *ls*

### ADVANCE USE

- *-a/* Creates a list of "hidden" files
- Use: *ls -a/*

### COMMAND: *mv*

- Moves a file
- Use: *mv filea.cpp fileb.cpp*

### ADVANCE USE

- It works also with directories
- Use: *mv directorysource directorytarget*

### COMMAND: *mv*

- Moves a file
- Use: *mv filea.cpp fileb.cpp*

### ADVANCE USE

- It works also with directories
- Use: *mv directorysource directorytarget*

### COMMAND: *cp*

- Copies(*cp*) a file
- Use: *cp filea.cpp fileb.cpp*

### ADVANCE USE

- It works also with directories
- Use: *cp -R directorysource directorytarget*
- *-R* means "recursive"



### COMMAND: *cp*

- Copies(*cp*) a file
- Use: *cp filea.cpp fileb.cpp*

### ADVANCE USE

- It works also with directories
- Use: *cp -R directorysource directorytarget*
- *-R* means "recursive"

### COMMAND: *rm*

- Removes files or directories
- Use: *rm file.cpp*

### ADVANCE USE

- It works also with directories
- Use: *rm -rf directory*
- *-rf* makes the removal recursive and silent

### COMMAND: *rm*

- Removes files or directories
- Use: *rm file.cpp*

### ADVANCE USE

- It works also with directories
- Use: *rm -rf directory*
- *-rf* makes the removal recursive and silent

## MORE COMPLEX USE OF THE SHELL

### STANDARD INPUT, STANDARD OUTPUT, STANDARD ERROR

- Everything is a file
- Devices can have a input stream, output stream...etc
- These data stream can be seen as "pipes" where data flows

### REDIRECTIONS AND PIPES

- test
- standard output
- standard output
- A pipe is a flow connections represented by the symbol

### EXAMPLES

- `ps aux | grep gnome`

## MORE COMPLEX USE OF THE SHELL

### STANDARD INPUT, STANDARD OUTPUT, STANDARD ERROR

- Everything is a file
- Devices can have a input stream, output stream...etc
- These data stream can be seen as "pipes" where data flows

### REDIRECTIONS AND PIPES

- test
- standard output
- standard output
- A pipe is a flow connections represented by the symbol

### EXAMPLES

- *ps aux — grep gnome*

## MORE COMPLEX USE OF THE SHELL

### STANDARD INPUT, STANDARD OUTPUT, STANDARD ERROR

- Everything is a file
- Devices can have a input stream, output stream...etc
- These data stream can be seen as "pipes" where data flows

### REDIRECTIONS AND PIPES

- test
- standard output
- standard output
- A pipe is a flow connections represented by the symbol

### EXAMPLES

- *ps aux — grep gnome*