Introduction to Linux and C++

49274 Advanced Robotics

Overview

- We will use Linux for the assignments
- The FEIT computer labs use Red Hat Linux, but we need to use Ubuntu Linux
- We will use an Ubuntu container on the Red Hat systems
- If you want to install Ubuntu Linux on your own computers use version 18.04

Using the FEIT computers

- First make sure you are in Linux: *Ctrl+Alt* then *Ctrl+Alt+F1*
- Enter your student number in the *Username* field
- Enter your password
- Open a terminal:

  Applications -> System Tools -> Terminal

- Start a shell in the Singularity container:

```
singularity shell /images/singularity_containers/ros-melodic-ar.sif
```

- The Singularity image is on all the Linux computers on level B1

Linux

Using Linux

- The main applications you will need are:
  - Terminal (found in “System Tools”)
  - Text editor (found in “Favorites”)
- You can open a file manager from the “Places” menu

Useful terminal commands

Command	Description	Example
`cd <directory>`	Change directory	`cd ~/catkin_ws`
`ls`	List files	`ls`
`mkdir <directory>`	Make directory	`mkdir new_directory`
`rm <file>`	Remove file	`rm new_file`
`rm -rf <directory>`	Remove directory	`rm new_directory`
`man <command>`	Manual for a command	`man ls`

Linux paths

- Paths can be relative or absolute
- Absolute paths:
  - `~/` is your home directory
  - `/` is the system root directory
- Relative path:
  - `./` is the current directory
  - `../` is the directory above your current location
  - Anything not beginning with `/` or `~/`

Other things

- You can press the up and down arrows on your keyboard to scroll through previous commands
- *Tab* will autocomplete commands, paths and some command parameters.
- *Ctrl*+*R* will search through your command history
- *Ctrl*+*C* will kill a running program

Additional resources

- Lynda: Learning Linux Command Line
  - 1. Command-Line Basics
  - 2. Files, Folders, and Permissions

C++

Tutorials

- Lynda: Learning C++ (Beginner)
- Lynda: C++ Essential Training (Intermediate)
- C++ Language - C++ Tutorials

If statement

```
double a = 0.3;

if (a >= 0.5)
{
    doSomething();
}
else
{
    doSomethingElse();
}
```

For loop

```
for (int i = 0; i < 10; ++i)
{
    printf("%d\n", i);
}
```

C++ References

- C++ references are a way of avoiding copying
- They are defined with an “&” after the type
- They are commonly used to pass data into functions:

```
void initialiseState(std::vector<double>& state)
```

- References can also be “const”, the data can’t be changed:

```
void printState(const std::vector<double>& state)
```

STL Containers

- std::vector dynamic array of values of the same type
- The range-based for loop can be convenient for vectors:

```
std::vector<double> state(10);
for (auto& s : state)
{
    s = 1.;
}
```

- “auto” selects the correct type for you
- Make sure you use a reference if you want to modify data

Accessing a vector

```
std::vector<double> v;

// Reserve storage for 10 elements
v.reserve(10);

// Add some elements
v.push_back(0.4);
v.push_back(0.6);
v.push_back(0.2);

// Access the second element
printf("%.2f\n", v[1]);
```

Accessing a vector

```
std::vector<double> v;

// Resize vector to 10 elements
v.resize(10);

// Add some elements
v[0] = 0.4;
v[1] = 0.6;
v[2] = 0.2;

// Access the second element
printf("%.2f\n", v[1]);
```

Writing to the Console

- You can print static text with “printf” from cstdio:

```
printf("Text string");
```

- “printf” can also print variables:

```
printf("int a: %i, double b: %.2f\n", a, b);
```

- Another way:

```
std::cout << “Text string” << std::endl;
```