



INFO 802
Master Advanced Mechatronics
Luc Marechal

2025

 ROS
Environment Setup
Tutorial 1

Objectives

At the end of this tutorial, you are excepted to :

- Understand the very basics of Linux system commands
- Be able to create a ROS catkin workspace
- Be able to create a ROS package
- Know what is and how to source a file

Introduction to Command Line

The command line, or **shell**, provides an interface between the user and the internals of a computer.

Among shells, **bash** is most widely used

Shell	Name	Description
sh	Bourne shell	Popular, ubiquitous shell developed in 1977, still guaranteed on all Unixes
csh	C shell	Improves on sh
ksh	Korn shell	Backward-compatible with sh, but extends and borrows from other shells
→ bash	Bourne again shell	Free software replacement for sh, much evolved
tcsh	Tenex C shell	Updated and extended C shell

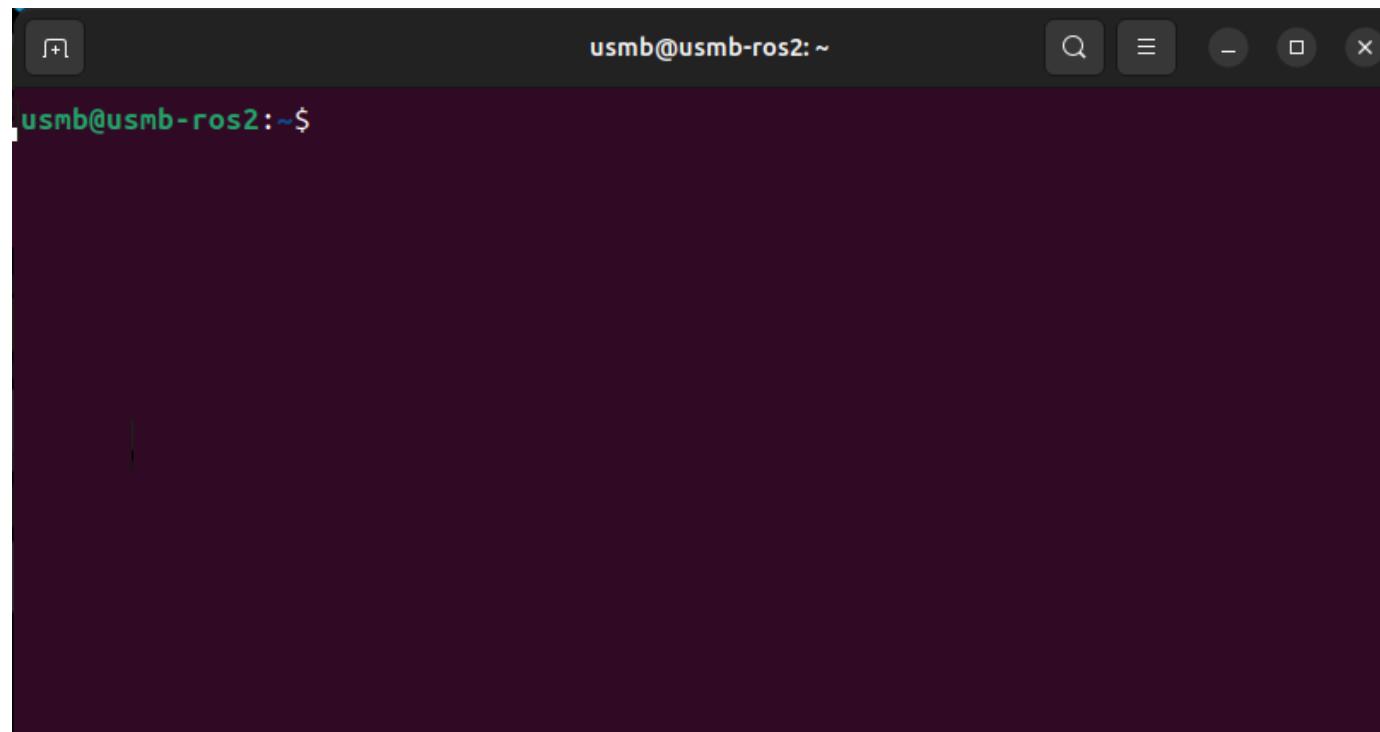
More info

Effective Computation in Physics
by Anthony Scopatz

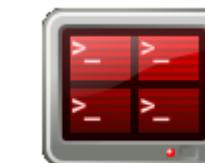
Navigating the Shell

You can access the shell by opening a **terminal**

It is used to run commands and navigate through the file system. This command line is also sometimes called the **prompt**.



Terminal



Terminator

More info

Effective Computation in Physics
by Anthony Scopatz

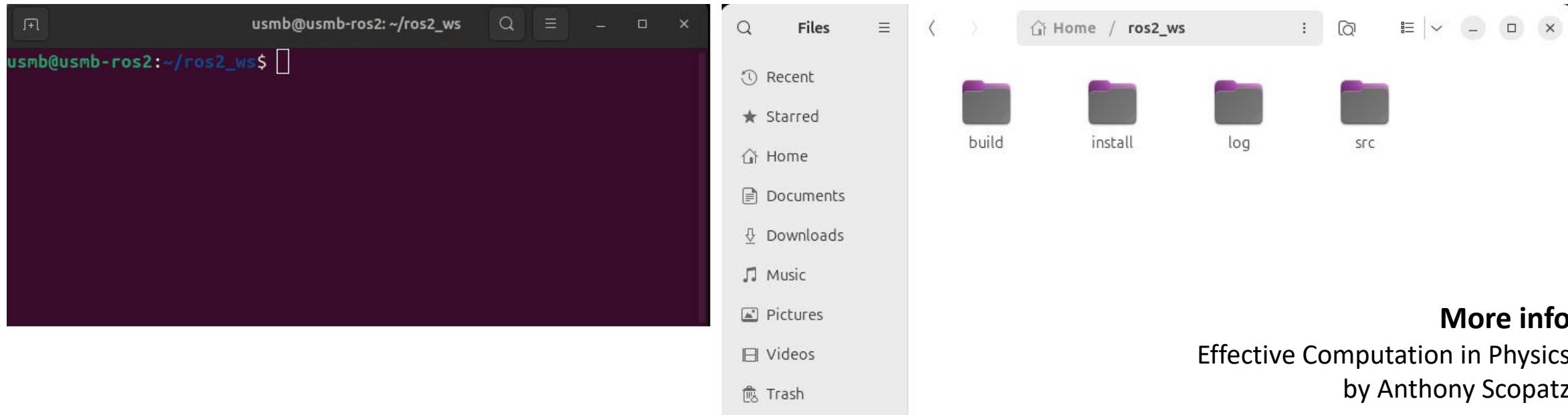
Home Directory

The shell starts your session from a special directory called your **home directory**.

The tilde (~) character can be used as a shortcut to your home directory

In a Terminal:

~/ is a shortcut for /home/username



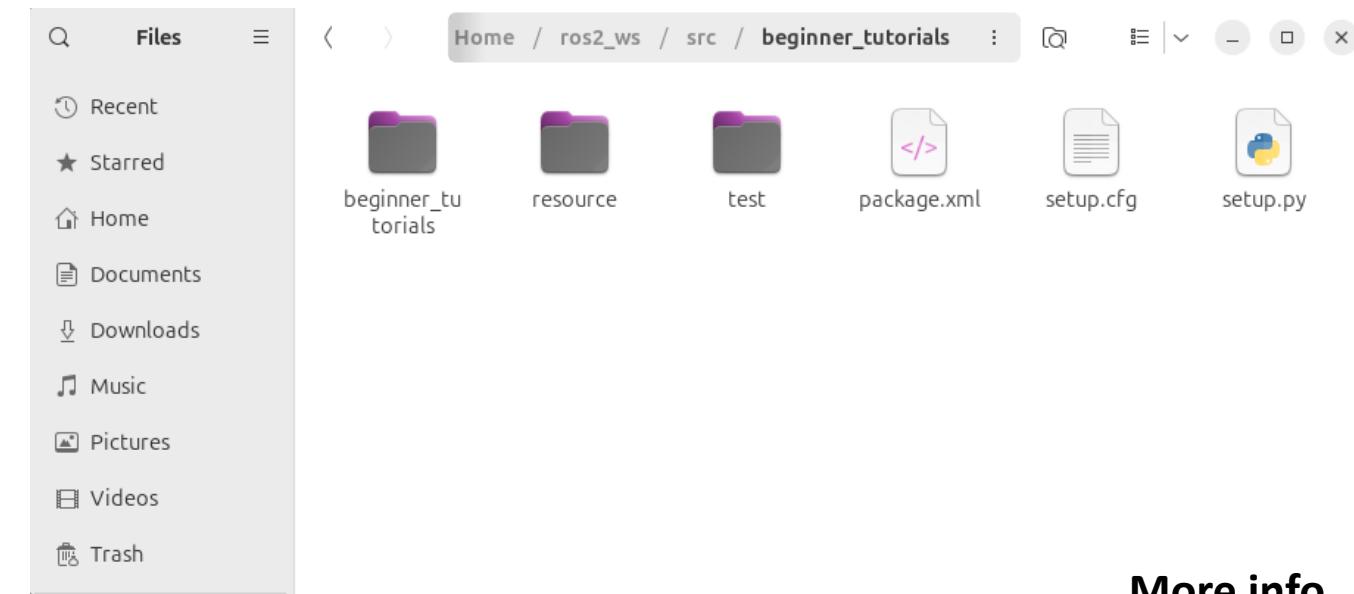
More info

Effective Computation in Physics
by Anthony Scopatz

Changing Directories (cd)

The `cd` command allows to change directories.

```
usmb@usmb-ros2:~/ros2_ws/src/beginner_tutorials$ cd src
usmb@usmb-ros2:~/ros2_ws/src$ cd beginner_tutorials/
usmb@usmb-ros2:~/ros2_ws/src/beginner_tutorials$
```



[More info](#)

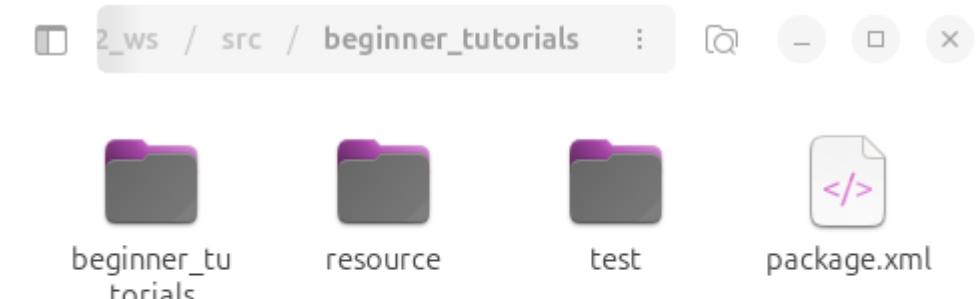
Effective Computation in Physics
by Anthony Scopatz

Autocompletion

Start writing the 2 first letters of the name of a file or folder and press



```
usmb@usmb-ros2:~/ros2_ws/src/beginner_tutorials$ cd beg
```



```
usmb@usmb-ros2:~/ros2_ws/src/beginner_tutorials$ cd beginner_tutorials/
```

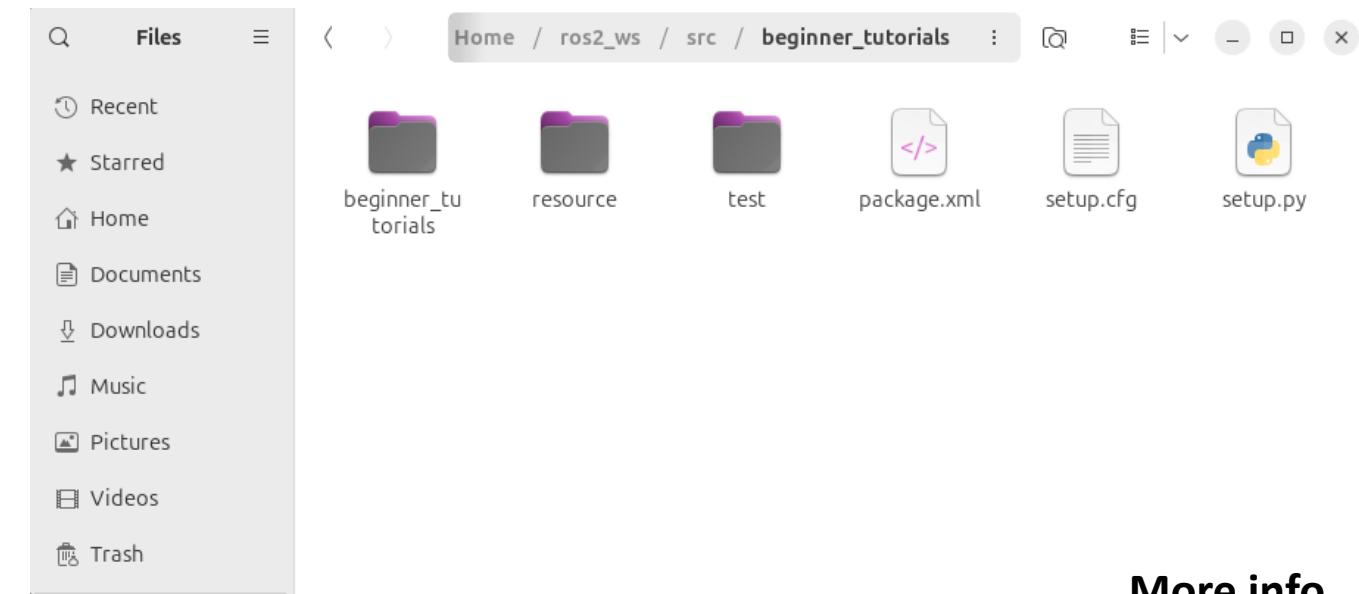
More info

Effective Computation in Physics
by Anthony Scopatz

Listing the Contents (ls)

The **ls** command allows the user to print out a list of all the files and subdirectories in a directory.

```
usmb@usmb-ros2:~/ros2_ws/src/beginner_tutorials$ ls
beginner_tutorials  package.xml  resource  setup.cfg  setup.py  test
```



More info

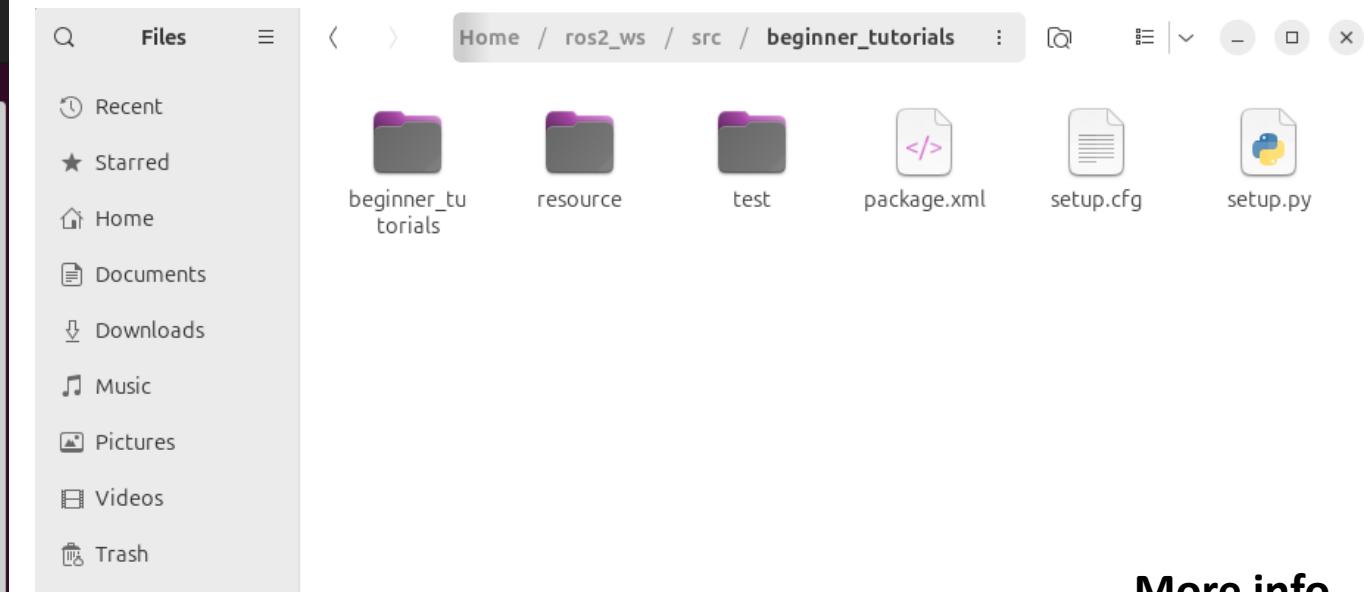
Effective Computation in Physics
by Anthony Scopatz

Listing the Contents (tree)

The `tree` command is used to display the directory structure

```
usmb@usmb-ros2:~/ros2_ws/src/beginner_tutorials$ tree
.
├── beginner_tutorials
│   ├── HelloWorldNode.py
│   └── __init__.py
├── package.xml
└── resource
    └── beginner_tutorials
        ├── setup.cfg
        └── setup.py
└── test
    ├── test_copyright.py
    ├── test_flake8.py
    └── test_pep257.py

4 directories, 10 files
```



More info

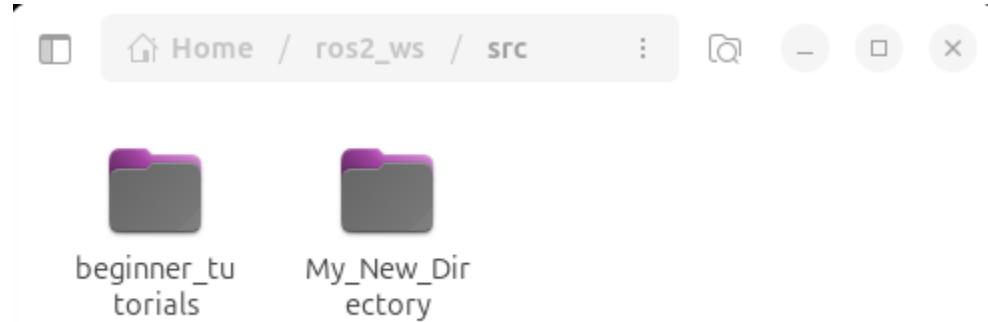
Effective Computation in Physics
by Anthony Scopatz

Making Directories (mkdir)

You can make new directories anywhere, not just in your current working directory with the `mkdir` (make directory) command.



```
usmb@usmb-ros2:~/ros2_ws$ mkdir ~/ros2_ws/src/My_New_Directory
usmb@usmb-ros2:~/ros2_ws$
```



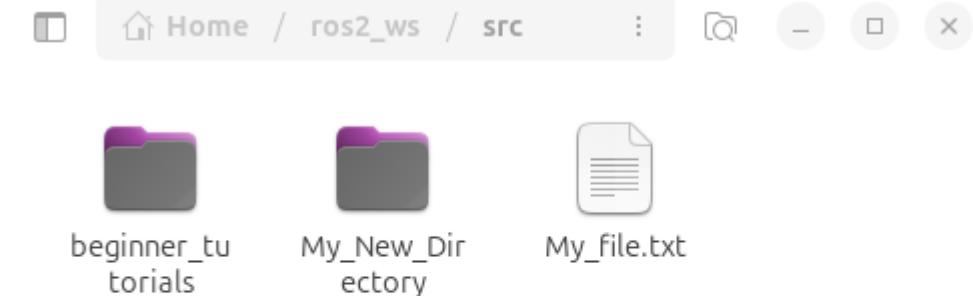
More info

Effective Computation in Physics
by Anthony Scopatz

Deleting Files and Directories (rm)

Files and directories can be deleted using the `rm` (remove) command

```
usmb@usmb-ros2:~/ros2_ws/src$ rm My_file.txt
usmb@usmb-ros2:~/ros2_ws/src$ rm -r My_New_Directory
usmb@usmb-ros2:~/ros2_ws/src$
```



More info

Effective Computation in Physics
by Anthony Scopatz

Command Line Shortcuts

Syntax	Meaning
/	The root, or top-level, directory of the filesystem (also used for separating the names of directories in paths)
~	The home directory
.	This directory
..	The parent directory of this directory
.../..	The parent directory of the parent directory of this directory

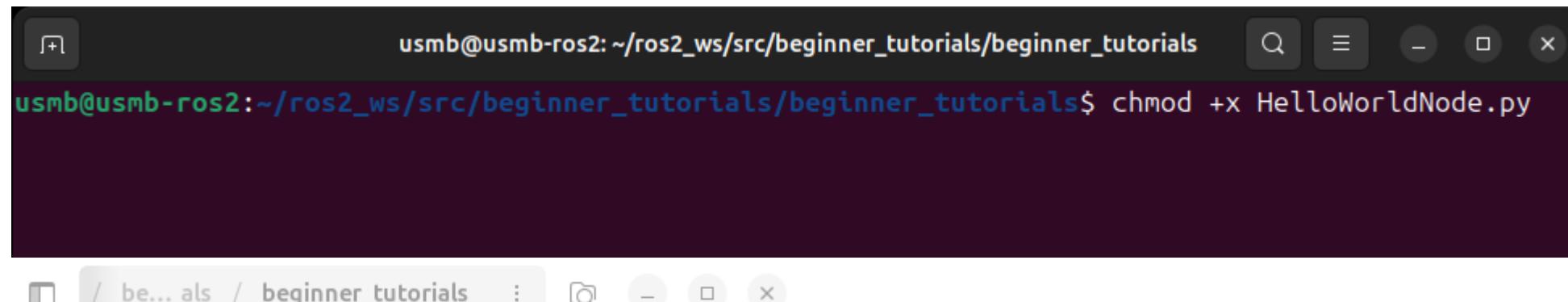
More infoEffective Computation in Physics
by Anthony Scopatz

Setting Permissions (chmod)

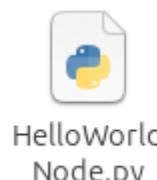
The **chmod** command configures access to files and directories.

The command allows 3 types of access to files (and directories):

- r : Read. The file can be opened, and its content viewed
- w : Write. The file can be edited, modified, and deleted
- x : Execute, If the file is a script or a program, it can be run (executed)



A screenshot of a terminal window titled "usmb@usmb-ros2: ~/ros2_ws/src/beginner_tutorials/beginner_tutorials". The command "chmod +x HelloWorldNode.py" is being typed into the terminal. The terminal has a dark background and light-colored text. The window title bar and some icons are visible at the top and bottom.



HelloWorld
Node.py



__init__.py



NodeHello
World.py

More info

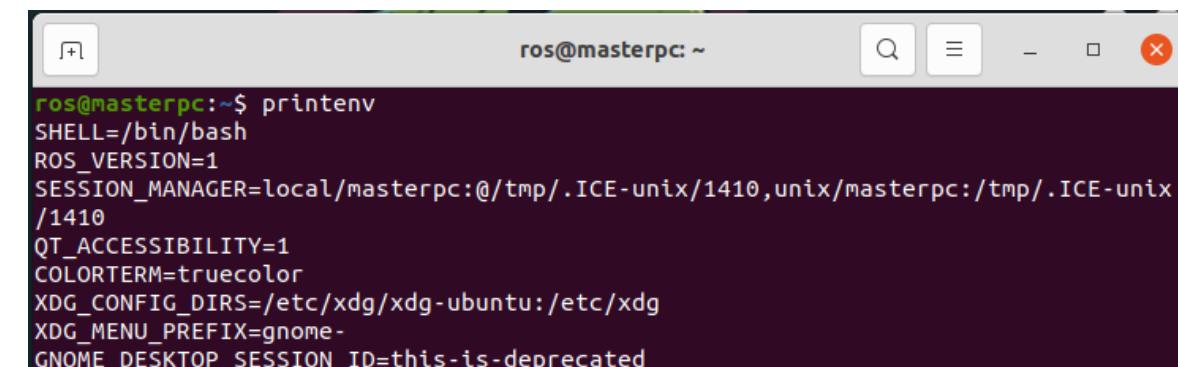
Effective Computation in Physics
by Anthony Scopatz

ROS Environment

Set up the environment

- Whenever you open a Terminal, a process takes place to gather and compile information that should be available to the shell process and its child processes.
- It obtains **Shell and Environmental Variables** for these settings from a variety of different files and settings on the system.
- We can see a list of all of our environmental variables by using the env or printenv commands.

```
$ printenv
```



```
ros@masterpc:~$ printenv
SHELL=/bin/bash
ROS_VERSION=1
SESSION_MANAGER=local/masterpc:@/tmp/.ICE-unix/1410,unix/masterpc:/tmp/.ICE-unix/1410
QT_ACCESSIBILITY=1
COLORTERM=truecolor
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
XDG_MENU_PREFIX=gnome-
GNOME_DESKTOP_SESSION_ID=this-is-deprecated
```

ROS Environment

Set up the environment

- In order to work properly, ROS uses the `setup.bash` and `setup.sh` files
- It is located in in the following directory: `/opt/ros/noetic/`
- The main function of these files is to set environment variables used by ROS and other apps.
- During the installation of ROS, you will see that you are prompted to `source` one of several `setup.*sh` files, or even add this 'sourcing' to your shell startup script
- If you are ever having problems finding or using your ROS packages make sure that you have your environment properly setup. Sourcing these `setup.*sh` files might help sometimes.

ROS Environment

Set up the environment:

- You will need to run `source /opt/ros/noetic/setup.bash` on every new shell you open to have access to the ROS commands, unless you add this line to your bash startup file (`~/.bashrc`)
- This will allow you to run `roscore` from any directory in your terminal window. To do so, we will modify the `.bashrc`.

Edit `.bashrc` file

```
> gedit ~/.bashrc
```

add the line at the bottom

```
> source ~/catkin_ws/devel/setup.bash
```



```
\s*//[^;]*\s*alert\$//'\\"')"\n\n# Alias definitions.\n# You may want to put all your additions into a separate file like\n# ~/.bash_aliases, instead of adding them here directly.\n# See /usr/share/doc/bash-doc/examples in the bash-doc package.\n\nif [ -f ~/.bash_aliases ]; then\n    . ~/.bash_aliases\nfi\n\n# enable programmable completion features (you don't need to enable\n# this, if it's already enabled in /etc/bash.bashrc and /etc/profile\n# sources /etc/bash.bashrc).\nif ! shopt -oq posix; then\n    if [ -f /usr/share/bash-completion/bash_completion ]; then\n        . /usr/share/bash-completion/bash_completion\n    elif [ -f /etc/bash_completion ]; then\n        . /etc/bash_completion\n    fi\n\n#source /opt/ros/kinetic/setup.bash\nsource ~/catkin_ws/devel/setup.bash\n\nsh ▾ Tab Width: 8 ▾ Ln 116, Col 5 ▾ INS
```

Bash (Unix shell)

- Bash is the Unix basic **shell** used in the terminal. (the \$ character is the default prompt.)
- The shell is an interface between the user and the operating system.
- It uses either a **command-line interface (CLI)** or graphical user interface (GUI) to control the computer.
- The CLI used in Ubuntu is **Terminal**



```
PS C:\> $PSVersionTable
Name          Value
----          -----
PSVersion     5.1.15063.786
PSEdition    Desktop
PSCompatibleVersions {1.0, 2.0, 3.0, 4.0...}
BuildVersion  10.0.15063.786
CLRVersion   4.0.30319.42000
WSManStackVersion 3.0
PSRemotingProtocolVersion 2.3
SerializationVersion 1.1.0.1
```

A screenshot of a Windows PowerShell window. The title bar says "Windows PowerShell". The command "PS C:\> \$PSVersionTable" is entered, and the output shows various PowerShell version details like PSVersion, PSEdition, and BuildVersion.

an example of CLI shell for Windows is PowerShell

```
mark@linux-desktop: /tmp/tutorial
File Edit View Search Terminal Help
Setting up tree (1.7.0-5) ...
Processing triggers for man-db (2.8.3-2) ...
mark@linux-desktop:/tmp/tutorial$ tree
.
├── another
├── combined.txt
├── dir1
└── dir2
    ├── dir3
    │   ├── test_1.txt
    │   ├── test_2.txt
    │   └── test_3.txt
    └── dir4
        └── dir5
            └── dir6
└── folder
    └── output.txt

8 directories, 5 files
mark@linux-desktop:/tmp/tutorial$
```

A screenshot of a Linux terminal window titled "mark@linux-desktop: /tmp/tutorial". The window shows the output of the "tree" command, which displays a directory structure with multiple levels of sub-directories and files. The terminal window has a standard Xfce-style title bar with icons for minimize, maximize, and close.

.bashrc

- `.bashrc` is a script file hidden in the `/home /<username>` directory.
- When you open a new terminal window by pressing **CTRL** + **ALT** + **T** or simply to open a new terminal tab, bash reads and executes commands from `~/.bashrc`, if that file exists.
- In particular, it reads the environment variables that are in the file.



Edit the file

```
> nano ~/.bashrc
```

Remarks

`nano` is an easy to use command line text editor for Unix OS

`~` is a shortcut for the `/home/<username>` directory

`.file` means the file is hidden

A screenshot of a terminal window titled "ros@masterpc: ~". The window shows the command "nano 4.8 /home/ros/.bashrc" and its output. The file contains several lines of shell code:

```
GNU nano 4.8 /home/ros/.bashrc
source /opt/ros/noetic/setup.bash
#source ~/catkin_ws/devel/setup.bash

export ROS_MASTER_URI=http://192.168.1.77:11311
export ROS_HOSTNAME=192.168.1.77
export TURTLEBOT3_MODEL=burger

export TURTLEBOT3BURGER1=192.168.0.229
export TURTLEBOT3BURGER2=192.168.0.230
export TURTLEBOT3WAFFLE=192.168.0.231
```

The terminal window includes standard Linux navigation keys at the bottom: Get Help, Write Out, Where Is, Cut Text, Justify, Exit, Read File, Replace, Paste Text, To Spell.

In this `~/.bashrc` file, the variable:

- `TURTLEBOT3_MODEL` has the value "burger"
- `TURTLEBOT3BURGER1` has the value "192.168.0.229"

ROS catkin workspace

Any ROS project begins with making a workspace.

In this workspace, you will put all the things related to this particular project.

Here we will create : - a workspace named : **catkin_ws**

- a package named : **beginner_tutorials_pkg**

in which we will create our nodes

Ubuntu

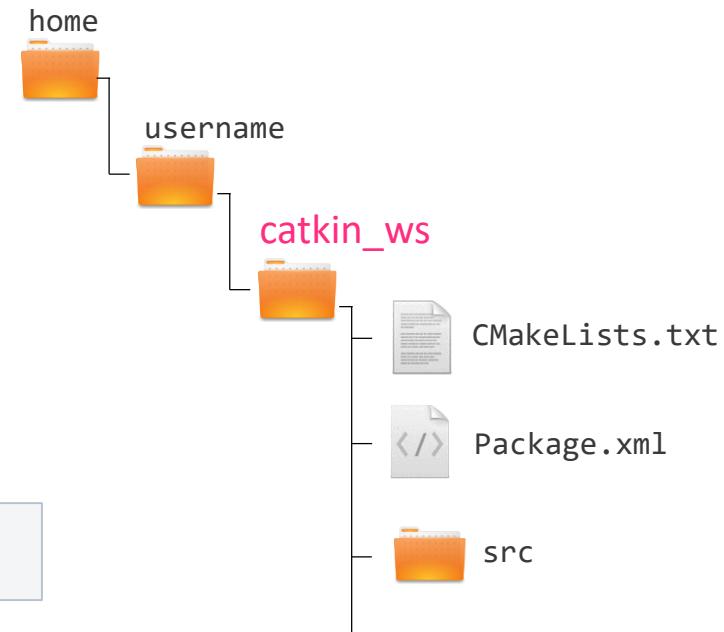
Your ROS workspace will be created in your **home** directory

In a Terminal:

`~/` is a shortcut for `/home/username`

The ROS workspace will then be created at this location :

`~/catkin_ws` = `/home/username/catkin_ws`



Exercice 1 – Create your catkin workspace

1 – Create a catkin ROS Workspace named : *catkin_ws*

Explain each command

```
> mkdir -p ~/catkin_ws/src  
  
> cd ~/catkin_ws/src  
  
> catkin_init_workspace  
  
> cd ~/catkin_ws  
  
> catkin_make
```

--> create a new folder named *catkin_ws* and inside a folder named *src*

--> navigate to folder *~/catkin_ws/src*

--> *catkin_init_workspace*

--> navigate to folder *~/catkin_ws*

--> build any packages located in *~/catkin_ws/src*.
always call *catkin_make* in the root of your catkin workspace

More info

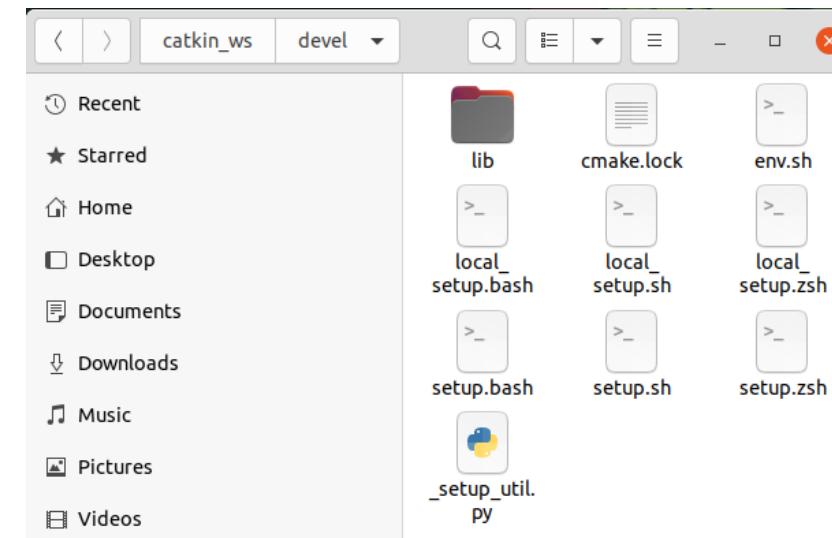
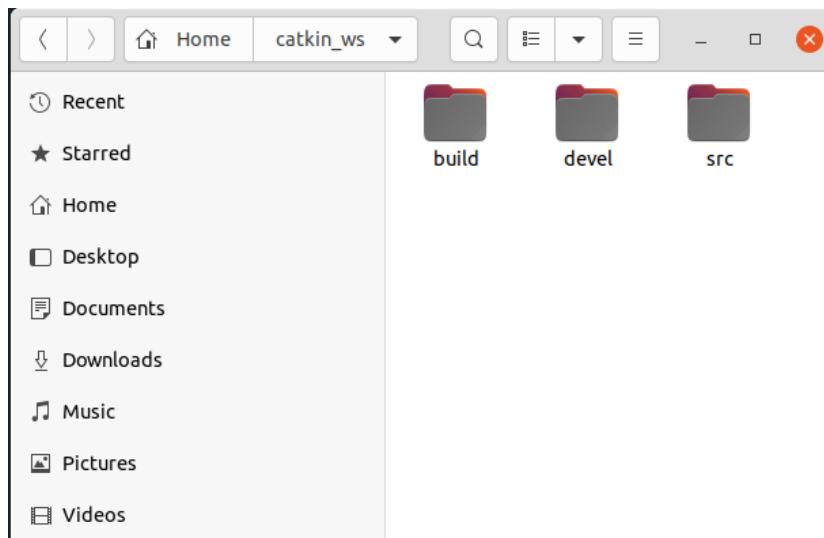
http://wiki.ros.org/catkin/Tutorials/create_a_workspace

Exercice 1 – Create your catkin workspace

If you look in your current directory you should now have a 'build' and 'devel' folder.

Inside the 'devel' folder you can see that there are now several setup.*sh files.

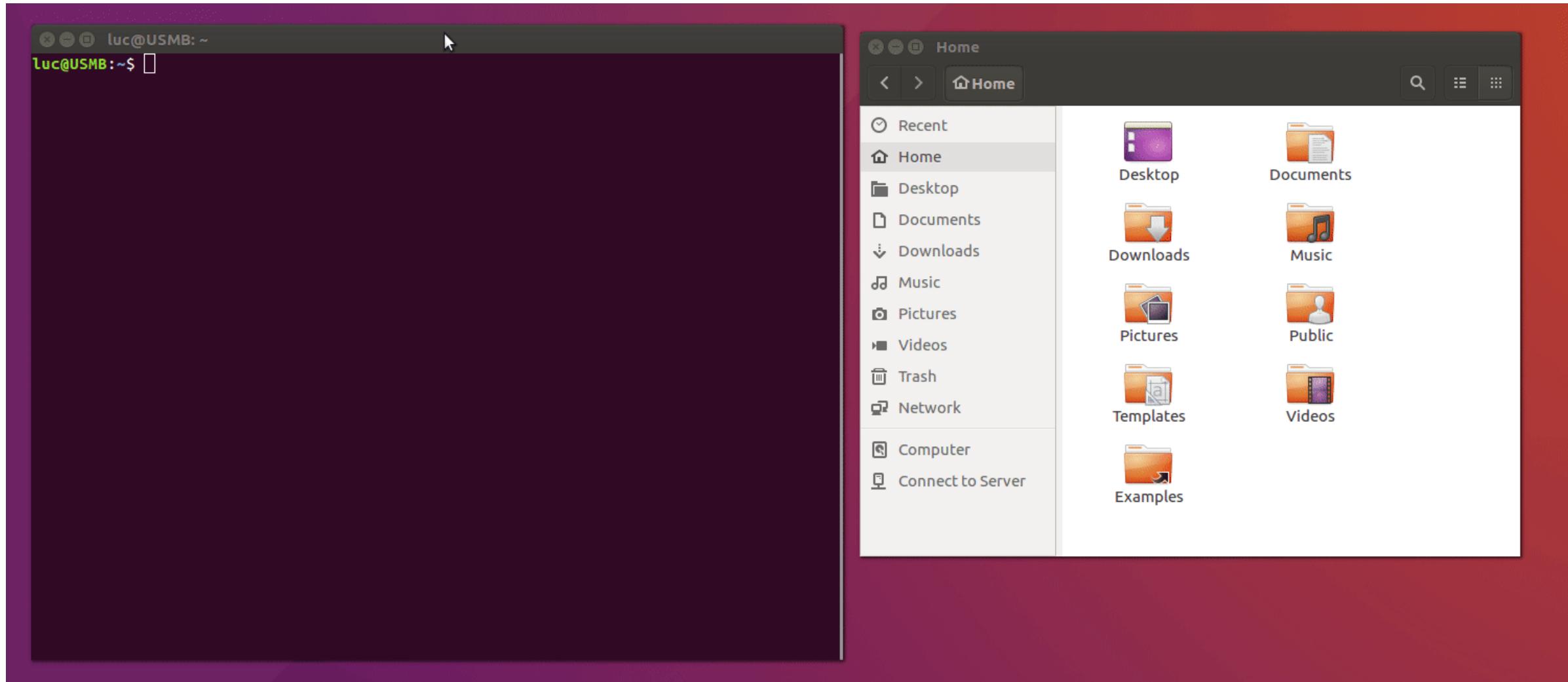
Sourcing any of these files will overlay this workspace on top of your environment.



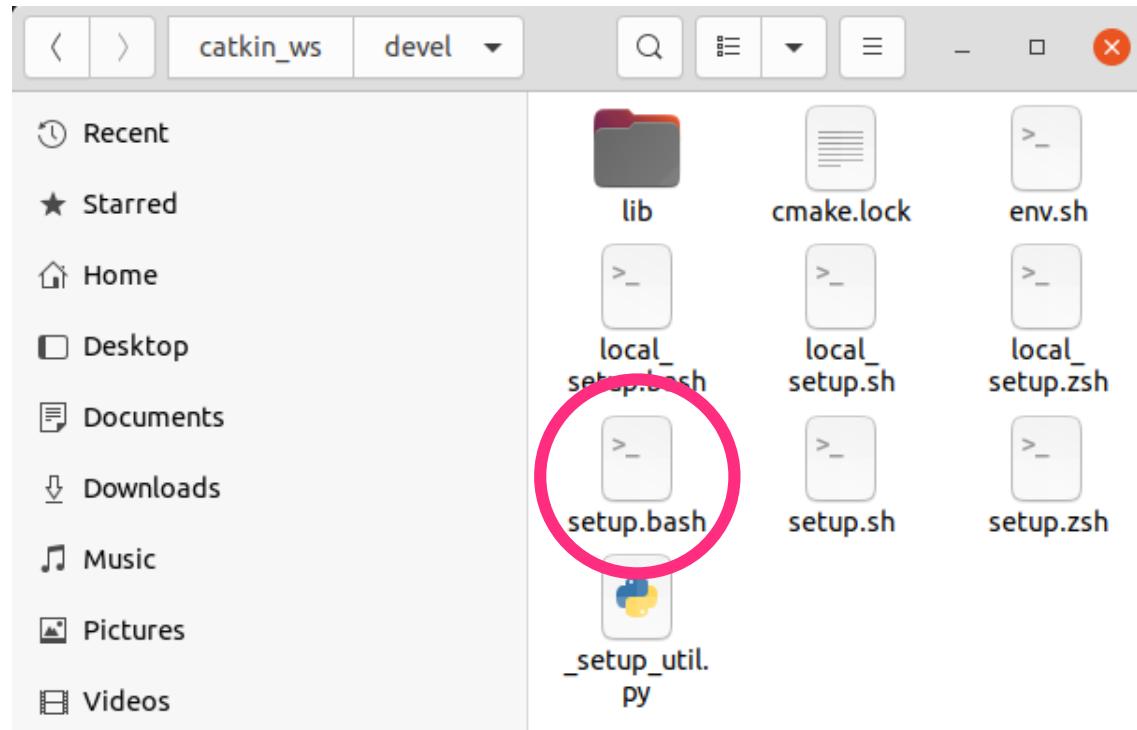
--> Automatically generated when you
create a catkin workspace

More info
http://wiki.ros.org/catkin/Tutorials/create_a_workspace

Exercice 1 – Create your catkin workspace: *Catkin_ws*



What does "sourcing a file" mean in Linux?



Before continuing, you must **source** the **setup.bash** file

This allows the terminal to understand where it needs to look to execute the ROS and catkin commands.

The **setup.bash** file is merely adding environment variables to your path to allow ROS to function

Linux Command : *Source*

- When a file is sourced, the lines of code in the file are executed as if they were printed at the command line.
- It updates functions and variables in the file for the current shell
- Any changes in /home/<username>/.bashrc file will only be taken into account after sourcing

Sourcing a file

```
> source file_name.sh
```

Or

```
> . file_name.sh
```

A screenshot of a terminal window titled "ros@masterpc: ~". The window contains the following text:

```
ros@masterpc:~$ nano ~/.bashrc
ros@masterpc:~$ source ~/.bashrc
ros@masterpc:~$
```

The terminal has a dark background with light-colored text. The title bar is dark with light text. The window has standard Linux-style window controls (minimize, maximize, close) in the top right corner.

sourcing the bashrc file

Source your environment

Every time you create a new workspace or a package or a node you must source your environment.

2 – source the setup.bash file

Explain what it does

```
> source ~/catkin_ws/devel/setup.bash
```

--> It adds the workspace to your ROS environment

3 – Check your workspace is properly overlayed by the setup script, make sure *ROS_PACKAGE_PATH* environment variable includes the directory you're in.

```
> echo $ROS_PACKAGE_PATH
```

Exercice 2 – Create a ROS Package

1 – Create a ROS package named : *beginner_tutorials_pkg*

```
> cd ~/catkin_ws/src
> catkin_create_pkg beginner_tutorials_pkg rospy std_msgs
```

Catkin function

Package name

Dependencies

(i.e other packages that we will use and
that already exist elsewhere)

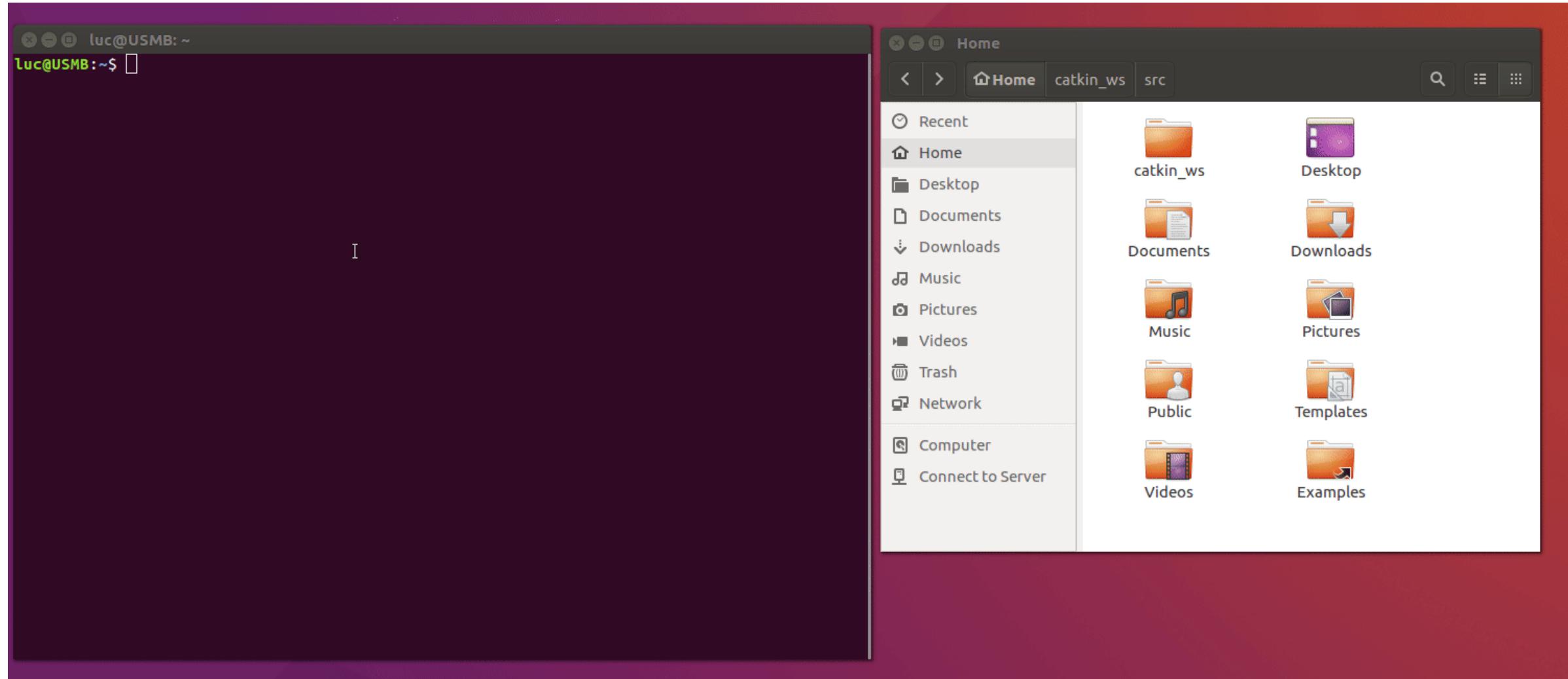
2 - Whenever you build a new package, **source** your environment

```
> source ~/catkin_ws/devel/setup.bash
```

More info

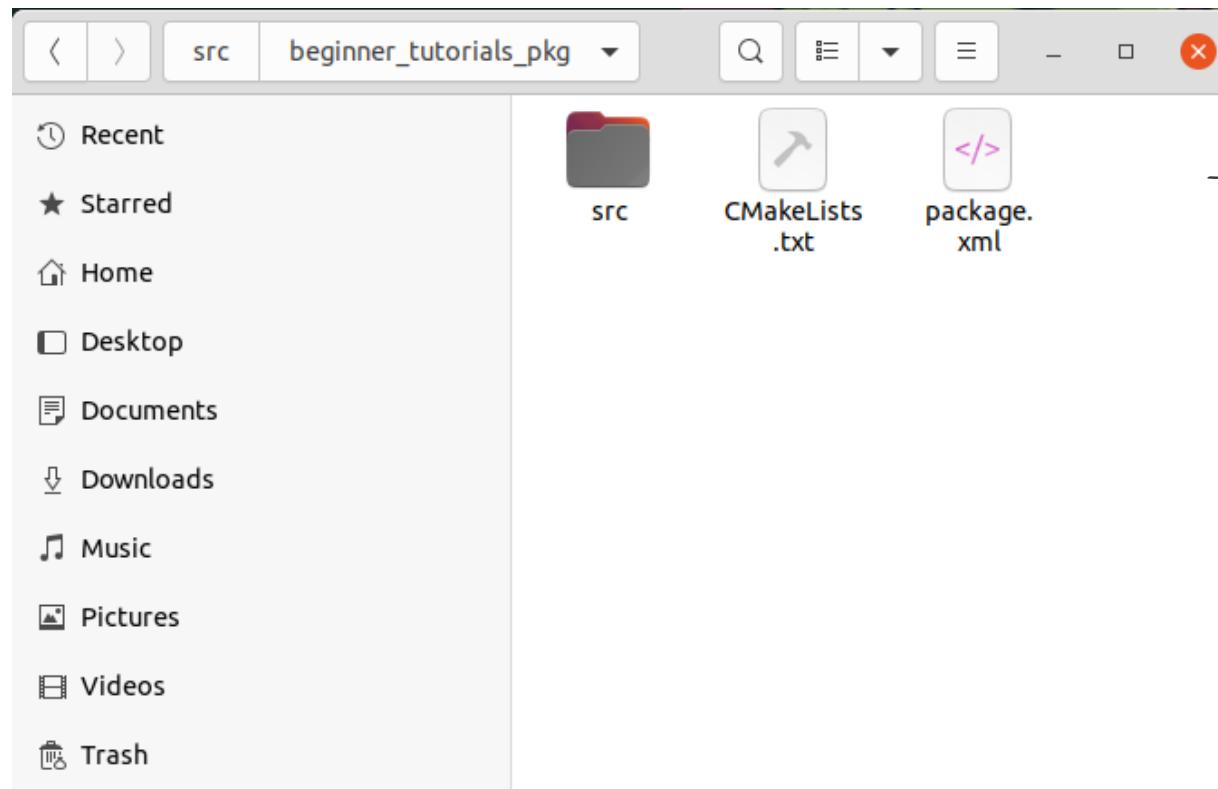
<http://wiki.ros.org/Packages>

Creating a ROS package : *beginner_tutorials*



ROS Package

package.xml and CMakeLists.txt files



-> Automatically generated when you create a package

ROS Package

package.xml

- The package.xml defines properties of the package

- <name> Package name
- <version> Version numbers
- <description> Description of the content
- <maintainer> Person maintaining the package
- <licence> Type of licence (usually BSD)

▪ Dependencies on other catkin packages

The dependencies are split into :

build_depend, buildtool_depend, exec_depend, test_depend

- and more

```
<?xml version="1.0"?>
<package format="2">
<name>beginner_tutorials_pkg</name>
<version>0.1.0</version>
<description>A ROS packages for beginner</description>
<maintainer email="luc.marechal@univ-smb.fr">Luc</maintainer>
<license>BSD</license>

<url type="website">https://github.com/my_project/ros_...</url>
<author email="luc.marechal@univ-smb.fr">Luc Mare</author>

<buildtool_depend>catkin</buildtool_depend>

<build_depend>rospy</build_depend>
<build_depend>std_msgs</build_depend>

<run_depend>rospy</run_depend>
<run_depend>std_msgs</run_depend>
</package>
```

This file must be included with the catkin package's root folder

If dependencies are missing or incorrect, you may be able to build from source and run tests on your own machine, but your package will not work correctly when released to the ROS community.

ROS Package

CMakeLists.txt

The CMakeLists.txt is the input to the CMakebuild system

1. Required CMake Version (cmake_minimum_required)
2. Package Name (project())
3. Find other CMake/Catkin packages needed for build (find_package())
4. Message/Service/Action Generators (add_message_files(), add_service_files(), add_action_files())
5. Invoke message/service/action generation (generate_messages())
6. Specify package build info export (catkin_package())
7. Libraries/Executables to build (add_library()/add_executable()/target_link_libraries())
8. Tests to build (catkin_add_gtest())
9. Install rules (install())

CMakeLists.txt

```
cmake_minimum_required(VERSION 2.8.3)
project(ros_package_template)

## Use C++11
add_definitions(--std=c++11)

## Find catkin macros and libraries
find_package(catkin REQUIRED
COMPONENTS
    roscpp
    sensor_msgs
)
...
```

More info

<http://wiki.ros.org/catkin/CMakeLists.txt>

ROS Package

CMakeLists.txt Example

```
cmake_minimum_required(VERSION 2.8.3)
project(husky_highlevel_controller)
add_definitions(--std=c++11)

find_package(catkin REQUIRED
    COMPONENTS roscpp sensor_msgs
)
```

```
catkin_package(
    INCLUDE_DIRS include
    # LIBRARIES
    CATKIN_DEPENDS roscpp sensor_msgs
    # DEPENDS
)
```

```
include_directories(include ${catkin_INCLUDE_DIRS})
```

```
add_executable(${PROJECT_NAME} src/${PROJECT_NAME}_node.cpp
src/HuskyHighlevelController.cpp)
```

```
target_link_libraries(${PROJECT_NAME} ${catkin_LIBRARIES})
```

Use the same name as in the package.xml

We use C++11 by default

List the packages that your package requires to build (have to be listed in package.xml)

Specify build export information

- INCLUDE_DIRS: Directories with header files
- LIBRARIES: Libraries created in this project
- CATKIN_DEPENDS: Packages dependent projects also need
- DEPENDS: System dependencies dependent projects also need (have to be listed in package.xml)

Specify locations of header files

Declare a C++ executable

Specify libraries to link the executable against

What is the .bashrc file in Linux ?

1– What is the .bashrc file ? --> it is a script file that's executed every time you open a Terminal
it contains your preferences, configurations and environmental variables.

2 – Where is located your .bashrc file? --> In the home folder: ~/

3 – Edit your .bashrc file and add *your country name* in the following environment variable to your system:

MY_COUNTRY

```
> nano ~/.bashrc
```

4 – Source your environment

```
> source ~/.bashrc
```

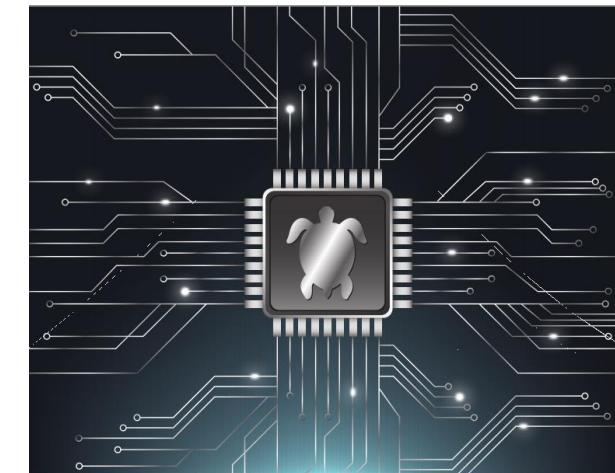
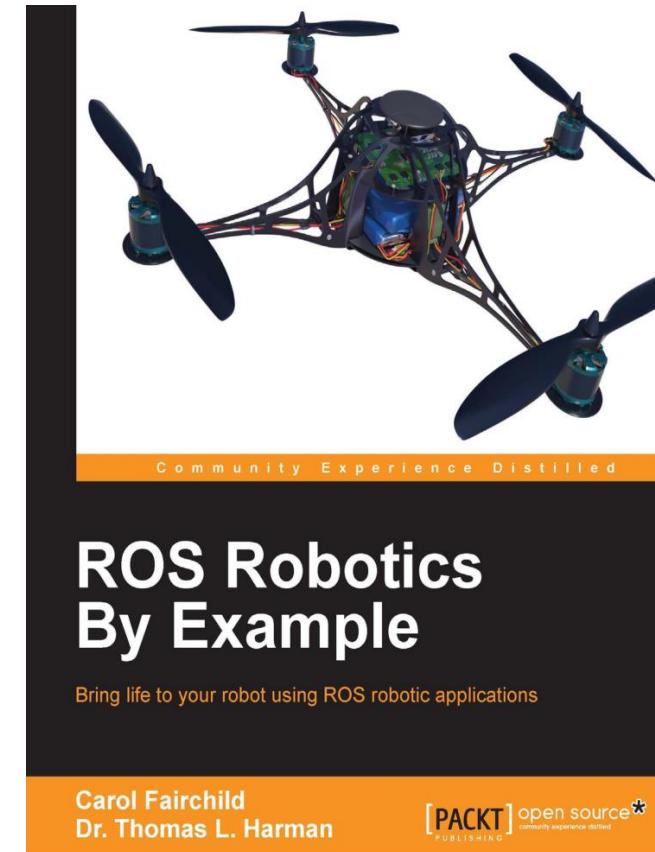
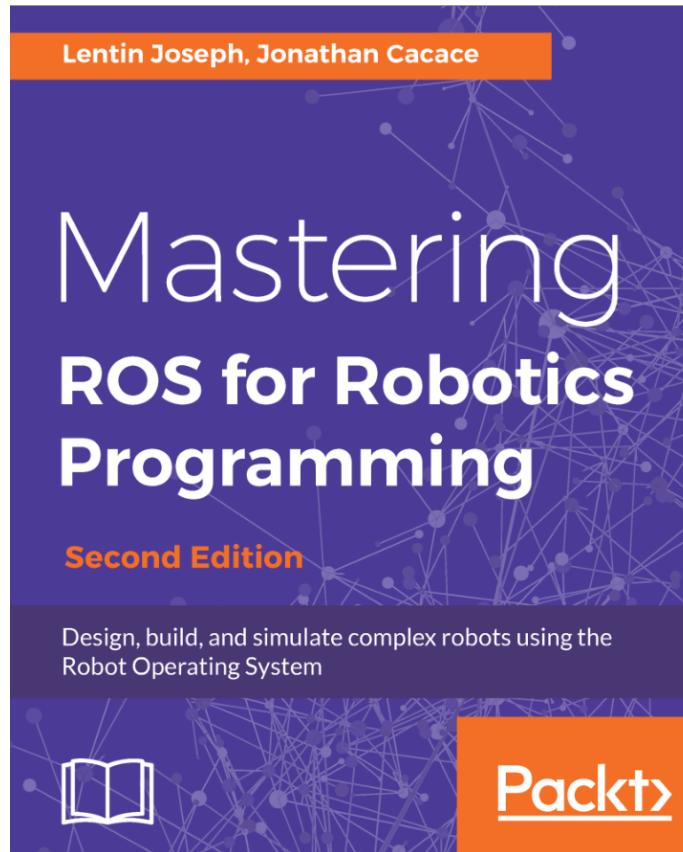
5 – Check that your variable exists with the command: *echo*

```
> echo $MY_COUNTRY
```

Further References

- **ROS Wiki**
 - <http://wiki.ros.org/>
- **Installation**
 - <http://wiki.ros.org/ROS/Installation>
- **Tutorials**
 - <http://wiki.ros.org/ROS/Tutorials>
- **Available packages**
 - <http://www.ros.org/browse/>
- **ROS Cheat Sheet**
 - <https://www.clearpathrobotics.com/ros-robot-operating-system-cheat-sheet/>
 - https://kapeli.com/cheat_sheets/ROS.docset/
- **ROS Best Practices**
 - https://github.com/leggedrobotics/ros_best_practices/wiki
- **ROS Package Template**
 - https://github.com/leggedrobotics/ros_best_practices/tree/master/ros_package_template

Relevant books



Contact Information

Université Savoie Mont Blanc

Polytech' Annecy Chambéry
Chemin de Bellevue
74940 Annecy
France

<https://www.polytech.univ-savoie.fr>

Lecturer

Luc Marechal (luc.marechal@univ-smb.fr)
SYMME Lab (Systems and Materials for Mechatronics)



SYMME