

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

Ce document à pour objectif de synthétiser l'installation et l'utilisation de ROS2. Cette simplification se base sur le tutoriel de ROS2 disponible sur le site officiel et sur la série de tutoriels réalisés par Robotics Back-End sur Youtube. Il est conseillé mais pas obligatoire d'avoir des notions en commandes Linux.

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## Sommaire

I. Prérequis et informations.....	3
II. Installation de ROS2.....	4
1 Installation de (UTF-8).....	4
2 Configuration des sources.....	4
3 Installation des packages de ROS2.....	5
4 Configuration de l'environnement ROS2.....	6
5 Test de l'installation de ROS2.....	6
III. Création d'un "Workspace" (dossier projet).....	7
1 Installation du compilateur colcon.....	7
2 Création d'un Workspace.....	8
IV. Création d'un "Package" .....	10
V. Programmer un Nœud (Node).....	13
1 Mise en place de VisualStudioCode.....	13
2 Création d'un Nœud.....	16

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## I. Prérequis et informations

- Dans ce document, il existe un code couleur pour repérer les lignes de codes à entrer dans un **terminal**, dans un **fichier script bash**, ou dans un **fichier python**.
- Pour commencer à utiliser ou installer ROS2, il est conseillé d'utiliser l'outil «**terminator**» disponible sur Linux dans l'Ubuntu Software (icône avec 4 terminaux rouges). Cet outil permet de créer plusieurs terminaux et de les afficher sur une seule fenêtre en les glissant les uns sur les autres pour les organiser.

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## II. Installation de ROS2

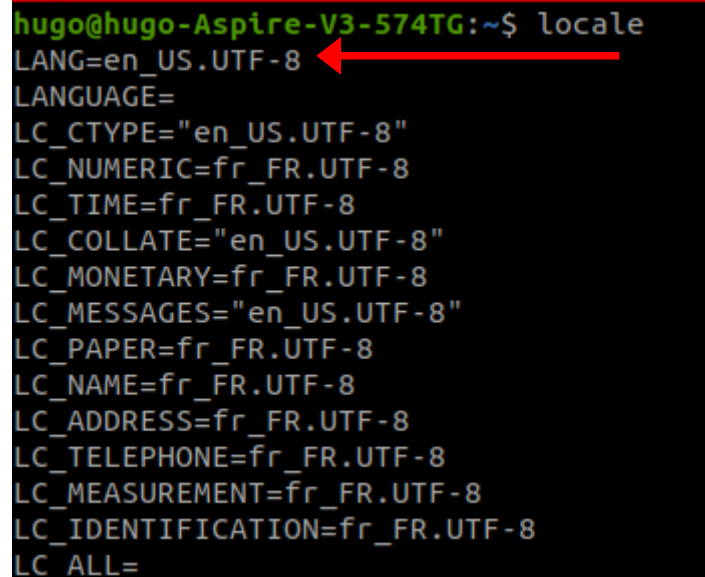
Dans cette rubrique, nous partons du principe que la distribution de Linux Ubuntu 22.04 est installée sur votre ordinateur. Ainsi nous effectuons l'installation de ROS2 de la manière suivante.

Ouvrez un terminal et écrivez les commandes suivantes (il est possible de copier/coller l'intégralité des commandes dans le terminal, il les exécute à la suite).

### 1 Installation de (UTF-8)

```
> sudo apt update && sudo apt install locales
> sudo locale-gen en_US en_US.UTF-8
> sudo update-locale LC_ALL=en_US.UTF-8 LANG=en_US.UTF-8
> export LANG=en_US.UTF-8
```

Nous observons l'installation avec la commande «locale»



```
hugo@hugo-Aspire-V3-574TG:~$ locale
LANG=en_US.UTF-8
LANGUAGE=
LC_CTYPE="en_US.UTF-8"
LC_NUMERIC=fr_FR.UTF-8
LC_TIME=fr_FR.UTF-8
LC_COLLATE="en_US.UTF-8"
LC_MONETARY=fr_FR.UTF-8
LC_MESSAGES="en_US.UTF-8"
LC_PAPER=fr_FR.UTF-8
LC_NAME=fr_FR.UTF-8
LC_ADDRESS=fr_FR.UTF-8
LC_TELEPHONE=fr_FR.UTF-8
LC_MEASUREMENT=fr_FR.UTF-8
LC_IDENTIFICATION=fr_FR.UTF-8
LC_ALL=
```

### 2 Configuration des sources

```
> sudo apt install software-properties-common
> sudo add-apt-repository universe

> sudo apt update && sudo apt install curl -y
> sudo curl -sSL https://raw.githubusercontent.com/ros/rosdistro/master/ros.key -o /usr/share/keyrings/ros-archive-keyring.gpg

> echo "deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/ros-archive-keyring.gpg]
http://packages.ros.org/ros2/ubuntu $(. /etc/os-release && echo $UBUNTU_CODENAME) main" | sudo tee
/etc/apt/sources.list.d/ros2.list > /dev/null
```

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

### 3 Installation des packages de ROS2

- **Mise à jour des «apt»:**

```
> sudo apt update  
> sudo apt upgrade
```

- **Installation de ROS2:**

Pour installer ROS2, il existe 2 choix:

- la version complète incluant les bibliothèques graphiques, des codes d'exemples, etc ...;
- la version «lite» avec uniquement le nécessaire pour faire fonctionner ROS2.

La deuxième installation est souvent utilisée pour des systèmes où les ressources sont «limitées» comme une RaspberryPI par exemple.

Installation de ROS2 Complet:

```
> sudo apt install ros-humble-desktop
```

Installation de ROS2 Limité:

```
> sudo apt install ros-humble-ros-base
```

Si vous vous rendez sur le site officiel d'installation de ROS2 Humble, ils présentent un «Development Tool» lors de l'installation de ROS2. Vous avez la possibilité de l'installer selon vos raisons. Dans ce document, nous ne l'utilisons pas.

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## 4 Configuration de l'environnement ROS2

Pour utiliser ROS2 il est nécessaire de «sourcer» son dossier d'installation afin d'utiliser les commandes de ROS2 dans les terminaux. Nous modifions le script nommé «bashrc» de la manière suivante.

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

Vous devez obtenir une fenêtre d'éditeur de texte qui s'ouvre. Nous inscrivons à la fin de ce fichier texte, la ligne suivante et enregistrons.

```
> source /opt/ros/humble/setup.bash
```

Pour actualiser le fichier .bashrc dans votre terminal exécuter la commande suivante.

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

**Note:** Vous pouvez aussi relancer votre terminal

## 5 Test de l'installation de ROS2

Pour vérifier l'installation de ROS2, il existe des nœuds de test. Lancez deux terminaux différents et exécutez les commandes suivantes.

Terminal 1:

```
> ros2 run demo_nodes_cpp talker
```

Terminal 2:

```
> ros2 run demo_nodes_cpp listener
```

Vous devez obtenir des résultats similaires (les terminaux discutent entre eux).

```
hugo@hugo-Aspire-V3-574TG:/$ ros2 run demo_nodes_cpp talker
[INFO] [1709118105.787267468] [talker]: Publishing: 'Hello World: 1'
[INFO] [1709118106.787377139] [talker]: Publishing: 'Hello World: 2'
[INFO] [1709118107.787513117] [talker]: Publishing: 'Hello World: 3'
[INFO] [1709118108.787651771] [talker]: Publishing: 'Hello World: 4'
[INFO] [1709118109.787786365] [talker]: Publishing: 'Hello World: 5'
[INFO] [1709118110.787862783] [talker]: Publishing: 'Hello World: 6'
[INFO] [1709118111.788059190] [talker]: Publishing: 'Hello World: 7'
[INFO] [1709118112.788239485] [talker]: Publishing: 'Hello World: 8'
[INFO] [1709118113.788377252] [talker]: Publishing: 'Hello World: 9'
[INFO] [1709118114.788507936] [talker]: Publishing: 'Hello World: 10'
^C[INFO] [1709118115.175352682] [rclcpp]: signal_handler(signum=2)
hugo@hugo-Aspire-V3-574TG:/$

hugo@hugo-Aspire-V3-574TG:~$ ros2 run demo_nodes_cpp listener
[INFO] [1709118107.787743960] [listener]: I heard: [Hello World: 3]
[INFO] [1709118108.787854354] [listener]: I heard: [Hello World: 4]
[INFO] [1709118109.787977387] [listener]: I heard: [Hello World: 5]
[INFO] [1709118110.788161044] [listener]: I heard: [Hello World: 6]
[INFO] [1709118111.788243708] [listener]: I heard: [Hello World: 7]
[INFO] [1709118112.788673962] [listener]: I heard: [Hello World: 8]
[INFO] [1709118113.788813338] [listener]: I heard: [Hello World: 9]
[INFO] [1709118114.788943806] [listener]: I heard: [Hello World: 10]
^C[INFO] [1709118116.023305558] [rclcpp]: signal_handler(signum=2)
hugo@hugo-Aspire-V3-574TG:~$
```

Pour arrêter l'exécution des nœuds vous pouvez presser les touches ctrl+c.

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

### III. Création d'un "Workspace" (dossier projet)

ROS2 nécessite une organisation de fichier particulière pour fonctionner et compiler. Dans un premier temps nous installons le compilateur de ROS2. Puis dans un second temps, nous créons un workspace.

#### 1 Installation du compilateur colcon

Le compilateur colcon permet de construire notre application ROS2. Cette installation ne doit pas être effectuée à chaque création d'un nouveau workspace.

Voici donc les commandes à écrire dans un terminal.

```
> sudo apt update  
> sudo apt install python3-colcon-common-extensions
```

Pour utiliser le compilateur aisément, nous sourçons le chemin d'accès de ce dernier dans le fichier .bashrc tel que :

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

Puis dans ce fichier, en dessous du «sourcing» de ROS2 effectué dans la partie « installation de ROS2», nous écrivons la ligne suivante et enregistrons.

```
> source /usr/share/colcon_argcomplete/hook/colcon_argcomplete.bash
```

Pour actualiser le fichier .bashrc dans votre terminal exécuter la commande suivante.

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

**Note:** Vous pouvez aussi relancer votre terminal

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## 2 Création d'un Workspace

Pour créer un workspace, vous devez connaître les commandes Linux suivantes:

- « `cd /chemin/d'accès` » : permet de se rendre dans un répertoire;
- « `cd ..` » : permet de retourner dans le répertoire précédent;
- « `ls /chemin/d'accès` » : permet de lister le contenu d'un répertoire;
- « `mkdir nom_de_répertoire` » : permet de créer un répertoire;
- « `rm nom_de_répertoire` » : supprime un répertoire vide;
- « `rm -rf nom_de_répertoire` » : **commande dangereuse** !\ supprime définitivement un répertoire vide ou non;

Si certaines commandes nécessitent les droits administrateurs (exemple: `rm -rf`) vous pouvez précéder la commande avec le mot «sudo» et entrer le mot de passe de votre session.

Exemple: `> sudo rm -rf mon_répertoire_a_supprimer`



Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

La création de l'espace de travail (Workspace) doit s'effectuer de la manière suivante.

- Nous créons un dossier sur le bureau s'appelant "ROS2\_Workspaces" permettant de stocker tout les différents espaces de travail.

***Note** : il est possible de renommer ce fichier et de le disposer dans un autre répertoire de l'ordinateur*

- Créez un dossier dans ce répertoire avec le nom de votre workspace (ici nous l'appellerons "App1\_ROS2")
- Dans ce dossier, créez un dossier "src".
- En restant dans le répertoire "App1\_ROS2/" tapez la commande «colcon build» et patientez. Vous obtenez l'arborescence suivante en exécutant la commande «ls».

```
hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2_Workspaces/App1_ROS2$ ls
build  install  log  src
```

- A présent, nous sourçons l'emplacement du fichier setup.bash disponible dans le répertoire «install» récemment créé, dans le fichier .bashrc à la suite des sourcing précédemment effectués.

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

```
> source ~/Desktop/ROS2_Workspaces/App1_ROS2/install/setup.bash
```

Le chemin d'accès peut être différent de celui-ci si vous avez modifié les éléments de création des dossiers du workspace (nom, répertoire etc...).

Vous devez obtenir des lignes de la même forme dans votre fichier .bashrc.

```
121 #Sourcing de ROS2
122 source /opt/ros/humble/setup.bash
123 #Sourcing de colcon
124 source /usr/share/colcon_argcomplete/hook/colcon-argcomplete.bash
125 #Sourcing des Workspaces
126 source ~/Desktop/ROS2_Workspaces/App1_ROS2/install/setup.bash
```

Nous actualisons les terminaux avec la commande suivante.

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

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## IV. Création d'un "Package"

Les packages permettent d'organiser les nœuds et les codes que nous allons créer par la suite. Ils permettent d'organiser les différents éléments d'un robot. Un package appelé «my\_robot\_camera» nous permet d'organiser l'écriture des nœuds liés au traitement d'image d'un robot par exemple.

Pour créer un package, rendez-vous dans le répertoire /src du workspace et écrivez la commande suivante, permettant de créer un package avec une base de code en python et en dépendance «rclpy».

```
> ros2 pkg create my_turtlesim_controller --build-type ament_python --dependencies rclpy
```

Une convention peut être employée pour renommer les packages. En voici une: «NomDuRobot\_NomDuPackage».

Dans notre cas, un exemple d'objectif est de contrôler le nœud «turtlesim\_node» (disponible dans les nœuds d'exemple de ROS2) pour exécuter des mouvements programmés par des nœuds personnalisés.

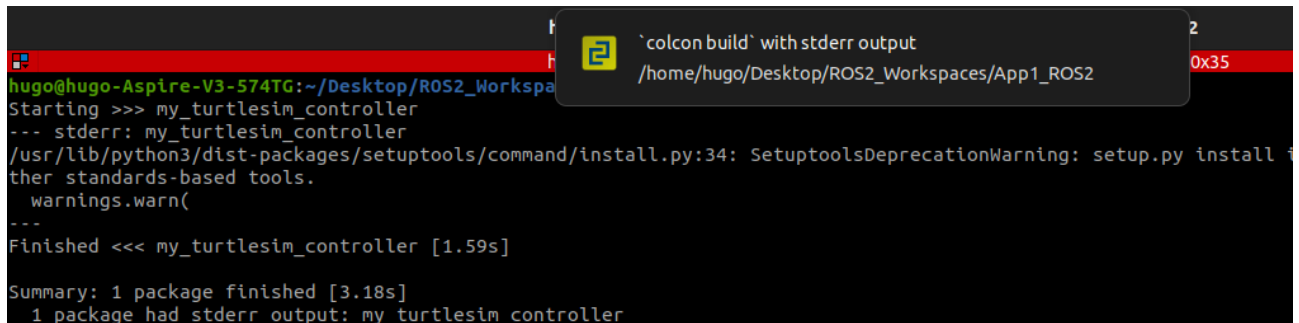
Retournons dans le répertoire principal de notre workspace ~/App1\_ROS2 (contenant src, install, build, log) et inscrivons la commande suivante.

```
> colcon build
```

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

Si vous ne rencontrez pas l'erreur suivante veuillez sauter cette étape.

- Résolution de l'erreur



```

hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2_Workspa
Starting >>> my_turtlesim_controller
--- stderr: my_turtlesim_controller
/usr/lib/python3/dist-packages/setuptools/command/install.py:34: SetuptoolsDeprecationWarning: setup.py install is deprecated. Use pip or other standards-based tools.
  warnings.warn(
---
Finished <<< my_turtlesim_controller [1.59s]

Summary: 1 package finished [3.18s]
1 package had stderr output: my_turtlesim_controller

```

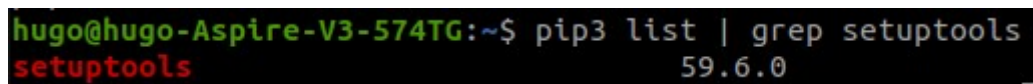
Sur un autre terminal installez le composant suivant.

> sudo apt install python3-pip

Nous listons les éléments de pip3 pour observer la version du composant «setuptools».

> pip3 list | grep setuptools

Nous obtenons le résultat suivant.



```

hugo@hugo-Aspire-V3-574TG:~$ pip3 list | grep setuptools
setuptools 59.6.0

```

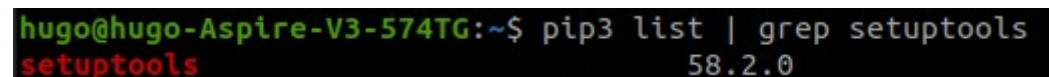
La version du setuptools doit être rétrogradé en 58.2.0 .

> pip3 install setuptools==58.2.0

Nous listons de nouveau les éléments de pip3 pour observer la version du composant «setuptools».

> pip3 list | grep setuptools

Nous obtenons le résultat suivant.



```

hugo@hugo-Aspire-V3-574TG:~$ pip3 list | grep setuptools
setuptools 58.2.0

```

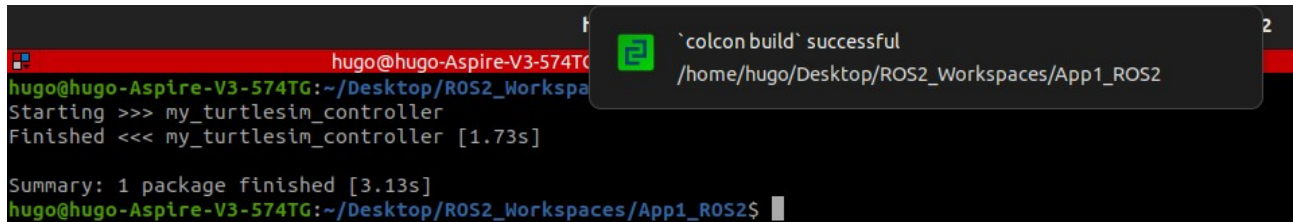
Nous pouvons retourner sur le terminal étant ouvert sur les répertoires du workspace et effectuer la commande

> colcon build

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

- Si vous ne rencontrez pas d'erreur

Si la commande fonctionne, vous devez obtenir le message suivant.



The image shows a terminal window with a dark background. The prompt is 'hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2\_Workspaces/App1\_ROS2\$'. The user has entered 'colcon build' and received a notification bubble that says 'colcon build` successful' and '/home/hugo/Desktop/ROS2\_Workspaces/App1\_ROS2'. Below this, the user has entered 'my\_turtlesim\_controller' and received a notification bubble that says 'Starting >>> my\_turtlesim\_controller' and 'Finished <<< my\_turtlesim\_controller [1.73s]'. The terminal output shows 'Summary: 1 package finished [3.13s]' and the prompt returns to 'hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2\_Workspaces/App1\_ROS2\$'.

```
hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2_Workspaces/App1_ROS2$ colcon build
colcon build` successful
/home/hugo/Desktop/ROS2_Workspaces/App1_ROS2

hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2_Workspaces/App1_ROS2$ my_turtlesim_controller
Starting >>> my_turtlesim_controller
Finished <<< my_turtlesim_controller [1.73s]

Summary: 1 package finished [3.13s]
hugo@hugo-Aspire-V3-574TG:~/Desktop/ROS2_Workspaces/App1_ROS2$
```

La commande «colcon build» compile tout les éléments du workspace et positionne des fichiers exécutables par ROS2 dans le répertoire «install» du workspace.

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## V. Programmer un Nœud (Node)

### 1 Mise en place de VisualStudioCode

Pour programmer des nœud nous utilisons des fichiers en python et l'IDE Visual Studio Code. Pour installer Vs-code, nous entrons la commande suivante.

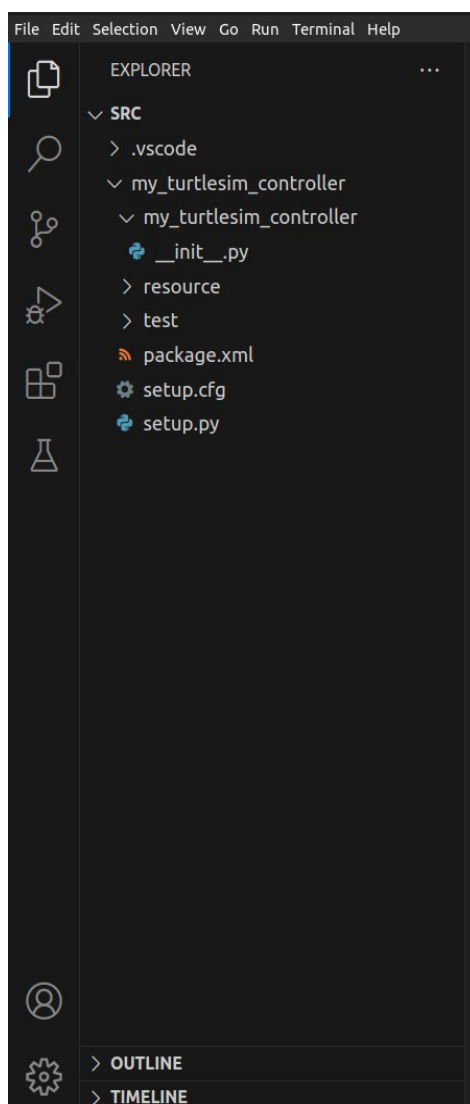
```
> sudo snap install code --classic
```

Rendez-vous dans le répertoire /src du workspace et entrer la commande suivante permettant d'exécuter Vs-code dans le répertoire courant.

```
> code .
```

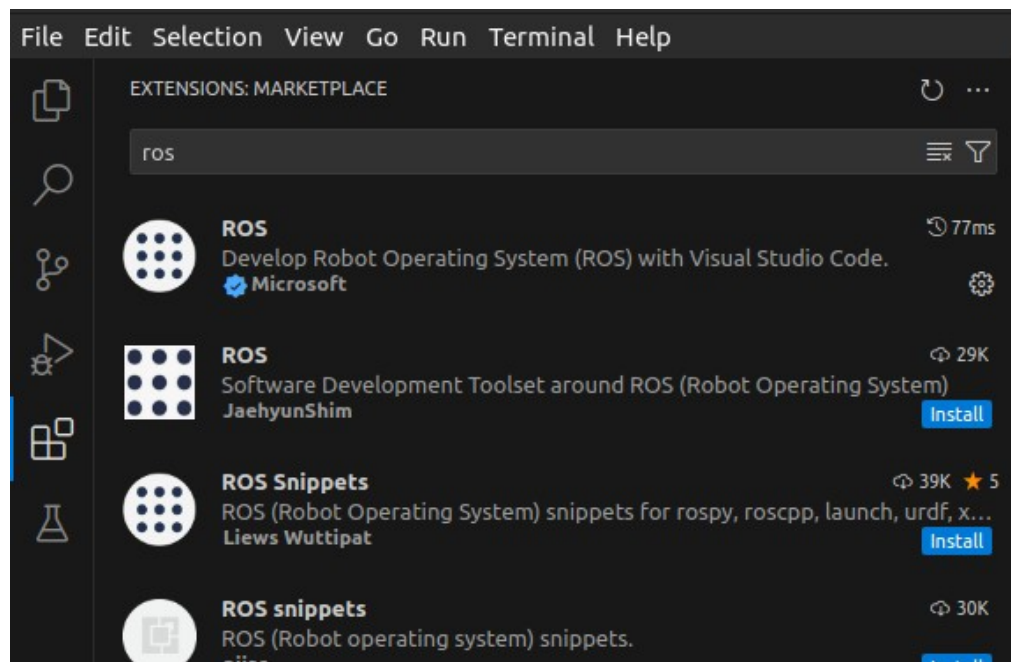
Vous obtenez une arborescence comme il suit.

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges



Pour faciliter la programmation nous installons différentes extensions intéressantes pour programmer les éléments de ROS2. Rendez vous dans l'onglet «extensions» de Vs-code, recherchez «ros» et installez la version avec la référence Microsoft (cette extension fonctionne pour ROS 1 et 2).

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges



Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

## 2 Création d'un Nœud

Une fois l'installation des extensions terminée, nous nous rendons à l'aide du terminal à l'emplacement du fichier `__init__.py`. Dans notre cas nous entrons la commande suivante.

```
> cd ~/Desktop/ROS2_Workspaces/App1_ROS2/src/my_turtlesim_controller/my_turtlesim_controller
```

Nous créons un fichier `.py` et le rendons exécutable de la manière suivante.

```
> touch my_first_node.py  
> chmod +x my_first_node.py
```

Ainsi nous pouvons retourner sur Vs-code et commencer à programmer un nœuds classique en ouvrant le fichier que nous venons de créer.

Pour programmer des nœuds, il faut avoir quelques notions de programmation orienté objet (POO).



Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

Voici un canva pouvant être repris pour débiter la programmation dans un fichier.

```
#!/usr/bin/env python3
import rclpy # Dépendance Python pour ROS
from rclpy.node import Node # Import de la notion de nodes

# 1er type de node appelé "MyNode"
class MyNode(Node):
    # Constructeur de la node
    def __init__(self):
        super().__init__("first_node") # définition de la node avec un nom arbitraire
        self.get_logger().info("Hello From ROS2") #Permet d'écrire dans la console

def main(args=None):
    rclpy.init(args=args) # initialisation des communications de ros

    #===== Zone de programmation =====#
    instance_node = MyNode() # Déclaration d'une instance de la Node MyNode

    rclpy.spin(instance_node) # Permet de garder la node active
    #=====#

    rclpy.shutdown() # eteindre les communications de ros
    pass

if __name__ == '__main__': # Permet d'appeller la fonction main
    main()
```

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges

Pour compiler les modifications, rendez-vous dans le dossier principal du workspace (contenant le dossier src, install, build, et log notamment) et écrivez dans un terminal la commande suivante.

```
> colcon build
> source ~/.bashrc
```

Cette commande comme telle, doit être exécutée après chaque modification de fichier pour les prendre en compte.

Cependant, il existe une astuce permettant de ne pas avoir à exécuter cette commande après chaque enregistrement de fichier.

Inscrivez plutôt la commande suivante.

```
> colcon build --symlink-install
> source ~/.bashrc
```

Cette commande indique au compilateur de prendre en compte les modifications automatiquement et de les compiler.

**(note: il est important de sourcer le .bashrc après la compilation, si vous rencontrez des soucis d'exécution des fichier avec la commande «ros2 run ...»)**

Pour prendre en compte un nœud par la commande «ros2 run», il faut se rendre dans le fichier **setup.py** et inscrire dans le tableau 'console\_scripts' les informations suivantes pour chaque nœud.

```
'console_scripts': [
    "nom_d'appel = nom_du_package.nom_du_fichier_node:fonction_du_fichier_à_executer",
    "nom_d'appel = nom_du_package.nom_du_fichier_node:fonction_du_fichier_à_executer",
    .
    .
    .
    "nom_d'appel = nom_du_package.nom_du_fichier_node:fonction_du_fichier_à_executer",
],
```

Voici un exemple dans notre cas.

```
'console_scripts': [
    "test_my_first_node = my_turtlesim_controller.my_first_node:main",
],
```

Attention, les noms d'appel doivent être différents pour chaque fichier. Ainsi pour exécuter les nœuds, vous devez écrire la commande suivante pour chaque nœuds .

```
> ros2 run nom_du_package nom_d'appel
```

Voici un exemple pour notre cas.

```
> ros2 run my_turtlesim_controller test_my_first_node
```

Hugo XAVIER	Guide d'utilisation de ROS2 Humble (Ubuntu 22.04)	01/03/2024
Stagiaire		PRISME Bourges