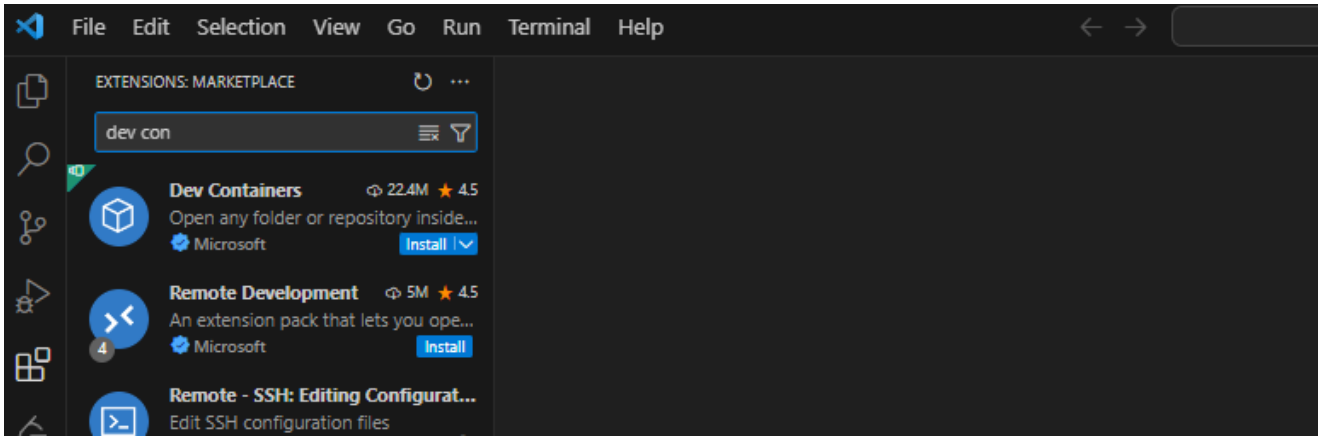
VS Code Docker integration

By JorgePRamos

In this guide we will show how to set up **Dev containers**, which allows us to develop in different OS with the full support of our IDE (linting, auto completion, plugins …)
Important note: we need Docker to be installed in our machine.

Preliminary steps

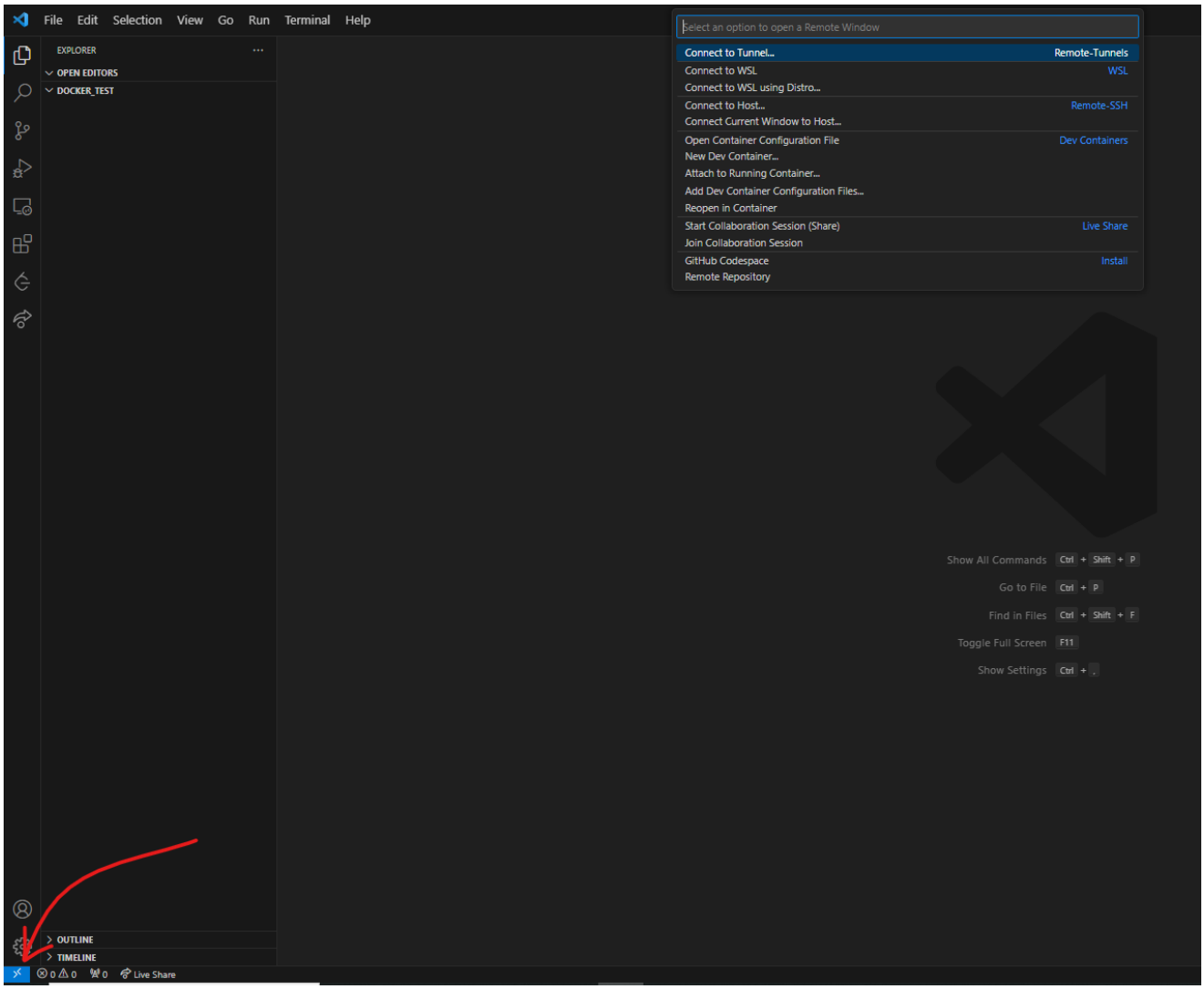
Before we need to have both installed: VS Code and the "Dev Containers" extension.



Note: A good practice is to install the "Remote Development" pack by Microsoft, which includes some additional extension that may be useful in our development as well as the "Dev Containers" extension.

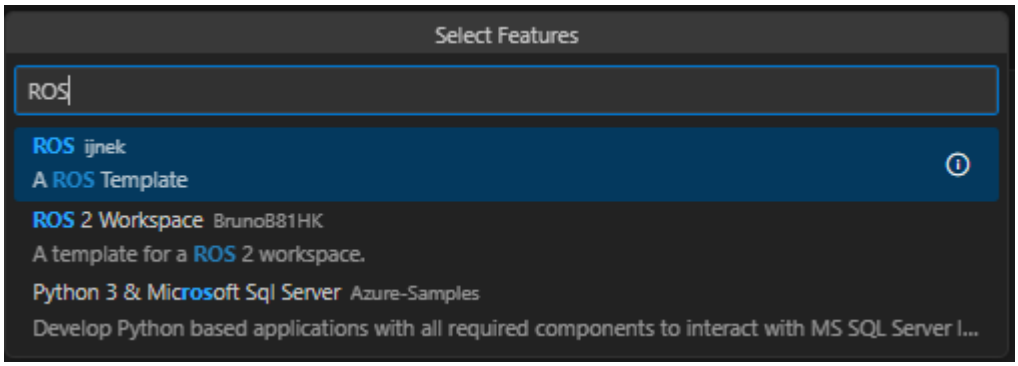
Dev Containers use

After the quick installation of the necessary extensions we are ready to start developing.
Lets create a new folder which will act as our development root.
From here we need to click the blue icon on the bottom left corner (remote session button), which will bring up a context menu about remote options.



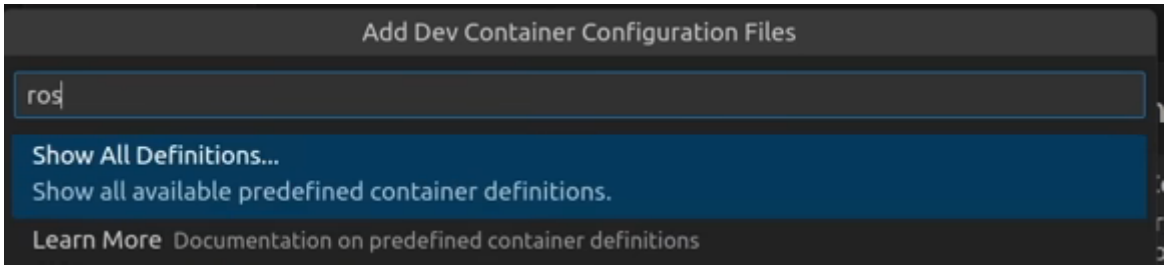
Here we will chose the option "Reopen in container" .
Now is we are presented with an extensive list of docker images.

For our example lets use a ROS2 image.

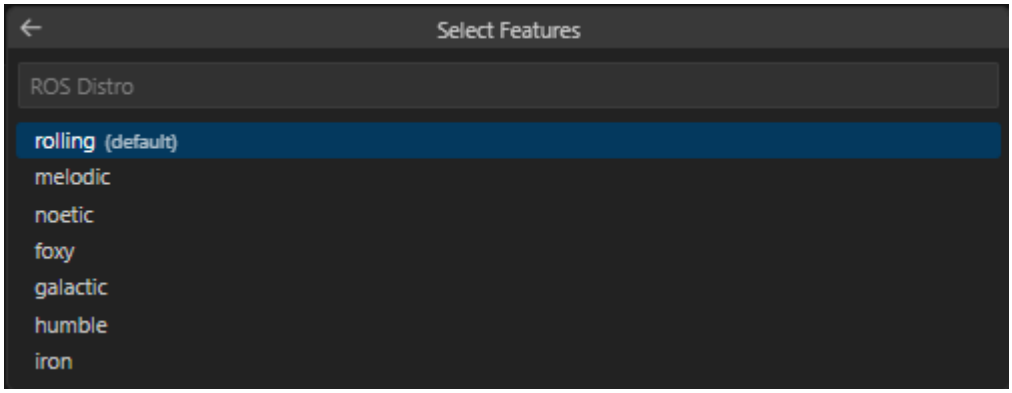


In our case we will chose the image provided by "ijnek" real name "Kenji Brameld" which is a renown ROS2 contributor.

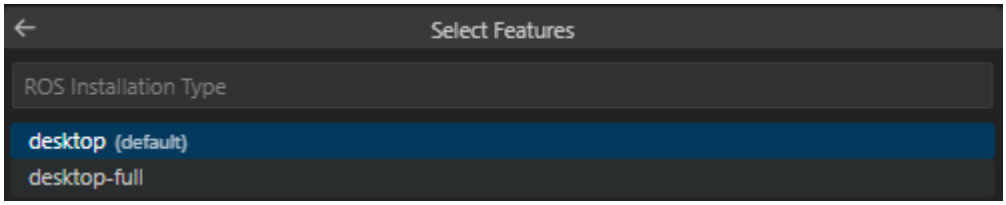
Note: If our search is unsuccessful we may want to to check the option "Show All Definitions".



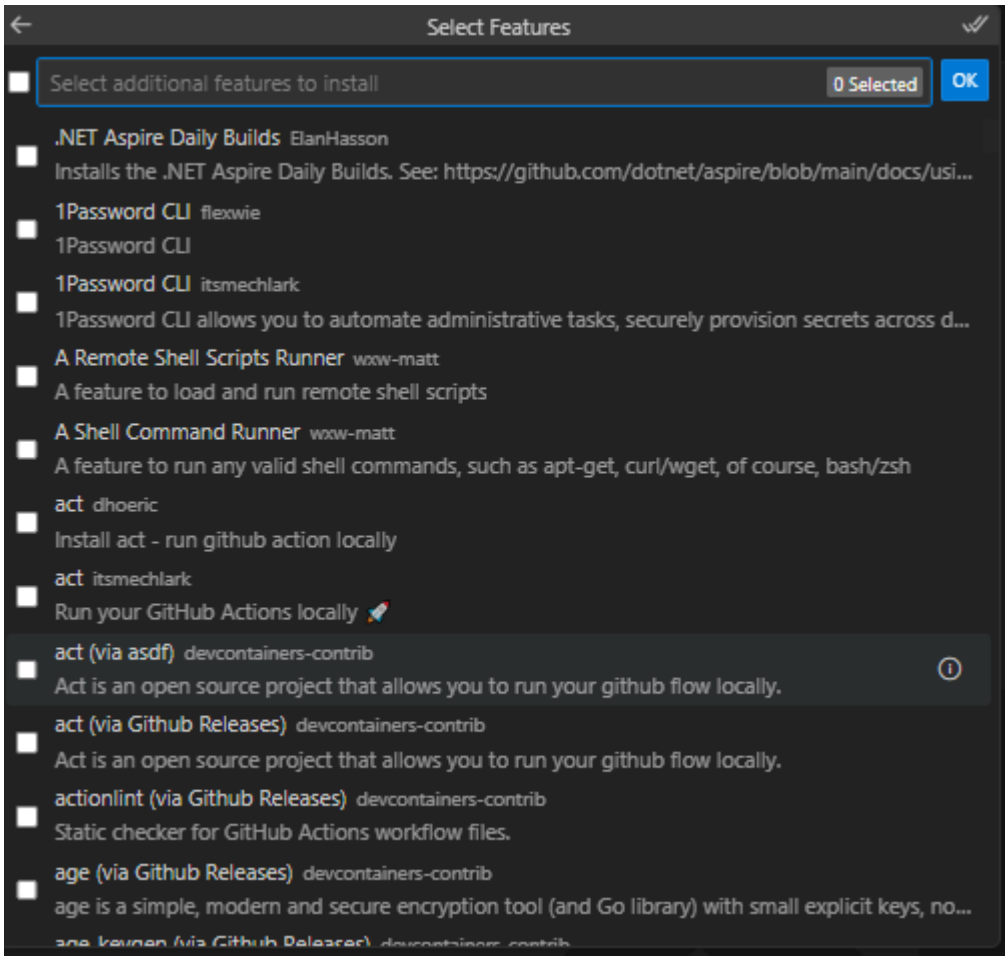
Then we can chosse our ditribution in ur case "humble"



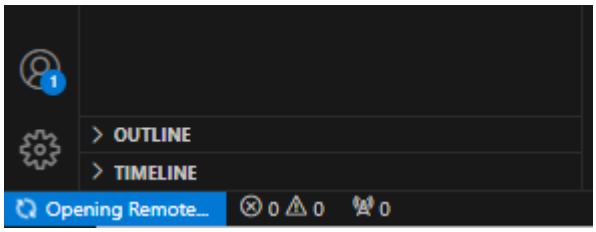
Then we need to select our installation type. in our case "desktop"



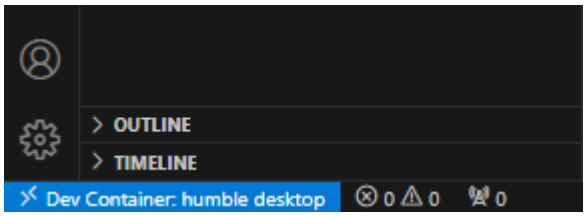
And lastly we will be prompted to install additional features by default which can be still be added after the creation of our container. In our case we will not choose anything.



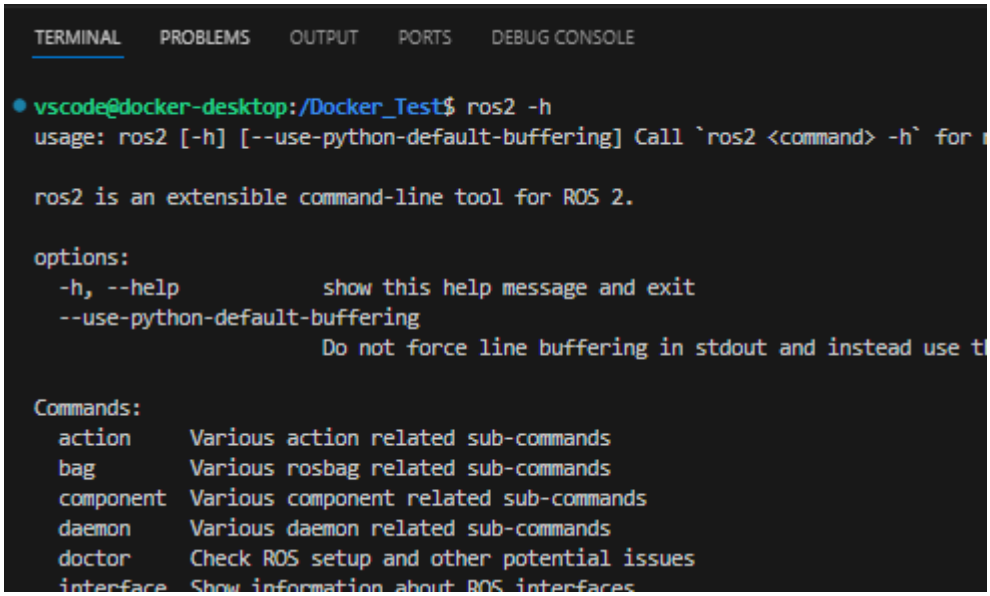
After this our container will be created and we can wait in the meantime and maybe grab a coffee our a tea, in our case coffee (the superior option) ;)



We will know that the build has finished when the bottom left icon changes to our containers name.

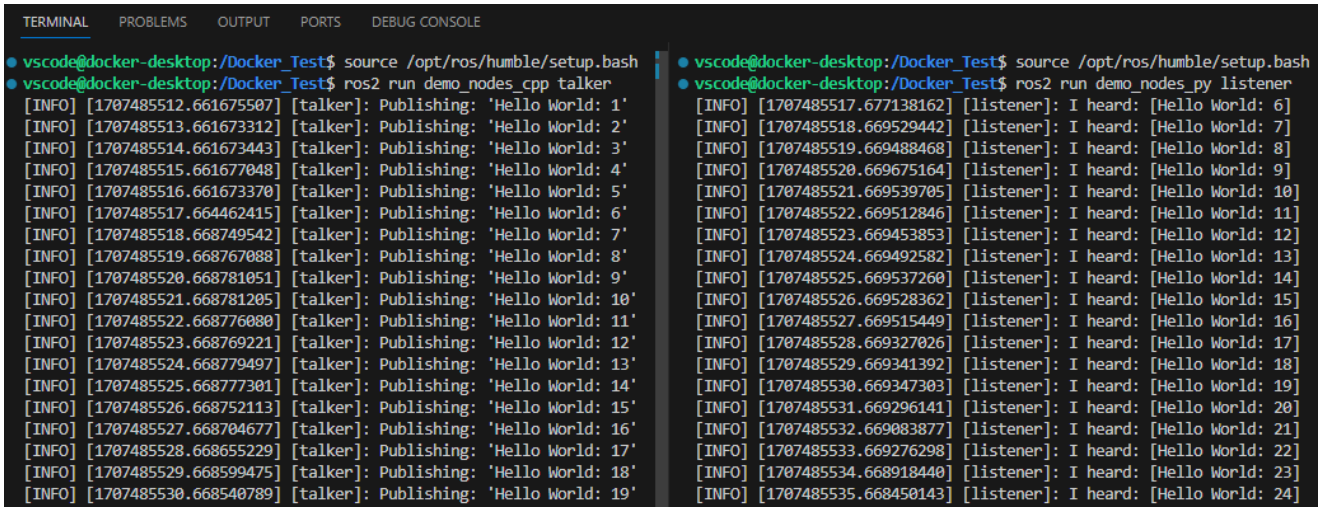


Now any terminal we open in Vs Code will be automatically inside of our container.



Talker - Listener example

Lets run a couple of ROS2 nodes to celebrate. Neat!



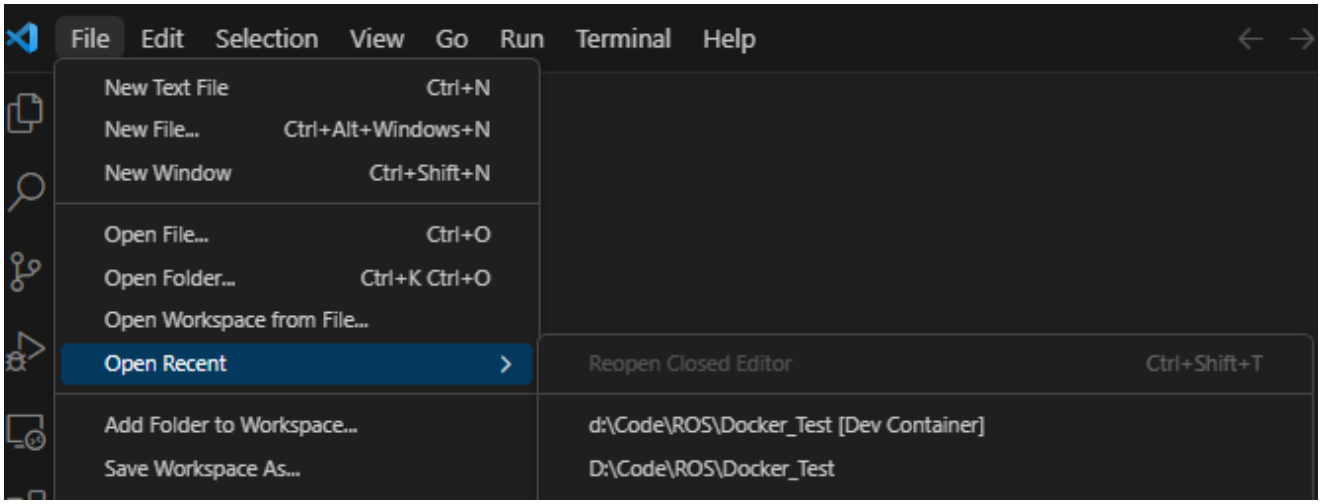
Working with the Dev Container

Extensions

Extensions work different when working inside of container and we have to understand the we will have local extension, being the ones you already have installed and remote extension which will reside specifically on our container and will aid us in development.

Reopening an existing container

We can can open back where we left selecting our container by:
File --> Open Recent --> [Container name]



GUI for our containers

In order to have access to GUI applications that we may run in our container like RViz or Gazebo, we must edit our container `.json` .

We will be adding the desktop lite feature
And forwarding a couple ports from our container

```
"features": {
  "ghcr.io/devcontainers/features/desktop-lite:1":{}
},
"forwardPorts": [6080,5901],
"portsAttributes": {
  "6080": {
    "label": "Desktop-Web"
  },
  "5901": {
```

```
      "label": "Desktop-VNC"

    }

  }```

**Important note:** Do not forget to separate with commas in order to follow the JSON syntax.

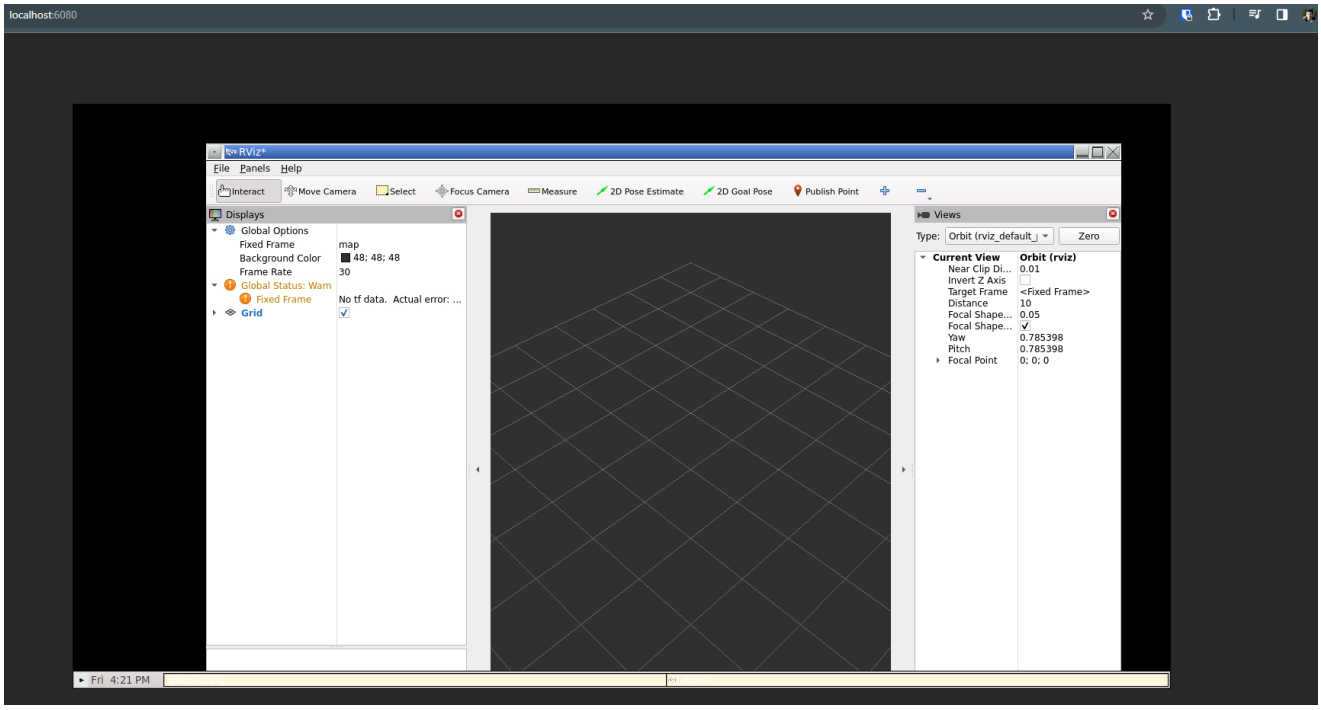
Last step is to delete the argument from `runArgs` named `"--network=host"` which can prevent the launch of desktop lite.

Once the changes are done we will either get prompted by Vs Code to rebuild our container or do ourselves.
Lets open the command pallet and look por the "Rebuild Container" option.
![[Pasted image 20240209170951.png]]

After its finished building we will notice 2 ports opened in Vs Code
![[Pasted image 20240209171536.png]]

If we open our browser of choice and enter: `localhost:6080` will bre greated with the following screen.
![[Pasted image 20240209171651.png]]

Click "Connect" and type `vscode` as the default password.
**Lo and behold our Linux desktop.**
Lets try starting `rviz` by running the follwing commmand in our terminal.
``` bash
source /opt/ros/humble/setup.bash
rviz2
```


```

From now on we can use our browser as our "virtual monitor" leaving behind those dark days of having 2 keyboards, two mouses and 2 screen to work directly on our ROS machine.