Mini guida: Executable e Node in ROS 2

Questa guida riassume le informazioni chiave su come funzionano gli **executables** e i **nodi** in ROS 2 e come usarli correttamente con `ros2 run`.

1️⃣ Differenza tra Executable e Node

Concetto	Descrizione
Executable	File binario compilato (C++ o Python). Può contenere uno o più nodi ROS 2, oppure nessun nodo e funzionare come programma normale.
Node	Entità logica attiva nel ROS graph. Viene creata da un executable tramite codice (es. `auto node = std::make_shared<rclcpp::Node>("nome_nodo");`). Consente la comunicazione tramite topic, servizi e azioni.

Esempi:

1. Executable con nodo ROS 2:

```
int main() {
  rclcpp::init(argc, argv);
  auto node = std::make_shared<rclcpp::Node>("talker_node");
  rclcpp::spin(node);
  rclcpp::shutdown();
  return 0;
}
```

2. Executable senza nodo ROS 2:

```
int main() {
  std::cout << "Programma senza ROS 2" << std::endl;
  return 0;
}
```

2️⃣ Eseguire un executable con ROS 2

Sintassi di base

```
ros2 run <nome_pacchetto> <nome_executable>
```

- `<nome_pacchetto>`: pacchetto ROS 2 che contiene l'executable - `<nome_executable>`: il nome del file binario definito in `CMakeLists.txt` con `add_executable(...)`

Cosa succede se l'executable non crea nodi

- L'executable viene eseguito normalmente.
- ROS 2 non registra alcun nodo nel ROS graph.
- `ros2 node list` non mostrerà nulla.

3️⃣ Requisiti perché `ros2 run` funzioni

1. **Definizione in CMake**

```
add_executable(talker src/publisher_member_function.cpp)
ament_target_dependencies(talker rclcpp std_msgs)
install(TARGETS talker DESTINATION lib/${PROJECT_NAME})
```

2. `add_executable`: crea il binario
3. `ament_target_dependencies`: collega le librerie necessarie
4. `install(TARGETS ...)`: rende l'executable visibile a ROS 2

5. **Build con colcon**

```
colcon build
```

6. Compila il codice
7. Copia gli eseguibili nell'install space

8. **Aggiornamento dell'ambiente**

```
source install/setup.bash
```

9. ROS 2 aggiorna le variabili (`PATH`, `AMENT_PREFIX_PATH`) per trovare gli executables

10. **Esecuzione**

```
ros2 run cpp_pubsub talker
```

11. ROS 2 cerca l'executable nel pacchetto installato e lo lancia
12. Se non è installato o non esiste, il comando fallisce

4️⃣ Sintesi pratica

Passo	Necessario per `ros2 run`?
`add_executable`	Sì, crea il binario
`ament_target_dependencies`	Sì, per compilare correttamente
`install(TARGETS …)`	Sì, rende l'executable visibile a ROS 2
`source install/setup.bash`	Sì, aggiorna l'ambiente

5️⃣ Nota finale

- Tutti i nodi ROS 2 sono contenuti in un executable, ma non tutti gli executables contengono nodi.
- Un executable senza nodi può comunque essere lanciato, ma non comparirà nel ROS graph.
- `ros2 run` **non compila il codice**: esegue solo l'executable già presente nell'install space.

Fine della mini guida