ROS 2 Odometry Variable Fetcher

This notebook uses a helper function to subscribe to the `/odom` topic and fetch one value for each major odometry variable.

Ensure your ROS 2 system is running and publishing to `/odom` before executing the cells.

In [ ]:
```python
#  Step 1: Import the helper function from script
from get_odom_variable import get_odom_variable
```

Step 2: Fetch Odometry Values

This will get a **single value** from the odometry message for each key variable.

In [ ]:
```python
# Position (meters)
pos_x = get_odom_variable('pose.pose.position.x')
pos_y = get_odom_variable('pose.pose.position.y')
pos_z = get_odom_variable('pose.pose.position.z')

# Orientation (quaternion)
orient_x = get_odom_variable('pose.pose.orientation.x')
orient_y = get_odom_variable('pose.pose.orientation.y')
orient_z = get_odom_variable('pose.pose.orientation.z')
orient_w = get_odom_variable('pose.pose.orientation.w')

# Linear velocity (m/s)
lin_x = get_odom_variable('twist.twist.linear.x')
lin_y = get_odom_variable('twist.twist.linear.y')
lin_z = get_odom_variable('twist.twist.linear.z')

# Angular velocity (rad/s)
ang_x = get_odom_variable('twist.twist.angular.x')
ang_y = get_odom_variable('twist.twist.angular.y')
ang_z = get_odom_variable('twist.twist.angular.z')
```

Step 3: Display the Results

In [ ]:
```python
print(f"\n Position:")
print(f"  x = {pos_x:.3f} m, y = {pos_y:.3f} m, z = {pos_z:.3f} m")

print(f"\n Orientation (quaternion):")
print(f"  x = {orient_x:.3f}, y = {orient_y:.3f}, z = {orient_z:.3f}, w = {o
```

```
print(f"\n Linear Velocity:")
print(f"  x = {lin_x:.3f} m/s, y = {lin_y:.3f} m/s, z = {lin_z:.3f} m/s")

print(f"\n Angular Velocity:")
print(f"  x = {ang_x:.3f} rad/s, y = {ang_y:.3f} rad/s, z = {ang_z:.3f} rad/
```

Step 4: Collect Variables Over Time

This cell collects `position.x` and `angular.z` over time and saves the data to a CSV file.

In [ ]:
```
import pandas as pd
import time

# Collect N samples at interval (seconds)
N = 10  # number of samples
interval = 1.0  # seconds between samples

log_data = []

for i in range(N):
    pos_x = get_odom_variable('pose.pose.position.x')
    ang_z = get_odom_variable('twist.twist.angular.z')
    timestamp = time.time()
    log_data.append({'time': timestamp, 'pos_x_m': pos_x, 'ang_z_rad_s': ang
    print(f"[{i+1}/{N}] Logged: pos_x = {pos_x:.3f}, ang_z = {ang_z:.3f}")
    time.sleep(interval)

df = pd.DataFrame(log_data)
```

Step 5: Save Data to CSV

In [ ]:
```
df.to_csv("odom_log.csv", index=False)
print(" Saved to odom_log.csv")
df.head()
```

Step 6: Live Graph with `rosshow`

`rosshow` is a ROS 2 tool for live plotting topics in the terminal.

Note: This will run a background subprocess. Make sure `rosshow` is installed:

`pip install rosshow`
To stop it, interrupt the cell (stop button in Jupyter).

In [ ]:
```
import subprocess

# Replace 'twist.twist.angular.z' with any field you want to graph
```

```
cmd = ["rosshow", "/odom", "twist.twist.angular.z"]
print(f" Launching: {' '.join(cmd)}\n(Press stop to exit)")
subprocess.run(cmd)
```