# 🐢 Turtlesim Discovery & Command Guide

## 🎯 Quick Start - No Coding Required

### Step 1: Install and Start Turtlesim

bash

```
# Install turtlesim (if not already installed)
sudo apt update
sudo apt install ros-humble-turtlesim

# Start turtlesim in Terminal 1
ros2 run turtlesim turtlesim_node
```

### Step 2: Discover Available Commands

#### 🎯 Discover Topics

bash

```
# List all available topics
ros2 topic list

# See topic info
ros2 topic info /turtle1/cmd_vel
ros2 topic info /turtle1/pose

# View message types
ros2 topic type /turtle1/cmd_vel
ros2 topic type /turtle1/pose

# See message structure
ros2 interface show geometry_msgs/msg/Twist
ros2 interface show turtlesim/msg/Pose
```

#### 🎯 Discover Services

bash

```
# List all available services
ros2 service list

# See service types
ros2 service type /spawn
ros2 service type /turtle1/set_pen
ros2 service type /reset

# View service message structure
ros2 interface show turtlesim/srv/Spawn
ros2 interface show turtlesim/srv/SetPen
ros2 interface show std_srvs/srv/Empty
```

### 🎯 Discover Parameters

```
bash
# List node parameters
ros2 param list

# Get parameter values
ros2 param get /turtlesim background_r
ros2 param get /turtlesim background_g
ros2 param get /turtlesim background_b

# See parameter descriptions
ros2 param describe /turtlesim background_r
```

---

# 🚀 Hands-on Turtlesim Commands

## Terminal 1: Turtlesim Node

```
bash
ros2 run turtlesim turtlesim_node
```

## Terminal 2: Manual Control

```bash
# Manual teleoperation
ros2 run turtlesim turtle_teleop_key

# Use arrow keys to control the turtle
```

## Terminal 3: Command Line Control

### 🎯 Topic Commands (Movement)

```bash
# Move forward
ros2 topic pub -1 /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"

# Turn left
ros2 topic pub -1 /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 1.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.0}}"

# Continuous movement (press Ctrl+C to stop)
ros2 topic pub --rate 1 /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 1.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.5}}"

# Monitor turtle position
ros2 topic echo /turtle1/pose
```

### 🎯 Service Commands (Configuration)

```bash
# Spawn a new turtle
ros2 service call /spawn turtlesim/srv/Spawn "{x: 2.0, y: 2.0, theta: 0.0, name: 'turtle2'}"

# Change pen color (red, width 5)
ros2 service call /turtle1/set_pen turtlesim/srv/SetPen "{r: 255, g: 0, b: 0, width: 5, off: 0}"

# Turn pen off (stop drawing)
ros2 service call /turtle1/set_pen turtlesim/srv/SetPen "{r: 0, g: 0, b: 0, width: 1, off: 1}"
```

```bash
# Teleport turtle
ros2 service call /turtle1/teleport_absolute turtlesim/srv/TeleportAbsolute "{x: 1.0, y: 1.0, theta: 1.57}"

# Clear drawing
ros2 service call /clear std_srvs/srv/Empty

# Reset everything
ros2 service call /reset std_srvs/srv/Empty
```

### 🎯 Parameter Commands (Appearance)

```bash
# Change background color to green
ros2 param set /turtlesim background_r 0
ros2 param set /turtlesim background_g 255
ros2 param set /turtlesim background_b 0

# Change background color to blue
ros2 param set /turtlesim background_r 0
ros2 param set /turtlesim background_g 0
ros2 param set /turtlesim background_b 255

# Reset to original background
ros2 param set /turtlesim background_r 69
ros2 param set /turtlesim background_g 86
ros2 param set /turtlesim background_b 255

# Dump all parameters to file
ros2 param dump /turtlesim > turtlesim_params.yaml

# Load parameters from file
ros2 param load /turtlesim turtlesim_params.yaml
```

---

## 🔍 Discovery Exercises

### Exercise 1: Explore Node Information

```bash
# See all running nodes
ros2 node list

# Get detailed info about turtlesim node
ros2 node info /turtlesim

# This shows:
# - Subscribers (who receives messages)
# - Publishers (who sends messages)
# - Services (what services it provides)
# - Actions (what actions it provides)
```

### Exercise 2: Monitor Real-time Data

```bash
# Terminal 1: Watch turtle position continuously
ros2 topic echo /turtle1/pose

# Terminal 2: Move turtle with teleop and watch values change
ros2 run turtlesim turtle_teleop_key
```

### Exercise 3: Service Discovery

```bash
# See what services each node provides
ros2 node info /turtlesim | grep Services

# Try calling unknown service to see error
ros2 service call /unknown_service std_srvs/srv/Empty
```

---

## 🎯 Fun Turtlesim Challenges

### Challenge 1: Draw a Square

bash

```
# Manual square drawing using services and topics
ros2 service call /turtle1/set_pen turtlesim/srv/SetPen "{r: 255, g: 0, b: 0, width: 3, off: 0}"
ros2 topic pub -1 /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 2.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"
ros2 topic pub -1 /turtle1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 0.0, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 1.57}}"

# Repeat 4 times
```

## Challenge 2: Multi-turtle Race

bash

```
# Spawn multiple turtles
ros2 service call /spawn turtlesim/srv/Spawn "{x: 1.0, y: 1.0, theta: 0.0, name: 'racer1'}"
ros2 service call /spawn turtlesim/srv/Spawn "{x: 1.0, y: 9.0, theta: 0.0, name: 'racer2'}"

# Make them race
ros2 topic pub --rate 1 /racer1/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 1.5, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"

ros2 topic pub --rate 1 /racer2/cmd_vel geometry_msgs/msg/Twist "{linear: {x: 1.2, y: 0.0, z: 0.0}, angular: {x: 0.0, y: 0.0, z: 0.0}}"
```

## Challenge 3: Color Changing Background

bash

```
# Create a colorful background animation
for i in {1..10}; do
    ros2 param set /turtlesim background_r $((RANDOM % 255))
    ros2 param set /turtlesim background_g $((RANDOM % 255))
    ros2 param set /turtlesim background_b $((RANDOM % 255))
    sleep 1
done
```