Course : ROS 2, Robot Operating System

Develop robotics applications

Practical course - 3d - 21h00 - Ref. ROZ

Price : 2470 CHF E.T.

★★★★★ 5 / 5

NEW

ROS 2 (Robot Operating System) is currently the most widely used framework for developing robotic applications. This course will help you understand its modular architecture and acquire the skills needed to design Python or C++ applications for controlling mobile robots and robotic arms, as well as integrating AI-based algorithms into a robotic system.

Teaching objectives

At the end of the training, the participant will be able to:

- Understanding the Modular Architecture of ROS 2 Applications
- Learn about debugging tools to make progress on your project
- Implementing Autonomous Navigation on a Mobile Robot
- Simulate sensors, actuators, and an environment in which they operate
- Integrating AI Processing into a Robot

Intended audience

Various developers, robotics engineers, embedded systems engineers, and technical project managers.

Prerequisites

Strong knowledge of Python and C++.

Course schedule

1 Introduction to ROS 2

- Overview of the ROS ecosystem.
- An overview of its technological building blocks and limitations.
- Community development and the project's communities.

Hands-on work

Manipulate existing ROS packages using the CLI to understand their architecture, inspect the runtime graph (rqt), and replay a recording (ros2 bag).

PARTICIPANTS

Various developers, robotics engineers, embedded systems engineers, and technical project managers.

PREREQUISITES

Strong knowledge of Python and C++.

TRAINER QUALIFICATIONS

The experts leading the training are specialists in the covered subjects. They have been approved by our instructional teams for both their professional knowledge and their teaching ability, for each course they teach. They have at least five to ten years of experience in their field and hold (or have held) decision-making positions in companies.

ASSESSMENT TERMS

The trainer evaluates each participant's academic progress throughout the training using multiple choice, scenarios, hands-on work and more.
Participants also complete a placement test before and after the course to measure the skills they've developed.

TEACHING AIDS AND TECHNICAL RESOURCES

• The main teaching aids and instructional methods used in the training are audiovisual aids, documentation and course material, hands-on application exercises and corrected exercises for practical training courses, case studies and coverage of real cases for training seminars.
• At the end of each course or seminar, ORSYS provides participants with a course evaluation questionnaire that is analysed by our instructional teams.
• A check-in sheet for each half-day of attendance is provided at the end of the training, along with a course completion certificate if the trainee attended the entire session.

2 Key Technical Concepts

- ROS Project Structure, URDF, Coordinate Systems, and TF2.
- RMW middleware, its implementations, and QoS control.
- The command-line interface (CLI).
- Tools for analysis and debugging: graph and data visualization (rqt, RViz2), log levels.
- Recording and Playback of Scenes (ros2bag)

Hands-on work
Manipulate existing ROS packages using the CLI to understand their architecture, inspect the runtime graph (rqt), and replay a recording (ros2 bag).

3 Developing Code with ROS

- Client libraries (RCL) for Python and C++.
- The Node, Lifecycle, and Composable node types, and their executors.
- The BT.cpp and Groot2 behavior trees.

Hands-on work
Create and compile simple Python and C++ nodes that use the RMW.

4 Integrating a ROS 2 System

- The characteristics of a system that is well integrated with ROS 2.
- Standard message types for common sensors (LIDAR, sonar, force/torque, etc.).
- The modularity of packages and message interfaces.
- Simulating sensors, actuators, and an environment using Gazebo.

Hands-on work
Observing the ROS 2 interface of the sensors and actuators of a simulated robot in the Gazebo simulator.

5 Nav2 Autonomous Navigation

- Navigation concepts: mapping, route planner, route tracker.
- Monte Carlo AMCL (Adaptive Monte Carlo Localization) Localization.
- Simultaneous Localization and Mapping (SLAM).

Hands-on work
Implementation of autonomous Lidar navigation controlled by a Python and C++ node.

6 Handling with the MoveIt 2 robotic arm and AI integration

- MoveIt 2 Architecture.
- Forward and inverse kinematics solvers, and optional constraints.
- Trajectory solvers with collision avoidance (OMPL, Open Motion Planning Library).
- Using code assistants to generate Python or C++ nodes.
- Integration of an AI algorithm into a node (DRL, TOD, DNN, etc.).

Hands-on work
Implementation of an arm controlled by a Python and C++ node.

TERMS AND DEADLINES
Registration must be completed 24 hours before the start of the training.

ACCESSIBILITY FOR PEOPLE WITH DISABILITIES
Do you need special accessibility accommodations? Contact Mrs. Fosse, Disability Manager, at psh-accueil@orsys.fr to review your request and its feasibility.

This programme is an original creation, developed by the teaching teams at ORSYS Formation. Any reproduction, representation, adaptation or use, in whole or in part, without the prior written authorisation of ORSYS, is strictly prohibited. ORSYS reserves the right to take any action necessary to protect its intellectual property rights.

Dates and locations

REMOTE CLASS
2026 : 14 Dec.

2027 : 31 Mar., 28 June, 13 Sep., 13 Dec.