+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 01/08/2026

Register

Jetbrains Rider Training

3 days (21 hours)

Overview

JetBrains Rider is a comprehensive IDE for developing in .NET (C#, ASP.NET Core) and working efficiently on multi-project solutions. It accelerates productivity through refactoring, advanced debugging, and integration of build and testing tools.

This training aims to make your development faster and more secure by mastering the key features of Rider: navigation, inspections, code generation, dependency management, and test execution. You will learn how to set up a consistent environment and standardize your practices on a team project.

The approach is 100% practical: guided demonstrations, workshops on a .NET solution, refactoring exercises, and debugging sessions. Deliverables include a configuration checklist, essential shortcuts, a workflow guide (Git, tests, run configurations), and a memo on optimizing IDE performance.

Objectives

- Configure Rider for a .NET project (SDK, plugins, settings).
- Navigate and analyze a solution with inspections and advanced search.
- Apply safe refactorings and automate code quality.
- Debug efficiently (breakpoints, watch, lightweight profiling).
- Run and diagnose tests, builds, and runtime configurations.

Target audience

- C#/.NET Developers
- ASP.NET Core developers (API, MVC)
- Unity developers using Rider
- Tech leads looking to standardize tools

Prerequisites

- Basic knowledge of C# and .NET concepts
- Understanding of Git and branch workflow
- Understanding of unit testing (xUnit/NUnit/MSTest)
- Familiarity with the command line

Technical prerequisites

- PC/Mac with at least 8 GB RAM (16 GB recommended)
- Windows, macOS, or Linux
- JetBrains Rider installed
- .NET SDK (version compatible with your projects) and access to a Git repository

JetBrains Rider training program

[Day 1 - Morning]

Getting started with Rider and configuring the .NET environment

- Installation and overview: solution, projects, explorer, tool windows
- Configuring the .NET SDK, templates, plugins, and essential shortcuts
- Understanding the structure of a solution: dependencies, references, NuGet
- Configuring the editor: formatting, inspections, code rules, and profiles
- Hands-on workshop: Create a .NET solution and apply a standardized editor configuration.

[Day 1 - Afternoon]

Navigation, productivity, and refactorings in C#

- Advanced navigation: symbol search, usages, hierarchy, recent files
- Code generation: constructors, properties, overrides, live templates
- Key refactorings: rename, extract method/interface, move, change signature
- Inspections and quick fixes: technical debt, nullability, simplification suggestions
- Hands-on workshop: Refactor an existing C# module using Rider inspections.

[Day 2 - Morning]

Debugging, testing, and code quality

- Rider debugger: conditional breakpoints, watch, evaluate, exceptions
- Execution analysis: threads, async/await, stack traces, and captured variables

- Running and organizing tests: NUnit/xUnit/MSTest, filters, runs, coverage
- Quality tools: inspections, code cleanup, duplication analysis
- Hands-on workshop: Diagnosing a bug and securing the fix with automated tests.

[Day 2 - Afternoon]

Integrated Git and team workflows

- Configuring Git in Rider: remotes, SSH, credential management
- Daily workflow: commit, amend, stash, interactive rebase, cherry-pick
- Resolving conflicts with the merge tool and advanced comparison
- Code review: diff, annotations, history, best practices for messages
- Hands-on workshop: Simulate a feature branch with rebase and conflict resolution up to merge.

[Day 3 - Morning]

Web development: ASP.NET Core, execution, and profiles

- Run/Debug configurations: profiles, environment variables, arguments, HTTPS
- Appsettings management: environments, secrets, user secrets
- Web productivity: controller/handler navigation, endpoints, Razor, JSON
- Tool integration: dotnet CLI, NuGet, restore/build/test from Rider
- Hands-on workshop: Configure and run an ASP.NET Core API with multiple environments (Dev/Staging).

[Day 3 - Afternoon]

Performance, profiling, and best practices for delivery

- Identifying hotspots: logs, traces, contention points, allocations
- .NET profiling: CPU, memory, snapshots, and interpreting results
- Cleanup and standardization: code style, analyzers, EditorConfig, warnings as errors
- Preparing for delivery: command line build, artifacts, reproducibility
- Hands-on workshop: Profiling a .NET application, fixing a performance bottleneck, and validating by measurement.

Target companies

This training is intended for both individuals and companies, large or small, wishing to train their teams in a new advanced IT technology or to acquire specific business knowledge or modern methods.

Positioning at the start of training

The positioning at the start of the training complies with Qualiopi quality criteria. Upon final registration, the learner receives a self-assessment questionnaire that allows us to assess their estimated level of knowledge of different types of technologies, their expectations and personal objectives for the upcoming training, within the limits imposed by the selected format. This questionnaire also allows us to anticipate certain connection or internal security issues within the company (intra-company or virtual classroom) that could be problematic for the monitoring and smooth running of the training session.

Teaching methods

Practical training: 60% practical, 40% theory. Training materials distributed in digital format to all participants.

Organization

The course alternates between theoretical input from the trainer, supported by examples and discussion sessions, and group work.

Assessment

At the end of the session, a multiple-choice questionnaire is used to verify that the skills have been correctly acquired.

Certification

A certificate will be issued to each trainee who has completed the entire training course.