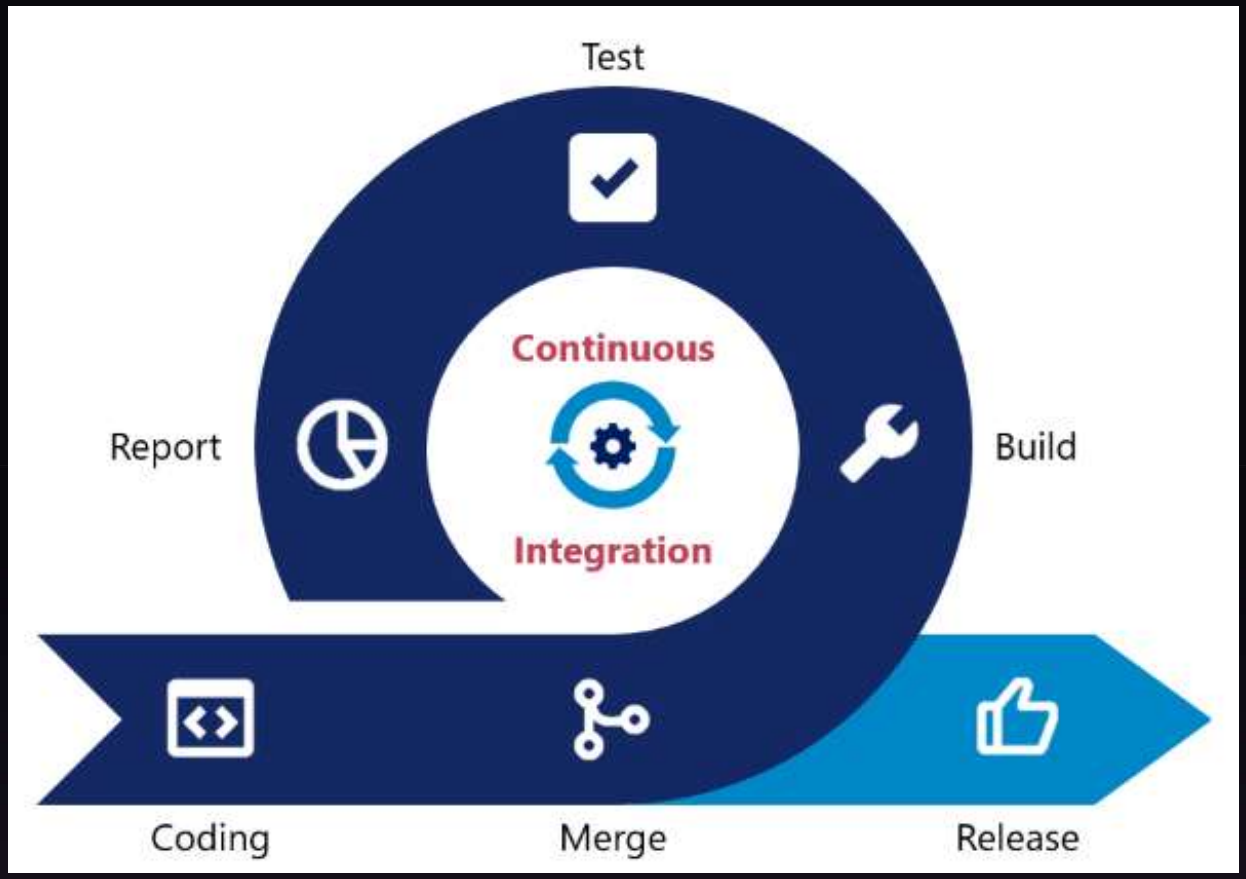

LearnYard

LearnYard

Continuous Integration



Source: https://unstop.com/blog/what-is-git

LearnYard

INTRODUCTION

Continuous Integration (CI) is a software development practice where developers regularly merge their code changes into a shared repository, usually multiple times a day. Each merge triggers an automated process to build and test the code, ensuring that new changes integrate smoothly with the existing codebase.

WHY IT IS NEEDED

- Early Detection of Errors: By integrating regularly, developers can detect and fix integration issues early, preventing them from escalating into larger problems.
- Automation of Testing: CI automates the testing process, ensuring that code quality remains high and that any bugs are caught early.
- Faster Development Cycles: CI allows for faster iterations by providing quick feedback to developers, leading to more agile and efficient development processes.
- Reduced Integration Problems: Regular integration reduces the complexity and time spent on integrating code at later stages.

HOW CI WORKS

- Version Control System (VCS): Developers commit their code to a shared repository (e.g., GitHub, GitLab).
- CI Server: A CI server (e.g., Jenkins, GitHub Actions) monitors the repository. When changes are detected, the CI server automatically triggers a series of tasks.
- Build Automation: The code is built, which includes compiling the code and generating artifacts.
- Automated Testing:

- Unit Tests: These tests check individual components or functions to ensure they work as expected.
- Integration Tests: These tests verify that different parts of the application work together as intended.
- End-to-End Tests: These tests simulate real user scenarios to ensure the system as a whole behaves correctly.

- Feedback: The results of the build and tests are reported back to the developers, usually in the form of success or failure notifications.

BENEFITS OF CI

- Improved Code Quality: Automated testing ensures that the codebase remains stable and bug-free.

- Faster Delivery: With CI, code changes can be deployed more quickly and frequently, leading to faster delivery of new features and fixes.

- Collaboration: CI encourages collaboration among team members by ensuring that everyone's code works well together.

- Reduced Risk: Early detection of issues reduces the risk of larger problems arising later in the development cycle.

CI WORKFLOW WITH GITHUB ACTIONS

- Triggering CI: CI can be triggered by various events, such as code push, pull request, or even on a schedule.

- Configuration Files: CI is configured using YAML files (e.g., `.github/workflows/ci.yaml`) that define the tasks to be performed.

- Steps in CI Pipeline:

  - Checkout Code: The first step is to checkout the code from the repository.
  - Set Up Environment: Install dependencies, set up the runtime environment, etc.
  - Run Tests: Execute the tests defined for the project.
  - Build Artifacts: If the tests pass, build the necessary artifacts for deployment.
  - Deploy: Optionally, deploy the code to a staging or production environment.

WHAT IS `pytest` TESTING IN THE `ci.yaml` FILE?

`pytest` Testing in the `ci.yaml` File: In the `ci.yaml` file provided for Continuous Integration (CI) using GitHub Actions, the `pytest` command is used to run tests on your Python code. `pytest` is a testing framework in Python that automatically discovers and runs tests defined in your codebase. When you run `pytest`, it looks for files that start with `test_` or end with `_test.py`, and it executes all the test functions inside those files.

In the context of the `ci.yaml` file:

- Unit Tests: If your project has unit tests, `pytest` will run them. Unit tests check individual units of your code (like functions or classes) to ensure they behave as expected.

- Integration Tests: If your project includes integration tests, `pytest` will run those as well. Integration tests verify that different parts of your application work together correctly.

- Test Discovery: `pytest` automatically discovers test files and functions, so you don't need to manually specify each test.

LearnYard

LearnYard