# CI/CD Overview

Context: What is CI/CD?

CI/CD is continuous integration/continuous deployment. It means integrating code changes/feature development into the main branch continuously as the team produces it, rather than all at once. Continuous deployment means “shipping” the changes by putting them into a deployable environment continuously as soon as the team has validated them.

## GitHub Actions

### What?

* Automate build, test, and deployment of features
* Github’s play at CI/CD functionality

### Why?

* Very easy to use/setup
* Easy to pick up if you already know Git (most people do)
* Tight, seamless integrations with GitHub; every Github event (for example, push, PR, etc.) can be linked to an GitHub Action – **is your source code already in Github?**
* Pay as you go pricing (similar to AWS cloud)
* Build a pilot with no cost; if you like it, can upgrade to a paid version with more minutes
* Virtual machine can be easily setup in any configuration
* Huge, dynamic community marketplace of “Git Actions” – use other “Actions” written by the community without needing to re-invent the wheel
* Github Actions is a fully managed service by GitHub, so you don’t need to know how to scale and operate the infrastructure to run it. [vs Jenkins]

# Other CI/CD Alternatives

## Jenkins:

### What?

* A free, open-source CI/CD automation server. Helps automate the parts of software development related to building, testing, and deploying to facilitate CI/CD.

### Challenges vis a vis Git Actions:

* Need to know how to scale and operate the infrastructure to run it
* Keep the plugins up-to-date
* Running a Jenkins server costs money even if you are not running any builds
* Reliance on plugins that come with updates that need to be installed from time to time
* Jenkins offers a sequential pipeline (one job after the other) vs. GitHub Actions which can run multiple jobs in parallel using the master-slave (coordinator and build node) architecture.
* Jenkins relies on a single server environment to run all the jobs, while, GitHub Actions relies on separate VM’s runners in parallel

**Key tradeoff: Managed vs unmanaged platform**

### References:

* <https://blog.bitsrc.io/github-actions-or-jenkins-making-the-right-choice-for-you-9ac774684c8>

## Teamcity

### What?

* Another on-premise CI/CD tool offered by JetBrains

### Why?

* Onpremise service; GitHub Actions is a cloud SAAS service
* Teamcity you **p**ay per ‘agent’; in GitHub, you pay per minute of usage.
* Teamcity has predictable pricing; GitHub has less predictable pricing, but there are maximum tier limits involved.

### References:

* <https://stackshare.io/stackups/github-actions-vs-teamcity>
* <https://knapsackpro.com/ci_comparisons/github-actions/vs/teamcity>

## Octopus

What?

Octopus Deploy is the first platform to enable your developers, release managers, and operations folks to bring all automation into a single place.

Live demo!!