

What is CI?

- CI stands for Continuous Integration
 - It is the practice where members of a team integrate their work frequently (usually at least once a day), and each integration is verified by an automated build (including tests!) to detect integration or other errors as quickly as possible.

The goal of CI is to quickly identify issues that might arise from multiple people working on multiple different parts of an application.

Think of it as very frequent communication between your code and everyone else's code.

Why CI?

- Could have helped mitigate NASA's infamous metric vs. imperial units issue with the 1999 Mars Orbiter
- Another easy example is a front end team and a back end team.
 - Your product leader decides on a design, delegates your PMs or EMs
 - Each team then goes off and builds to spec, but to their version of spec
 - A few months pass, each team completes their part, and now it is time to integrate them!
 - Easy for front end team to make different assumptions about the shape of the data than the backend team.

What is CD?

- CD stands for Continuous Delivery
 - It is a discipline where software is built in such a way that it can be released into production at any point in time.

CD is achieved through use of CI

CD is in place when:

- Your software can be deployed at any point in time
- Deployment is prioritized over new feature development
- Anyone can get fast, automated feedback on the production-readiness of their software whenever a change is made
- You can easily and on-demand do “push button: deployment of any version of the software

So what's the difference?

- CI allows for easily testing in-progress work, either between different boundaries, or between different layers in the software
 - Remember, “integration” is sort of a fuzzy term
- CD integrates CI as part of its practice, but isn't implemented by just having CI in place.

In general: CI is used pretty widely, where CD is something a specific team will choose to opt in to, and implemented CI does not mean you're now able to implement CD.

Popular tools used for CI/CD.

- [Jenkins](#)
- [Travis CI](#)
- [Circle CI](#)
- [Bamboo](#)

Examples of Interview Questions

- How is software deployed?
- How would you ensure software does not get released if tests do not pass?
- What is test coverage?

Vocabulary


Words you may hear in the job or work environment

- Coverage
- Deploy gates
- CI/CD
- Jobs
- Workers
- Builds

Live Coding Demo

- Testing Pull Requests automatically
- Preventing Pull Requests from merging that fail tests
- Deploying a web app automatically on merges to master

Exercise

- In Week 1 you wrote tests for this git repository:
<https://github.com/ahfarmer/emoji-search>. Use your fork of it to set up continuous integration with <https://travis-ci.org>
- Follow the directions in TravisCI's documentation for running builds on a nodejs app. <https://docs.travis-ci.com/user/languages/javascript-with-nodejs/>
- After you have successful builds on master change the settings for your repository to disallow merges to master without passing PR checks.
- Lastly add a badge on your project's readme that reflects whether the latest build on master is passing If a build fails the badge should change red. It looks like this: 

Suggested Reading

- <https://docs.travis-ci.com/user/for-beginners/>
- <https://travis-ci.com/plans>
- <https://help.github.com/en/articles/about-protected-branches>
- <https://martinfowler.com/articles/continuousIntegration.html>
- <https://martinfowler.com/bliki/ContinuousDelivery.html>
- <https://martinfowler.com/bliki/DeploymentPipeline.html>
- https://en.wikipedia.org/wiki/The_Mythical_Man-Month
- <https://www.simscale.com/blog/2017/12/nasa-mars-climate-orbiter-metric/>
-