Continuous Integration

By this time, you should be aware and comfortable with automated tests and the benefits. Up until now, we're relying on developer's discipline to always ensure tests pass locally before they deploy features. This is probably ok in small and high-trust teams, but in larger teams you typically see Continuous Integration (CI) introduced as part of the development process.

In his article on Continuous Integration, Martin Fowler defines continuous integration as:

> *Continuous Integration is a software development practice where members of a team integrate their work frequently, usually each person integrates at least daily - leading to multiple integrations per day.*

The problem that CI solves is to **force** integration on a regular basis. For example, you may have just finished a feature on `my_feature` branch and have had all the tests pass locally. You open a PR and your fellow coworkers are happy with the way the code looks. Github shows the PR can be automatically merged without conflict - so you do that, merge it back to the `master` branch and ready to deploy the feature!

The danger of this workflow is that while you are working on your feature, someone else could have just finished their feature and pushed to the master branch to Github, and your new feature never integrated with this new piece of code.

A Continuous Integration solution would watch your repository and force integration (running the entire test suite on the CI server) whenever necessary.

Once we have our CI server set up to monitor our repository, it'll pull the latest code and run the entire test suite every time we push code to any branch on Github (when you work with a CI server, you want to make sure that every push should always make the tests pass) and notify you on the results. The CI server also runs every time we merge our pull request to the `master branch` to ensure integration.

A CI server helps us catch integration errors before they reach production, and the earlier we catch errors, the less costly we can fix them.

A CI solution doesn't eliminate the necessity to run your tests locally - your will lose friends quickly if you constantly push code that "breaks the build" and have everyone notified. However, in cases of long running test suites, it is acceptable to run only the tests pertaining to the new feature built, and have the CI server to run the entire test suite to catch any potential regression. Most modern CI services allow you to run the tests in parallel (with a paid plan, typically) so it could run your tests several times faster.

You marked this topic or exercise as completed.