Software Testing, Documentation, and Licensing

Overview

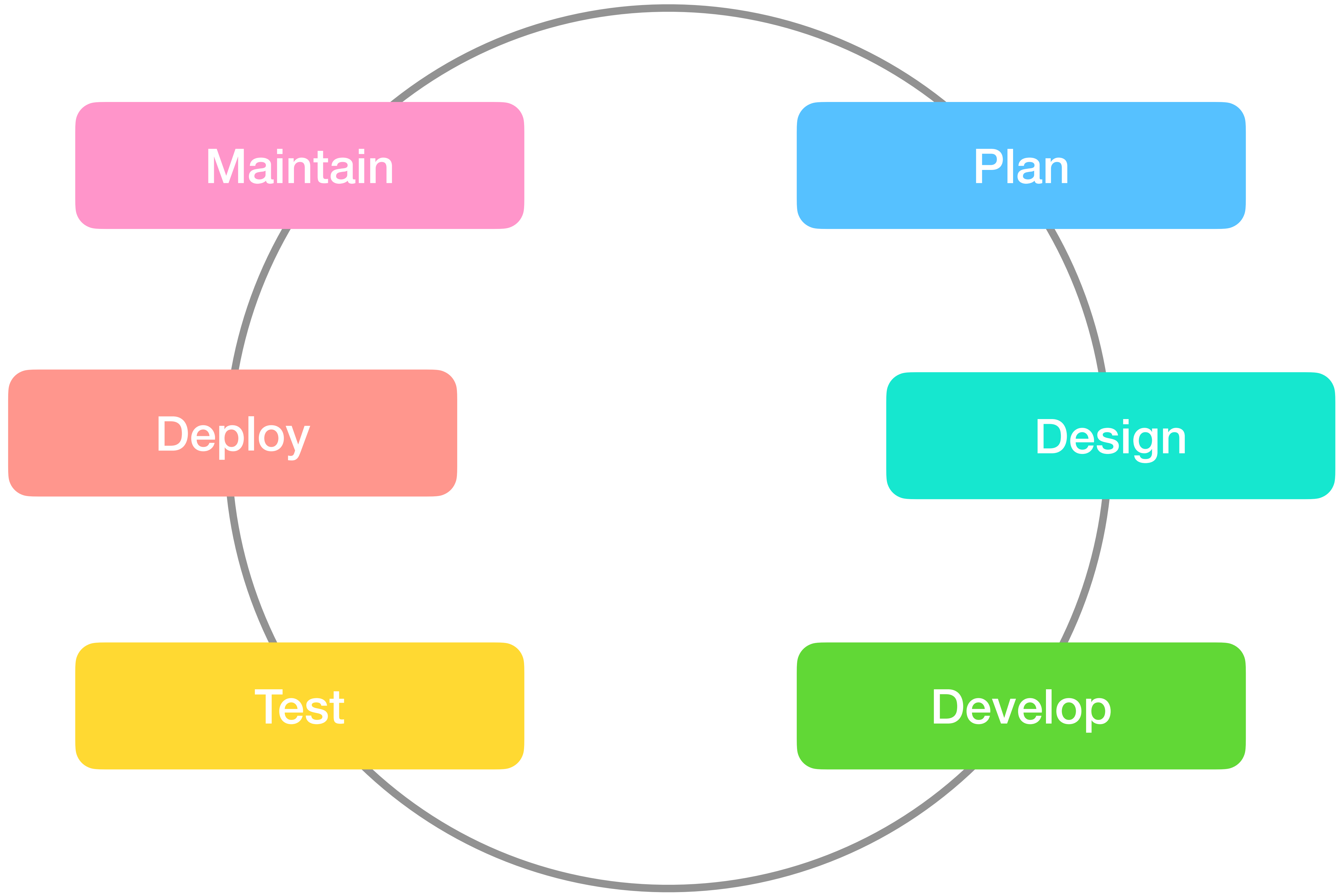
- You’ve written your code, it runs and works, and you are done, right?
- Nope! To effectively share your code in a way that will be lastingly useful, you also need to test and document.
- These are not just “overhead” tasks — they are a core part of software engineering, and code that lacks these properties is essentially impossible to maintain or build on in the long term.
- Last but not least, you need to choose an appropriate license for your code, and make sure you understand the licenses of your dependencies and the ecosystem in general.

Objectives

- Understand approaches toward and the purpose of software testing, and be able to write basic unit tests.
- Read and write quality comments, Pydoc, and READMEs.
- Recognize major open-source licenses and their significance for personal and professional use.

SDLC

- Software Development Life Cycle
    - A process used to produce high quality software.



Software Testing

- When you say “the code works”, what do you *really* mean?
- Generally you mean that, when the system/function is run with given input, the output/behavior is as expected.
- Software testing formally specifies this, and provides a framework for verifying that code really passes the tests.
- This helps you avoid *regressions-* "going backwards” with your code.
- The purpose is to catch bugs/defects and improve the quality of the code/product.

Types

- Functional Testing
    - Unit Testing
    - Integration Testing
    - System Testing
    - UI Testing
    - Regression Testing
    - User Acceptance Testing

Types

- Non-Functional Testing
    - Performance Testing
        - Load Testing
        - Stress Testing
    - Security Testing

How

- Manual Testing
- Automated Testing

Unit Testing

- The simplest possible test requires the simplest possible piece of code — a *unit*.
- What is a unit?
    - A function, a method, a class
    - For larger/more complicated code, it may be you write different unit tests for different cases of a function call, each passing different arguments in and testing for expected output.
- Unit tests are the most basic, well, unit of testing.
    - You can have “100% unit test coverage” and still miss things if you don’t test things combined.

Unit Testing

- They force you to think of your code in *units*.
    - A good unit test requires good code to test, and so you may find yourself refactoring your code in order to make it more testable.
- Embrace this! It’s one of the biggest advantages of proper software testing.

Documentation

- Documentation plays a particularly key role.
- It’s where you write those things that the code itself doesn’t quite say, but you had to think through to be able to come up with it.
- You read more code than you write - So when you write code, you should always remember that it’s not good enough for it to just *run*.
- It’s also important that your code can be *read* — that is, understood by another human, be it a coworker, or just you but in the future.
- It requires great focus to write complicated code, and you’ll quickly forget the details.
- Comments, Pydoc, and READMEs are how we save our human mental state, and share it with whoever works with our code in the future.

Software Licenses

- Protect our intellectual property
- Proprietary
- Open Source
  - There are two major “schools” of open source licenses - GPL and BSD/MIT.

GPL

- The GPL (General Public License, part of the GNU Project with Richard Stallman) is the more “aggressive” of the two.
- It takes the stance that source code shouldn’t just be available, but that people who use open source code should also make their source code available.
- This is referred to as “copyleft”.

BSD/MIT

- BSD/MIT-flavored licenses take a different approach
    - They put code out there, and presume that the original code writer isn't liable for bad things (i.e. you can't sue them),
    - And that you should include the license with your code and acknowledge the original author (but not necessarily release your own code the same way).