Week 9 Agenda

Week 9 Additional programming concepts

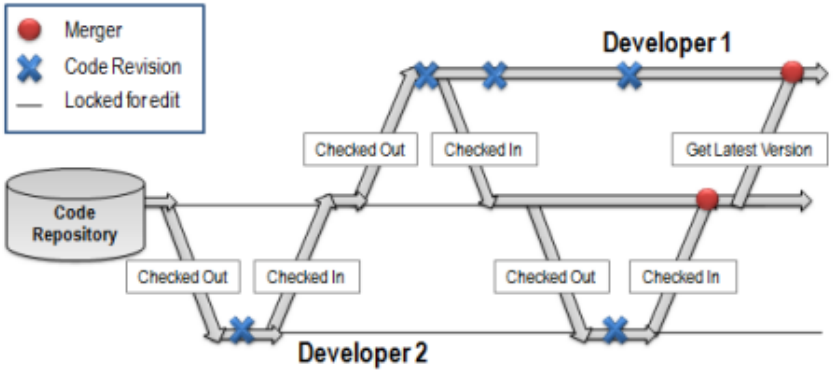
- Source Control Management
- Testing your software
- Software security topics
- Peer Reviews

Chapter 11 in Python Crash Course: A Hands-on, Project-Based Introduction to Programming

Source Control Management (SCM)

SCM solutions allow for efficient collaboration between members of a software development team by managing the source code of the project. SCM solutions provide file management and versioning of software artifacts.

A common aspect of SCM solutions is that they provide the ability of developers to work on different “branches” of the code base and when ready to merge those branches into a release. Below is a diagram which displays how SCM solutions work, courtesy of http://www.ni.com/white-paper/4114/en/.



Testing Your Software

Testing software helps ensure that a quality software solution which meets customer requirements is delivered. Software testing is used to reduce risk and improve software quality by finding defects.

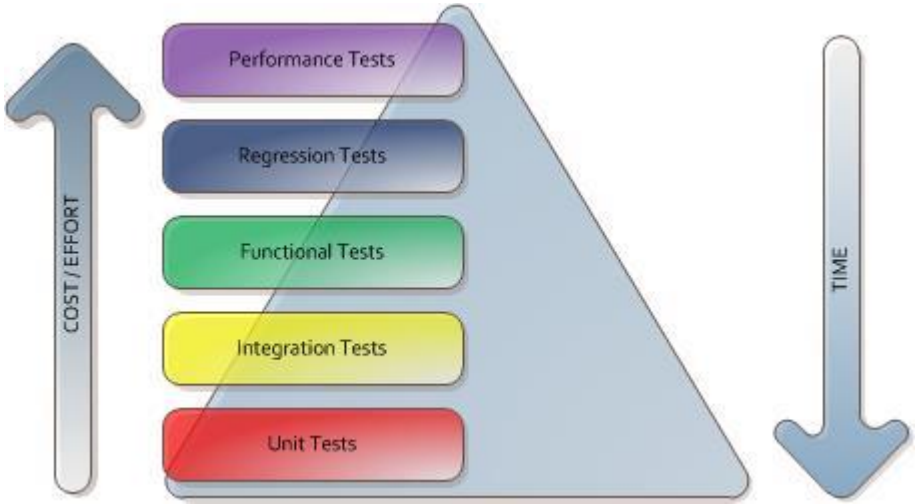
There are multiple forms of testing that take place during the software development lifecycle including:

- Unit Testing
- Integration Testing
- Regression Testing
- Performance Testing

In most environments development staff are responsible for unit and integration testing. Functional, regression, and performance testing are often performed by the Quality Assurance group (QA).

It is important to keep in mind that the further along in the development lifecycle we are the more expensive it is to fix vulnerabilities. For this reason unit testing is extremely important since it detects issues with the code very early in the software development lifecycle.

Testing Pyramid



Unit Testing

Unit testing is typically performed by the developer prior to checking their code into the source code management solution or automatically using a unit testing framework such as ***unittest*** for Python. Unit testing attempts to test the application's functionality at a basic level.

Unit testing attempts to test each individual unit of code in order to give you an adequate level of confidence that the application will function as intended once integrated. Fixing an issue before it's deployed to QA or Production environments reduces the cost of fixing identified issues.

Integration Testing

Integration testing often takes place once multiple software components are combined (a.k.a integrated). The idea of integration testing is to take the modules that were unit tested, combine them into the overall solution and run integration testing to test the overall application as a whole. This helps identify issues from a system wide perspective including the interface of the application.

Integration testing can take place between multiple components of the same software application or multiple applications which integrate. Integration testing is often performed by the development team but may also be performed by the quality assurance group.

Functional Testing

Functional testing is a form of testing which focuses on certain features or functional aspects of the application. For instance, in a stock trading application one functional test may focus solely on the stock buying function of the application.

In order to properly perform functional testing you typically follow this process:
- Use test data to identify inputs
- Determine what the expected outcome should be based on those inputs
- Run the test cases with the proper inputs
- Compare the expected results to the actual results

Regression Testing

Regression testing is a way to ensure that software which was previously developed, tested, and released still functions as intended once updates are made to the solution as a whole. Regression tests are often completed at the end of the development lifecycle right before delivering an application to production environments and may only take place after significant code rewrites.

The purpose of regression testing is to make sure that the application works as expected or to ensure that major code changes don't negatively affect the application's functionality. Other forms of testing may focus on testing specific changes or specific functionality.

Performance Testing

Performance Testing is typically performed on a regular basis to ensure that an application can meet real world conditions. Performance testing is typically performed on applications which must be capable of processing large volumes of data/transactions (i.e. credit card processing, stock trading solutions). Performance testing consists of multiple types of tests including, straight performance tests, load test, stress test, and capacity test.

Software Security

Software security is a discipline of software engineering that focuses on delivering secure software. Software Security takes place throughout the software development lifecycle and consists of activities such as design reviews, threat modeling, static analysis, dynamic analysis, and penetration testing. Many of these processes are dependent on the organization.

Software Security focuses on making sure that best practices to prevent software vulnerabilities are followed and that when vulnerabilities are located within software they are mitigated before an attacker can exploit this weakness.

There are various tools involved with securing software that focus on vulnerability detection and vulnerability mitigation including static/dynamic analysis tools, Web Application Firewalls (WAF), and Runtime Application Self-Protection.

Software Security focuses heavily on securing web and mobile applications. Even systems that aren't directly connected to the internet typically interact with systems that are connected to the internet and should be coded securely.

Common software vulnerabilities are detailed in the OWASP Top 10

OWASP Top 10 (2017)

OWASP Top 10 - 2017
A1:2017-Injection
A2:2017-Broken Authentication
A3:2017-Sensitive Data Exposure
A4:2017-XML External Entities (XXE) [NEW]
A5:2017-Broken Access Control [Merged]
A6:2017-Security Misconfiguration
A7:2017-Cross-Site Scripting (XSS)
A8:2017-Insecure Deserialization [NEW, Community]
A9:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging&Monitoring [NEW,Comm.]

OWASP Python Security Project

Python Security is a free, open source, OWASP project that aims at creating a hardened version of python that makes it easier for security professionals and developers to write applications more resilient to attacks and manipulations.

The project is designed to explore how web applications can be developed in python by approaching the problem from three different angles:

- Security in python: white-box analysis, structural and functional analysis
- Security of python: black-box analysis, identify and address security-related issues
- Security with python: develop security hardened python suitable for high-risk and high-security environments

More information is available here: http://www.pythonsecurity.org/

Peer Reviews

Peer reviews take many forms including code inspection, team reviews, code walkthroughs, and peer programming. Regardless of the form of peer review the fundamental concept is to allow other members of the development team to review code from team members before the code is delivered.

Using peer reviews in the development process can help ensure that software requirements are followed, software vulnerabilities don't exist, development standards are followed and that the code fits the needs of the business.

Peer review results should be documented and communicated to the developer responsible for the changes being reviewed.

References

The Basics of Compiled Languages, Interpreted Languages, and Just-in-Time Compiliers, https://www.upwork.com/hiring/development/the-basics-of-compiled-languages-interpreted-languages-and-just-in-time-compilers/

The End

You may close this Window and return to the course.