ProTech protechtraining.com *"Charting the Course ... ... to Your Success!"*

Introduction to Software Testing and Quality Management

Course Summary

Description

This is a three day overview course designed to provide students with a foundational understanding of the concepts, principles, and techniques of software testing and quality assurance. The course is lecture based with simple hands on examples, worked exercises, and breakout discussions. Specific techniques and topics – like test management, functional testing techniques, and software quality maturity – are covered in a unified manner but the mastery of specific techniques and skills are deferred to other more intensive courses on specific topics.

Objectives

By the end of this course, students will be able to:

- Defining software testing, quality control, assurance and management and how they are interdependent
- Quality management, good enough quality and risk analysis
- Quality control: setting testing goals and objectives – the fundamental dilemma: thoroughness versus cost and time constraints
- The three main areas of software testing – structural testing, functional testing and process testing
- Good versus bad testing – test validity, accuracy and reliability
- The psychology of testing: the testing mindset and software testing protocols and principles
- Fault models and test heuristics
- Levels of testing: Unit, Integration, System, and Acceptance
- Black box testing techniques: domain analysis, boundary analysis, Karnuagh-Veitch matrices, cause and effect graphing, decision tables and combinatorial testing
- White box testing techniques: flow analysis, path and decision coverage, data flow and transaction flow testing . Exploratory testing
- Model based testing: logic models, state models, and data model testing
- Testing metrics: coverage, confidence, completeness
- The testing process, managing the testing lifecycle, the testing maturity model
- Just in time testing, testing throughout the development lifecycle
- Quality Assurance and lifecycle testing in an Agile environment
- Integrating development and testing: test driven development and acceptance test driven development.
- Peer reviews: working with code inspections, design reviews and code walkthroughs
- Test management and documentation: testing as project management
- Testing standards and certifications
- Current trends in software quality and testing

Topics

- Review of Basic Testing Terms and Concepts
- Review of Basic Testing Techniques
- Object Oriented Development
- The Java Language and Architecture
- Java Structured Code
- The Java Class and Program Structure
- Java Classes, Objects, Instances and References
- Testing Java Class Design
- Inheritance
- Testing inheritance
- Exceptions and Exception Handling and Assertions
- Testing Exception Handling
- Java APIs
- The Java Class Libraries
- Code Smells
- Test Driven Development
- Acceptance Test Driven Development
- Code Reviews and Code Walkthroughs
- Java Standards and Best Practices

Introduction to Software Testing and Quality Management

Course Summary (cont'd)

Audience

This course is intended for those who need a broad integrated introduction into the fields of software testing and quality assurance. No prior knowledge of the subjects is assumed. The course is a prerequisite for the more advanced courses in the testing and quality curriculum but is also appropriate for non-testers who need a general understanding of the subject matter.

Prerequisites

There are no prerequisites for this course.

Duration

Three days

ProTech protechtraining.com *"Charting the Course ...*

... to Your Success!"

Introduction to Software Testing and Quality Management

Course Outline

I. Review of basic testing terms and concepts
An introductory module that is intended to establish a common baseline of terms, concepts and ideas that will be used in the rest of the course. Topics reviewed are coverage, types of testing, basic testing principles. The testing concepts of verification versus validation, completeness, test sensitivity, etc. This and the following module are not intended to teach the testing topics but rather are for the purpose of establishing a common baseline for the rest of the material.

II. Review of basic testing techniques.
Review of test execution, setting testing goals and integration into the development cycle. Testing within the waterfall and agile development processes. Role of the tester and quality with respect to developer. Planning the testing program

III. Object Oriented Development.
Basic review of the OO development paradigm. How it differs from the structured paradigm (under which software testing was developed as a discipline). Differences in OO versus structured development and systems design and how software testing has modified and extended standard practices and techniques to accommodate those differences.

IV. The Java language and architecture.
Introduction to how the Java language works. The virtual machine, JRE, and class libraries. The distributed and modular architecture of the language and the importance of configuration testing for Java apps. Overview of the "Java ecosystem" and the sandbox model. The Java philosophy and process for managing the distributed model. Common sorts of problems that developers overlook.

V. Java Structured code
In this module we look at the parts of the java programming language that are similar to most other programming languages: operators, data types, variables, scope. Specific features of Java that are "error prone" such as the difference between the bitwise and logical operators are covered. The basics of loops and flow control are reviewed as well as the specific techniques for testing those constructs.

VI. The Java Class and program structure
This module looks at the basic structure of Java classes focusing on instance variables and methods. Good java design techniques are covered such as designing to interfaces, data hiding, encapsulation, and the proper design of methods. Static methods and variables are discussed.

VII. Java Classes, objects, instances and references
In this module we look at the dynamics of using objects in Java: how they are created and referred to with reference variables. We look at the typical errors introduced by shallow versus deep copies and the sorts of typical problems that we see with reference variables, initialization and object versus class scoping of methods and variables. Specifically, we look at test strategies that allow for us identify and detect the sorts of errors that often are introduced by developers in managing object instances.

VIII. Testing Java Class Design
Continuing from the previous module, we examine typical ways Java classes can be poorly designed, and the impact that has on the quality of code. Examples are broken interfaces, bad choices in parameters and return values, and inappropriate modularization. Emphasis is on looking for the code smells that suggest structural problems and test strategies to revel those problems.

IX. Inheritance.
One of the major issues in moving to OO testing is dealing with the errors that are introduced by the bad use of inheritance. The basics of inheritance in OO languages, with emphasis on Java, is covered along with the best practices for use of inheritance. Bad inheritance design is a very common source of errors but is also one of the more subtle sources of errors as well. This module covers the mechanics of inheritance, overriding methods, shadowing variables, working with superclass methods and variables and the standard best practices for these (for example, this Liskov substitution principle).

X. Testing inheritance
Continuing from the previous module, this one covers the ways to explore and test for problems in inheritance, and how these can interact with the problems also covered in previous modules on testing class design.

Introduction to Software Testing and Quality Management

Course Outline (cont'd)

XI. Exceptions and exception handling and assertions

This module covers the Java exception handling mechanism and the best practices on how it should be used. The module covers the effective use or try-throw-catch blocks, assertions and exception classes and hierarchies. The problems of nested try blocks and re throwing exceptions are covered.

XII. Testing Exception Handling

Continuing from the previous module, this covers planning a set of exception tests and testing for exception failures, especially in complex exception handling cases.

XIII. Java APIs.

This module is a high level overview of the Java collections API and the Java Reflection API. Typical programmer problems encountered with these APIs are covered as well as suggested test methods.

XIV. The Java Class Libraries

An overview of the main Java class libraries: IO, RMI, Networking, JDBC etc. This module gives the testers a sense of the functionality of each module, typical patterns of use and typical types of testing one would want to cover for the most common types of errors in library use.

XV. Code Smells

This is an introduction into the Test Driven Development and refactoring idea of Code Smells, which are identifiable structural characteristics that usually illustrate poorly designed code. The occurrence of code smells can often identify specific areas of code to test and what sort of testing may be most useful at identifying code problems.

XVI. Test Driven Development

This module is an introduction to the procedures, techniques, and tools of TDD with typical examples of how it is used by developers to write Java code. The emphasis of the module is on the role of the tester and testing practices in TDD and how the quality of the tests used directly affect the quality of the code produced.

XVII. Acceptance Test Driven Development

Similar in structure and approach to the previous module, this module introduced the processes, tools and techniques or ATDD and the role of the tester in developing the acceptance tests using Gherkin as part of the development process

XVIII. Code Reviews and Code Walkthroughs

Peer reviews are an integral part of development processes for producing high quality code. How these work and how they are structured are covered in the module but from the perspective of the role of the tester and the optimal way testers can contribute when included in the review process.

XIX. Java Standards and Best Practices

There are a wide collection of Java standards and quasi-standards out in the Java community such as Java EE, Spring, Servelets, and many more. This module is a survey of what these are and what established best practice standards exist for the overall Java community