

The Art Of Unit Testing With Examples In Net

Fundamental testing methodologies applied to the popular Pythonlanguage Testing Python: Applying Unit Testing, TDD, BDD andAcceptance Testing is the most comprehensive book available ontesting for one of the top software programming languages in theworld. Python is a natural choice for new and experienceddevelopers, and this hands-on resource is a much needed guide toenterprise-level testing development methodologies. The book willshow you why Unit Testing and TDD can lead to cleaner, moreflexible programs. Unit Testing and Test-Driven Development (TDD) are increasinglymust-have skills for software developers, no matter what languagethey work in. In enterprise settings, it's critical for developersto ensure they always have working code, and that's what maketesting methodologies so attractive. This book will teach you themost widely used testing strategies and will introduce to you tostill others, covering performance testing, continuous testing, andmore. Learn Unit Testing and TDD—important developmentmethodologies that lie at the heart of Agile development Enhance your ability to work with Python to develop powerful,flexible applications with clean code Draw on the expertise of author David Sale, a leading UKdeveloper and tech commentator Get ahead of the crowd by mastering the underappreciated worldof Python testing Knowledge of software testing in Python could set you apart fromPython developers using outmoded methodologies. Python is a naturalfit for TDD and Testing Python is a must-read text foranyone who wants to develop expertise in Python programming. Master high quality software development driven by unit tests About This Book Design and implement robust system components by means of the de facto unit testing standard in Java Reduce defect rate and maintenance effort, plus simultaneously increase code quality and development pace Follow a step-by-step tutorial imparting the essential techniques based on real-world scenarios and code walkthroughs Who This Book Is For No matter what your specific background as a Java developer, whether you're simply interested in building up a safety net to reduce regressions of your desktop application or in improving your server-side reliability based on robust and reusable components, unit testing is the way to go. This book provides you with a comprehensive but concise entrance advancing your knowledge step-wise to a professional level. What You Will Learn Organize your test infrastructure and resources reasonably Understand and write well structured tests Decompose your requirements into small and independently testable units Increase your testing efficiency with on-the-fly generated stand-in components and deal with the particularities of exceptional flow Employ runners to adjust to specific test demands Use rules to increase testing safety and reduce boilerplate Use third party supplements to improve the expressiveness of your verification statements In Detail JUnit has matured to become the most important tool when it comes to automated developer tests in Java. Supported by all IDEs and build systems, it empowers programmers to deliver software features reliably and efficiently. However, writing good unit tests is a skill that needs to be learned; otherwise it's all too easy to end up in gridlocked development due to messed up production and testing code. Acquiring the best practices for unit testing will help you to prevent such problems and lead your projects to success with respect to quality and costs. This book explains JUnit concepts and best practices applied to the test first approach, a foundation for high quality Java components delivered in time and budget. From the beginning you'll be guided continuously through a practically relevant example and pick up background knowledge and development techniques step by step. Starting with the basics of tests organization you'll soon comprehend the necessity of well structured tests and delve into the relationship of requirement decomposition and the many-faceted world of test double usage. In conjunction with third-party tools you'll be trained in writing your tests efficiently, adapt your test case environment to particular demands and increase the expressiveness of your verification statements. Finally, you'll experience continuous integration as the perfect complement to support short feedback cycles and quality related reports for your whole team. The tutorial gives a profound entry point in the essentials of unit testing with JUnit and prepares you for test-related daily work challenges. Style and approach This is an intelligible tutorial based on an ongoing and non-trivial development example. Profound introductions of concepts and techniques are provided stepwise as the programming challenges evolve. This allows you to reproduce and practice the individual skills thoroughly.

Revised and updated with improvements conceived in parallel programming courses, The Art of Multiprocessor Programming is an authoritative guide to multicore programming. It introduces a higher level set of software development skills than that needed for efficient single-core programming. This book provides comprehensive coverage of the new principles, algorithms, and tools necessary for effective multiprocessor programming. Students and professionals alike will benefit from thorough coverage of key multiprocessor programming issues. This revised edition incorporates much-demanded updates throughout the book, based on feedback and corrections reported from classrooms since 2008 Learn the fundamentals of programming multiple threads accessing shared memory Explore mainstream concurrent data structures and the key elements of their design, as well as synchronization techniques from simple locks to transactional memory systems Visit the companion site and download source code, example Java programs, and materials to support and enhance the learning experience

The classic, landmark work on software testing The hardware and software of computing have changed markedly inthe three decades since the first edition of The Art of SoftwareTesting, but this book's powerful underlying analysis has stoodthe test of time. Whereas most books on software testing targetparticular development techniques, languages, or testing methods,The Art of Software Testing, Third Edition provides a briefbut powerful and comprehensive presentation of time-proven softwaretesting approaches. If your software development project is missioncritical, this book is an investment that will pay for itself withthe first bug you find. The new Third Edition explains how to apply the book'sclassic principles to today's hot topics including: Testing apps for iPhones, iPads, BlackBerrys, Androids, andother mobile devices Collaborative (user) programming and testing Testing for Internet applications, e-commerce, and agileprogramming environments Whether you're a student looking for a testing guide you'll usefor the rest of your career, or an IT manager overseeing a softwaredevelopment team, The Art of Software Testing, Third Editionis an expensive book that will pay for itself many times over.

With Examples in C#. Second Edition

Dependency Injection Principles, Practices, and Patterns

By Example

A Practical Approach

Obey the Testing Goat: Using Django, Selenium, and JavaScript

WORK EFFECT LEG CODE _p1

Write automated unit tests for the ABAP language. This book teaches programmers using simple examples and metaphors and explains the underlying concepts of writing effective automated unit tests. Many, if not most, ABAP programmers learned their programming and testing skills before the ABAP development environment provided an automated unit testing facility. Automated Unit Testing with ABAP: A Practical Approach offers hope and salvation to ABAP programmers who continue to toil with antiquated manual unit testing processes, taking them by the hand and lifting them out of that dungeon of despair with a modern and proven alternative. It begins by explaining how the xUnit family of automated testing frameworks provides a quick and effective means of insuring high-quality software. It then focuses on the ABAP Unit Testing Facility, the xUnit framework applicable specifically to the ABAP language, showing how it can be used to bring ABAP applications under automated testing control, from old legacy applications to those newly written. Whereas xUnit testing has been widely accepted with developers writing in many other programming languages, it is an unfortunate fact in the ABAP community that many programmers still are unfamiliar with xUnit concepts and do not know how to begin implementing automated unit testing into their development process. This book demonstrates how to refactor programs so they become designed for testability, showing how to use process encapsulation and test isolation to facilitate automated testing, including a thorough explanation of test-driven development and the use of test doubles. The book: Shows how to write automated unit tests for ABAP Instills ABAP programmers with the confidence to refactor poorly written code Explains how an automated testing harness facilitates rapid software development Teaches how to utilize test-driven development (TDD) with ABAP Offers advice and tips on the best ways to write automated unit tests What You Will Learn Become familiar with the xUnit approach to testing Know the ABAP statements that interfere with running automated unit tests and how to accommodate them Understand what it means to isolate code for testing and how this is achieved Gain the confidence to refactor poorly written code Make ABAP programs designed for testability Reap the benefits of spending less time manually unit testing ABAP programs Use test-driven development (TDD) with ABAP programming Use configurable test doubles in ABAP Who This Book Is For ABAP programmers who remain unfamiliar with the automated unit testing facility and those who already use it but want to improve their skill writing and using automated tests. The book addresses the reluctance and trepidation felt by procedural ABAP programmers who need to know some object-oriented concepts to use this facility, expands their horizons, and helps them step through the doorway leading to a different approach to program design.

Summary Effective Unit Testing is written to show how to write good tests—tests that are concise and to the point, expressive, useful, and maintainable. Inspired by Roy Osherove's bestselling *The Art of Unit Testing*, this book focuses on tools and practices specific to the Java world. It introduces you to emerging techniques like behavior-driven development and specification by example, and shows you how to add robust practices into your toolkit. About *Testing Test the components before you assemble them into a full application, and you'll get better software. For Java developers, there's now a decade of experience with well-crafted tests that anticipate problems, identify known and unknown dependencies in the code, and allow you to test components both in isolation and in the context of a full application. About this Book Effective Unit Testing teaches Java developers how to write unit tests that are concise, expressive, useful, and maintainable. Offering crisp explanations and easy-to-absorb examples, it introduces emerging techniques like behavior-driven development and specification by example. Programmers who are already unit testing will learn the current state of the art. Those who are new to the game will learn practices that will serve them well for the rest of their career. Purchase of the print book comes with an offer of a free PDF, ePub, and Kindle eBook from Manning. Also available is all code from the book. About the Author Lasse Koskela is a coach, trainer, consultant, and programmer. He hacks on open source projects, helps companies improve their productivity, and speaks frequently at conferences around the world. Lasse is the author of *Test Driven*, also published by Manning. What's Inside A thorough introduction to unit testing Choosing best-of-breed tools Writing tests using dynamic languages Efficient test automation Table of Contents PART 1 FOUNDATIONS The promise of good tests In search of good Test doubles PART 2 CATALOG Readability Maintainability Trustworthiness PART 3 DIVERSIONS Testable design Writing tests in other JVM languages Speeding up test execution*

The Art of Unit Testing, Third Edition takes you below the surface and understand how unit testing can transform the way you deliver and maintain software. *The Art of Unit Testing, Third Edition* guides you step by step from your first simple unit tests to building complete test sets that are maintainable, readable, and trustworthy. Now with examples using JavaScript and Node, this expanded third edition teaches you to write unit tests for async and observable code, as well as methods of faking functions, modules and refactors. You'll get to grips with complex subjects such as mocks and stubs, explore how to handle tricky legacy codebases, and save yourself time with tests that function for both frontend and backend applications. *The Art of Unit Testing, Third Edition* is an essential guide for any unit tester who wants to feel confident writing code that's easier to create, maintain, and adapt. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. An encyclopedia designed especially to meet the needs of elementary, junior high, and senior high school students.

Java Unit Testing with JUnit 5

How Tests Drive the Code

Fundamentals of Computer Programming with C#

Working Effectively with Unit Tests

Not as Hard as You Think

Dive into TDD to create flexible, maintainable, and production-ready .NET Core applications

The Art of Unit Testing, Second Edition guides you step by step from writing your first simple tests to developing robust test sets that are maintainable, readable, and trustworthy. You'll master the foundational ideas and quickly move to high-value subjects like mocks, stubs, and isolation, including frameworks such as Moq, FakeItEasy and Typemock Isolator. You'll explore test patterns and organization, working with legacy code, and even "untestable" code. Along the way, you'll learn about integration testing and techniques and tools for testing databases and other technologies. About This Book You know you should be unit testing, so why aren't you doing it? If you're new to unit testing, or if you're just not getting enough payoff for the effort you put into it, keep reading. The Art of Unit Testing, Second Edition guides you step by step from writing your first simple unit tests to building complete test sets that are maintainable, readable, and trustworthy. You'll move quickly to more complicated subjects like mocks and stubs, while learning to use isolation (mocking) frameworks like Moq, FakeItEasy, and Typemock Isolator. You'll explore test patterns and organization, refactor code applications, and learn how to test "untestable" code. Along the way, you'll learn about integration testing and techniques for testing with databases. The examples in the book use C#, but will benefit anyone using a statically typed language such as Java or C++. What's Inside Create readable, maintainable, trustworthy tests Fakes, stubs, mock objects, and isolation (mocking) frameworks Simple dependency injection techniques Refactoring legacy code About the Author Roy Osherove has been coding for over 15 years, and he consults and trains teams worldwide on the gentle art of unit testing and test-driven development. His blog is at ArtOfUnitTesting.com.

2nd edition of the step-by-step guide that helps developers to write test sets that are maintainable, readable and trustworthy.

Hands-on guidance to creating great test-driven development practice Test-driven development (TDD) practice helps developers recognize a well-designed application, and encourages writing a test before writing the functionality that needs to be implemented. This hands-on guide provides invaluable insight for creating successful test-driven development processes. With source code and examples featured in both C# and .NET, the book walks you through the TDD methodology and shows how it is applied to a real-world application. You'll witness the application built from scratch and details each step that is involved in the development, as well as any problems that were encountered and the solutions that were applied. Clarifies the motivation behind test-driven development (TDD), what it is, and how it works Reviews the various steps involved in developing an application and the testing that is involved prior to implementing the functionality Discusses unit testing and refactoring Professional Test-Driven Development with C# shows you how to create great TDD processes right away.

Unit testing. You've heard the term. Probably a lot. You know you should probably figure out how it works, since everyone's always talking about it and a lot of companies require developers to know it. But you don't really know it and you're worried that you'll look uninformed if you cop to not knowing it. Well, relax. This book assumes you have absolutely no idea how it works and walks you through the practice from the very beginning. You'll learn the basics, but more importantly, you'll learn the business value, the path to walk not to get frustrated, what's testable and what isn't, and, and everything else that a practical unit testing newbie could possibly want to know.

Pragmatic Unit Testing in C# with NUnit

Refactoring Test Code

C# and .NET Core Test-Driven Development

los Unit Testing by Example

Growing Self Organizing Teams

Testing Python

As iOS apps become increasingly complex and business-critical, iOS developers must ensure consistently superior code quality. This means adopting best practices for creating and testing iOS apps. Test-Driven Development (TDD) is one of the most powerful of these best practices. Test-Driven iOS Development is the first book 100% focused on helping you successfully implement TDD and unit testing in an iOS environment. Long-time iOS/Mac developer Graham Lee helps you rapidly integrate TDD into your existing processes using Apple 's Xcode 4 and the OCUnit unit testing framework. He guides you through constructing an entire Objective-C iOS app in a test-driven manner, from initial specification to functional product. Lee also introduces powerful patterns for applying TDD in iOS development, and previews powerful automated testing capabilities that will soon arrive on the iOS platform. Coverage includes Understanding the purpose, benefits, and costs of unit testing in iOS environments Mastering the principles of TDD, and applying them in areas from app design to refactoring Writing usable, readable, and repeatable iOS unit tests Using OCUnit to set up your Xcode project for TDD Using domain analysis to identify the classes and interactions your app needs, and designing it accordingly Considering third-party tools for iOS unit testing Building networking code in a test-driven manner Automating testing of view controller code that interacts with users Designing to interfaces, not implementations Testing concurrent code that typically runs in the background Applying TDD to existing apps Preparing for Behavior Driven Development (BDD) The only iOS-specific guide to TDD and unit testing, Test-Driven iOS Development covers both essential concepts and practical implementation.

Summary The Art of Unit Testing, Second Edition guides you step by step from writing your first simple tests to developing robust test sets that are maintainable, readable, and trustworthy. You'll master the foundational ideas and quickly move to high-value subjects like mocks, stubs, and isolation, including frameworks such as Moq, FakeItEasy, and Typemock Isolator. You'll explore test patterns and organization, working with legacy code, and even "untestable" code. Along the way, you'll learn about integration testing and techniques and tools for testing databases and other technologies. About This Book You know you should be unit testing, so why aren't you doing it? If you're new to unit testing, or if you find unit testing tedious, or if you're just not getting enough payoff for the effort you put into it, keep reading. The Art of Unit Testing, Second Edition guides you step by step from writing your first simple unit tests to building complete test sets that are maintainable, readable, and trustworthy. You'll move quickly to more complicated subjects like mocks and stubs, while learning to use isolation (mocking) frameworks like Moq, FakeItEasy, and Typemock Isolator. You'll explore test patterns and organization, refactor code applications, and learn how to test "untestable" code. Along the way, you'll learn about integration testing and techniques for testing with databases. The examples in the book use C#, but will benefit anyone using a statically typed language such as Java or C++. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. What's Inside Create readable, maintainable, trustworthy tests Fakes, stubs, mock objects, and isolation (mocking) frameworks Simple dependency injection techniques Refactoring legacy code About the Author Roy Osherove has been coding for over 15 years, and he consults and trains teams worldwide on the gentle art of unit testing and test-driven development. His blog is at ArtOfUnitTesting.com. Table of Contents PART 1 GETTING STARTED The basics of unit testing A first unit test PART 2 CORE TECHNIQUES Using stubs to break dependencies Interaction testing using mock objects Isolation (mocking) frameworks Digging deeper into isolation frameworks PART 3 THE TEST CODE Test hierarchies and organization The pillars of good unit tests PART 4 DESIGN AND PROCESS Integrating unit testing into the organization Working with legacy code Design and testability

Including numerous examples throughout, this book guides you step-by-step from writing your first simple unit tests to building complete test sets that are maintainable, readable, and trustworthy. --

Get more out of your legacy systems: more performance, functionality, reliability, and manageability Is your code easy to change? Can you get nearly instantaneous feedback when you do change it? Do you understand it? If the answer to any of these questions is no, you have legacy code, and it is draining time and money away from your development efforts. In this book, Michael Feathers offers start-to-finish strategies for working more effectively with large, untested legacy code bases. This book draws on material Michael created for his renowned Object Mentor seminars: techniques Michael has used in mentoring to help hundreds of developers, technical managers, and testers bring their legacy systems under control. The topics covered include Understanding the mechanics of software change: adding features, fixing bugs, improving design, optimizing performance Getting legacy code into a test harness Writing tests that protect you against introducing new problems Techniques that can be used with any language or platform—with examples in Java, C++, C, and C# Accurately identifying where code changes need to be made Coping with legacy systems that aren't object-oriented Handling applications that don't seem to have any structure This book also includes a catalog of twenty-four dependency-breaking techniques that help you work with program elements in isolation and make safer changes.

The Art of Multiprocessor Programming, Revised Reprint

Unit Test Frameworks

with examples in C#

The Art of Unit Testing, Second Edition

The Art of Unit Testing

xUnit Test Patterns

Assembly is a low-level programming language that's one step above a computer's native machine language. Although assembly language is commonly used for writing device drivers, emulators, and video games, many programmers find its somewhat unfriendly syntax intimidating to learn and use. Since 1996, Randall Hyde's *The Art of Assembly Language* has provided a comprehensive, plain-English, and patient introduction to 32-bit x86 assembly for non-assembly programmers. Hyde's primary teaching tool, High Level Assembler (or HLA), incorporates many of the features found in high-level languages (like C, C++, and Java) to help you quickly grasp basic assembly concepts. HLA lets you write true low-level code while enjoying the benefits of high-level language programming. As you read *The Art of Assembly Language*, you'll learn the low-level theory fundamental to computer science and turn that understanding into real, functional code. You'll learn how to: –Edit, compile, and run HLA programs –Declare and use constants, scalar variables, pointers, arrays, structures, unions, and namespaces –Translate arithmetic expressions (integer and floating point) –Convert high-level control structures This much anticipated second edition of *The Art of Assembly Language* has been updated to reflect recent changes to HLA and to support Linux, Mac OS X, and FreeBSD. Whether you're new to programming or you have experience with high-level languages, *The Art of Assembly Language, 2nd Edition* is your essential guide to learning this complex, low-level language.

If you program in C++ you've been neglected. Test-driven development (TDD) is a modern software development practice that can dramatically reduce the number of defects in systems, produce more maintainable code, and give you the confidence to change your software to meet changing needs. But C++ programmers have been ignored by those promoting TDD--until now. In this book, Jeff Langr gives you hands-on lessons in the challenges and rewards of doing TDD in C++. Modern C++ Programming With Test-Driven Development, the only comprehensive treatment on TDD in C++ provides you with everything you need to know about TDD, and the challenges and benefits of implementing it in your C++ systems. Its many detailed code examples take you step-by-step from TDD basics to advanced concepts. As a veteran C++ programmer, you're already writing high-quality code, and you work hard to maintain code quality. It doesn't have to be that hard. In this book, you'll learn: how to use TDD to improve legacy C++ systems how to identify and deal with troublesome system dependencies how to do dependency injection, which is particularly tricky in C++ how to use testing tools for C++ that aid TDD new C++11 features that facilitate TDD As you grow in TDD mastery, you'll discover how to keep a massive C++ system from becoming a design mess over time, as well as particular C++ trouble spots to avoid. You'll find out how to prevent your tests from being a maintenance burden and how to think in TDD without giving up your hard-won C++ skills. Finally, you'll see how to grow and sustain TDD in your team. Whether you're a complete unit-testing novice or an experienced tester, this book will lead you to mastery of test-

driven development in C++. What You Need A C++ compiler running under Windows or Linux, preferably one that supports C++11. Examples presented in the book were built under gcc 4.7.2. Google Mock 1.6 (downloadable for free: it contains Google Test as well) or an alternate C++ unit testing tool. Most examples in the book are written for Google Mock, but it isn't difficult to translate them to your tool of choice. A good programmer's editor or IDE. Cmake, preferably. Of course, you can use your own preferred make too. CMakeLists.txt files are provided for each project. Examples provided were built using cmake version 2.8.9. Various freely-available third-party libraries are used as the basis for examples in the book. These include: cURL JsonCpp Boost (filesystem, date_time/gregorian, algorithm, assign) Several examples use the boost headers/libraries. Only one example uses cURL and JsonCpp.

Explore the new way of building and maintaining test cases with Java test driven development (TDD) using JUnit 5. This book doesn't just talk about the new concepts, it shows you ways of applying them in TDD and Java 8 to continuously deliver code that excels in all metrics. Unit testing and test driven development have now become part of every developer's skill set. For Java developers, the most popular testing tool has been JUnit, and JUnit 5 is built using the latest features of Java. With Java Unit Testing with JUnit 5, you'll master these new features, including method parameters, extensions, assertions and assumptions, and dynamic tests. You'll also see how to write clean tests with less code. This book is a departure from using older practices and presents new ways of performing tests, building assertions, and injecting dependencies. What You Will Learn Write tests the JUnit 5 way Run your tests from within your IDE Integrate tests with your build and static analysis tools Migrate from JUnit 4 to JUnit 5 Who This Book Is For Java developers both with and without any prior unit testing experience.

Automated testing is a cornerstone of agile development. An effective testing strategy will deliver new functionality more aggressively, accelerate user feedback, and improve quality. However, for many developers, creating effective automated tests is a unique and unfamiliar challenge. xUnit Test Patterns is the definitive guide to writing automated tests using xUnit, the most popular unit testing framework in use today. Agile coach and test automation expert Gerard Meszaros describes 68 proven patterns for making tests easier to write, understand, and maintain. He then shows you how to make them more robust and repeatable—and far more cost-effective. Loaded with information, this book feels like three books in one. The first part is a detailed tutorial on test automation that covers everything from test strategy to in-depth test coding. The second part, a catalog of 18 frequently encountered "test smells," provides trouble-shooting guidelines to help you determine the root cause of problems and the most applicable patterns. The third part contains detailed descriptions of each pattern, including refactoring instructions illustrated by extensive code samples in multiple programming languages.

A Practical Guide
Occupational Outlook Handbook
Automated Unit Testing with ABAP
Practical Unit Testing with JUnit and Mockito
The Bulgarian C# Book
The Art of Assembly Language, 2nd Edition

Most people who write software have at least some experience with unit testing—even if they don't call it that. If you have ever written a few lines of throwaway code just to try something out, you've built a unit test. On the other end of the software spectrum, many large-scale applications have huge batteries of test cases that are repeatedly run and added to throughout the development process. What are unit test frameworks and how are they used? Simply stated, they are software tools to support writing and running unit tests, including a foundation on which to build tests and the functionality to execute the tests and report their results. They are not solely tools for testing; they can also be used as development tools on a par with preprocessors and debuggers. Unit test frameworks can contribute to almost every stage of software development and are key tools for doing Agile Development and building big-free code. Unit Test Frameworks covers the usage, philosophy, and architecture of unit test frameworks. Tutorials and example code are platform-independent and compatible with Windows, Mac OS X, Unix, and Linux. The companion CD includes complete versions of JUnit, CppUnit, NUnit, and XMLUnit, as well as the complete set of code examples.

*The Art of Unit Testing with examples in C#*Simon and Schuster

Learn how to apply a test-driven development process by building ready C# 7 and .NET Core applications. Key Features Create tests to quickly detect and resolve issues when writing portable code Uncover code integration issues that improve code quality using continuous integration Set up and use data-driven unit testing to verify your code Book Description This book guides developers to create robust, production-ready C# 7 and .NET Core applications through the practice of test-driven development process. In C# and .NET Core Test-Driven Development, you will learn the different stages of the TDD life cycle, basics of TDD, best practices, and anti-patterns. It will teach you how to create an ASP.NET Core MVC sample application, write testable code with SOLID principles and set up a dependency injection for your sample application. Next, you will learn the xUnit testing framework and learn how to use its attributes and assertions. You'll see how to create data-driven unit tests and mock dependencies in your code. You will understand the difference between running and debugging your tests on .NET Core on LINUX versus Windows and Visual Studio. As you move forward, you will be able to create a healthy continuous integration process for your sample application using GitHub, TeamCity, Cake, and Microsoft VSTS. By the end of this book, you will have learned how to write clean and robust code through the effective practice of TDD, set up CI build steps to test and build applications as well as how to package application for deployment on NuGet. What you will learn Write flexible, maintainable, and verifiable code for .NET Core Write testable code using SOLID principles and dependency injections Recognize the characteristics of a good unit test Structure and group your unit test Use mock objects to handle dependencies Set up an end-to-end continuous integration process Who this book is for This book is for .NET developers who would like to build efficient applications by implementing principles of test-driven development. C# programming and working knowledge of VS is assumed.

Radically improve your testing practice and software quality with new testing styles, good patterns, and reliable automation. Key Features A practical and results-driven approach to unit testing Refine your existing unit tests by implementing modern best practices Learn the four pillars of a good unit test Safely automate your testing process to save time and money Spot which tests need refactoring, and which need to be deleted entirely Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About The Book Great testing practices maximize your project quality and delivery speed by identifying bad code early in the development process. Wrong tests will break your code, multiply bugs, and increase time and costs. You owe it to yourself—and your projects—to learn how to do excellent unit testing. Unit Testing Principles, Patterns and Practices teaches you to design and write tests that target key areas of your code including the domain model. In this clearly written guide, you learn to develop professional-quality tests and test suites and integrate testing throughout the application life cycle. As you adopt a testing mindset, you'll be amazed at how better tests cause you to write better code. What You Will Learn Universal guidelines to assess any unit test Testing to identify and avoid anti-patterns Refactoring tests along with the production code Using integration tests to verify the whole system This Book Is Written For For readers who know the basics of unit testing. Examples are written in C# and can easily be applied to any language. About the Author Vladimir Khorikov is an author, blogger, and Microsoft MVP. He has mentored numerous teams on the ins and outs of unit testing. Table of Contents: PART 1 THE BIGGER PICTURE 1 : The goal of unit testing 2 ; What is a unit test? 3 ; The anatomy of a unit test PART 2 MAKING YOUR TESTS WORK FOR YOU 4 ; The four pillars of a good unit test 5 ; Mocks and test fragility 6 ; Styles of unit testing 7 ; Refactoring toward valuable unit tests PART 3 INTEGRATION TESTING 8 ; Why integration testing? 9 ; Mocking best practices 10 ; Testing the database PART 4 UNIT TESTING ANTI-PATTERNS 11 ; Unit testing anti-patterns

Code Better, Sleep Better

Test Driven Development with JUnit 5

The Art of Lean Software Development

The Complete Guide to Software Testing

Professional Test Driven Development with C#

Developing Real World Applications with TDD

Fearlessly change the design of your iOS code with solid unit tests. Use Xcode's built-in test framework XCTest and Swift to get rapid feedback on all your code - including legacy code. Learn the tricks and techniques of testing all iOS code, especially view controllers (UIViewController), which are critical to iOS apps. Learn to isolate and replace dependencies in legacy code written without tests. Practice safe refactoring that makes these tests possible, and watch all your changes get verified quickly and automatically. Make even the boldest code changes with complete confidence. Manual code and UI testing get slower the deeper your navigation hierarchy goes. It can take several taps just to reach a particular screen, never mind the actual workflow tests. Automatic unit testing offers such rapid feedback that it can change the rules of development. Bring testing to iOS development, even for legacy code. Use XCTest to write unit tests in Swift for all your code. iOS developers typically reserve unit tests for their model classes alone. But that approach skips most of the code common to iOS apps, especially with UIViewControllers. Learn how to unit test these view controllers to expand your unit testing possibilities. Since good unit tests form the bedrock for safe refactoring, you're empowered to make bold changes. Learn how to avoid the most common mistakes Swift programmers make with the XCTest framework. Use code coverage to find holes in your test suites. Learn how to identify hard dependencies. Reshape the design of your code quickly, with less risk and less fear.

Software testing is indispensable and is one of the most discussed topics in software development today. Many companies address this issue by assigning a dedicated software testing phase towards the end of their development cycle. However, quality cannot be tested into a buggy application. Early and continuous unit testing has been shown to be crucial for high quality software and low defect rates. Yet current books on testing ignore the developer's point of view and give little guidance on how to bring the overwhelming amount of testing theory into practice. Unit Testing in Java represents a practical introduction to unit testing for software developers. It introduces the basic test-first approach and then discusses a large number of special issues and problem cases. The book instructs developers through each step and motivates them to explore further. Shows how the discovery and avoidance of software errors is a demanding and creative activity in its own right and can build confidence early in a project. Demonstrates how automated tests can detect the unwanted effects of small changes in code within the entire system. Discusses how testing works with persistency, concurrency, distribution, and web applications. Includes a discussion of testing with C++ and Smalltalk.

The free book "Fundamentals of Computer Programming with C#" is a comprehensive computer programming tutorial that teaches programming, logical thinking, data structures and algorithms, problem solving and high quality code with lots of examples in C#. It starts with the first steps in programming and software development like variables, data types, conditional statements, loops and arrays and continues with other basic topics like methods, numeral systems, strings and string processing, exceptions, classes and objects. After the basics this fundamental programming book enters into more advanced programming topics like recursion, data structures (lists, trees, hash-tables and graphs), high-quality code, unit testing and refactoring, object-oriented principles (inheritance, abstraction, encapsulation and polymorphism) and their implementation the C# language. It also covers fundamental topics that each good developer should know like algorithm design, complexity of algorithms and problem solving. The book uses C# language and Visual Studio to illustrate the programming concepts and explains some C# / .NET specific technologies like lambda expressions, extension methods and LINQ. The book is written by a team of developers lead by Svetlin Nakov who has 20+ years practical software development experience. It teaches the major programming concepts and way of thinking needed to become a good software engineer and the C# language in the meantime. It is a great start for anyone who wants to become a skillful software engineer. The books does not teach technologies like databases, mobile and web development, but shows the true way to master the basics of programming regardless of the languages, technologies and tools. It is good for beginners and intermediate developers who want to put a solid base for a successful career in the software engineering industry. The book is accompanied by free video lessons, presentation slides and mind maps, as well as hundreds of exercises and live examples. Download the free C# programming book, videos, presentations and other resources from <http://Introprogramming.info>. Title: Fundamentals of Computer Programming with C# (The Bulgarian C# Programming Book) ISBN: 9789544007737 ISBN-13: 978-954-400-773-7 (9789544007737) ISBN-10: 954-400-773-3 (9544007733) Author: Svetlin Nakov & Co. Pages: 1132 Language: English Published: Sofia, 2013 Publisher: Faber Publishing, Bulgaria Web site: <http://www.introprogramming.info> License: CC-Attribution-Share-Alike Tags: free, programming, book, computer programming, programming fundamentals, ebook, book programming, C#, CSharp, C# book, tutorial, C# tutorial;

programming concepts, programming fundamentals, compiler, Visual Studio, .NET, .NET Framework, data types, variables, expressions, statements, console, conditional statements, control-flow logic, loops, arrays, numeral systems, methods, strings, text processing, StringBuilder, exceptions, exception handling, stack trace, streams, files, text files, linear data structures, list, linked list, stack, queue, tree, balanced tree, graph, depth-first search, DFS, breadth-first search, BFS, dictionaries, hash tables, associative arrays, sets, algorithms, sorting algorithm, searching algorithms, recursion, combinatorial algorithms, algorithm complexity, OOP, object-oriented programming, classes, objects, constructors, fields, properties, static members, abstraction, interfaces, encapsulation, inheritance, virtual methods, polymorphism, cohesion, coupling, enumerations, generics, namespaces, UML, design patterns, extension methods, anonymous types, lambda expressions, LINQ, code quality, high-quality code, high-quality methods, code formatting, self-documenting code, code refactoring, problem solving, problem solving methodology, 9789544007737, 9544007733

Presents a guide to unit testing with the NUnit library in C# along with providing information on writing code, detecting and fixing problems, testing pieces of code, and testing with a team.

with examples in JavaScript

Testing with JUnit

Xctest Tips and Techniques Using Swift

A guide for Java developers

Unit Testing in Java

Tools for High-Quality Software Development

By taking you through the development of a real web application from beginning to end, the second edition of this hands-on guide demonstrates the practical advantages of test-driven development (TDD) with Python. You'll learn how to write and run tests before building each part of your app, and then develop the minimum amount of code required to pass those tests. The result? Clean code that works. In the process, you'll learn the basics of Django, Selenium, Git, jQuery, and Mock, along with current web development techniques. If you're ready to take your Python skills to the next level, this book—updated for Python 3.6—clearly demonstrates how TDD encourages simple designs and inspires confidence. Dive into the TDD workflow, including the unit test/code cycle and refactoring Use unit tests for classes and functions, and functional tests for user interactions within the browser Learn when and how to use mock objects, and the pros and cons of isolated vs. integrated tests Test and automate your deployments with a staging server Apply tests to the third-party plugins you integrate into your site Run tests automatically by using a Continuous Integration environment Use TDD to build a REST API with a front-end Ajax interface

Ed Yourdan called it a bible for project managers. You'll gain a new perspective on software testing as a life cycle activity, not merely as something that happens at the end of coding. An invaluable aid for the development of testing standards and the evaluation of testing effectiveness.

Summary Dependency Injection Principles, Practices, and Patterns teaches you to use DI to reduce hard-coded dependencies between application components. You'll start by learning what DI is and what types of applications will benefit from it. Then, you'll work through concrete scenarios using C# and the .NET framework to implement DI in your own projects. As you dive into the thoroughly-explained examples, you'll develop a foundation you can apply to any of the many DI libraries for .NET and .NET Core. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology Dependency Injection (DI) is a great way to reduce tight coupling between software components. Instead of hard-coding dependencies, such as specifying a database driver, you make those connections through a third party. Central to application frameworks like ASP.NET Core, DI enables you to better manage changes and other complexity in your software. About the Book Dependency Injection Principles, Practices, and Patterns is a revised and expanded edition of the bestselling classic Dependency Injection in .NET. It teaches you DI from the ground up, featuring relevant examples, patterns, and anti-patterns for creating loosely coupled, well-structured applications. The well-annotated code and diagrams use C# examples to illustrate principles that work flawlessly with modern object-oriented languages and DI libraries. What's Inside Refactoring existing code into loosely coupled code DI techniques that work with statically typed OO languages Integration with common .NET frameworks Updated examples illustrating DI in .NET Core About the Reader For intermediate OO developers. About the Authors Mark Seemann is a programmer, software architect, and speaker who has been working with software since 1995, including six years with Microsoft. Steven van Deursen is a seasoned Java technologies: JUnit and Mockito. It presents a range of techniques necessary to write high quality unit tests - e.g. mocks, parametrized tests and matchers. It also discusses trade-offs related to the choices we have to make when dealing with some real-life code issues. The book stresses the importance of writing readable and maintainable unit tests, and puts a lot of stress on code quality. It shows how to achieve testable code and to eliminate common mistakes by following the Test Driven Development approach. Every topic discussed in the book is illustrated with code examples, and each chapter is accompanied by some exercises. By reading this book you will: Grasp the role and purpose of unit tests Write high-quality, readable and maintainable unit tests Learn how to use JUnit and Mockito (but also other useful tools) Avoid common pitfalls when writing unit tests Recognize bad unit tests, and fix them in no time Develop code following the Test Driven Development (TDD) approach Use mocks, stubs and test-spies intelligently Measure the quality of your tests using code coverage and mutation testing Learn how to improve your tests' code so it is an asset and not a burden Test collections, expected exceptions, time-dependent methods and much more Customize test reports so that they show you what you really need to know Master tools and techniques your team members have never even heard of (priceless!)) Nowadays every developer is expected to write unit tests. While simple in theory, in practice writing high-quality unit tests can turn out to be a real challenge. This book will help.

Test-Driven iOS Development

A Practical and Incremental Approach

Working Effectively with Legacy Code

Modern C++ Programming with Test-Driven Development

Test-Driven Development with Python

Essential Cell Biology provides a readily accessible introduction to the central concepts of cell biology, and its lively, clear writing and exceptional illustrations make it the ideal textbook for a first course in both cell and molecular biology. The text and figures are easy-to-follow, accurate, clear, and engaging for the introductory student. Molecular detail has been kept to a minimum in order to provide the reader with a cohesive conceptual framework for the basic science that underlies our current understanding of all of biology, including the biomedical sciences. The Fourth Edition has been thoroughly revised, and covers the latest developments in this fast-moving field, yet retains the academic level and length of the previous edition. The book is accompanied by a rich package of online student and instructor resources, including over 130 narrated movies, an expanded and updated Question Bank. Essential Cell Biology, Fourth Edition is additionally supported by the Garland Science Learning System. This homework platform is designed to evaluate and improve student performance and allows instructors to select assignments on specific topics and review the performance of the entire class, as well as individual students, via the instructor dashboard. Students receive immediate feedback on their mastery of the topics, and will be better prepared for lectures and classroom discussions. The user-friendly system provides a convenient way to engage students while assessing progress. Performance data can be used to tailor classroom discussion, activities, and lectures to address students' needs precisely and efficiently. For more information and sample material, visit <http://garlandscience.rocketmix.com/>.

This guide for programmers teaches how to practice Test Driven Development (TDD), also called Test First Development. Contrary to the accepted approach to testing, when you practice TDD you write tests for code before you write the code being tested. This text provides examples in Java.

A practical, example-driven guide to using, automating, and integrating JavaScript Unit tests for the busy and conscientious JavaScript developer striving for excellence and success. JavaScript Unit Testing is a must have guide for every web developer, designer, architect, and JavaScript coder seeking to ensure the highest quality of their web applications and JS code. Knowledge of JavaScript is assumed.

This succinct book explains how you can apply the practices of Lean software development to dramatically increase productivity and quality. Based on techniques that revolutionized Japanese manufacturing, Lean principles are being applied successfully to product design, engineering, the supply chain, and now software development. With The Art of Lean Software Development, you'll learn how to adopt Lean practices one at a time rather than taking on the entire methodology at once. As you master each practice, you'll see significant, measurable results. With this book, you will: Understand Lean's origins from Japanese industries and how it applies to software development Learn the Lean software development principles and the five most important practices in detail Distinguish between the Lean and Agile methodologies and understand their similarities and differences Determine which Lean principles you should adopt first, and how you can gradually incorporate more of the methodology into your process Review hands-on practices, including descriptions, benefits, trade-offs, and roadblocks Learn how to sell these principles to management The Art of Lean Software Development is ideal for busy people who want to improve the development process but can't afford the disruption of a sudden and complete transformation. The Lean approach has been yielding dramatic results for decades, and with this book, you can make incremental changes that will produce immediate benefits. "This book presents Lean practices in a clear and concise manner so readers are motivated to make their software more reliable and less costly to maintain. I recommend it to anyone looking for an easy-to-follow guide to transform how the developer views the process of writing good software."-- Bryan Wells, Boeing Intelligence & Security Sysms Mission System "If you're new to Lean software development and you're not quite sure where to start, this book will help get your development process going in the right direction, one step at a time."-- John McClenning, software development lead, Aclara

Applying Unit Testing, TDD, BDD and Acceptance Testing

The Art of Software Testing

The Art of Unit Testing, Third Edition

Unit Testing Principles, Practices, and Patterns

Starting to Unit Test

The World Book Encyclopedia

This book details Jay Fields' strong opinions on the best way to test, while acknowledging alternative styles and various contexts in which tests are written. Whether you prefer Jay Fields' style or not, this book will help you write better Unit Tests.

From the Preface: Over a dozen years ago I read Refactoring for the first time; it immediately became my bible. While Refactoring isn't about testing, it explicitly states: If you want to refactor, the essential precondition is having solid tests. At that time, if Refactoring deemed it necessary, I unquestionably complied. That was the beginning of my quest to create productive unit tests. Throughout the 12+ years that followed reading Refactoring I made many mistakes, learned countless lessons, and developed a set of guidelines that I believe make unit testing a productive use of programmer time. This book provides a single place to examine those mistakes, pass on the lessons learned, and provide direction for those that want to test in a way that I've found to be the most productive. The book does touch on some theory and definition, but the main purpose is to show you how to take tests that are causing you pain and turn them into tests that you're happy to work with.

"The art of unit testing, second edition guides you step by step from writing your first simple tests to developing robust test sets that are maintainable, readable, and trustworthy. You'll master the foundational ideas and quickly move to high-value subjects like mocks, stubs, and isolation, including frameworks such as Moq, FakeItEasy and Typemock Isolator. You'll explore test patterns and organization, working with legacy code, and even 'untestable' code. Along the way, you'll learn about integration testing and techniques and tools for testing databases and other technologies."--Resource description page.

Write clean code that works with the help of this groundbreaking software method. Example-driven teaching is the basis of Beck's step-by-step instruction that will have readers using TDD to further their projects.

Are you an architect? Scrum Master? team leader? project manager? If you are any of these, you will find that leadership, done right, is a very tough job. This book deals with the hard parts. Not with tools, but with people. Here is the manifesto that drives this book: For us as team leaders, the goal and the way we measure our work is the overall growth in skills of self-organization and self-maintenance in each member of our team and the team as a whole. To that end: We accept that the team's needs from us change continuously based on their skills for handling the current reality of work, so we embrace a continuously changing leadership style over a one-style-fits-all leadership approach. We believe in challenging ourselves and our teams to always get better, so: * We create slack time for the team to learn and be challenged. * We embrace taking risks for our team over staying safe. * We embrace fear and discomfort while learning new skills over keeping people within their comfort zone. * We embrace experimentation as a constant practice over maintaining the status quo: * With people * With tools * With processes * With the environment * We believe our core practice is leading people, not wielding machines, so: * We embrace spending more time with our team than in meetings. * We embrace treating software problems as people problems. * We learn people skills and communication techniques. About the notes The second part of this book allows a peek into the minds of some of the best leaders, consultants, and managers as they give advice to a new team leader. Hear from Johanna Rothman, Kevlin Henney, Dan North, Uncle Bob Martin, and many others about the one thing they would like to teach you if you ever become a team leader, Scrum Master, project manager, or architect.

Test-driven Development

Notes to a Software Team Leader

With Examples in C#

Essential Cell Biology

JavaScript Unit Testing

Effective Unit Testing