what is specification by example

what is specification by example is a modern approach to defining requirements and ensuring software quality through concrete examples rather than abstract documentation. This method emphasizes collaboration between stakeholders, developers, and testers to create clear, executable specifications that guide the development process. Specification by example helps bridge the gap between business needs and technical implementation by using real-world scenarios to describe desired behavior. It supports agile development practices by promoting communication, reducing misunderstandings, and improving the accuracy of requirements. By using concrete examples, teams can automate tests, verify functionality continuously, and deliver software that meets expectations more reliably. This article explores the concept of specification by example, its benefits, core practices, and how it integrates with software development workflows. The following sections provide a comprehensive overview to better understand this powerful methodology.

Understanding Specification by Example

Specification by example is a collaborative approach to defining software requirements through detailed examples that illustrate how the system should behave in various scenarios. Instead of relying solely on traditional textual requirements or lengthy documentation, this methodology uses concrete examples to clarify expectations. These examples are often structured as acceptance criteria or test cases that can be executed to verify the system’s functionality.

Origins and Definition

The term “specification by example” was popularized by Gojko Adzic in his book on the subject, where it is described as a technique that uses realistic examples to guide software development. This technique integrates well with agile and behavior-driven development (BDD) frameworks, focusing on communication and collaboration among stakeholders to define requirements precisely.

How It Differs from Traditional Requirements

Traditional requirements documentation tends to be abstract, ambiguous, and prone to misinterpretation. Specification by example replaces vague descriptions with specific scenarios that demonstrate desired behaviors. This shift reduces misunderstandings between business analysts, developers, and testers, resulting in higher quality software that meets user needs.

Key Components

Specification by example consists of several critical components:

- **Examples:** Concrete instances demonstrating how the system should behave.
- **Collaboration:** Joint creation of specifications by business and technical teams.
- **Automation:** Using examples as automated tests to verify functionality.
- **Living Documentation:** Specifications evolve alongside the system and remain up-to-date.

Key Benefits of Specification by Example

Implementing specification by example offers numerous advantages that improve the software development lifecycle and product quality. These benefits stem from the clarity, precision, and continuous validation that example-based specifications provide.

Improved Communication

Specification by example fosters clear communication between stakeholders. By working together on concrete examples, misunderstandings are minimized, which leads to a shared understanding of what the software should achieve. This collaboration ensures that business goals align closely with technical implementation.

Reduced Ambiguity and Errors

Because specifications are based on real scenarios, ambiguity is drastically reduced. This clarity helps developers build the right functionality and testers design accurate test cases. As a result, the number of defects and rework decreases significantly.

Accelerated Feedback and Testing

Examples used as acceptance criteria can be automated and integrated into continuous integration pipelines. This automation enables rapid feedback on whether new code meets the defined specifications, supporting faster release cycles and more reliable software.

Living Documentation

Unlike traditional requirements documents that quickly become outdated, specifications by example serve as living documentation. They evolve with the product, providing an always-current reference for the team, which simplifies maintenance and onboarding.

Core Practices and Process

Specification by example relies on a set of core practices that ensure its effective application in software development projects. These practices emphasize collaboration, clarity, and continuous validation.

Collaborative Specification Workshops

Successful specification by example begins with collaborative workshops where stakeholders come together to define examples. These sessions encourage dialogue between business experts, developers, and testers to capture requirements precisely.

Defining Concrete Examples

Examples should be specific, relevant, and easy to understand. They typically describe input conditions, actions taken, and expected outcomes. This format provides a clear basis for acceptance testing and development.

Automating Acceptance Tests

The examples become acceptance tests that can be automated. Tools like Cucumber or SpecFlow enable teams to write executable specifications in natural language, bridging the gap between documentation and code.

Maintaining Specifications as Living Artifacts

Specifications must be updated regularly to reflect changes in requirements or functionality. Continuous collaboration ensures that these examples remain accurate and useful throughout the product lifecycle.

Iterative Refinement

Specification by example supports an iterative approach where examples are refined over time. This refinement improves clarity and coverage as the team gains deeper understanding of the system.

Tools and Techniques for Specification by Example

Various tools and methodologies facilitate the implementation of specification by example, enabling teams to write, automate, and manage executable specifications effectively.

Behavior-Driven Development (BDD) Frameworks

BDD tools such as Cucumber, SpecFlow, and JBehave allow teams to write specifications in a domain-specific language that non-technical stakeholders can understand. These frameworks convert plain-text examples into automated tests.

Collaborative Platforms

Platforms like Jira, Confluence, and other agile project management tools support collaboration and documentation of examples. They often integrate with testing frameworks to maintain traceability between requirements and tests.

Living Documentation Generators

Tools that generate living documentation from executable specifications help keep documentation synchronized with the codebase. This documentation acts as a reliable reference for current system behavior.

Example Mapping

Example mapping is a technique used to organize and discover examples during specification workshops. It helps categorize rules, examples, and questions, making the specification process more structured and efficient.

Challenges and Best Practices

Despite its benefits, specification by example can present challenges if not implemented thoughtfully. Understanding these challenges and following best practices ensures successful adoption.

Common Challenges

- **Initial Learning Curve:** Teams may require training to understand and adopt the methodology effectively.
- **Maintaining Specifications:** Without discipline, specifications can become outdated or inconsistent.
- **Collaboration Barriers:** Lack of stakeholder engagement can limit the quality and usefulness of examples.
- **Tooling Complexity:** Selecting and integrating appropriate tools can be difficult.

Best Practices

- **Engage All Stakeholders:** Ensure business, development, and testing teams collaborate regularly.
- **Focus on Clarity:** Write simple, precise examples that are easy to understand and automate.
- **Integrate Automation Early:** Automate acceptance tests to gain rapid feedback and maintain quality.
- **Keep Specifications Up-to-Date:** Treat specifications as living documents that evolve with the product.
- **Use Structured Techniques:** Employ methods like example mapping to systematically discover and manage examples.

Frequently Asked Questions

What is Specification by Example?

Specification by Example is a collaborative approach to defining requirements and functional specifications through realistic examples, which serve as the basis for automated tests and ensure shared understanding among stakeholders.

How does Specification by Example improve software development?

Specification by Example improves software development by fostering clear communication

between business and technical teams, reducing misunderstandings, enabling early validation of requirements, and supporting automated acceptance testing.

What are the key components of Specification by Example?

The key components of Specification by Example include collaborative specification workshops, concrete examples that illustrate requirements, living documentation maintained through automation, and automated acceptance tests derived from those examples.

How is Specification by Example related to Behavior-Driven Development (BDD)?

Specification by Example and Behavior-Driven Development (BDD) are closely related; both use concrete examples to define behavior, promote collaboration, and support automated testing. BDD often uses Specification by Example as a technique to write executable specifications in a structured format like Gherkin.

What tools support Specification by Example practices?

Tools that support Specification by Example include Cucumber, SpecFlow, FitNesse, and Concordion, which enable writing executable specifications from examples and integrating them with automated testing frameworks.

Additional Resources

1. *Specification by Example: How Successful Teams Deliver the Right Software*
This book by Gojko Adzic introduces the concept of Specification by Example, a collaborative approach to defining requirements and functional tests with realistic examples. It emphasizes communication between stakeholders, developers, and testers to reduce misunderstandings and deliver software that meets business needs. The book includes case studies and practical techniques to implement this methodology effectively.

2. *Bridging the Communication Gap: Specification by Example and Agile Acceptance Testing*
Written by Gojko Adzic, this book focuses on improving communication between business and technical teams through Specification by Example. It explores how to write clear acceptance tests and use examples to clarify requirements, reducing defects and rework. The book provides actionable advice for teams adopting agile practices and test-driven development.

3. *Discover to Deliver: Agile Product Planning and Analysis*
By Ellen Gottesdiener and Mary Gorman, this book covers collaborative techniques for discovering product requirements, including principles aligned with Specification by Example. It helps product owners and teams understand user needs deeply and produce clear, testable specifications. The authors focus on delivering valuable software through iterative feedback and shared understanding.

4. *Agile Testing: A Practical Guide for Testers and Agile Teams*
Lisa Crispin and Janet Gregory provide comprehensive guidance on testing within agile projects, including the use of Specification by Example for acceptance testing. The book explains how testers can collaborate with developers and business stakeholders to create executable specifications. It highlights the importance of continuous testing and automation in agile environments.

5. *Example-Driven Software Development: A Practical Guide to Specification by Example*
This book offers a hands-on approach to implementing Specification by Example in software projects. It explains how to use concrete examples to specify requirements, automate tests, and improve collaboration across teams. The practical tips and real-world examples help teams reduce ambiguity and increase software quality.

6. *Specification by Example Patterns*
This resource provides a collection of patterns and best practices for applying Specification by Example effectively. It covers how to structure examples, manage evolving requirements, and integrate with agile workflows. The book is ideal for teams looking to deepen their understanding and refine their use of collaborative specification techniques.

7. *Lean-Agile Acceptance Test-Driven Development: Better Software Through Collaboration*
Author Ken Pugh explores how acceptance test-driven development (ATDD) complements Specification by Example to improve software delivery. The book details processes for writing acceptance tests collaboratively and using them to guide development. It emphasizes lean and agile principles to enhance team collaboration and product quality.

8. *Behaviour-Driven Development with Cucumber: Specification by Example for Agile Teams*
This book focuses on the practice of Behaviour-Driven Development (BDD), closely related to Specification by Example, using the Cucumber tool. It guides teams on writing executable specifications in plain language that serve as both requirements and automated tests. The book is practical for teams adopting BDD to improve communication and reduce defects.

9. *Agile Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise*
Dean Leffingwell discusses lean and agile approaches to requirements management, including the use of Specification by Example. The book covers methods to capture and communicate requirements effectively at all levels of an organization. It provides strategies for aligning development work with business goals through collaborative specification.

What Is Specification By Example

what is specification by example: Specification by Example Gojko Adzic, 2011-06-02 Summary Specification by Example is an emerging practice for creating software based on realistic examples, bridging the communication gap between business stakeholders and the dev teams building the software. In this book, author Gojko Adzic distills interviews with successful teams worldwide, sharing how they specify, develop, and deliver software, without defects, in short iterative delivery cycles. About the Technology Specification by Example is a collaborative method for specifying requirements and tests. Seven patterns, fully explored in this book, are key to making the method effective. The method has four main benefits: it produces living, reliable documentation; it defines expectations clearly and makes validation efficient; it reduces rework; and, above all, it assures delivery teams and business stakeholders that the software that's built is right for its purpose. About the Book This book distills from the experience of leading teams worldwide effective ways to specify, test, and deliver software in short, iterative delivery cycles. Case studies in this book range from small web startups to large financial institutions, working in many processes including XP, Scrum, and Kanban. This book is written for developers, testers, analysts, and business people working together to build great software. 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. What's Inside Common process patterns How to avoid bad practices Fitting SBE in your process 50+ case studies ========== Table of Contents Part 1 Getting started Part 2 Key process patterns Part 3 Case studies Key benefits Key process patterns Living documentation Initiating the changes Deriving scope from goals Specifying collaboratively Illustrating using examples Refining the specification Automating validation without changing specifications Validating frequently Evolving a documentation system uSwitch RainStor Iowa Student Loan Sabre Airline Solutions ePlan Services Songkick Concluding thoughts

what is specification by example: *Writing Great Specifications* Kamil Nicieja, 2017-10-25 Summary Writing Great Specifications is an example-rich tutorial that teaches you how to write good Gherkin specification documents that take advantage of the benefits of specification by example. Foreword written by Gojko Adzic. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology The clearest way to communicate a software specification is to provide examples of how it should work. Turning these story-based descriptions into a well-organized dev plan is another matter. Gherkin is a human-friendly, jargon-free language for documenting a suite of examples as an executable specification. It fosters efficient collaboration between business and dev teams, and it's an excellent foundation for the specification by example (SBE) process. About the Book Writing Great Specifications teaches you how to capture executable software designs in Gherkin following the SBE method. Written for both developers and non-technical team members, this practical book starts with collecting individual feature stories and organizing them into a full, testable spec. You'll learn to choose the best scenarios, write them in a way that anyone can understand, and ensure they can be easily updated by anyone.management. What's Inside Reading and writing Gherkin Designing story-based test cases Team Collaboration Managing a suite of Gherkin documents About the Reader Primarily written for developers and architects, this book is accessible to any member of a software design team. About the Author Kamil Nicieja is a seasoned engineer, architect, and project manager with deep expertise in Gherkin and SBE. Table of contents Introduction to specification by example and Gherkin PART 1 - WRITING EXECUTABLE SPECIFICATIONS WITH EXAMPLES The specification layer and the automation layer Mastering the Given-When-Then template The basics of scenario outlines Choosing examples for scenario outlines The life cycle of executable specifications Living documentation PART 2 - MANAGING SPECIFICATION SUITES Organizing scenarios into a specification suite Refactoring features into abilities and business needs Building a domain-driven specification suite Managing large projects with bounded contexts

what is specification by example: ATDD by Example Markus Gärtner, 2013 With Acceptance Test-Driven Development (ATDD), business customers, testers, and developers can collaborate to produce testable requirements that help them build higher quality software more rapidly. However,

ATDD is still widely misunderstood by many practitioners. ATDD by Example is the first practical, entry-level, hands-on guide to implementing and successfully applying it. ATDD pioneer Markus Gärtner walks readers step by step through deriving the right systems from business users, and then implementing fully automated, functional tests that accurately reflect business requirements, are intelligible to stakeholders, and promote more effective development. Through two end-to-end case studies, Gärtner demonstrates how ATDD can be applied using diverse frameworks and languages. Each case study is accompanied by an extensive set of artifacts, including test automation classes, step definitions, and full sample implementations. These realistic examples illuminate ATDD's fundamental principles, show how ATDD fits into the broader development process, highlight tips from Gärtner's extensive experience, and identify crucial pitfalls to avoid. Readers will learn to Master the thought processes associated with successful ATDD implementation Use ATDD with Cucumber to describe software in ways businesspeople can understand Test web pages using ATDD tools Bring ATDD to Java with the FitNesse wiki-based acceptance test framework Use examples more effectively in Behavior-Driven Development (BDD) Specify software collaboratively through innovative workshops Implement more user-friendly and collaborative test automation Test more cleanly, listen to test results, and refactor tests for greater value If you're a tester, analyst, developer, or project manager, this book offers a concrete foundation for achieving real benefits with ATDD now-and it will help you reap even more value as you gain experience.

what is specification by example: *Software Methodologies* Capers Jones, 2017-07-06 This comprehensive reference uses a formal and standard evaluation technique to show the strengths and weakness of more than 60 software development methodologies such as agile, DevOps, RUP, Waterfall, TSP, XP and many more. Each methodology is applied to an application of 1000 function points using the Java language. Each methodology produces a characteristic set of results for development schedules, productivity, costs, and quality. The intent of the book is to show readers the optimum kinds of methodologies for the projects they are concerned with and to warn them about counter indications and possible harm from unsuitable methodologies.

what is specification by example: Logic Program Synthesis from Incomplete Information Pierre Flener, 2012-12-06 Program synthesis is a solution to the software crisis. If we had a program that develops correct programs from specifications, then program validation and maintenance would disappear from the software life-cycle, and one could focus on the more creative tasks of specification elaboration, validation, and maintenance, because replay of program development would be less costly. This monograph describes a novel approach to Inductive Logic Programming (ILP), which cross-fertilizes logic programming and machine learning. Aiming at the synthesis of recursive logic programs only, and this from incomplete information, we take a software engineering approach that is more appropriate than a pure artificial intelligence approach. This book is suitable as a secondary text for graduate level courses in software engineering and artificial intelligence, and as a reference for practitioners of program synthesis.

what is specification by example: Agile Processes in Software Engineering and Extreme Programming Giovanni Cantone, Michele Marchesi, 2014-06-30 This book contains the refereed proceedings of the 15th International Conference on Agile Software Development, XP 2014, held in Rome, Italy, in May 2014. Because of the wide application of agile approaches in industry, the need for collaboration between academics and practitioners has increased in order to develop the body of knowledge available to support managers, system engineers, and software engineers in their managerial/economic and architectural/project/technical decisions. Year after year, the XP conference has facilitated such improvements and provided evidence on the advantages of agile methodologies by examining the latest theories, practical applications, and implications of agile and lean methods. The 15 full papers, seven short papers, and four experience reports accepted for XP 2014 were selected from 59 submissions and are organized in sections on: agile development, agile challenges and contracting, lessons learned and agile maturity, how to evolve software engineering teaching, methods and metrics, and lean development.

what is specification by example: *Agile Software Development - An Overview* K Amuthabala,

Shantala Devi Patil, Thirumagal E, Thanuja K, 2023-10-05 This textbook has been meticulously crafted with a singular purpose: offering a comprehensive and practical guide to Agile Software Development. In the forthcoming chapters, we will delve into theintricacies of Agile methodologies, explore their underlying principles, and investigate the compelling reasons behind their prominence in the software development industry. Section I: Introduction to Iterative Development, Evolutionary, and Adaptive Development, Our journeybegins with an exploration of fundamental concepts: Iterative Development, Evolutionary Development,and Adaptive Development. These approaches break free from conventional linear development processesand prioritize flexibility, risk management, and client-driven planning. This chapter will discuss the meritsof time-boxed iterative development, evolutionary requirements analysis, incremental delivery, and theultimate goal of evolutionary delivery. Section II: Serves as a bridge between theory and practice within the Agile realm. Here, we define AgileDevelopment, categorize various methodologies, and delve deep into the Agile Manifesto and its guidingprinciples. Additionally, we explore Agile project management, emphasizing the crucial role ofcommunication, feedback, and the human element. The chapter culminates in an exploration of specificAgile methods and a balanced discussion of the ongoing discourse surrounding Agile Hype. Section III: Motivation and Evidence, Understanding the motivation underpinning Agile is fundamental toappreciating its significance. In Chapter 3, we illuminate the imperatives for change in software projectsand how iterative development addresses these challenges. We critique the limitations of the traditionalWaterfall model and provide a comprehensive review of supporting evidence, including research findings,historical project data, and expert opinions, all converging to fortify the case for iterative development. Section IV: Fundamentals of DevOps and Technical View, Agile methodologies extend beyond softwaredevelopment into the realm of DevOps. Chapter 4 introduces the foundational principles of DevOps and itspivotal role in contemporary development practices. We delve into the building blocks of DevOps, thevital metrics and measurement perspective, and the process view that fosters seamless collaborationbetween development and operations teams. The section IV concludes with an in-depth exploration of thetechnical facets, including topics like automatic releasing, infrastructure as code, and specification byexample, enriched by real-world case studies. Upon completing this textbook, you will comprehensively comprehend Agile Software Development andDevOps. Whether you are a student embarking on a career in software development or an industryprofessional looking to stay at the forefront of the field, the knowledge and insights provided here will equip you with the tools to excel in the dynamic world of software development. Let us embark on this enlightening journey together, embracing agility, adaptability, and excellence in software development.

what is specification by example: Succeeding with Agile Mike Cohn, 2010 Proven, 100% Practical Guidance for Making Scrum and Agile Work in Any Organization This is the definitive, realistic, actionable guide to starting fast with Scrum and agile-and then succeeding over the long haul. Leading agile consultant and practitioner Mike Cohn presents detailed recommendations, powerful tips, and real-world case studies drawn from his unparalleled experience helping hundreds of software organizations make Scrum and agile work. Succeeding with Agile is for pragmatic software professionals who want real answers to the most difficult challenges they face in implementing Scrum. Cohn covers every facet of the transition: getting started, helping individuals transition to new roles, structuring teams, scaling up, working with a distributed team, and finally, implementing effective metrics and continuous improvement. Throughout, Cohn presents Things to Try Now sections based on his most successful advice. Complementary Objection sections reproduce typical conversations with those resisting change and offer practical guidance for addressing their concerns. Coverage includes Practical ways to get started immediately-and get good fast Overcoming individual resistance to the changes Scrum requires Staffing Scrum projects and building effective teams Establishing improvement communities of people who are passionate about driving change Choosing which agile technical practices to use or experiment with Leading self-organizing teams Making the most of Scrum sprints, planning, and quality techniques Scaling Scrum to distributed, multiteam projects Using Scrum on projects with complex sequential

processes or challenging compliance and governance requirements Understanding Scrum's impact on HR, facilities, and project management Whether you've completed a few sprints or multiple agile projects and whatever your role-manager, developer, coach, ScrumMaster, product owner, analyst, team lead, or project lead-this book will help you succeed with your very next project. Then, it will help you go much further: It will help you transform your entire development organization.

what is specification by example: *Students' Guide to Programming Languages* Malcolm Bull, 2016-06-06 Students' Guide to Programming Languages introduces programming languages, emphasizing why they are needed, how they are defined and constructed, and where and how they are used. With greater access to computers at work, at school, and in the home, more and more people are now able to write programs. Only a small number of these people recognize the underlying features of the programming languages they are using, and even fewer people appreciate the features that are common to most programming languages. This book demonstrates how most programming languages are based upon the same concepts and how knowledge of these concepts can benefit the analyst and the programmer. When specifying computer solutions to real problems, the systems analyst and the programmer must be able to stand back from the particular problem in hand and visualize a solution that is independent of the constraints and limitations imposed by the programming language itself. The text helps in achieving these goals. The book as well is suitable for college students following BTEC and City and Guilds courses in computer studies and IT topics, including professional commercial and end-users.

what is specification by example: Java 2 by Example Geoff Friesen, Jeff Friesen, 2002 Java 2 by Example, Second Edition gives novice programmers in-depth coverage of both object-oriented programming and Java fundamentals. It starts with an overview of Java, including a survey of development tools beginners should use. The book explains the basics of the Java language, including operators, expressions, statements and more; and Object-Oriented Programming with classes and objects, inheritance, and dynamic methods. The author includes a chapter applying the concepts of OOP to object-oriented analysis and design methods. Later chapters demonstrate organizing data in collections and utilizing Java's built-in mathematical functions. Along the way, readers learn from hundreds of examples explaining every concept. Plus, each chapter ends with a series of review questions to ensure that readers are caught up - with answers provided in an appendix.

what is specification by example: *Software Testing Automation* Saeed Parsa, 2023-03-24 This book is about the design and development of tools for software testing. It intends to get the reader involved in software testing rather than simply memorizing the concepts. The source codes are downloadable from the book website. The book has three parts: software testability, fault localization, and test data generation. Part I describes unit and acceptance tests and proposes a new method called testability-driven development (TsDD) in support of TDD and BDD. TsDD uses a machine learning model to measure testability before and after refactoring. The reader will learn how to develop the testability prediction model and write software tools for automatic refactoring. Part II focuses on developing tools for automatic fault localization. This part shows the reader how to use a compiler generator to instrument source code, create control flow graphs, identify prime paths, and slice the source code. On top of these tools, a software tool, Diagnoser, is offered to facilitate experimenting with and developing new fault localization algorithms. Diagnoser takes a source code and its test suite as input and reports the coverage provided by the test cases and the suspiciousness score for each statement. Part III proposes using software testing as a prominent part of the cyber-physical system software to uncover and model unknown physical behaviors and the underlying physical rules. The reader will get insights into developing software tools to generate white box test data.

what is specification by example: Emerging Innovations in Agile Software Development Ghani, Imran, 2016-01-26 Agile is a relatively recent methodology used in the development process of a project. Therefore, it is important to share new emerging knowledge with researchers and professionals interested in adopting an agile mindset. Emerging Innovations in Agile Software

Development focuses on the use of agile methodologies to manage, design, develop, test and maintain software projects. Emphasizing research-based solutions for contemporary software development, this publication is designed for use by software developers, researchers, and graduate-level students in software engineering and project management programs.

what is specification by example: *The Agile Guide to Business Analysis and Planning* Howard Podeswa, 2021-04-05 How Product Owners and Business Analysts can maximize the value delivered to stakeholders by integrating BA competencies with agile methodologies This book will become a staple reference that both product owners and business analysis practitioners should have by their side. -- From the Foreword by Alain Arseneault, former IIBA Acting President & CEO [This book] is well organized in bite-sized chunks and structured for ready access to the essential concepts, terms, and practices that can help any agile team be more successful. -- Karl Wiegers The Agile Guide to Business Analysis and Planning provides practical guidance for eliminating unnecessary errors and delays in agile product development through effective planning, backlog refinement and acceptance criteria specification ---with hard-to-find advice on how and when to analyze the context for complex changes within an agile approach---including when to use Journey Maps, Value Stream Mapping, Personas, Story Maps, BPMN, Use Cases and other UML models. Renowned author and consultant Howard Podeswa teaches best practices drawn from agile and agile-adjacent frameworks, including ATDD, BDD, DevOps, CI/CD, Kanban, Scrum, SAFe, XP, Lean Thinking, Lean Startup, Circumstance-Based Market Segmentation, and theories of disruptive innovation. He offers a comprehensive agile roadmap for analyzing customer needs and planning product development, including discussion of legacy business analysis tools that still offer immense value to agile teams. Using a running case study, Podeswa walks through the full agile product lifecycle, from visioning through release and continuous value delivery. You learn how to carry out agile analysis and planning responsibilities more effectively, using tools such as Kano analysis, minimum viable products (MVPs), minimum marketable features (MMFs), story maps, product roadmaps, customer journey mapping, value stream mapping, spikes, and the definition of ready (DoR). Podeswa presents each technique in context: what you need to know and when to apply each tool. Read this book to Master principles, frameworks, concepts, and practices of agile analysis and planning in order to maximize value delivery throughout the product's lifecycle Explore planning and analysis for short-term, long-term, and scaled agile initiatives using MVPs and data-informed learning to test hypotheses and find high-value features Split features into MMFs and small stories that deliver significant value and enable quick wins Refine, estimate, and specify features, stories, and their acceptance criteria, following ATDD/BDD guidance Address the unique analysis and planning challenges of scaled agile organizations Implement 13 practices for optimizing enterprise agility Supported by 175+ tools, techniques, examples, diagrams, templates, checklists, and other job aids, this book is a complete toolkit for every practitioner. Whatever your role, you'll find indispensable guidance on agile planning and analysis responsibilities so you can help your organization respond more nimbly to a fast-changing environment. Register your book for convenient access to downloads, updates, and/or corrections as they become available. See inside book for details.

what is specification by example: BDD in Action John Smart, 2014-09-29 Summary BDD in Action teaches you the Behavior-Driven Development model and shows you how to integrate it into your existing development process. First you'll learn how to apply BDD to requirements analysis to define features that focus your development efforts on underlying business goals. Then, you'll discover how to automate acceptance criteria and use tests to guide and report on the development process. Along the way, you'll apply BDD principles at the coding level to write more maintainable and better documented code. Purchase of the print book includes a free eBook in PDF, Kindle, and ePub formats from Manning Publications. About the Technology You can't write good software if you don't understand what it's supposed to do. Behavior-Driven Development (BDD) encourages teams to use conversation and concrete examples to build up a shared understanding of how an application should work and which features really matter. With an emerging body of best practices and sophisticated new tools that assist in requirement analysis and test automation, BDD has become a

hot, mainstream practice. About the Book BDD in Action teaches you BDD principles and practices and shows you how to integrate them into your existing development process, no matter what language you use. First, you'll apply BDD to requirements analysis so you can focus your development efforts on underlying business goals. Then, you'll discover how to automate acceptance criteria and use tests to guide and report on the development process. Along the way, you'll apply BDD principles at the coding level to write more maintainable and better documented code. No prior experience with BDD is required. What's Inside BDD theory and practice How BDD will affect your team BDD for acceptance, integration, and unit testing Examples in Java, .NET, JavaScript, and more Reporting and living documentation About the Author John Ferguson Smart is a specialist in BDD, automated testing, and software lifecycle development optimization. Table of Contents PART 1: FIRST STEPS Building software that makes a difference BDD—the whirlwind tour PART 2: WHAT DO I WANT? DEFINING REQUIREMENTS USING BDD Understanding the business goals: Feature Injection and related techniques Defining and illustrating features From examples to executable specifications Automating the scenarios PART 3: HOW DO I BUILD IT? CODING THE BDD WAY From executable specifications to rock-solid automated acceptance tests Automating acceptance criteria for the UI layer Automating acceptance criteria for non-UI requirements BDD and unit testing PART 4: TAKING BDD FURTHER Living Documentation: reporting and project management BDD in the build process

what is specification by example: Software Testing for Managers Ross Radford, 2024-10-07 Software leaders, directors, and managers of all types need to know about software testing. It can be a tough climb up the mountain of technical jargon. Engineers seem to be speaking a language all their own sometimes. Most books on testing are deep in the weeds with technical terms and techniques that simply aren’t applicable even to technical managers. This book provides a high-level perspective on broad topics in a friendly, easy-to-absorb style. Get started and up to speed quickly with immediately useful, actionable guidance. Guidance on team structure, best practices and even common pitfalls will save you time and money, while automation and code reuse will provide exponential value. There’s a gap of knowledge between engineers and their managers; they are almost speaking different languages and the jargon can be confusing. There’s a lot to know about the world of testing. Test from the Top delivers quick, concise guidance to bridge the gap! It offers clear, actionable steps and is a must have for busy leaders who need quick answers. What You Will Learn: How and where to integrate testing in the software development lifecycle Testing terminology and concepts from a management perspective Common pitfalls of testing, how to avoid wasted time How to hire test-aware teams The value in reusing test code for more generalized automation Who This Book is for: Software managers, Lead Software Engineers, Tech Directors, CTOs, Project Managers, software leaders of all kinds. These leaders understand the value of testing, but have not yet built out extensive automation or team structure. Either new to testing concepts or modernizing systems or looking to improve software quality. Assumed to have a working knowledge of the Software Development Lifecycle and basic project management (no specific methodology required).

what is specification by example: *Software Testing Strategies* Matthew Heusser, Michael Larsen, 2023-12-22 Unlock the true potential of software testing to achieve seamless software performance with this comprehensive guide Key Features Gain a solid understanding of software testing and master its multifaceted strategies Empower yourself to effectively overcome software testing challenges Develop actionable real-world testing skills for succeeding in any role Purchase of the print or Kindle book includes a free PDF eBook Book DescriptionIn today’s world, software is everywhere—from entertainment apps to mission-critical systems that support our health, finance, and infrastructure. Testing plays a vital role in ensuring these systems work reliably. Whether you're a software developer, hobbyist, or IT professional, this book will guide you in mastering the art of testing. It’s about asking the right What if? questions, uncovering vulnerabilities, and ensuring software performs as expected throughout its lifecycle. Testing isn't just about automation; it’s a human-driven, creative process that requires skill, and a deep understanding of software behavior.

With practical examples and expert insights, this book helps you craft your own test strategies and explore novel approaches to problem-solving in the testing world. With its help, you'll hone your testing skills with techniques and methodologies rather than tool-based solutions. Authored by experts Matt Heusser and Michael Larson, the book provides valuable strategies for making testing both effective and engaging. Matt is known for his leadership in project rescue initiatives, while Michael's work in accessibility testing has helped shape industry standards. By the end of this book, you'll be equipped to enhance your testing practices and ensure high-quality software in an ever-evolving tech landscape.What you will learn Explore accessibility, functional testing, performance testing, and more as an integral part of testing Find out how to implement a wide range of testing approaches Develop the skills needed to create effective testing strategies tailored to your project's needs Discover how to prioritize and execute the most impactful test ideas Gain insight into when and how to apply different testing elements Defend your chosen testing strategy with a comprehensive understanding of its components Who this book is for This book is for a broad spectrum of professionals engaged in software development, including programmers, testers, and DevOps specialists. Tailored to those who aspire to elevate their testing practices beyond the basics, the book caters to anyone seeking practical insights and strategies to master the nuanced interplay between human intuition and automation. Whether you are a seasoned developer, meticulous tester, or DevOps professional, this comprehensive guide offers a transformative roadmap to become an adept strategist in the dynamic realm of software quality assurance.

what is specification by example: The Handbook of Artificial Intelligence Avron Barr, Edward A. Feigenbaum, 2014-05-12 The Handbook of Artificial Intelligence, Volume II focuses on the improvements in artificial intelligence (AI) and its increasing applications, including programming languages, intelligent CAI systems, and the employment of AI in medicine, science, and education. The book first elaborates on programming languages for AI research and applications-oriented AI research. Discussions cover scientific applications, teiresias, applications in chemistry, dependencies and assumptions, AI programming-language features, and LISP. The manuscript then examines applications-oriented AI research in medicine and education, including ICAI systems design, intelligent CAI systems, medical systems, and other applications of AI to education. The manuscript explores automatic programming, as well as the methods of program specification, basic approaches, and automatic programming systems. The book is a valuable source of data for computer science experts and researchers interested in conducting further research in artificial intelligence.

what is specification by example: Cranked Steve Fenton, Martin Milsom, Dan Horrocks, Divya Murari, 2014-06-15 Cranked helps teams and organisations to effectively deliver software in a changeable or uncertain environment. This book will teach you all about the values, activities and practices that you need to know to delight your customers with your software product. With the techniques in this book you can: - Improve product quality - Release faster and with less errors - Focus on value - Deliver more features - Increase motivation and job satisfaction - Make your customers and end-users happy If you are already working in an agile or lean team, Cranked could accelerate you to the next level. If you are switching to agile or lean - Cranked will help you to avoid common problems in failed agile adoptions. Cranked can be used in any size of organisation to solve complex software development problems.

what is specification by example: Crafting Test-Driven Software with Python Alessandro Molina, 2021-02-18 Get to grips with essential concepts and step-by-step explanations to apply TDD practices to your Python projects while keeping your test suite under control Key FeaturesBuild robust Python applications using TDD and BDD methodologiesTest Python web applications using WebTest and web frameworksLeverage PyTest to implement stringent testing mechanisms to ensure fault-tolerant applicationsBook Description Test-driven development (TDD) is a set of best practices that helps developers to build more scalable software and is used to increase the robustness of software by using automatic tests. This book shows you how to apply TDD practices effectively in Python projects. You'll begin by learning about built-in unit tests and Mocks before covering rich

frameworks like PyTest and web-based libraries such as WebTest and Robot Framework, discovering how Python allows you to embrace all modern testing practices with ease. Moving on, you'll find out how to design tests and balance them with new feature development and learn how to create a complete test suite with PyTest. The book helps you adopt a hands-on approach to implementing TDD and associated methodologies that will have you up and running and make you more productive in no time. With the help of step-by-step explanations of essential concepts and practical examples, you'll explore automatic tests and TDD best practices and get to grips with the methodologies and tools available in Python for creating effective and robust applications. By the end of this Python book, you will be able to write reliable test suites in Python to ensure the long-term resilience of your application using the range of libraries offered by Python for testing and development. What you will learnFind out how tests can make your life easier as a developer and discover related best practicesExplore PyTest, the most widespread testing framework for PythonGet to grips with the most common PyTest plugins, including coverage, flaky, xdist, and pickedWrite functional tests for WSGI web applications with WebTestRun end-to-end tests for web applications using Robot FrameworkUnderstand what test-driven development means and why it is importantDiscover how to use the range of tools available in PythonBuild reliable and robust applicationsWho this book is for This book is for Python developers looking to get started with test-driven development and developers who want to learn about the testing tools available in Python. Developers who want to create web applications with Python and plan to implement TDD methodology with PyTest will find this book useful. Basic knowledge of Python programming is required.

what is specification by example: Behavior-Driven Development in Practice Richard Johnson, 2025-06-14 Behavior-Driven Development in Practice Behavior-Driven Development in Practice is an authoritative guide to mastering BDD as both a philosophy and a set of pragmatic techniques. This comprehensive volume begins by delving into the foundational principles of BDD, emphasizing the pivotal role of communication, shared language, and executable specifications in bridging the gap between stakeholders, developers, and testers. Through detailed exploration of techniques such as example mapping and specification by example, readers gain a nuanced understanding of how to translate complex requirements into clear, testable scenarios that foster genuine collaboration and mutual understanding. The book moves beyond theory into hands-on guidance, offering structured approaches for expressive scenario design, robust automation architecture, and the effective use of popular BDD frameworks such as Cucumber, SpecFlow, and Behave. It thoroughly addresses the realities of modern software development—including working with distributed teams, integrating with CI/CD pipelines, supporting non-functional requirements, and maintaining living documentation. Through practical patterns, anti-patterns, and advanced data-driven strategies, practitioners learn how to scale BDD for enterprise environments, ensure traceability, and meet audit and compliance demands without sacrificing agility or collaboration. Concluding with forward-looking perspectives, Behavior-Driven Development in Practice explores the future of BDD in the age of AI augmentation, cloud-native architectures, and domain-driven design synergies. Drawing on research, industry case studies, and community innovations, the book not only offers a blueprint for successful BDD adoption and continuous improvement but also inspires teams to realize the socio-technical benefits of transparent, outcome-focused software development. Whether you are just starting with BDD or seeking to optimize and future-proof your organizational practices, this book is an indispensable resource for building high-quality software that truly meets business needs.

Related to what is specification by example

Automating Visual Testing with Appraise (InfoQ7y) Developing applications where the look and feel is key for success might benefit from automated visual testing. Appraise, an open source tool on GitHub licensed under MIT, applies the approach of

Automating Visual Testing with Appraise (InfoQ7y) Developing applications where the look and feel is key for success might benefit from automated visual testing. Appraise, an open source tool on

GitHub licensed under MIT, applies the approach of

Back to Home: http://www.speargroupllc.com