

**preface**

**preface to the first edition**

**acknowledgments**

**about this book**

**about the authors**

**about the cover illustration**

## **Part 1 JUnit essentials**

### **1. Chapter 1 JUnit jump-start**

**1.1. Proving it works**

**1.2. Starting from scratch**

**1.3. Understanding unit testing frameworks**

**1.4. JUnit design goals**

**1.5. Setting up JUnit**

**1.6. Testing with JUnit**

**1.7. Summary**

### **2. Chapter 2 Exploring core JUnit**

**2.1. Exploring core JUnit**

**2.2. Running parameterized tests**

**2.3. JUnit test runners**

**2.4. Composing tests with a suite**

**2.5. Summary**

## **3. Chapter 3 Mastering JUnit**

**3.1. Introducing the controller component**

**3.2. Let's test it!**

**3.3. Testing exception handling**

**3.4. Timeout testing**

**3.5. Introducing Hamcrest matchers**

**3.6. Setting up a project for testing**

**3.7. Summary**

## **4. Chapter 4 Software testing principles**

**4.1. The need for unit tests**

**4.2. Test types**

**4.3. Black box versus white box testing**

**4.4. Summary**

## **Part 2 Different testing strategies**

### **5. Chapter 5 Test coverage and development**

**5.1. Measuring test coverage**

**5.2. Writing testable code**

**5.3. Test-driven development**

**5.4. Testing in the development cycle**

**5.5. Summary**

### **6. Chapter 6 Coarse-grained testing with stubs**

**6.1. Introducing stubs**

**6.2. Stubbing an HTTP connection**

**6.3. Stubbing the web server's resources**

**6.4. Stubbing the connection**

**6.5. Summary**

## **7. Chapter 7 Testing with mock objects**

**7.1. Introducing mock objects**

**7.2. Unit testing with mock objects**

**7.3. Refactoring with mock objects**

**7.4. Mocking an HTTP connection**

**7.5. Using mocks as Trojan horses**

**7.6. Introducing mock frameworks**

**7.7. Summary**

## **8. Chapter 8 In-container testing**

**8.1. Limitations of standard unit testing**

**8.2. The mock objects solution**

**8.3. In-container testing**

**8.4. Comparing stubs, mock objects, and in-container testing**

**8.5. Summary**

# **Part 3 JUnit and the build process**

## **9. Chapter 9 Running JUnit tests from Ant**

**9.1. A day in the life**

**9.2. Running tests from Ant**

**9.3. Introducing and installing Ant**

**9.4. Ant targets, projects, properties, and tasks**

**9.5. Putting Ant to the task**

**9.6. Dependency management with Ivy**

**9.7. Creating HTML reports**

**9.8. Batching tests**

**9.9. Summary**

## **10. Chapter 10 Running JUnit tests from Maven2**

**10.1. Maven's features**

**10.2. Setting up a Maven project**

**10.3. Introduction to Maven plug-ins**

**10.4. The bad side of Maven**

**10.5. Summary**

## **11. Chapter 11 Continuous integration tools**

**11.1. A taste of continuous integration**

**11.2. CruiseControl to the rescue**

**11.3. Another neat tool—Hudson**

**11.4. Benefits of continuous integration**

**11.5. Summary**

## **Part 4 JUnit extensions**

### **12. Chapter 12 Presentation-layer testing**

**12.1. Choosing a testing framework**

**12.2. Introducing HtmlUnit**

**12.3. Writing HtmlUnit tests**

**12.4. Integrating HtmlUnit with Cactus**

**12.5. Introducing Selenium**

**12.6. Generating Selenium tests**

**12.7. Running Selenium tests**

**12.8. Writing Selenium tests**

**12.9. HtmlUnit versus Selenium**

**12.10. Summary**

## **13. Chapter 13 Ajax testing**

**13.1. Why are Ajax applications difficult to test?**

**13.2. Testing patterns for Ajax**

**13.3. Functional testing**

**13.4. JavaScript testing**

**13.5. RhinoUnit versus JsUnit**

**13.6. Checking best practices with JSLint**

**13.7. Testing services with HttpClient**

**13.8. Testing Google Web Toolkit applications**

**13.9. Summary**

## **14. Chapter 14 Server-side Java testing with Cactus**

**14.1. What is Cactus?**

**14.2. Testing with Cactus**

**14.3. Testing servlets and filters**

**14.4. Testing JSPs**

**14.5. Testing EJBs**

**14.6. What is Cargo?**

**14.7. Executing Cactus tests with Ant**

**14.8. Executing Cactus tests with Maven2x**

**14.9. Executing Cactus tests from the browser**

**14.10. Summary**

## **15. Chapter 15 Testing JSF applications**

**15.1. Introducing JSF**

**15.2. Introducing the sample application**

**15.3. Typical problems when testing JSF applications**

**15.4. Strategies for testing JSF applications**

**15.5. Testing the sample application with JSFUnit**

**15.6. Using HtmlUnit with JSFUnit**

**15.7. Performance testing for your JSF application**

**15.8. Summary**

## **16. Chapter 16 Testing OSGi components**

**16.1. Introducing OSGi**

**16.2. Our first OSGi service**

**16.3. Testing OSGi services**

**16.4. Introducing JUnit4OSGi**

**16.5. Summary**

## **17. Chapter 17 Testing database access**

**17.1. The database unit testing impedance mismatch**

**17.2. Introducing DbUnit**

**17.3. Using datasets to populate the database**

**17.4. Asserting database state with datasets**

**17.5. Transforming data using ReplacementDataSet**

**17.6. Creating datasets from existing database data**

**17.7. Advanced techniques**

**17.8. Database access testing best practices**

**17.9. Summary**

## **18. Chapter 18 Testing JPA-based applications**

**18.1. Testing multilayered applications**

**18.2. Aspects of JPA testing**

**18.3. Preparing the infrastructure**

**18.4. Testing JPA entities mapping**

**18.5. Testing JPA-based DAOs**

**18.6. Testing foreign key names**

**18.7. Summary**

## **19. Chapter 19 JUnit on steroids**

**19.1. Introduction**

**19.2. Transparent mock usage**

**19.3. DbUnit integration**

**19.4. Assertions made easy**

**19.5. Using reflection to bypass encapsulation**

**19.6. Summary**

## **Appendix A: : Differences between JUnit 3 and JUnit 4**