



Whitebox Testing

Franz-Josef Elmer

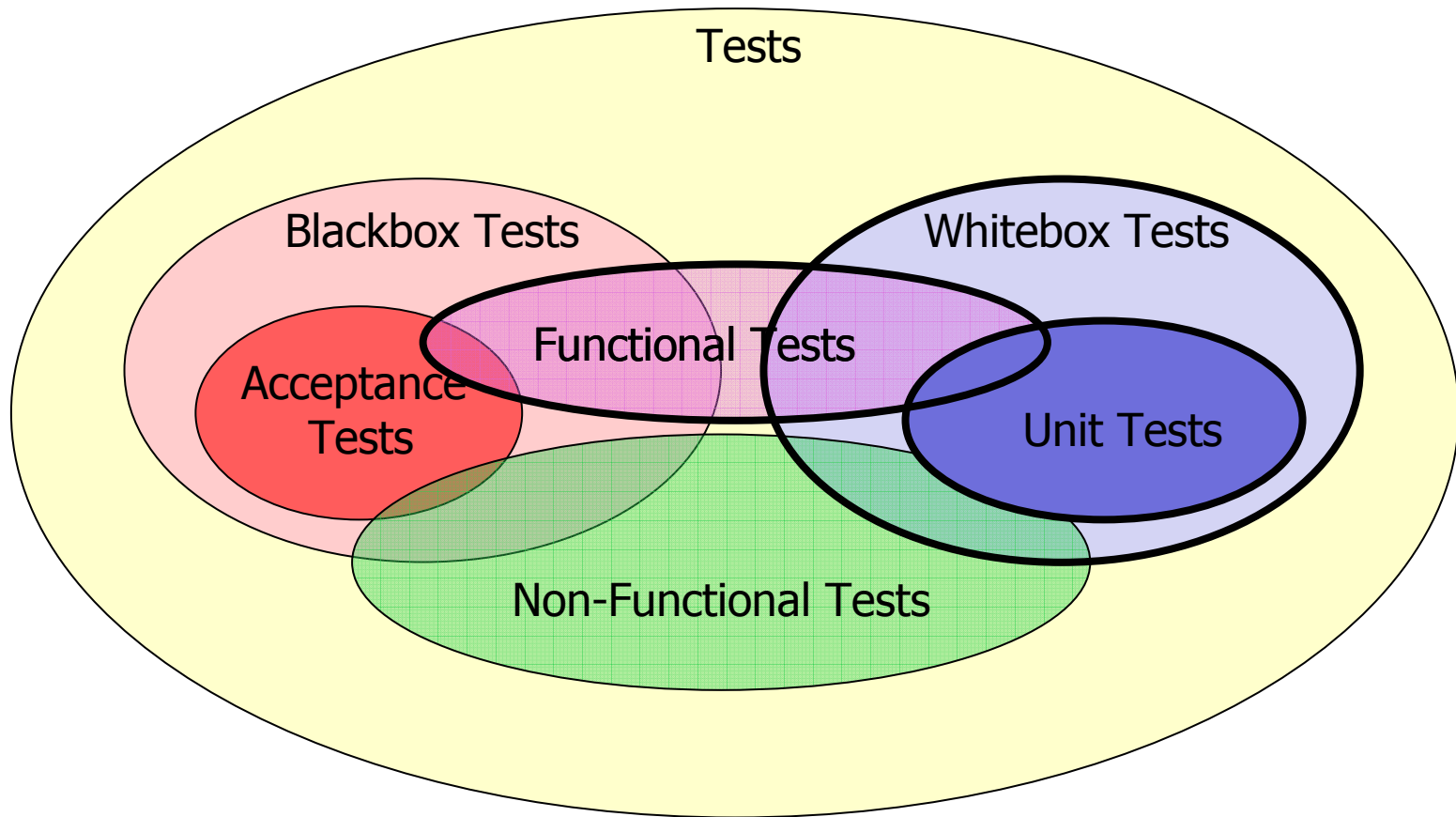


Gliederung

- Begriffe und Konzepte
- JUnit: Unit Tests in Java
- JUnit Demo
- HttpUnit: Funktionale Tests von Web Sites
- HttpUnit Demo am Beispiel Chronos



Begriffe





Unit Test

- Automatisches Test Programm für
 - eine Funktion
 - ein Modul
 - eine Klasse
- Test liefert
 - OK oder
 - Fehlermeldung
- Test Suite
 - Gruppe von Tests
 - Test Suite ist selber ein Test



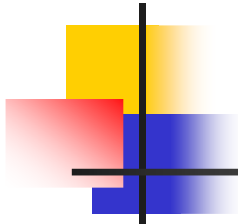
Framework für Unit Tests

Sprache	Framework	Homepage
Java	JUnit	www.junit.org
C++	CppUnit	http://sourceforge.net/projects/cppunit
C#	NUnit, csUnit	www.nunit.org www.csunit.org
PL/SQL	utPLSQL	http://utplsql.sourceforge.net/

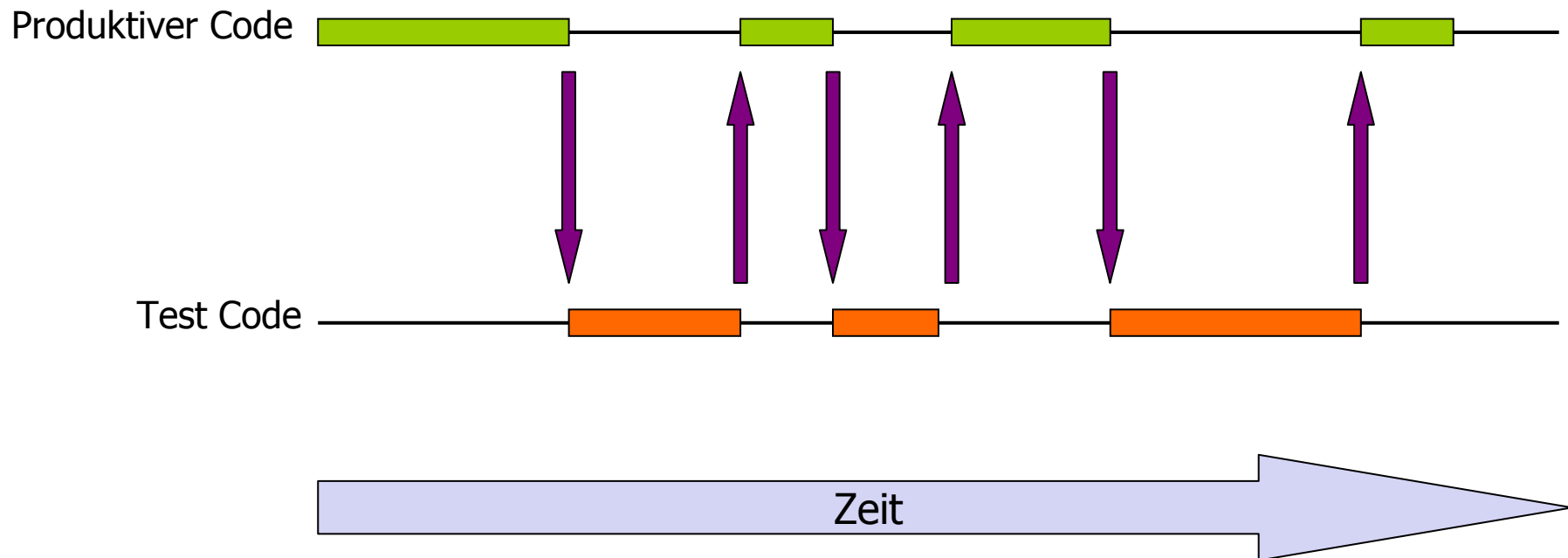


Konzepte im Unit Testing

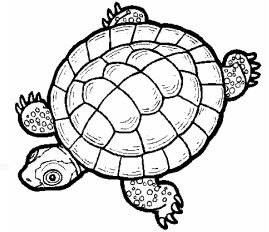
- "Code a little, test a little"
- Mock Objects
- Refactoring
- Test Driven Development



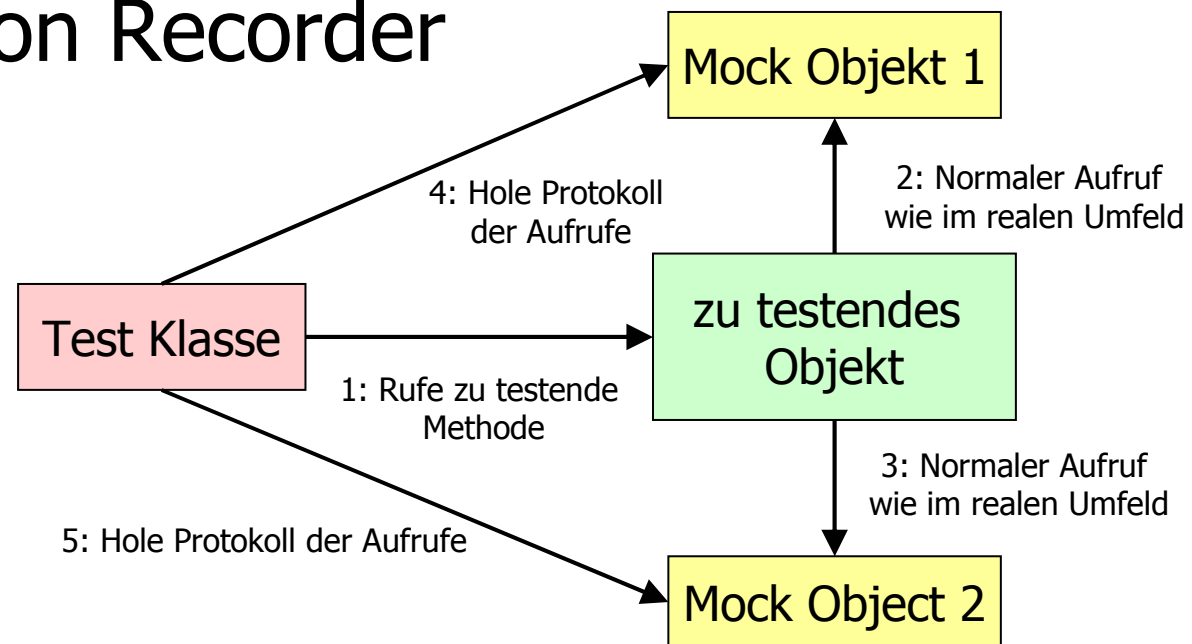
Code a little, test a little



Mock Objects

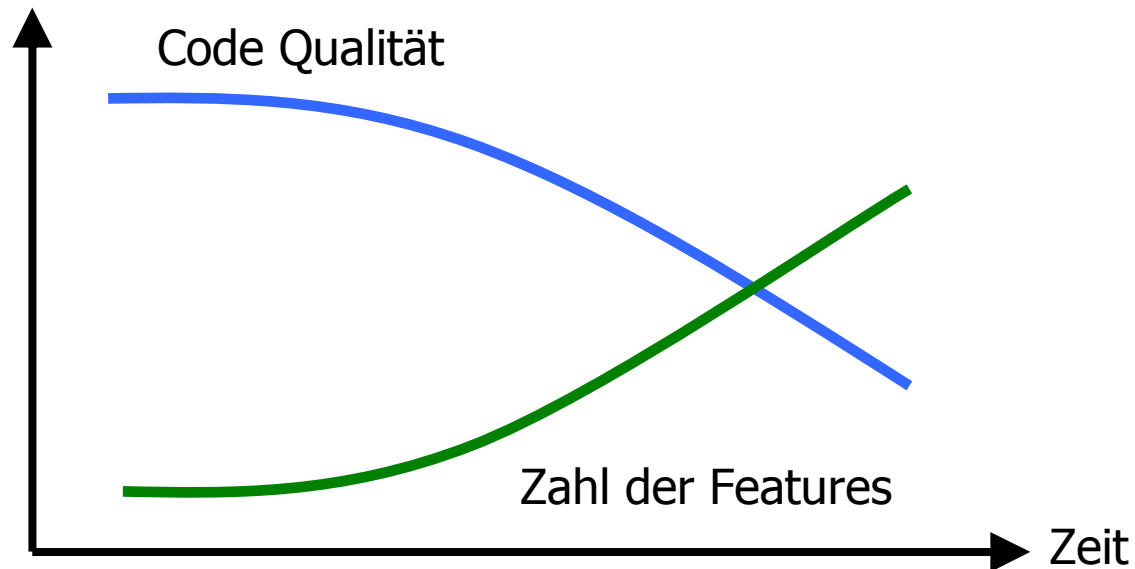


- Mock Turtle aus "Alice in Wonderland"
- Täuschen ein Umgebung vor
- Interaction Recorder



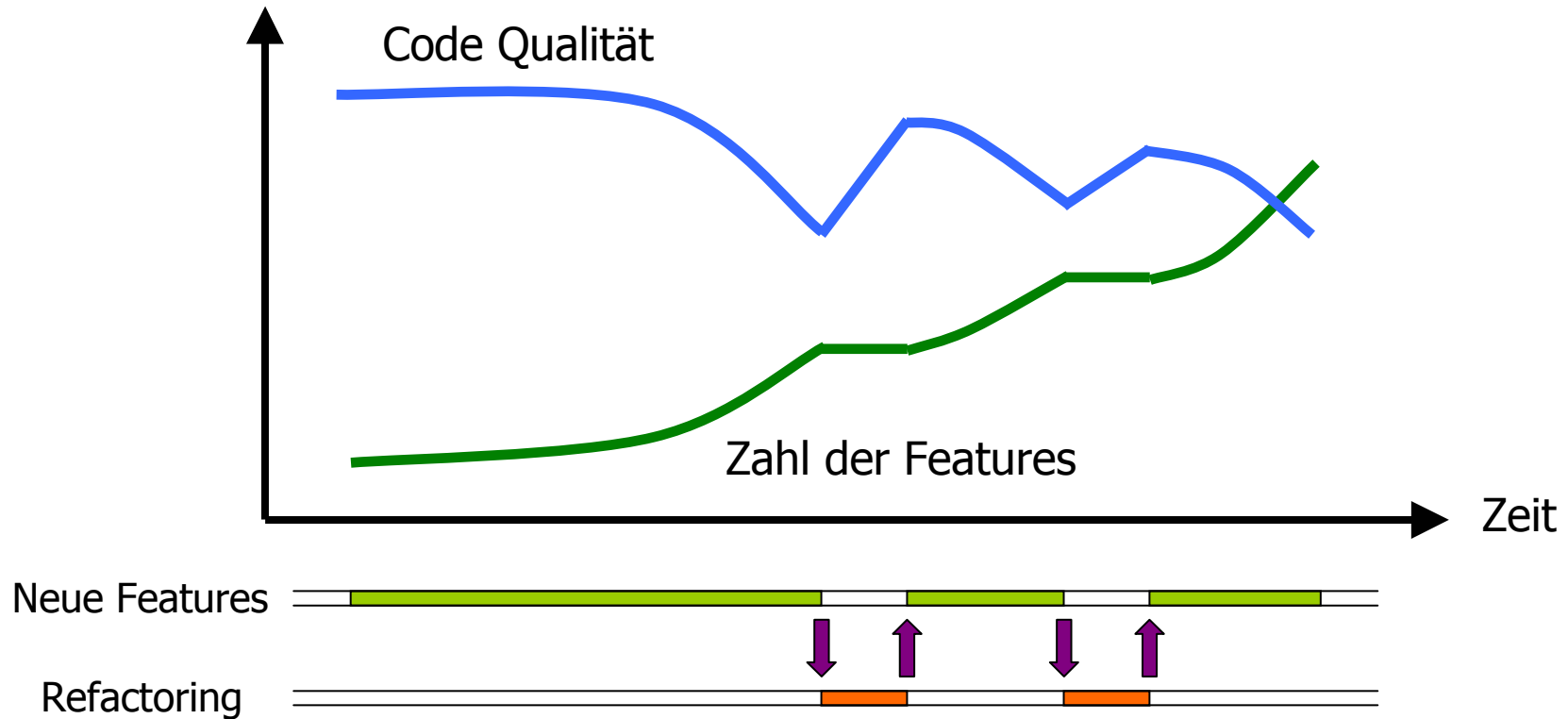
Refactoring

- Allgemeine Erfahrung:



Refactoring

- Code umbauen ohne neue Features zu implementieren



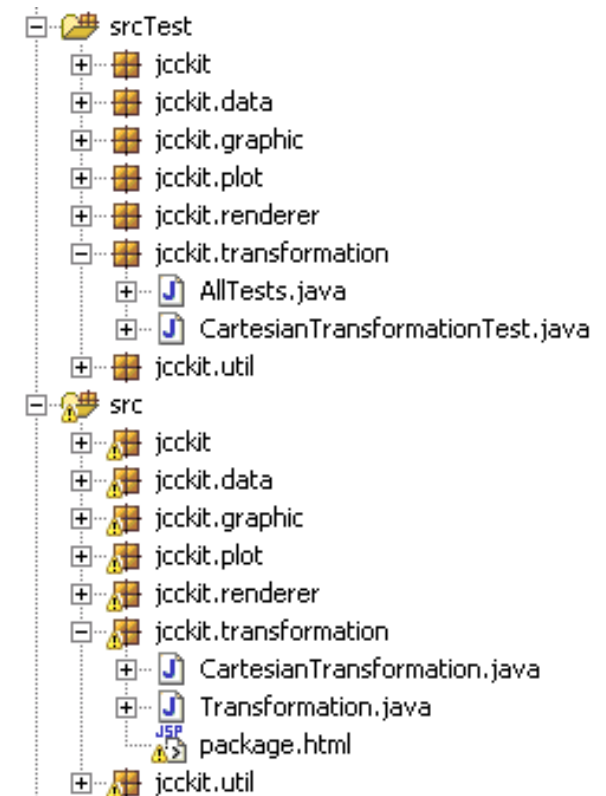


Test Driven Development

- Testcode **vor** funktionalem Code schreiben
- Testcode ist funktionale Spezifikation
- Besseres Design des funktionalen Codes

JUnit

- Basis Klasse: **TestCase**
- Testmethoden:
public void test<X>()
- Namenskonvention:
<ClassToBeTested>Test
- Test Container: **TestSuite**
- Namenskonvention:
AllTests





HttpUnit

- Java Bibliothek um JUnit Tests für Web Sites zu schreiben
- Klasse **WebConversation** simuliert einen Web Browser
- Klasse **WebResponse** repräsentiert eine HTML Page
- Beispiel:

```
WebResponse wr = new WebConversation().getResponse("http://www.google.com");
```



HttpUnit

- Programmatischer Zugriff auf all HTML Elemente wie z.B.:
 - Links
 - Buttons
 - Textfelder
 - Auswahllisten
- Eingeschränkte JavaScript Unterstützung (keine DOM Manipulationen)