

Qualitätsmanagement

Softwareprojekt 2013
Daniel Jahn

Inhalt

- Validation und Verifikation
- Tests
- JUnit
- ECLEmma
- Checkstyle
- Jenkins
- Quellen

Validation und Verifikation

- Überprüfen, ob das System die Anforderungen realisiert (Verifikation)
- Überprüfen, ob die tatsächlichen Anforderungen realisiert wurden (Validation)
- Validation durch Prototyping

Verifikationsverfahren

- Durch Testen der Software:
 - White- und Black-Box Test
 - Integrationstest
 - Systemtest

Ziele der Verifikation

- Nachweis der korrekten Funktionsweise aller Funktionen eines Systems bezogen auf eine vorgegebene Spezifikation
- Verifikationsprozess muss auch verifiziert werden
- Technische Anforderungen müssen auch überprüft werden

Tests

- **White-Box Test:**
 - Anweisungsüberdeckung,
 - Pfadüberdeckung,
 - Kantenüberdeckung beim Kontrollflussgraphen
 - Bedingungsüberdeckung
 - Code Coverage
- **Black-Box Test:**
 - Äquivalenzklassenbildung,
 - Test spezieller Werte,
 - Grenzwertanalyse

JUnit

- Testwerkzeug für Java
- Automatisierte Unit-Tests

Integrationstests

- Test einzelner Module garantiert nicht die Funktionsweise des kompletten Systems
- Dabei werden immer größere Subsysteme getestet

ECLEmma

- Plugin zum analysieren der Code Coverage
- Ergebnisse werden direkt im Source-Code angezeigt

Instruktionen (C0 Coverage)

- Gibt Informationen darüber, welche Instruktionen vom Code aufgerufen wurden

```
public boolean collideBarrier(Barrier b) {  
    Rectangle r1 = new Rectangle((int) x, (int) y,  
        (int) icon.getIconWidth(), (int) icon.getIconHeight());  
    Rectangle r2 = new Rectangle((int) b.getX(), (int) b.getY(),  
        (int) b.getWidth(), (int) b.getHeight());  
    return r1.intersects(r2);  
}  
  
/**  
 * method to inverse the speed on x-axis  
 */  
public void inverseX() {  
    this.speedX = -getSpeedX();  
}  
  
/**  
 * method to inverse the speed on y-axis  
 */  
public void inverseY() {  
    this.speedY = -getSpeedY();  
}
```

Verzweigungen (C1 Coverage)

- Testet die Pfade für alle if und switch statements

```
public Hit hitBarrier(Barrier b) {  
    if (this.collideBarrier(b)) {  
        if (this.x + img.getWidth(null) <= b.getX() + 2)  
            return Hit.Left;  
        else if (this.x >= b.getX() + b.getWidth() - 2)  
            return Hit.Right;  
        else if (this.y + img.getHeight(null) <= b.getY() + 2)  
            return Hit.Up;  
        else if (this.y >= b.getY() + b.getHeight() - 2)  
            return Hit.Down;  
    }  
    return Hit.noHit;  
}
```

Checkstyle

- Tool zum Einhalten eines Coding-Standards
- Standard: Sun-Checks
- Ergebnisse auch direkt im Source-Code

Jenkins

- Build Management Tool
- Erstellen von Builds mit Dokumentation über den Erfolg

Quellen

- Offizielle Seiten der Tools
- Softwaretechnik Skript