

# ITU-VOOP-F2002

## Test & Effektivitet

Lars Thorup

## Dagens overblik

- Opsamling fra sidst
- Test
- Continuous Integration
- Effektivitet
- Til næste gang

# Opsamling fra sidst

- Design Patterns

# Hvorfor teste automatisk

- Det er hurtigere (når man er kommet ind i vanen)
  - Fordi man finder fejl med det samme
  - Fordi en fejl aldrig kommer igen
  - Fordi man slipper for at debugge
  - Fordi andre tør bruge ens nyeste kode hele tiden
- Koden bliver bedre
  - Fordi man tør rette i den
- Projektet er lettere at styre
  - Fordi du hele tiden ved præcis hvor langt du er
  - Fordi du kan release med kort varsel når som helst

# JUnit

- Assert-kald
  - assert(bool)
  - assert(msg, bool)
  - assertEquals(expected, actual)
- Test Suites
- Også til "alle" andre programmeringsprog.
- Også for GUI!

5

# JUnit, TestCase

```
• package dk.itu.voop.conference.common;

import junit.framework.*;

public class DrawingTest extends TestCase {
    public DrawingTest(String name) {
        super(name);
    }
    public static void main(String args[]) {
        junit.textui.TestRunner.run(suite());
    }
    public static Test suite() {
        return new TestSuite(DrawingTest.class);
    }
    public void testAdd() {
        Drawing drawing = new Drawing();
        assertEquals("Shape Count", 0, drawing.getShapeCount());
        LineShape lineShape = new LineShape(anchor, end);
        drawing.add(lineShape);
        assertEquals("Shape Count", 1, drawing.getShapeCount());
    }
}
```

6

# JUnit, TestSuite

- ```
package dk.itu.voop.conference.common;

import junit.framework.*;

public class CommonTestSuite extends TestCase {
    public CommonTestSuite(String name) {
        super(name);
    }
    public static Test suite() {
        TestSuite suite = new TestSuite();
        suite.addTest(LineShapeTest.suite());
        suite.addTest(DrawingTest.suite());
        return suite;
    }
    public static void main(String args[]) {
        junit.textui.TestRunner.run(suite());
    }
}
```

7

## Demo

- StringUtil.java og StringUtilTest.java
- update fra cvs...
- test at alting bygger (N tests ialt)
- skriv StringUtilTest.java
- byg og se at stadig N tests ialt
- tilføj StringUtilTest i CommonTestSuite
- byg og se at nu N+1 tests og 1 der fejler
- implementer StringUtil.java
- byg og se at det virker

8

## Continuous Integration: Why?

- We always know what we have so far
- Everybody is responsible for total system quality
- Realistic planning and prioritization is possible
- Last-minute development is possible
- Blind alleys are spotted fast

## Continuous Integration: How?

- Choose a programming task, only a few hours work
- Update with latest from CVS
- Write test program for the given task
- Run local test suite to ensure that new test fails
- Write code
- Verify code with compiler
- Verify code with local test suite
- Verify code with subsequent project tests
- Update with latest from CVS, repeat previous steps
- Commit changes to CVS

## Continuous Integration, eksempel (1)

- **build.xml:**

```
<project name="conference" default="compile">

  <property name="builddir" value="build"/>

  <target name="init">
    <tstamp/>
    <mkdir dir="${builddir}"/>
  </target>

  <target name="compile" depends="init">
    <javac destdir="${builddir}">
      <src path="."/>
      <include name="dk/itu/voop/conference/**"/>
    </javac>
  </target>

  <target name="clean">
    <delete dir="${builddir}"/>
  </target>
</project>
```

11

## Continuous Integration, eksempel (2)

- **build.xml:**

```
<target name="test" depends="compile">
  <junit printsummary="yes" haltonfailure="no" haltonerror="no">
    <classpath path="${builddir}"/>
    <formatter type="plain" usefile="false"/>
    <test name="dk.itu.voop.conference.ConferenceTestSuite"/>
  </junit>
</target>
```

12

## Effektivitet

- analysere latency ved IPC-kald.
- overveje caching

## Til næste gang

- Udarbejd testprogrammer og programskelet for tegnekonferencesystemet
- Læs Design Patterns:
  - Command
  - Strategy
  - Decorator
  - Flyweight
  - Builder
  - Proxy