

```

package player.playeragent;

import static org.junit.Assert.*;
import edu.neu.ccs.demeterf.demfgen.lib.List;
import gen.Constraint;
import gen.Derivative;
import gen.FinishedProduct;
import gen.Literal;
import gen.RelationNr;
import gen.Type;
import gen.TypeInstance;
import gen.Variable;

import org.junit.Test;

import player.Util;

public class FinishAgentTest {

    private final Type type;
    private final FinishAgent agent;
    private final FinishedProduct fp;
    private final Derivative d;

    public FinishAgentTest() {
        type = new Type(List.create(new TypeInstance(new RelationNr(22))));
        agent = new FinishAgent();
        Derivative tmp = new Derivative(null, null, null, type);
        d = new DeliverAgent().deliverRawMaterial(tmp);
        fp = agent.finishDerivative(d);
    }

    @Test
    public void testAllVariablesAssigned() {
        FinishedProduct fp = agent.finishDerivative(d);
        for (Constraint c : d.optraw.inner().instance.cs) {
            for (final Variable v : c.vs) {
                assertTrue("No assignement for " + v,
                    fp.ip.assignment.literals.contains(new List.Pred<Literal>() {
                        public boolean huh(Literal l) {
                            return l.var.equals(v);
                        }
                    }));
            }
        }
    }

    @Test
    public void testAllVariablesInRawMaterial() {
        for (final Literal l : fp.ip.assignment.literals) {
            d.optraw.inner().instance.cs.contains(new List.Pred<Constraint>() {
                public boolean huh(Constraint c) {
                    return c.vs.contains(l.var);
                }
            });
        }
    }

    @Test
    public void testQuality() {
        double q = fp.quality.val;
        assertTrue("Quality should be > 0 and <= 1 but is " + q, q > 0 && q <= 1);
        double breakEven = Util.breakEvenPoint(type);
        assertTrue("Quality should be > break even point of " + breakEven +
            " but is " + q, q > breakEven);
    }
}

```