

Software Quality



Requirements fulfilment

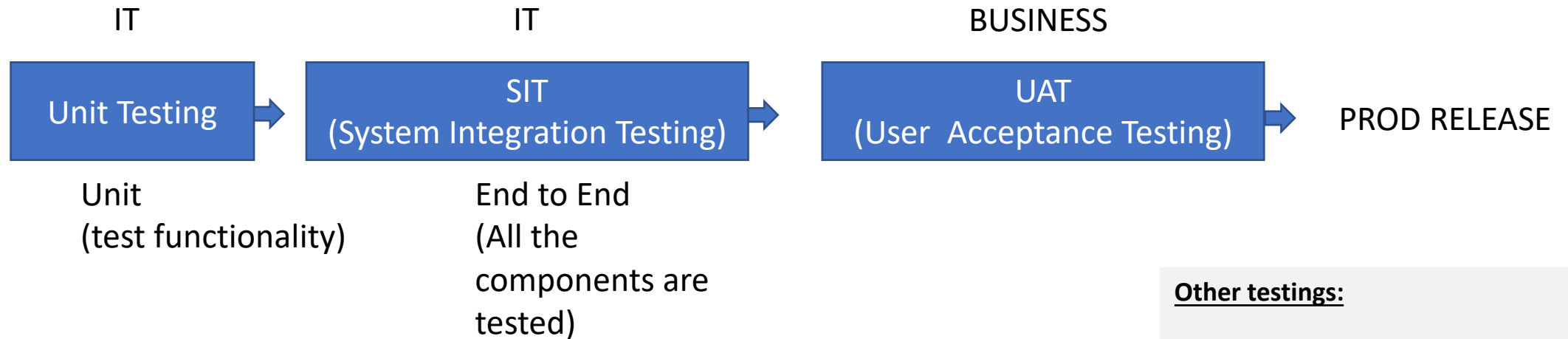
Using Requirements Create Test Cases:

Sample Test case format, (Use excel)

Title, Input, Actual Output, Expected Output, Result (OK or NG)

Sample Test case for Tax Calculation: (Requirement for Gross income Of < 1000 the TAX will be 10%)

1. Calculate tax for Salary less than 1000, 700, 630, OK



Stakeholders (each stakeholder has different interest)

- Primary (Business Owners, Sponsors)
- Secondary (Developers IT, some interest, Compliance)
- Other ()

Different Testing for Different Stakeholders

Other testings:

- Security Testing
- Load Testing
(Cloud scaling of load)
- Performance Test
(Throughput, Response time)
- Regression Testing

Unit testing of Tax Calculator Class

Steps:

1. Write TaxCalculator
2. Write TaxCalculatorTest Unit test
3. Run unit test and check the results

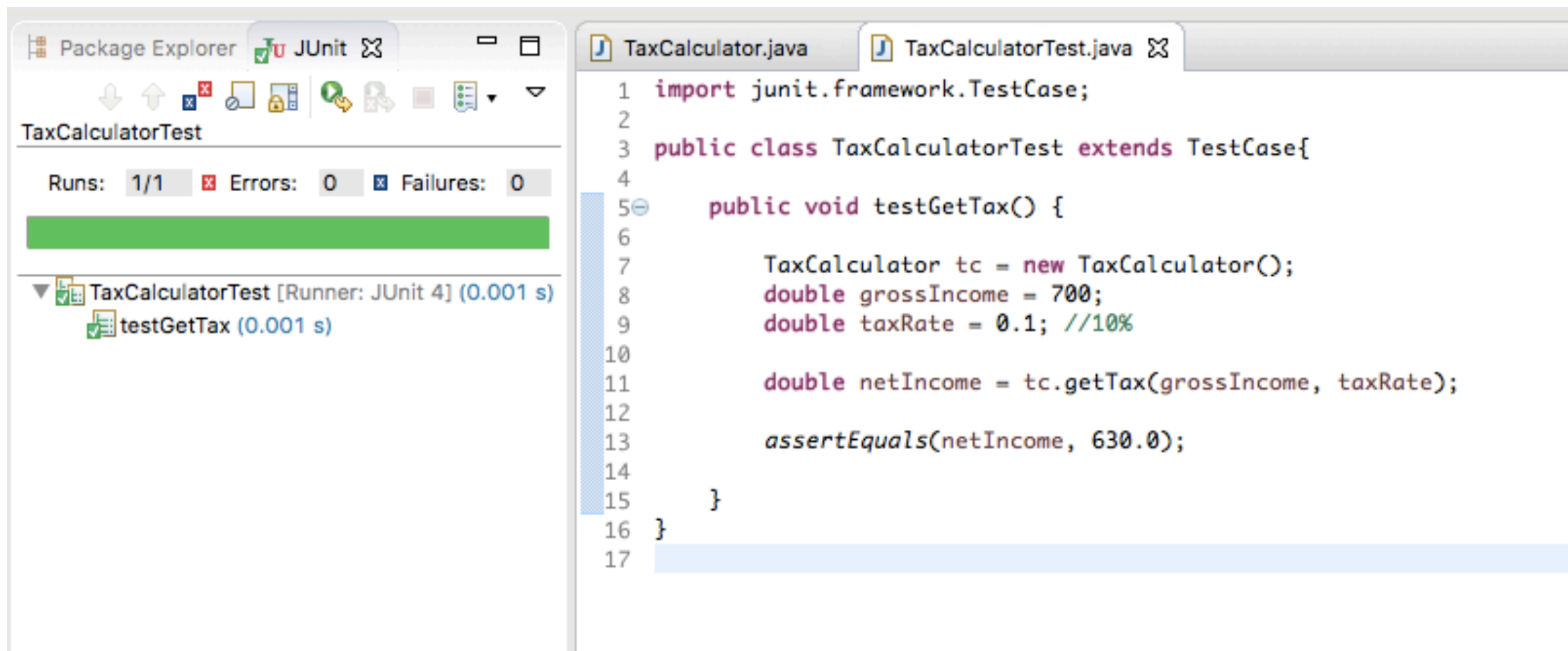
1. TaxCalculator.java

```
2 public class TaxCalculator {
3
4     /**
5      * Method to calculate tax for the specified Gross income and tax rate
6      * @param grossIncome Gross income to calculate the tax
7      * @param taxRate tax rate apply on Gross income
8      * @return net income after deducting tax amount from the gross income
9      */
10    public double getTax(double grossIncome, double taxRate) {
11
12        double taxAmount = 0;
13        double netIncome = 0;
14
15        taxAmount = grossIncome * taxRate;
16        netIncome = grossIncome - taxAmount;
17
18        return netIncome;
19    }
20 }
21
```

2. TaxCalculatorTest.java

```
1 import junit.framework.TestCase;
2
3 public class TaxCalculatorTest extends TestCase{
4
5     public void testGetTax() {
6
7         TaxCalculator tc = new TaxCalculator();
8         double grossIncome = 700;
9         double taxRate = 0.1; //10%
10
11         double netIncome = tc.getTax(grossIncome, taxRate);
12
13         assertEquals(netIncome, 630.0);
14
15     }
16 }
```

3. Run unit test and check the results



The screenshot displays an IDE interface with two main panels. The left panel, titled 'Package Explorer', shows a project named 'TaxCalculatorTest'. Below the project name, it indicates 'Runs: 1/1', 'Errors: 0', and 'Failures: 0'. A green progress bar is visible. Underneath, a tree view shows 'TaxCalculatorTest [Runner: JUnit 4] (0.001 s)' expanded, revealing a single test 'testGetTax (0.001 s)' with a green checkmark icon. The right panel shows the source code for 'TaxCalculatorTest.java'. The code is as follows:

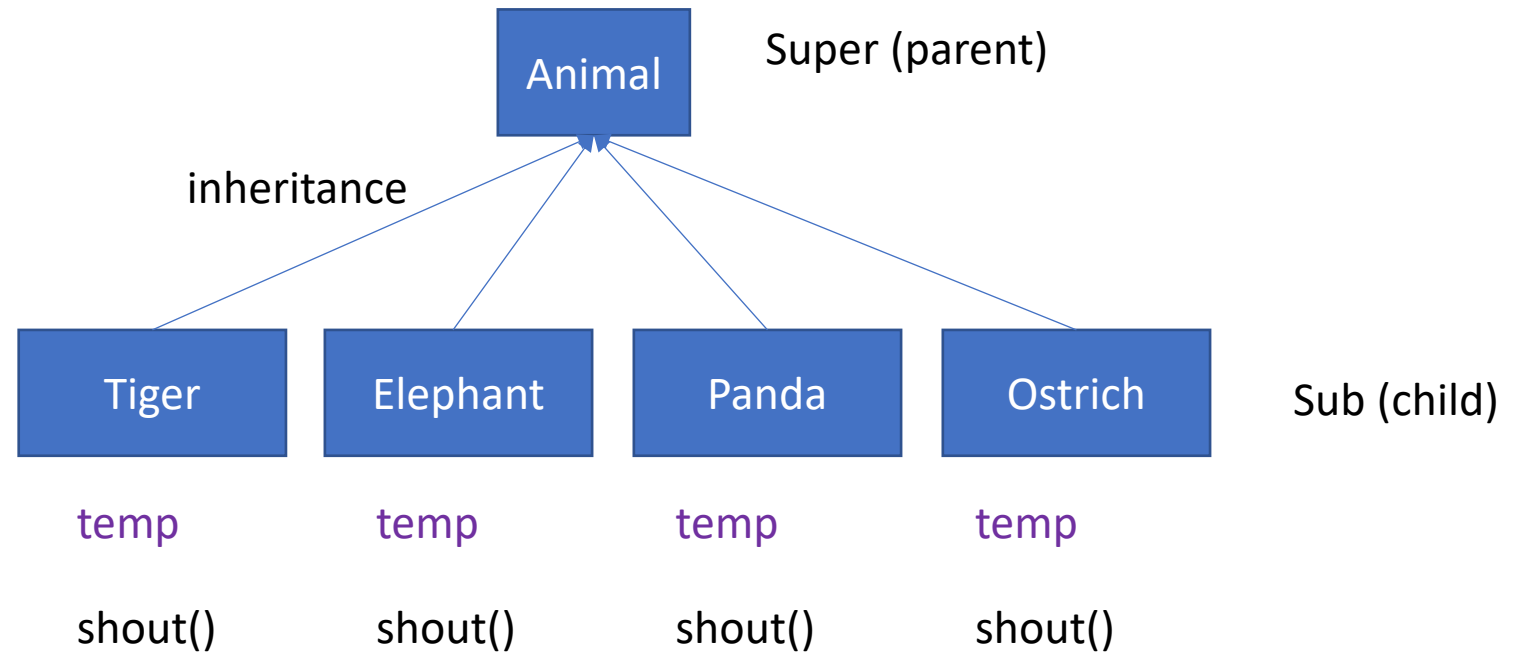
```
1 import junit.framework.TestCase;
2
3 public class TaxCalculatorTest extends TestCase{
4
5     public void testGetTax() {
6
7         TaxCalculator tc = new TaxCalculator();
8         double grossIncome = 700;
9         double taxRate = 0.1; //10%
10
11         double netIncome = tc.getTax(grossIncome, taxRate);
12
13         assertEquals(netIncome, 630.0);
14
15     }
16 }
17
```

Inheritance

Monitor Zoo:

Application to check health of the animals

(Check the room humidity, Check the body temperature)



Design with Inheritance

