
# Test Driven Development

Chamil Jeewantha

ZONE24x7

ICTA
ideas actioned

# What is a Unit Test?

- Select the **smallest piece** of testable software in the application
- **Isolate it** from the rest of the code
- **Determine** whether it **behaves** exactly **as you expect**
- Each unit is **tested separately** before integrating them into modules to test the interfaces between modules

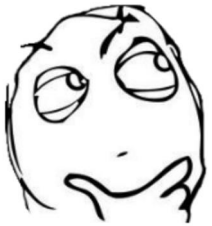
Ref: https://msdn.microsoft.com/en-us/library/aa292197(VS.71).aspx

# Characteristics of a Good Unit Test

- Automated
- Thorough
- Repeatable
- Independent
  - Test only one thing
  - Should not rely on each other
- Fast
- Professional (Readable, Maintainable, Trustworthy)

# Isolating A Unit for Testing

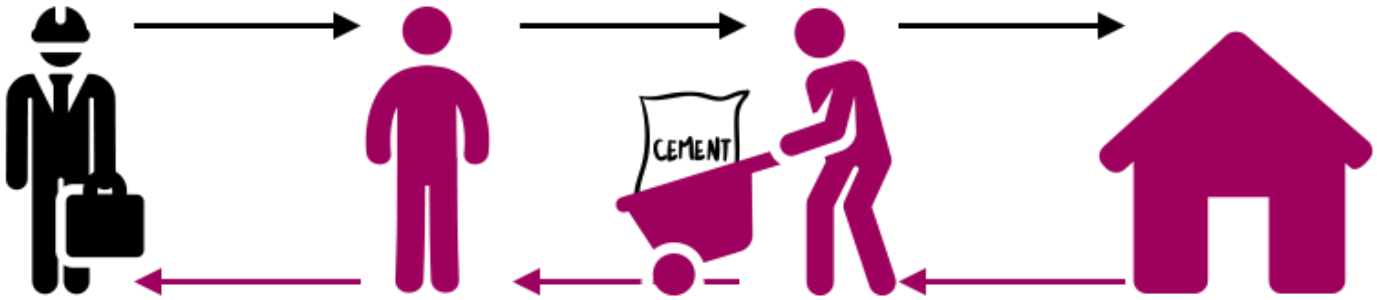
Is this Site Manager good?



Site Manager

He should by cement from the shop named by me

# Test Flow



# How to Test

- Create a dummy bag of cement
- Create a dummy hardware shop
- Instruct the dummy hardware to send the dummy bag of cement if somebody asks for a bag of cement.
- Give the address of dummy hardware to the site manager.
- Ask for a bag of cement from the site manager
- Verify whether he has called the given hardware
- Verify whether we receive the dummy cement bag back.

## How to Test

```
BagOfCement dummyCement = mock(BagOfCement.class);
HwShop dummyHw = mock(HwShop.class);
When(dummyHw.buyCement()).thenReturn(dummyCement);
siteManager.setHwShop(dummyHw);

BagOfCement output = siteManager.askCement();

assertThat(output, is(dummyCement));
verify(dummyHw).buyCement();
```

# How To Test?

```
public class AndCriteria implements Criteria{

   private final Criteria criteria1;
   private final Criteria criteria2;

   public AndCriteria(Criteria criteria1, Criteria criteria2) {
      this.criteria1 = criteria1;
      this.criteria2 = criteria2;
   }


   public List filter(List values) {
      List filteredList = criteria1.filter(values);
      return criteria2.filter(filteredList);
   }
}
```


ZONE24x7 ICTA ideas actioned

# Example: Requirement of AndCriteria

- The AndCriteria should be an implementation of Criteria
- Should filter a given list by first expression and return a filteredList
- Should filter the filteredList by the second expression and return the finalList

# Example: Tests for AndCriteria

1. Should_**FilterThrough1StAnd2ndExpressions**_When_**AListIsGiven**

2. Should_**ThrowException**_When_**NullIsProvided**

# Example: AndCriteria : Test 1

```
@Test
public void
Should_FilterThrough1StAnd2ndExpressions_When_AListIsGiven(){
    List input = // dummy list
    List lst1 = // dummy list
    List lst2 = // dummy list

   Criteria expr1 = // dummy criteria -> filter(input) returns dummy list (lst1)
   Criteria expr2 = // dummy criteria -> filter(lst1) returns dummy list (lst2)

   AndCriteria criteria = new AndCriteria(expr1, expr2);
   List output = criteria.filter(input);
   assertThat(output, is(lst2));
}
```

# Example: AndCriteria : Test 2

```
@Test (expected = IllegalArgumentException.class)
public void Should_ThrowException_When_NullIsProvided() {
    List input = null
   AndCriteria criteria = new AndCriteria(expr1, expr2);
   criteria.filter(input);
}
```


ZONE24x7 ICTA ideas actioned

# Example 2: Evaluate Expression

- The user should provide an expression with relevant value mappings to its variables.
  - Java Evaluate ((a+b)*3)-2  a=5 b=8

- The program should assign a & b values to this expression and evaluate it.

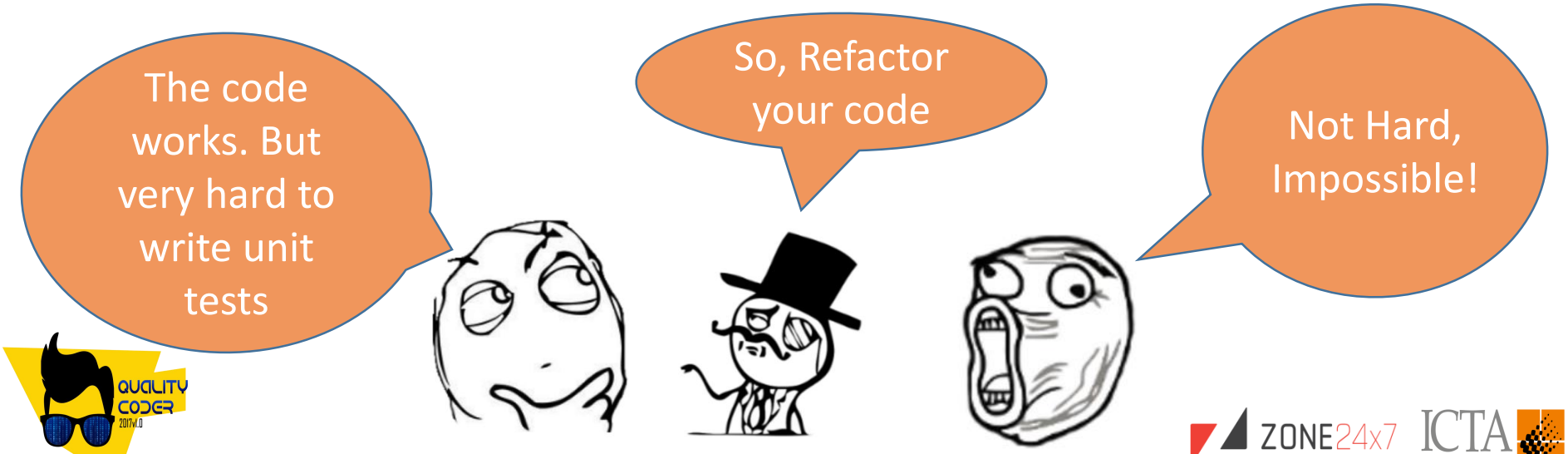
# Code for Evaluating an Expression

```
class Evaluate{
    public static void main(String[] args){
        String expr = ... // assign variables with values
        // evaluate the expr
        double value = // the output value of the evaluation
        System.out.println(value);
    }
}
```

Can you write a "Good" automated test?

# Common Complaints About Unit Testing

- Deadline is near, **no time** to write tests
- Writing tests **takes longer than** the **production code**
- **Hard to keep** the test suite **up to date**
- **One line of code change breaks 100s of tests**



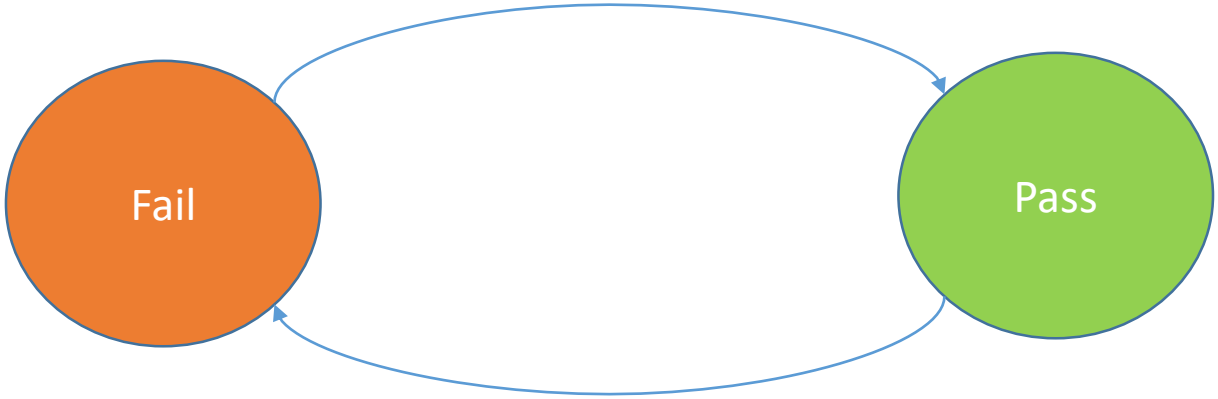
# Solution

Test First Development (TFD) ?


ZONE24x7 ICTA ideas actioned

# Rules: TFD

1. **Write** a(nother) unit **test that fails**
2. **Write** the **minimum production code** until all the tests pass
3. **Repeat** until all your work is done.



# Test First Benefits (Vs Test Late)

All the benefits of Unit testing

**+**

- Write **non-testable codes are impossible**
- Test-first **forces** you to **plan before you code**
- It's **faster than writing** code **without tests**
- It **saves you from lengthy code**
- It **guides you** to **build** good, **SOLID units**
- It **increases your confidence** (**refactor without fear**)
- Acts as a **real-time progress bar**

## What is TDD ?

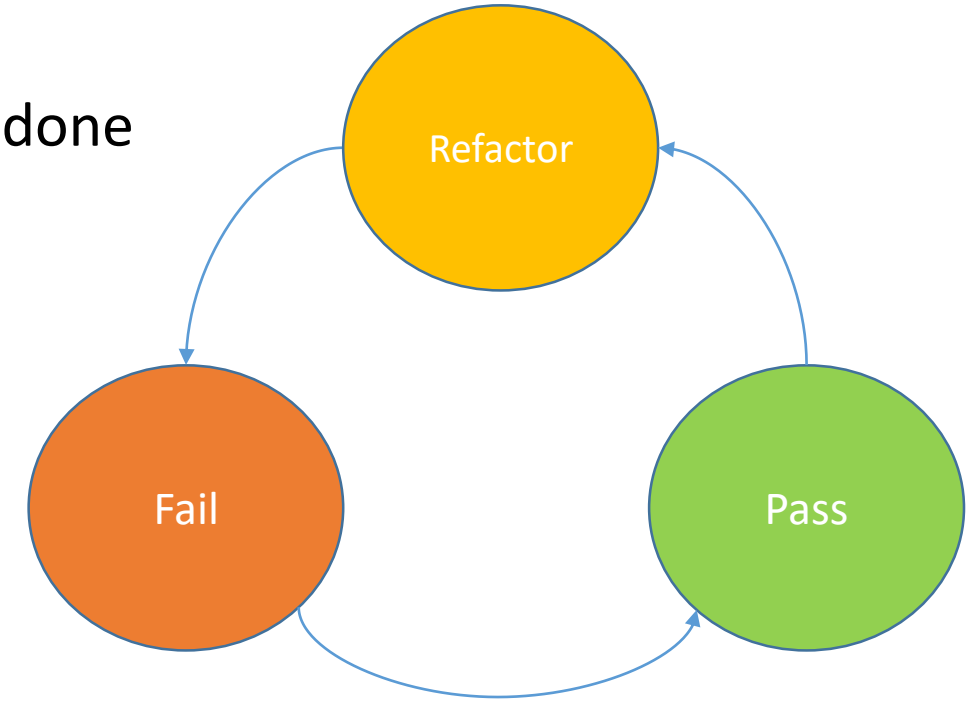
# TDD = TFD + Refactoring

- TDD = Test Driven Development
- TFD = Test First Development


ZONE24x7 ICTA ideas actioned

# Rules: TDD

1. Write a(nother) unit test that fails
2. Write the minimum production code until all the tests pass
3. **Refactor your code**
4. Repeat until all your work is done



# DEMO

Get your hands Dirty with TDD


ZONE24x7 ICTA ideas actioned

## Tools

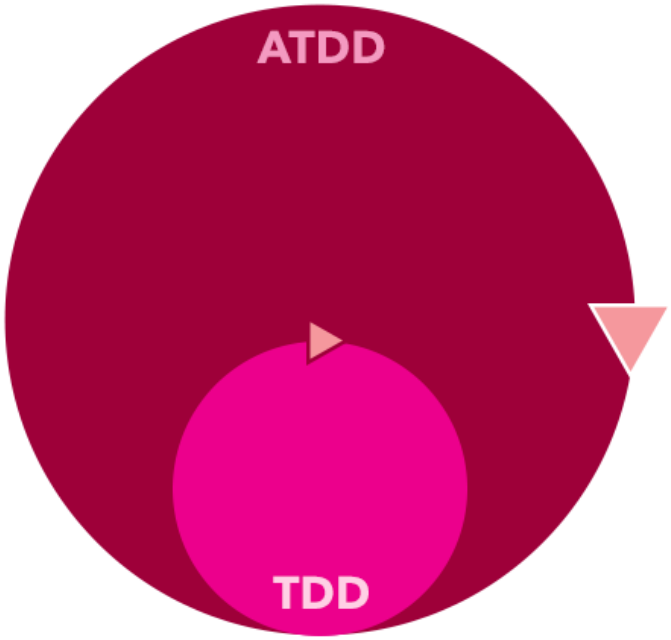
- IDE (IntelliJ)
- Test Runner (Junit)
- Mock libraries (Mockito)
- Verification (Hamcrest)

# TDD - FAQ


ZONE24x7 ICTA ideas actioned

# ATDD vs TDD

- **TDD** to **drive the design**
- **ATDD** to make sure all the **requirements are implemented**



## Additional time due to TDD?

- Beginner
  - Lots of time thinking where to start
- Experienced Developer
  - Initially 15 - 17% more
- Big time saving later for both


ICTA
ideas actioned

# Thank You!

## Lets practice…

Chamil Jeewantha


ZONE24x7

ICTA
ideas actioned