



Test-Driven Development

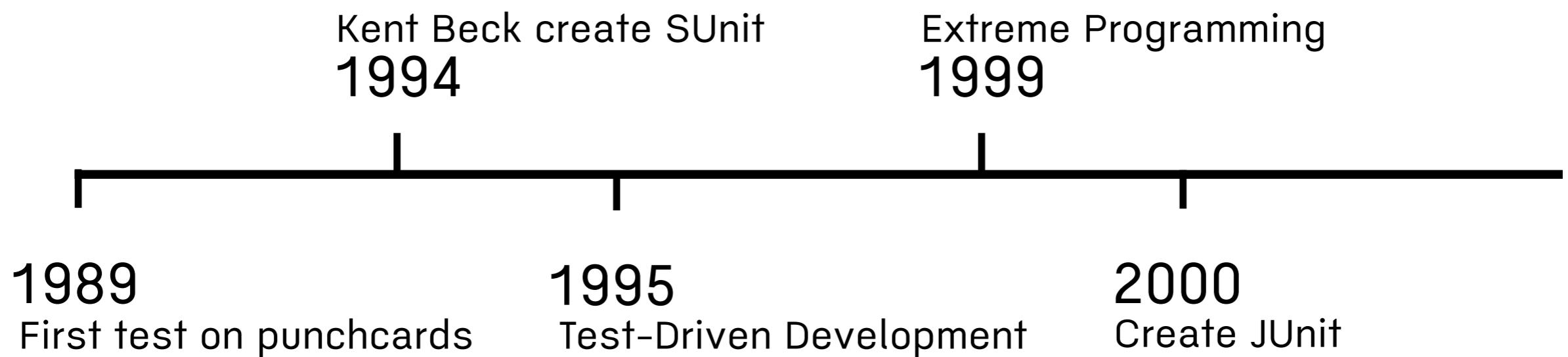
in Java

SPRINT3R

Siam Chamnankit Co., Ltd., Odd-e (Thailand) Co., Ltd. and Alliance

What is TDD ?

Testing timeline

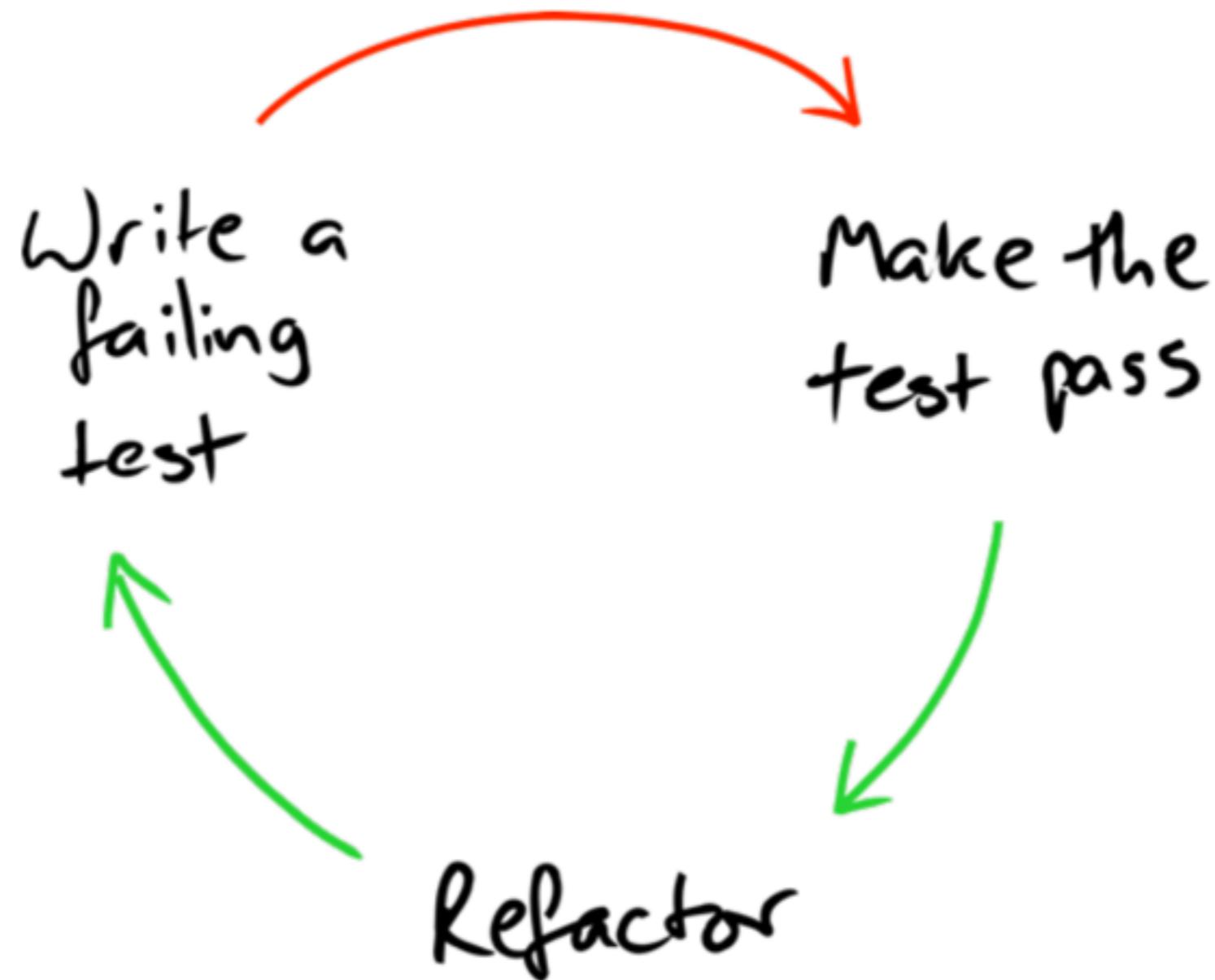


Simple TDD Rules

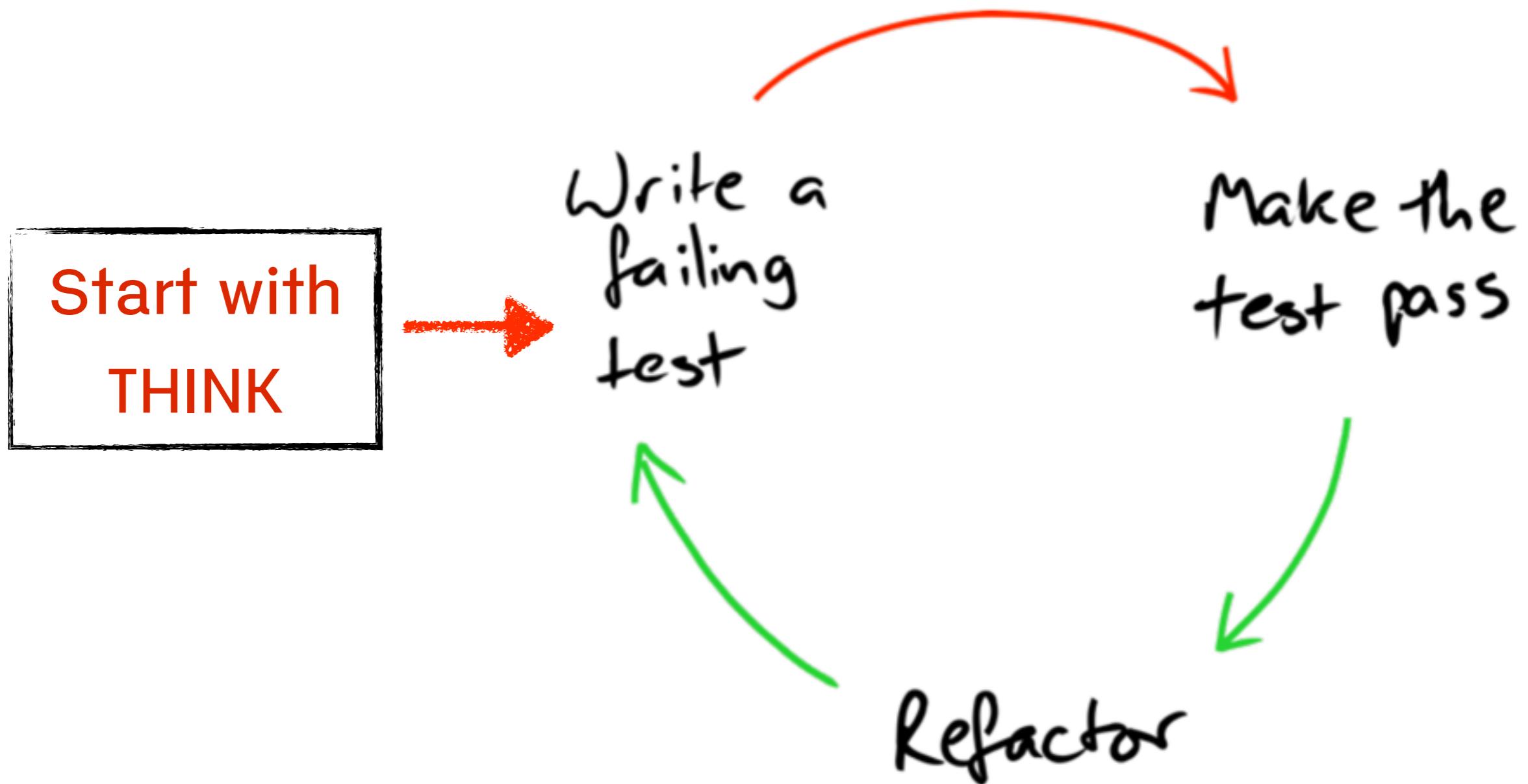
Write a failing automated test before
you write any code

Remove duplication

TDD Cycle



Improve TDD Cycle



วิธีการทำให้ Test Pass

Fake or Hard code

Triangulation

Use obvious implementation

Why TDD ?

ทำไมต้อง TDD

เพื่อทำความเข้าใจกับปัญหาต่างๆ

คุณเข้าใจปัญหาหรือไม่ ?

ทำไมต้อง TDD

Software **เปลี่ยนแปลงอยู่เสมอ**

คนเรามัก**โน้ม** และ **ทำผิดพลาดอยู่เสมอ**

ทำไมต้อง TDD

Test นั้นช่วยคุณสร้างสิ่งที่ **น้อม**
และสร้างความคาดหวังขึ้นมา

ทำไมต้อง TDD

บอกคุณว่า ผิดตรงไหน

ไม่ต้องเสียเวลา **debug**

ทำไมต้อง TDD

ได้รับ feedback ที่รวดเร็ว

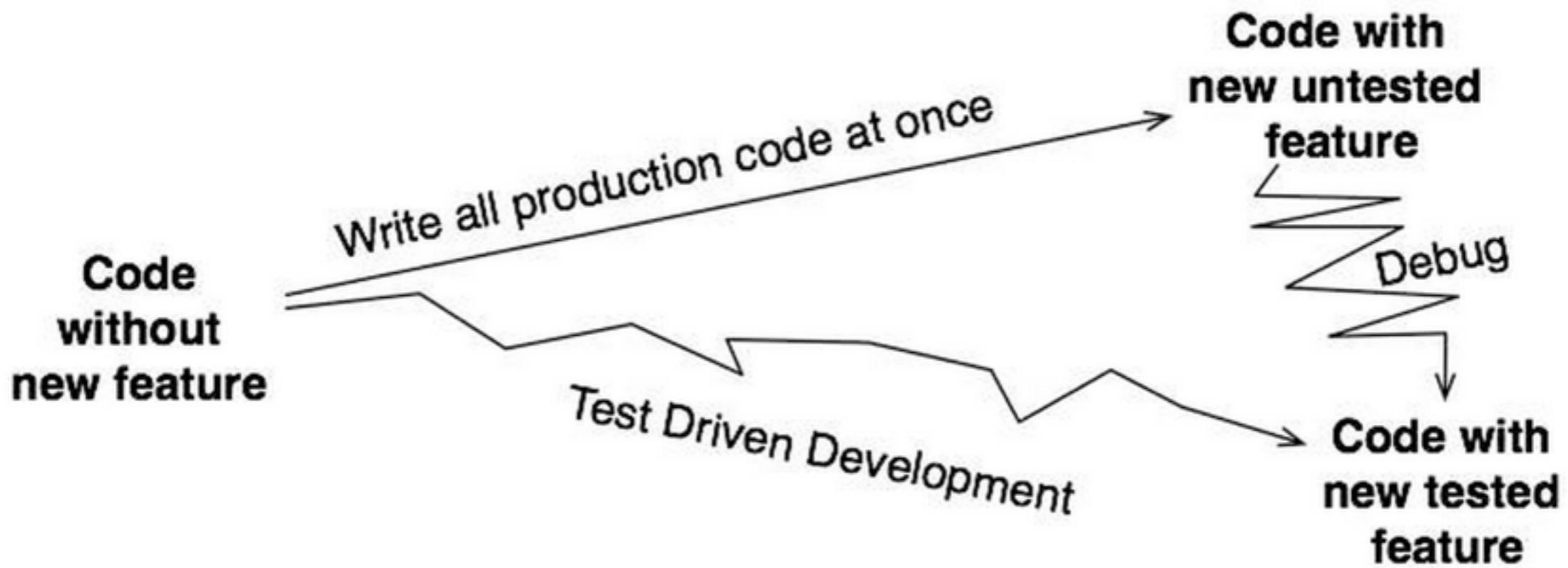
Many small changes vs One Big Change

ทำไมต้อง TDD

ให้ developer ทำงานเป็นทีมได้ดีขึ้น

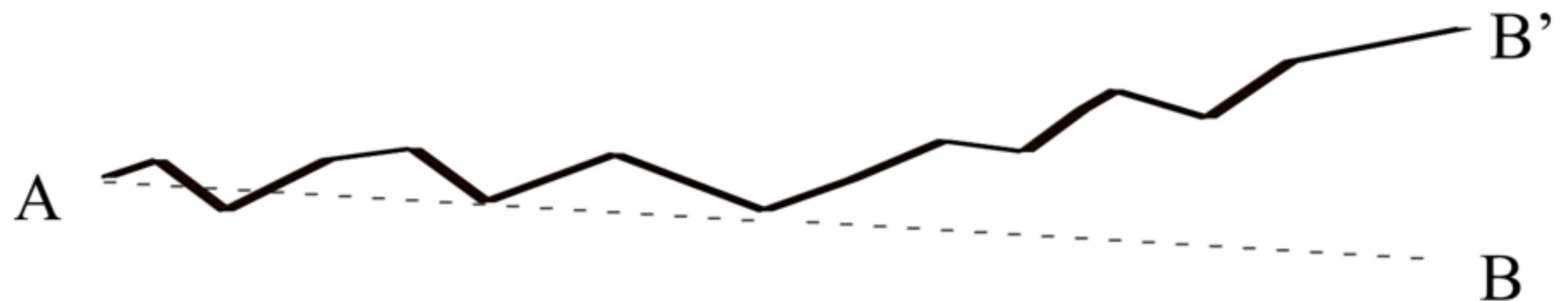
TDD vs DLP

DLP = Debug Later Programming



รูปจาก <http://www.renaissancesoftware.net/>

Remember :: Small step



Proactive vs Reactive

นำไปสู่หาก่อน

ทดสอบทีหลัง

ตรวจสอบ req ก่อนทำ

แก้ไขตาม defect ที่แจ้ง

ตรวจสอบ code

debug code

Test ที่ดีต้อง FIRST

Fast
Independent
Repeatable
Self-validating
Timely

สิ่งที่ไม่ใช่ Unit test

External service

File system

Database

WebService

API calls

สิ่งที่ไม่ใช่ Unit test

Over specification

Compare screen image

Compare HTML

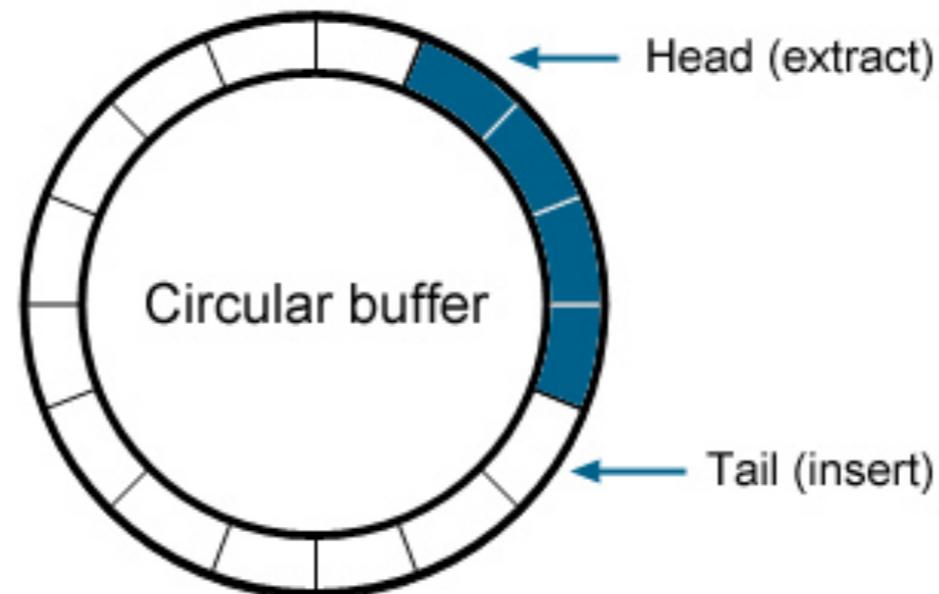
Workshop #0



Workshop #1

Circular Buffer

http://en.wikipedia.org/wiki/Circular_buffer



Unit

with **JUnit**



Anatomy JUnit 4

Test Name

```
public class FizzBuzzTest {
```

Annotation

```
@Test
```

Test Case Name

```
public void sayFizzWhenNumberIsDevidedByThree() {
```

```
    FizzBuzz fizzBuzz = new FizzBuzz();
```

```
    String actualResult = fizzBuzz.say(3);
```

```
    assertEquals("Fizz", actualResult);
```

```
}
```

```
}
```

Test Naming



“What’s in a name?”

That which we call a rose

by any other name would smell

as sweet

Romeo and Juliet

Guide to Test Writing

Don't say “**test**”, say “**should**”

Don't use the word “test”



Use the word “should”

```
transferShouldDeductSumFromSourceAccountBalance()  
transferShouldAddSumLessFeesToDestinationAccountBalance()  
depositShouldAddAmountToAccountBalance()
```

Guide to Test Writing

Don't test your class, test **behaviour**

Guide to Test Writing

Test class names are important too

Structure your test well

Tests are **deliverable** too

Test Structure



The ratio of time spent (code)
versus writing is
over 10 to 1

Robert C. Martin, Clean Code

Good Unit Test

```
@Test  
public void sayFizzWhenNumberIsDevidedByThree() {  
Arrange   FizzBuzz fizzBuzz = new FizzBuzz();  
Act       String actualResult = fizzBuzz.say(3);  
Assert    assertEquals("Fizz", actualResult);  
}
```



Setup Pattern

```
@Test  
public void sayFizzWhenNumberIsDevidedByThree() {  
    FizzBuzz fizzBuzz = new FizzBuzz();  
  
    String expectedResult = fizzBuzz.say(3);  
  
    assertEquals("Fizz", expectedResult);  
}  
  
@Test  
public void sayBuzzWhenNumberIsDevidedByFive() {  
    FizzBuzz fizzBuzz = new FizzBuzz();  
  
    String expectedResult = fizzBuzz.say(5);  
  
    assertEquals("Buzz", expectedResult);  
}
```

Setup Pattern

Inline
Setup

```
@Test
public void sayFizzWhenNumberIsDevivedByThree() {
    FizzBuzz fizzBuzz = new FizzBuzz();

    String actualResult = fizzBuzz.say(3);

    assertEquals("Fizz", actualResult);
}

@Test
public void sayBuzzWhenNumberIsDevivedByFive() {
    FizzBuzz fizzBuzz = new FizzBuzz();

    String actualResult = fizzBuzz.say(5);

    assertEquals("Buzz", actualResult);
}
```

Setup Pattern

Delegate
Setup

```
@Test  
public void sayFizzWhenNumberIsDividedByThree() {  
    FizzBuzz fizzBuzz = setup();  
    String actualResult = fizzBuzz.say(3);  
    assertEquals("Fizz", actualResult);  
}
```

```
@Test  
public void sayBuzzWhenNumberIsDividedByFive() {  
    FizzBuzz fizzBuzz = setup();  
    String actualResult = fizzBuzz.say(5);  
    assertEquals("Buzz", actualResult);  
}
```

```
private FizzBuzz setup() {  
    FizzBuzz fizzBuzz = new FizzBuzz();  
    return fizzBuzz;  
}
```

Setup Pattern

Implicit Setup

```
@Before  
public void initial() {  
    fizzBuzz = new FizzBuzz();  
}
```

```
@Test  
public void sayFizzWhenNumberIsDevindedByThree() {  
    String actualResult = fizzBuzz.say(3);  
    assertEquals("Fizz", actualResult);  
}
```

```
@Test  
public void sayBuzzWhenNumberIsDevindedByFive() {  
    String actualResult = fizzBuzz.say(5);  
    assertEquals("Buzz", actualResult);  
}
```

Implicit Teardown

```
@After  
public void clearResources(){  
    fizzBuzz = null;  
}
```

Ignore Testcase

```
@Ignore("Pending implemetation")
@Test
public void sayFizzWhenNumberIsDevidedByThree() {
    FizzBuzz fizzBuzz = new FizzBuzz();
    String actualResult = fizzBuzz.say(3);
    assertEquals("Fizz", actualResult);
}
```

Workshop #2

LCD Digits



Workshop #3

Kata Range

<http://codingdojo.org/cgi-bin/index.pl?KataRange>

Handle Exception

Traditional approach

```
@Test  
public void invalidWhenNumberLessThanOne() {  
    try {  
        fizzBuzz.say(0);  
        fail();  
    } catch (IllegalStateException expected) {  
    }  
}
```

Expected Annotation

```
@Test(expected=IllegalStateException.class)
public void invalidWhenNumberLessThanOne() {
    fizzBuzz.say(0);
}
```

ExpectedException Rule

```
@Rule  
public ExpectedException thrown = ExpectedException.none();  
  
@Test  
public void invalidWhenNumberLessThanOne() {  
    thrown.expect(IllegalStateException.class);  
    fizzBuzz.say(0);  
}
```

Data-Driven with JUnit

@Parameterized

Using parameterized

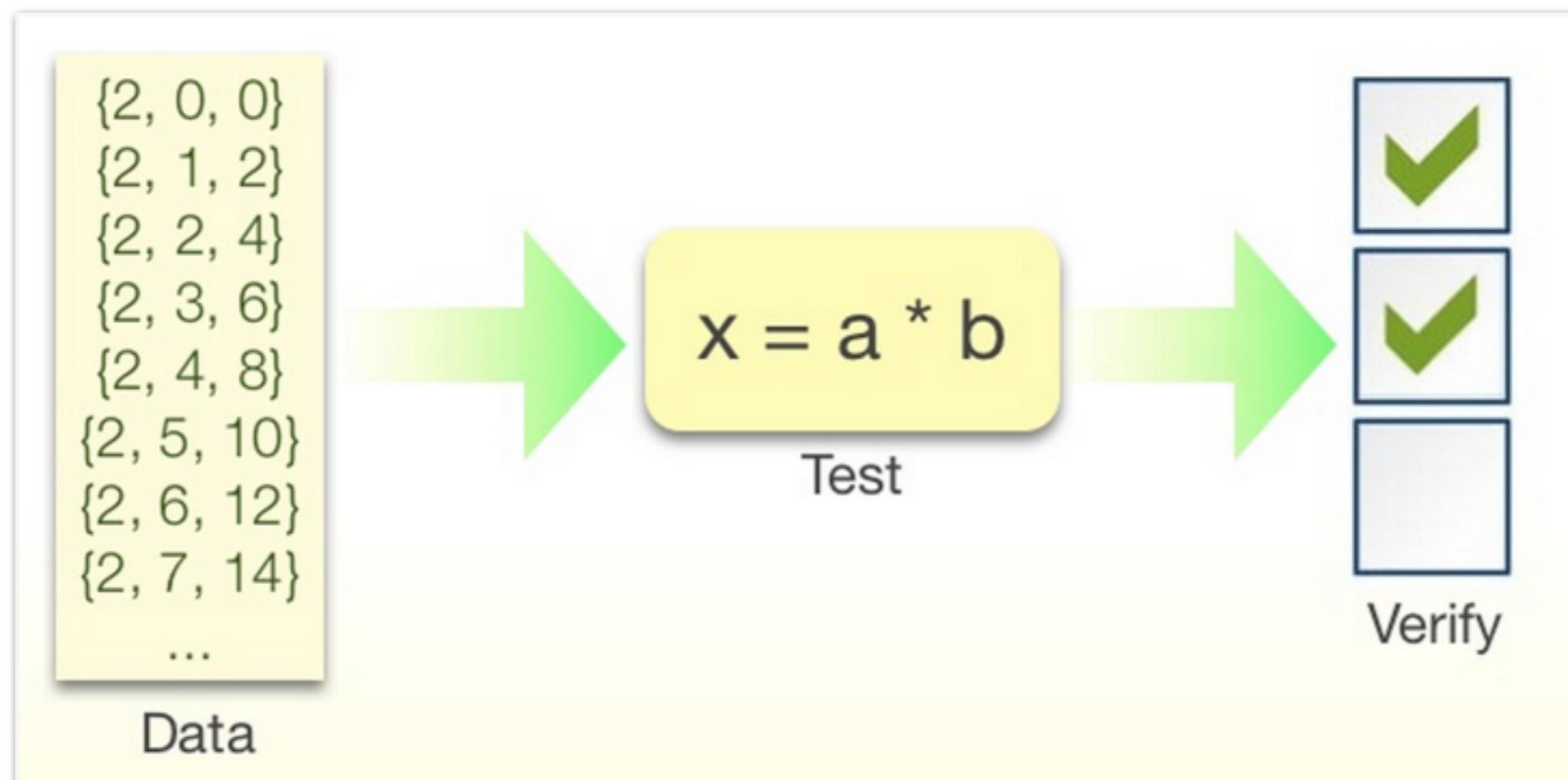
Set of test data

Expected result

Define test that uses the test data

Verify result against expected result

Data-Driven Development



Demo

@Parameterized
with Calculate Grade

Input	Expected Result
80	A
70	B
60	C
50	D
40	F

Step 1 :: Test Data

Input	Expected Result
80	A
70	B
60	C
50	D
40	F

Step 2 :: Add Runner

```
@RunWith(Parameterized.class)
public class CalculateGradeTest {

    private int score;
    private String expectedGrade;

    public CalculateGradeTest(int score, String expectedGrade) {
        this.score = score;
        this.expectedGrade = expectedGrade;
    }

    @Test
    public void convertScoreToGrade() {
    }

}
```

Step 3 :: Matching fields

```
@RunWith(Parameterized.class)
public class CalculateGradeTest {

    private int score;
    private String expectedGrade;

    public CalculateGradeTest(int score, String expectedGrade) {
        this.score = score;
        this.expectedGrade = expectedGrade;
    }

    @Test
    public void convertScoreToGrade() {
    }

}
```

Step 4 :: Define test case

```
@RunWith(Parameterized.class)
public class CalculateGradeTest {

    private int score;
    private String expectedGrade;

    public CalculateGradeTest(int score, String expectedGrade) {
        this.score = score;
        this.expectedGrade = expectedGrade;
    }

    @Test
    public void convertScoreToGrade() {
    }
}
```

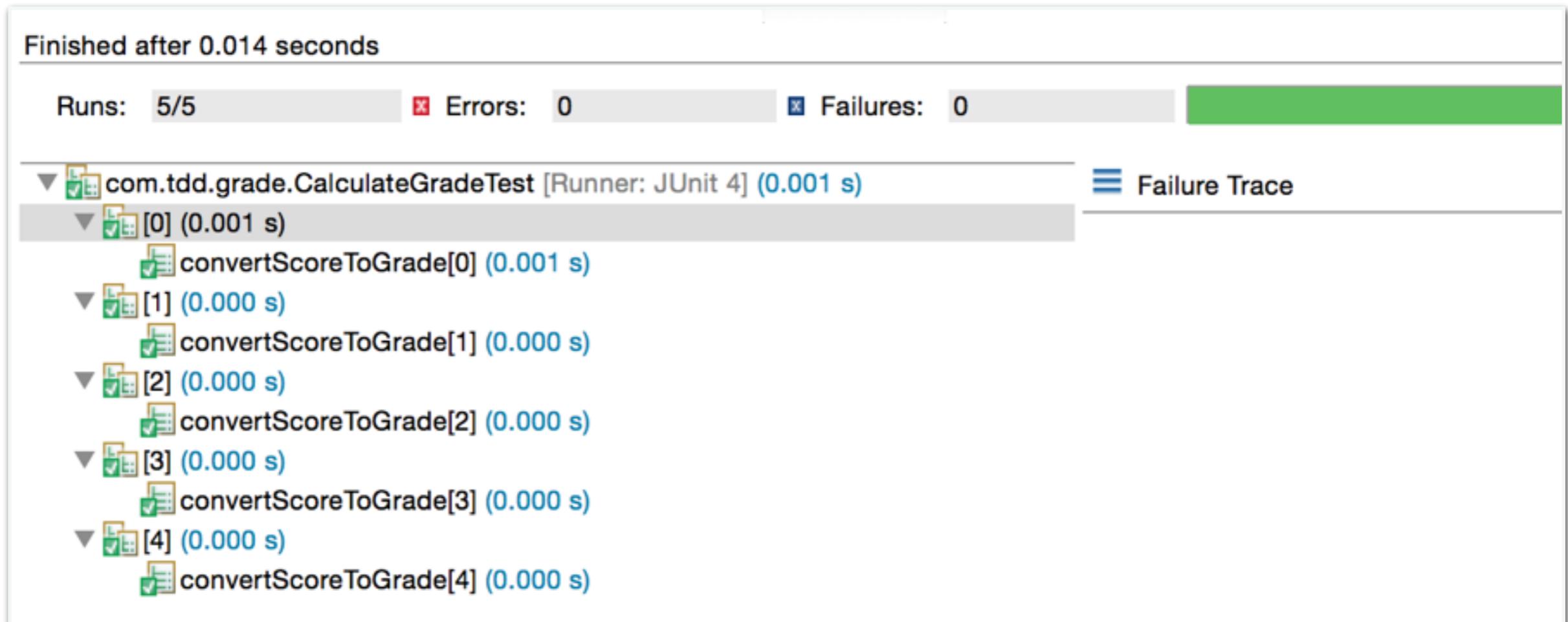
Step 5 :: Add @parameters

```
@RunWith(Parameterized.class)
public class CalculateGradeTest {

    private int score;
    private String expectedGrade;

    @Parameters
    public static Collection<Object[]> data(){
        return Arrays.asList(new Object[][]{
            {80, "A"},
            {70, "B"},
            {60, "C"},
            {50, "D"},
            {40, "F"}
        });
    }
}
```

Step 6 :: Test Result



Workshop

@Parameterized with Calculator

Operand 1	Operator	Operand 2	Expected
1	+	1	2
2	+	2	4
1	-	1	0
2	-	1	1
3	*	1	3
3	*	2	6

TDD in Real world

Design Principle



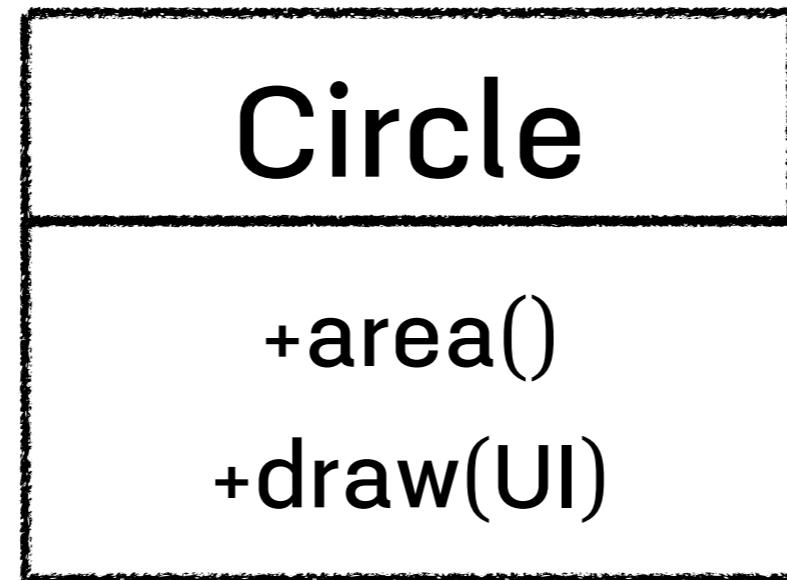
SOLID
Software development is not a Jenga game.



Single Responsibility Principle

Just because you *can* doesn't mean you *should*.

Single Responsibility Principle



Circle

+area()

CircleUI

+draw(UI)

Workshop #4

Single Responsibility Principle



Open-Closed Principle

Open-chest surgery isn't needed when putting on a coat.

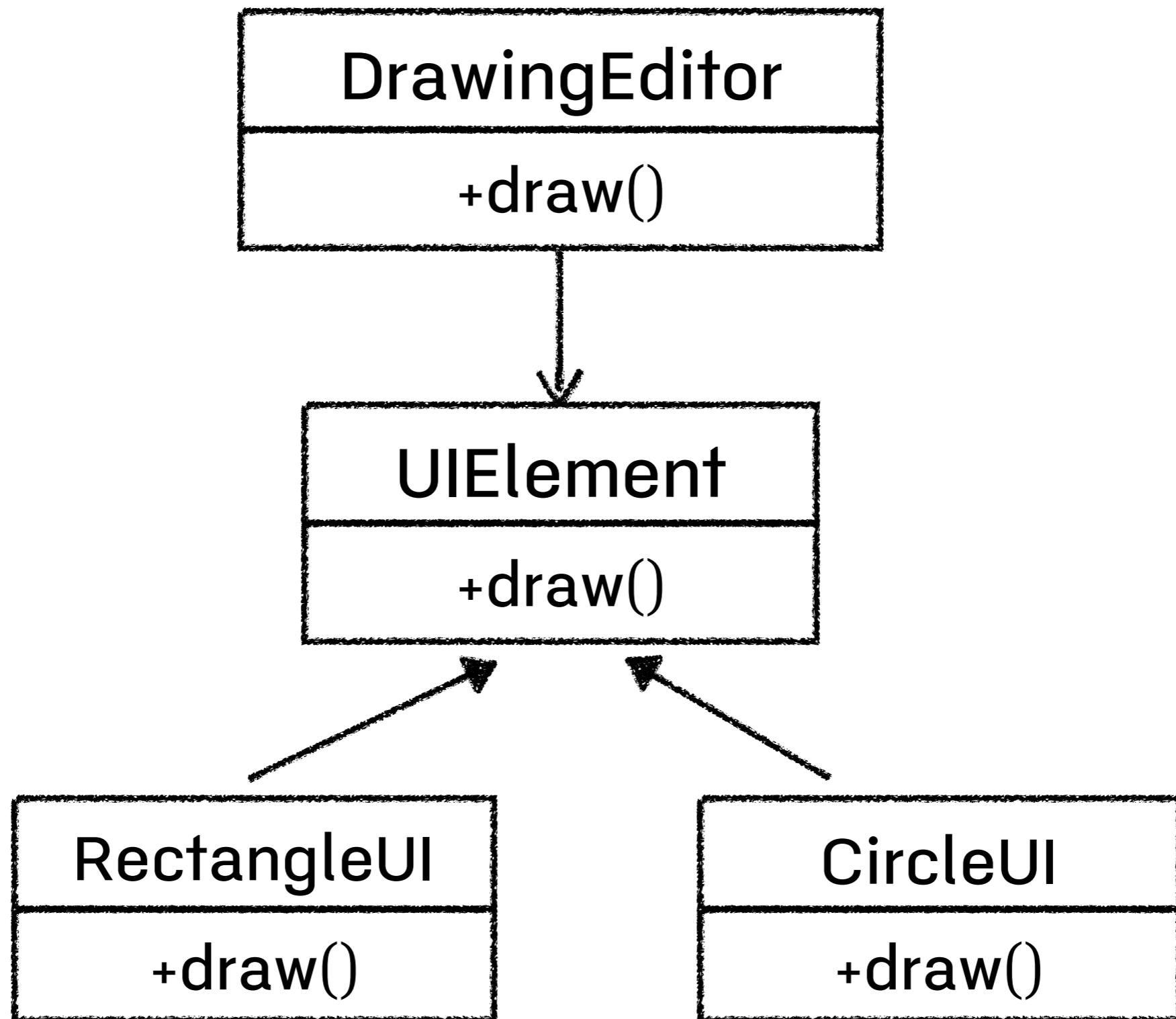
Open/Closed Principle

DrawingEditor

+draw()

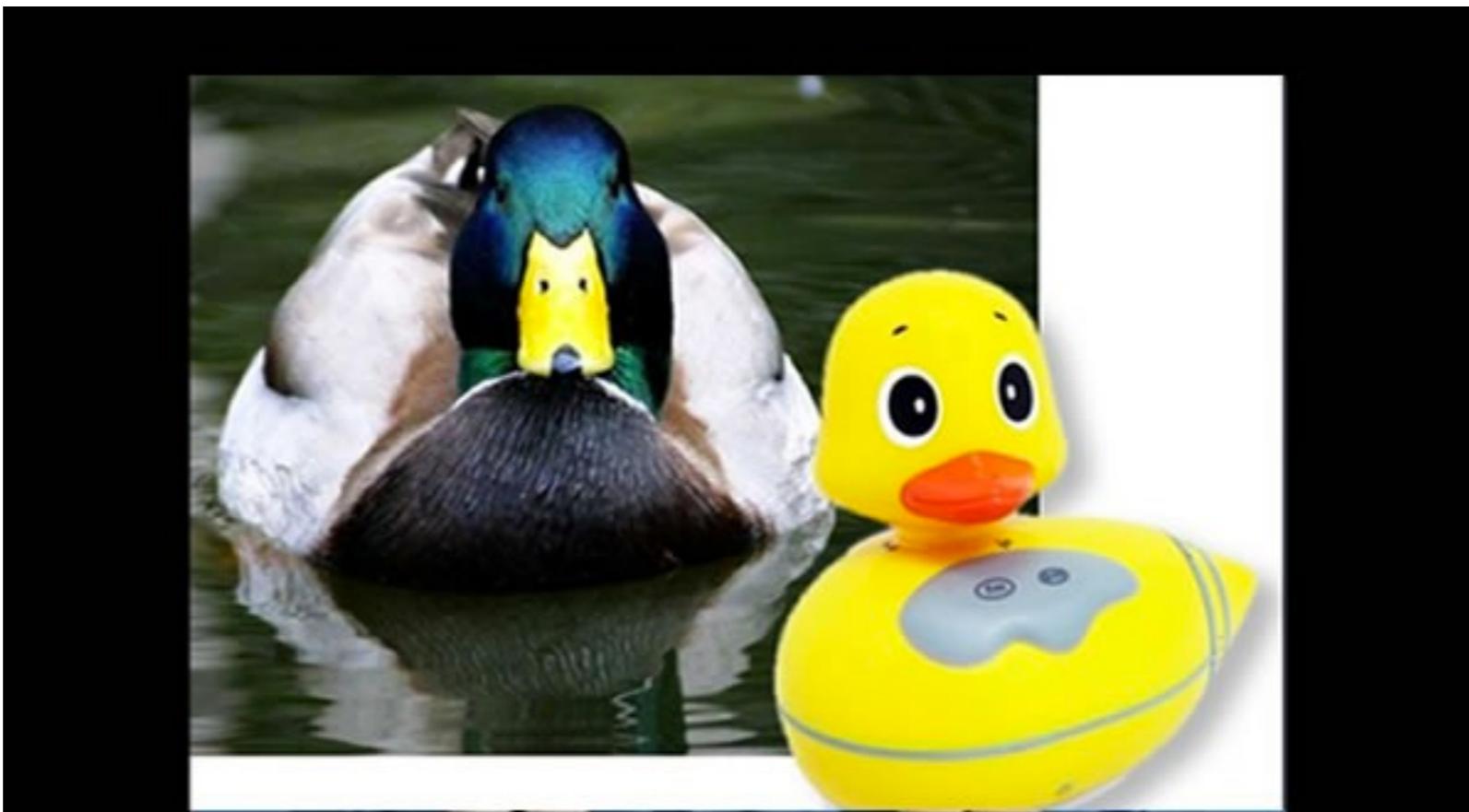
-drawCircle()

-drawRectangle()



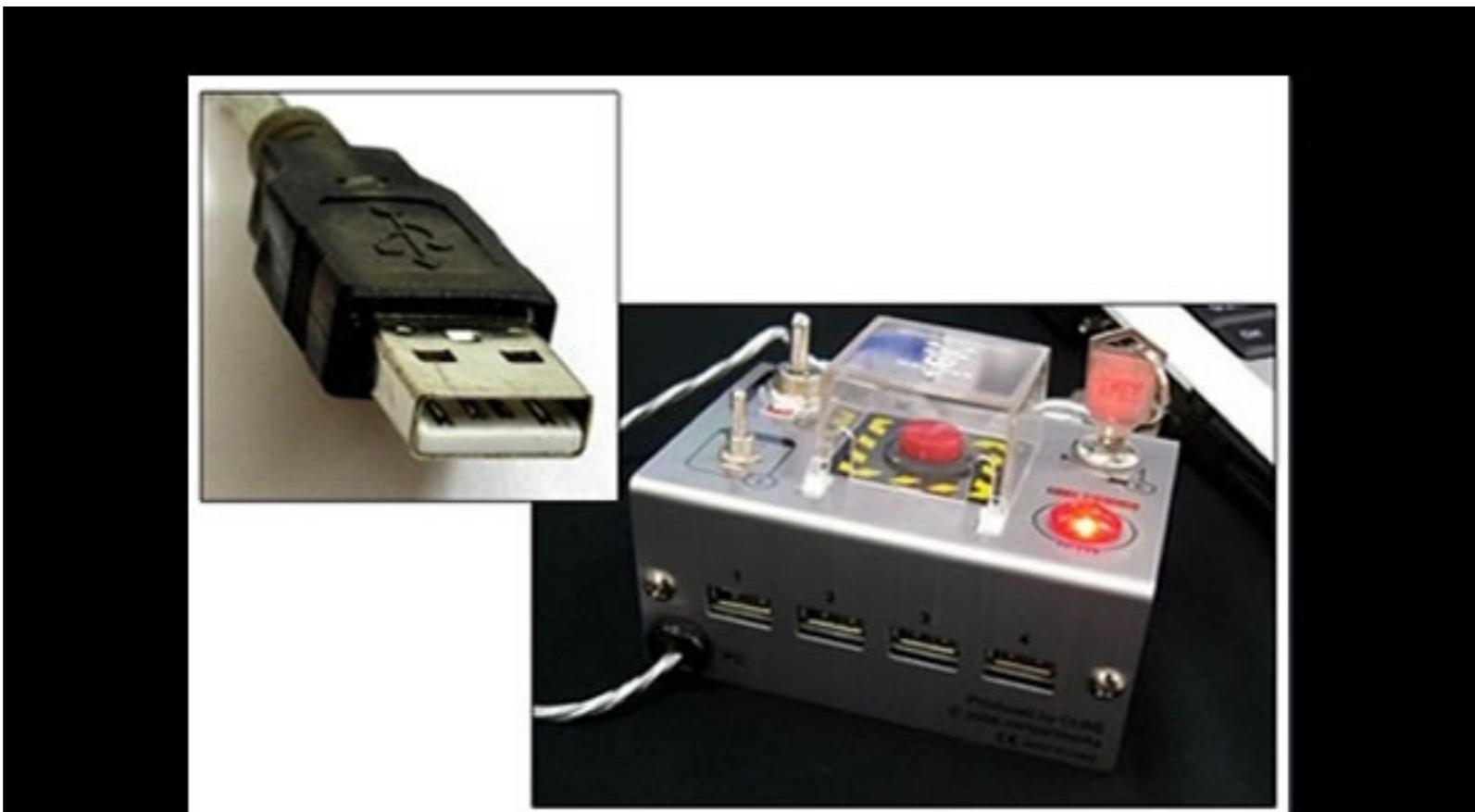
Workshop #5

Open/Closed Principle



Liskov Substitution Principle

If it looks like a duck and quacks like a duck but needs batteries,
you probably have the wrong abstraction.



Interface Segregation Principle

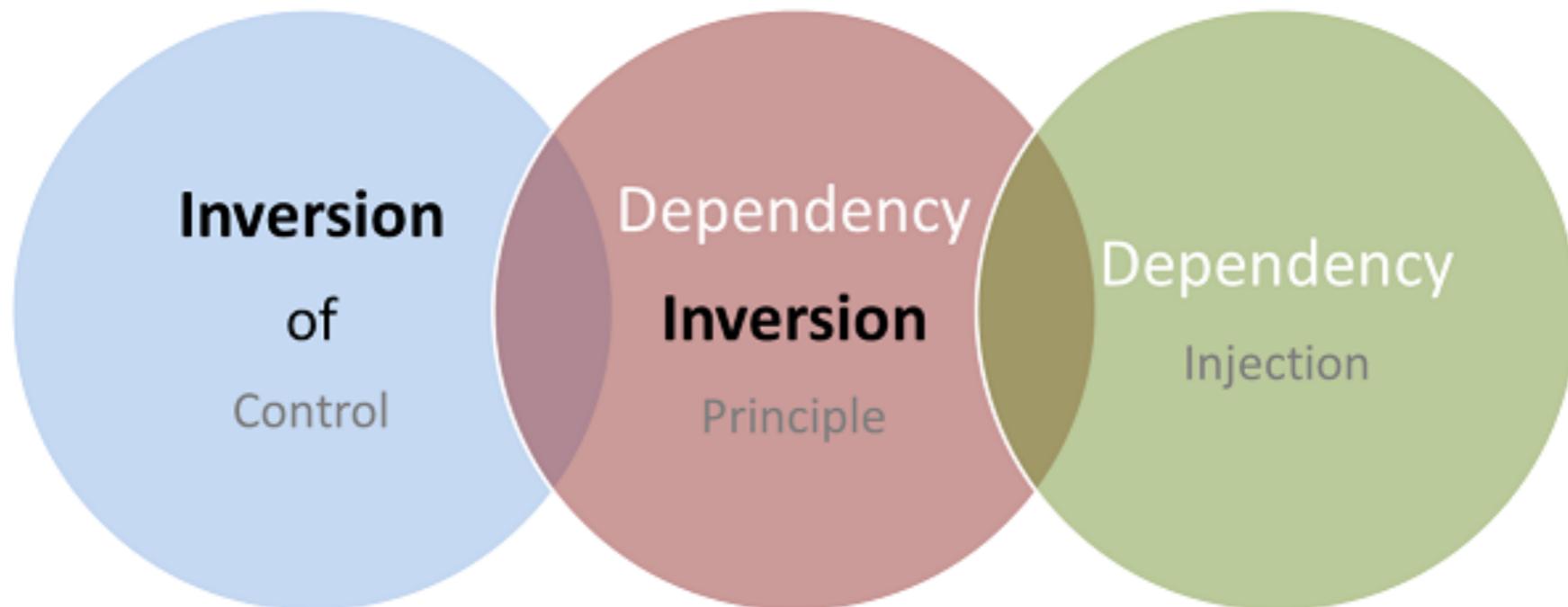
You want me to plug this in *where?*



Dependency Inversion Principle

Would you solder a lamp directly
to the electrical wiring in a wall?

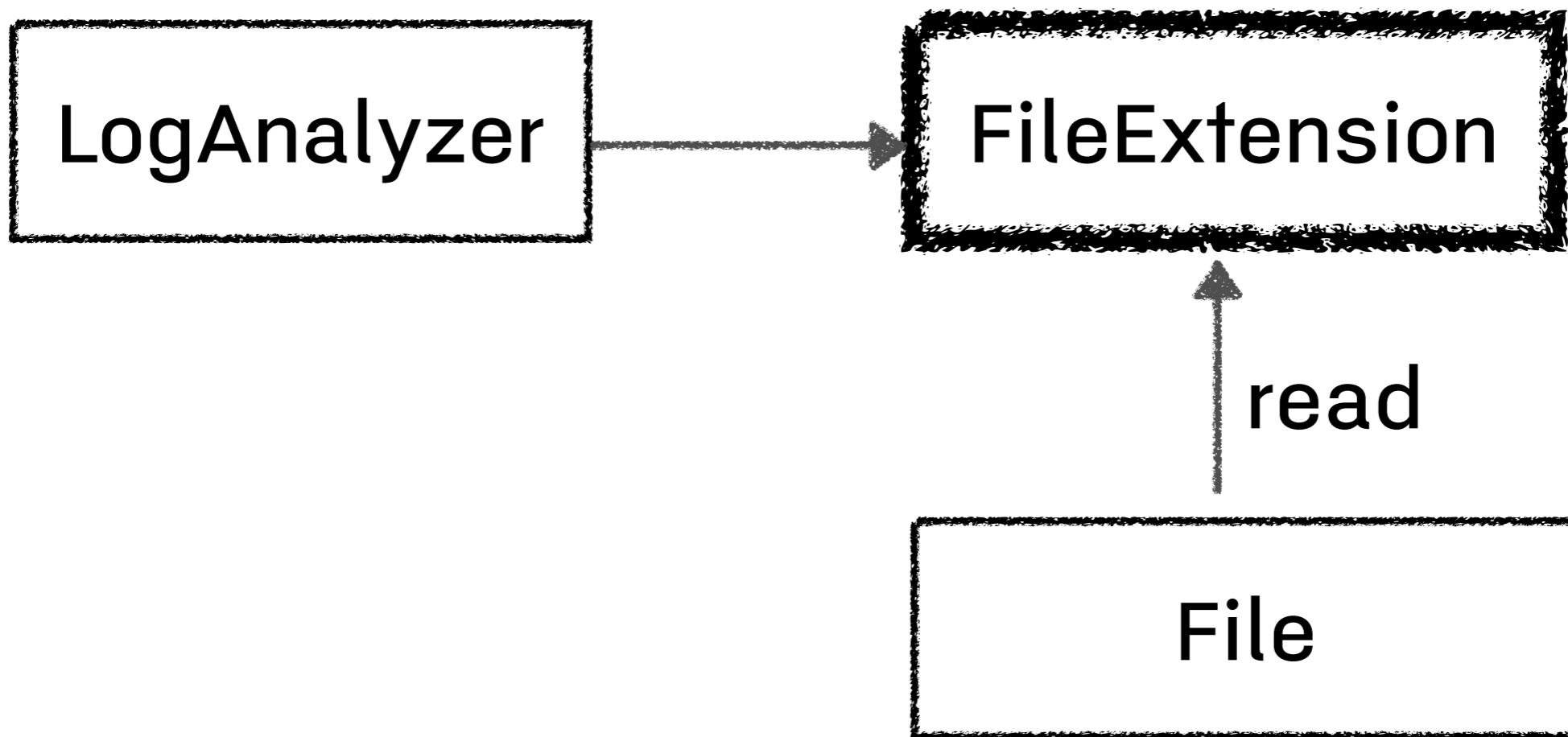
Dependency Inversion Principle



Dependency Inversion Principle



Move to new class



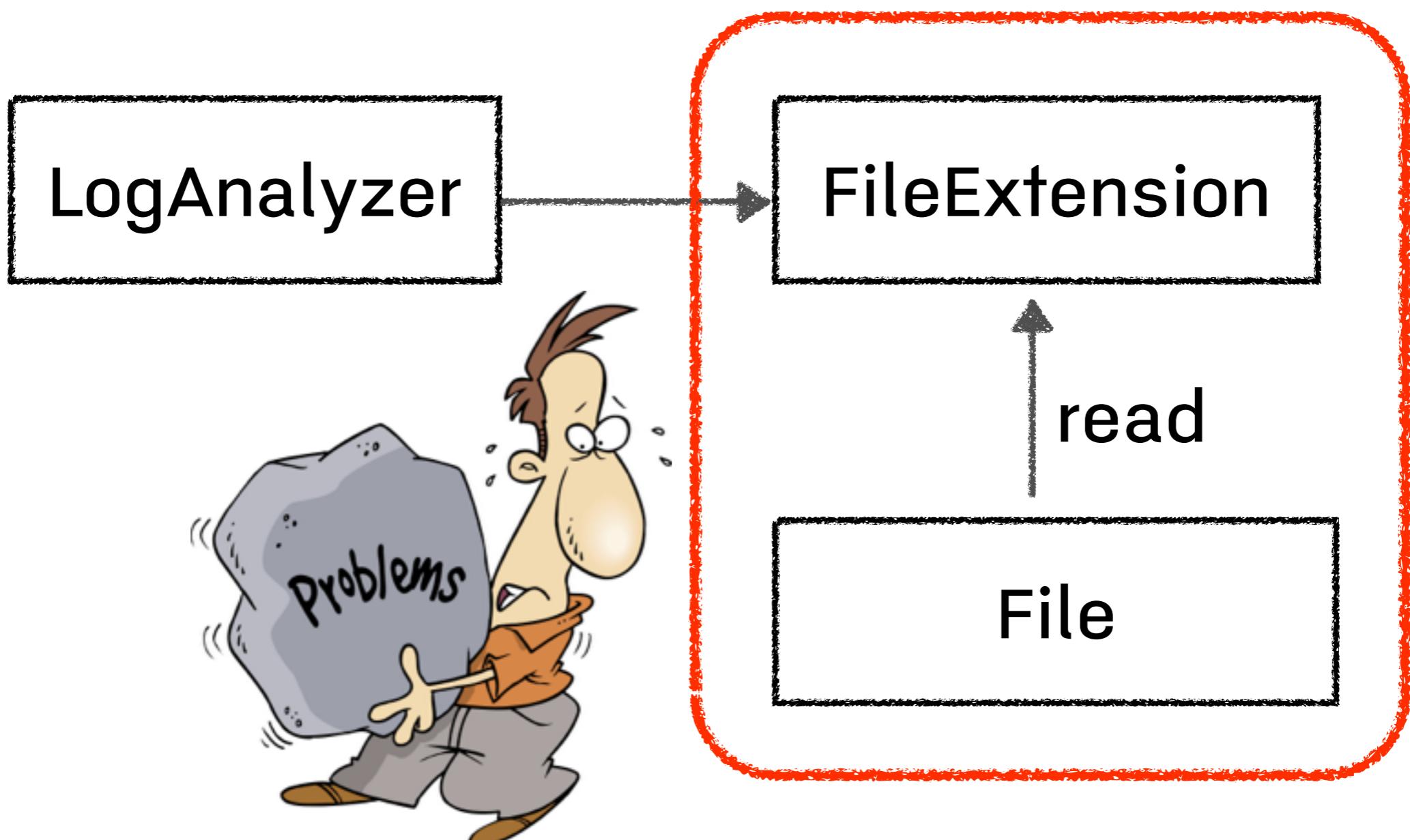
Workshop #6

Discuss about FileExtension.java

Problem ?

Fast
Independent
Repeatable
Self-validating
Timely

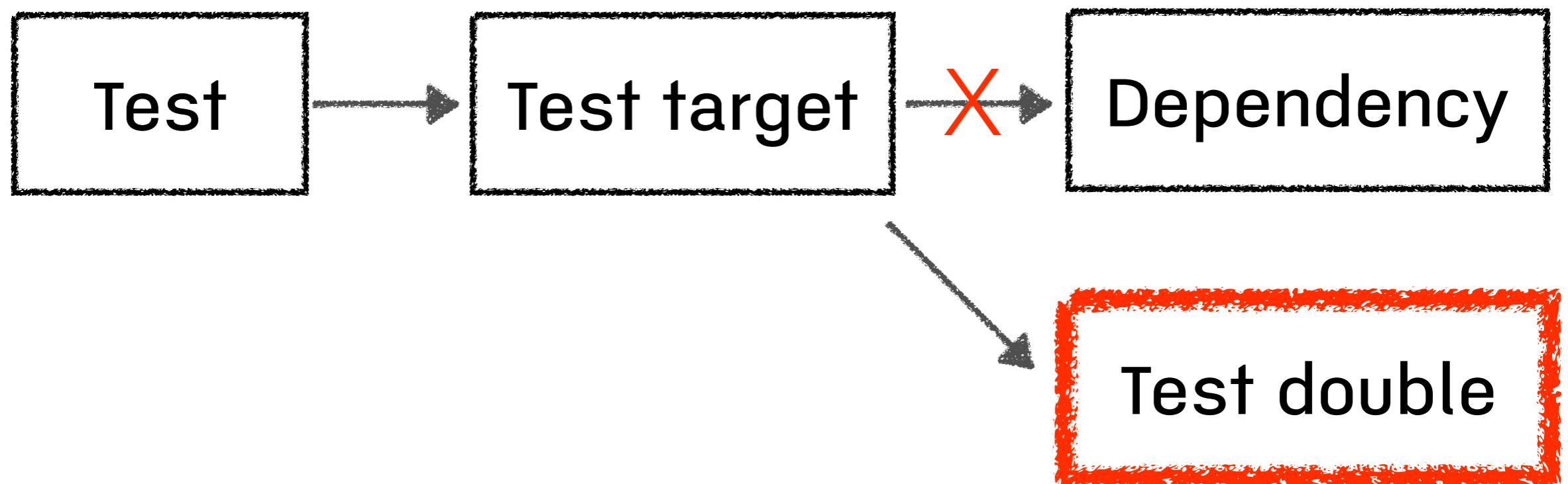
Dependency with File



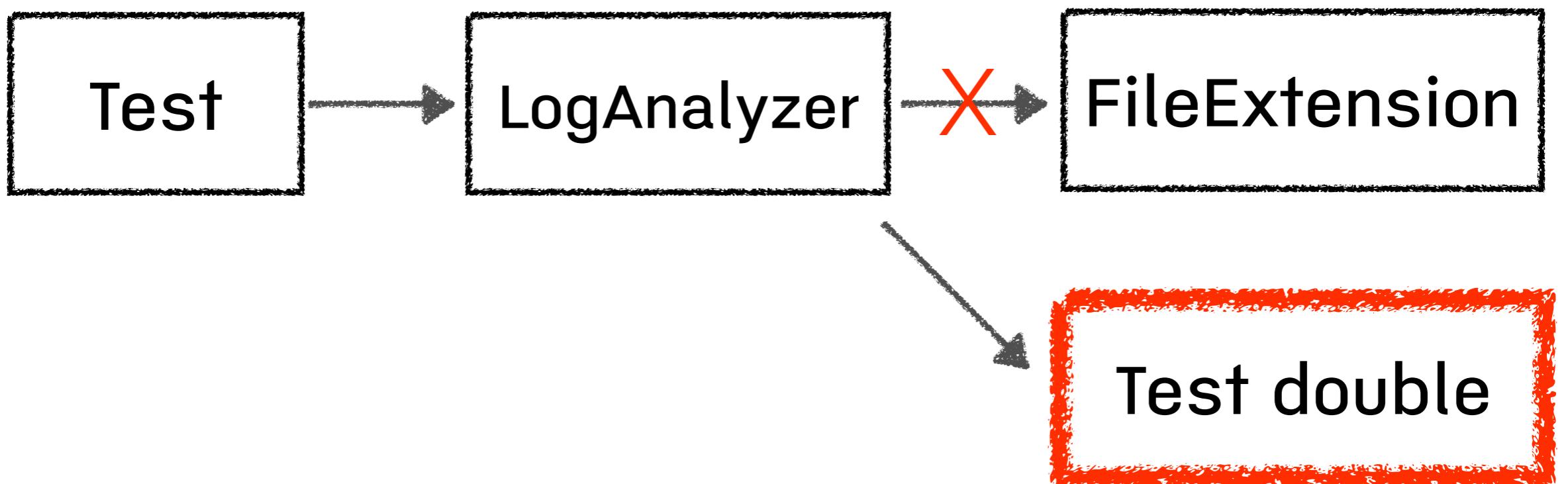
Discuss

How to Break dependency ?

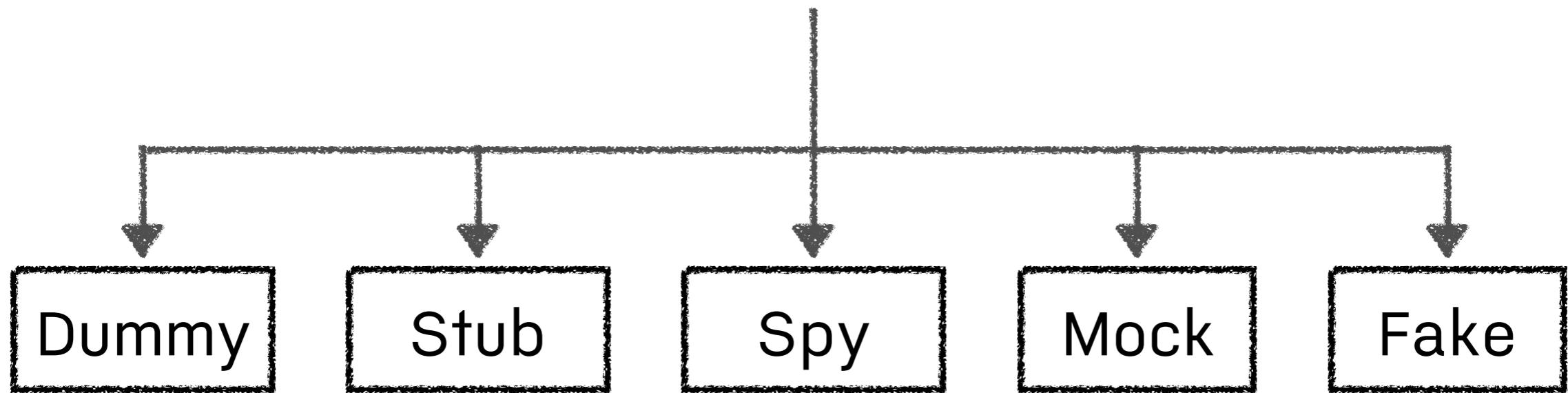
Isolate Dependency



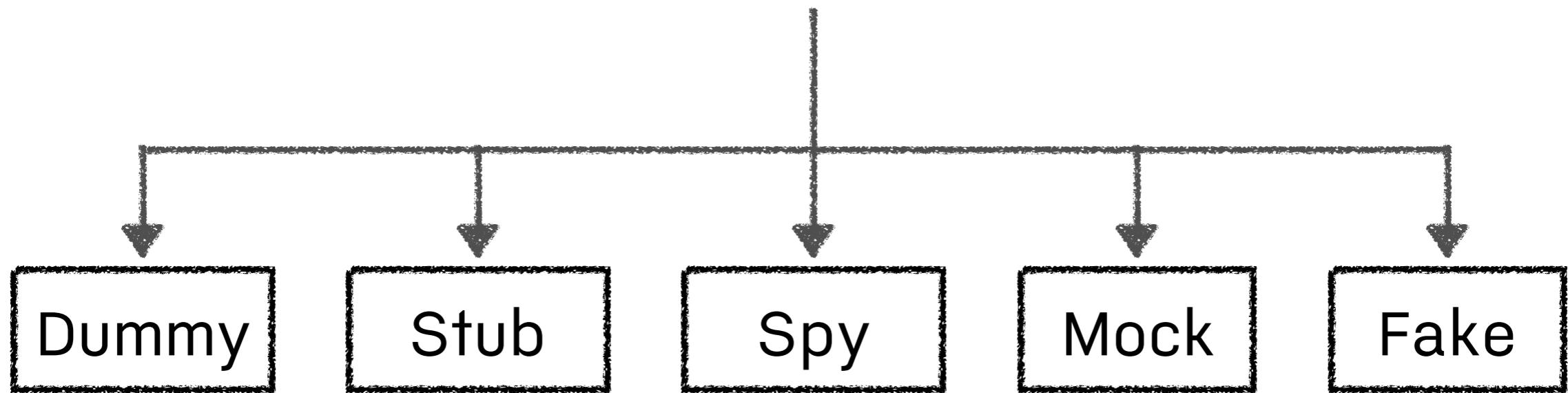
Isolate Dependency



Test Double

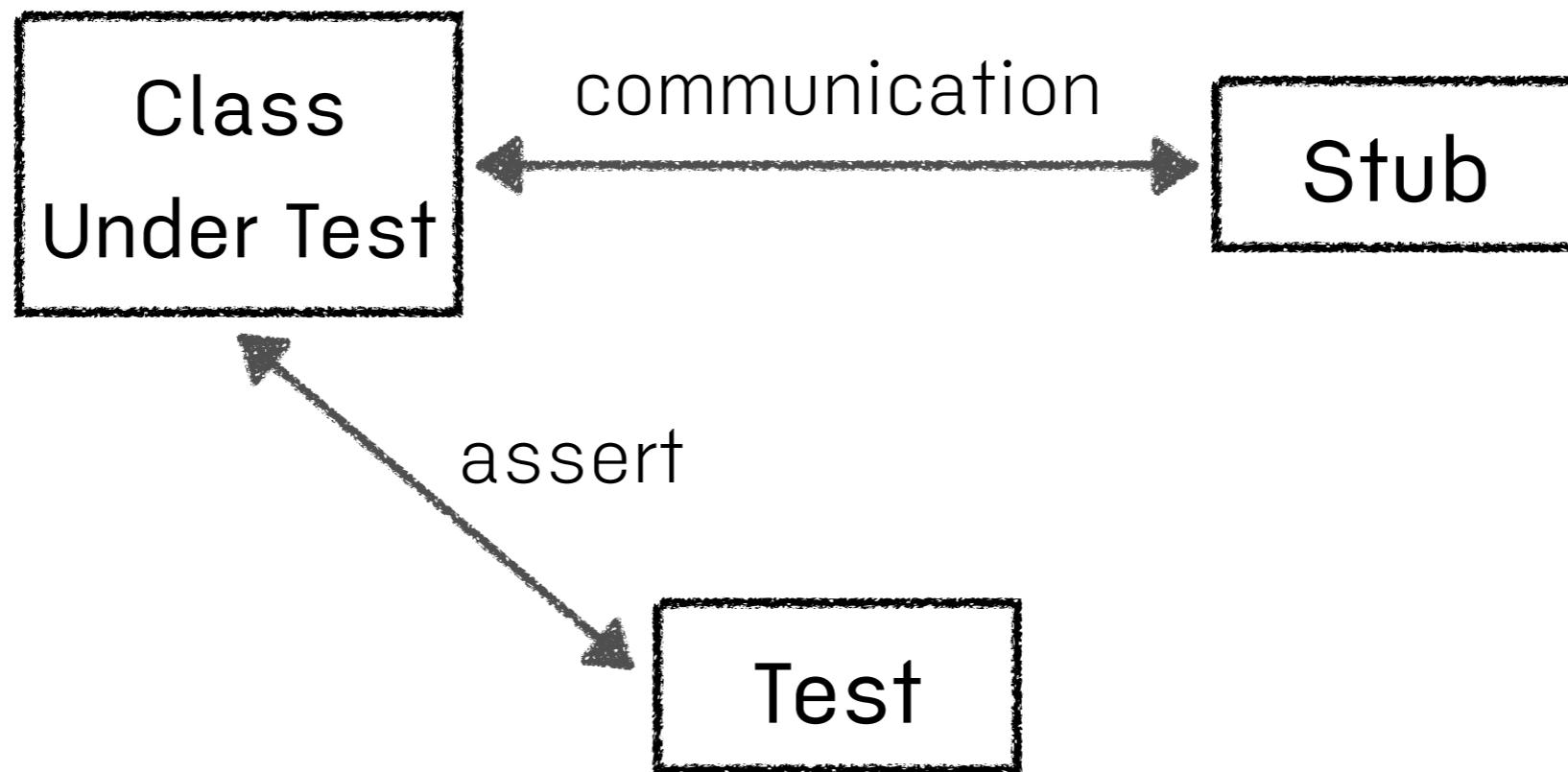


Test Double



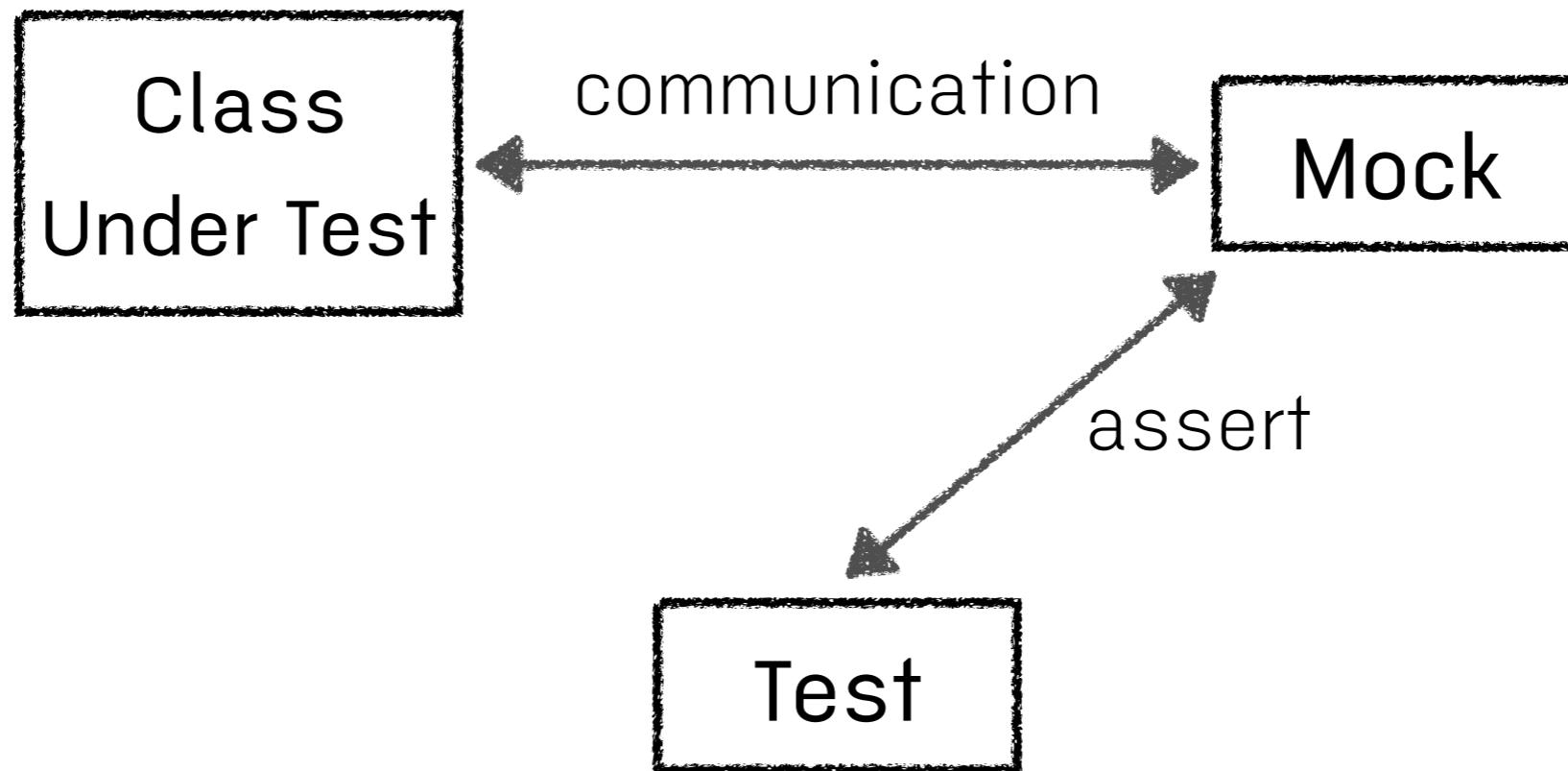
Configuration & **Hard code**

Stub



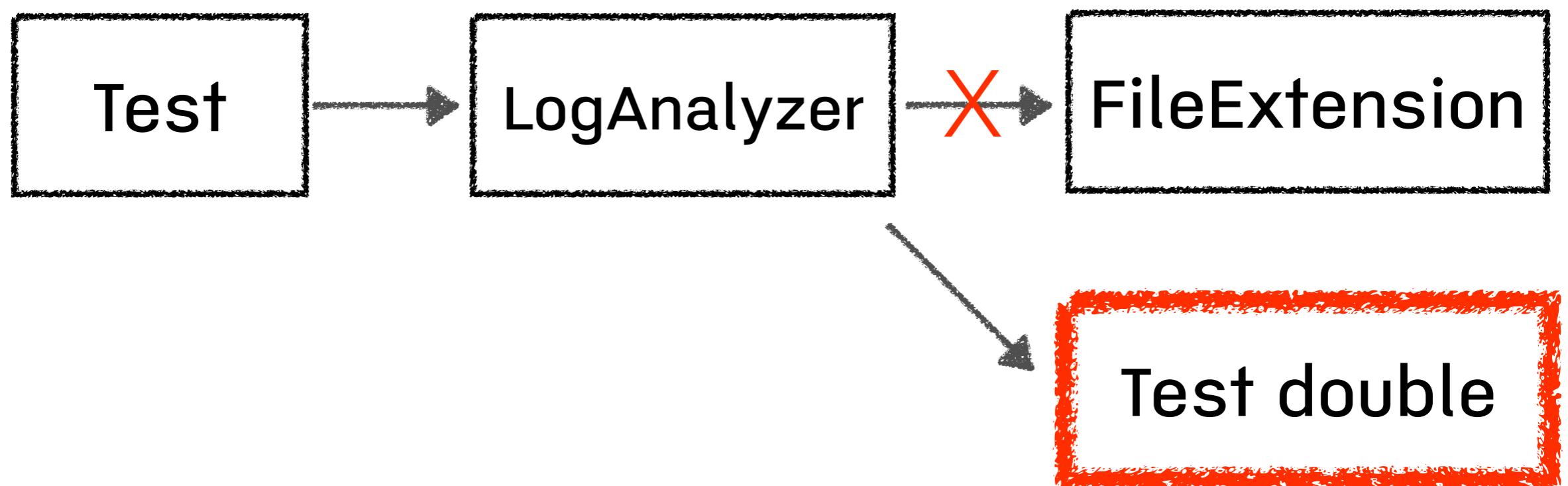
Make sure the test run smoothly

Mock object



To verify that the test pass

How to use Test double ?



Demo & Discuss

Break dependency with File System

Problem ?

```
public class LogAnalyzer {  
  
    private String fileName;  
  
    public LogAnalyzer(String fileName) {  
        this.fileName = fileName;  
    }  
  
    public boolean checkFileExtension() {  
        FileExtension fileExtension = new FileExtension(this.fileName);  
        return fileExtension.isValid();  
    }  
}
```

Problem ?

```
public class LogAnalyzer {  
  
    private String fileName;  
  
    public LogAnalyzer(String fileName) {  
        this.fileName = fileName;  
    }  
  
    public boolean checkFileExtension() {  
        FileExtension fileExtension = new FileExtension(this.fileName);  
        return fileExtension.isValid();  
    }  
}
```

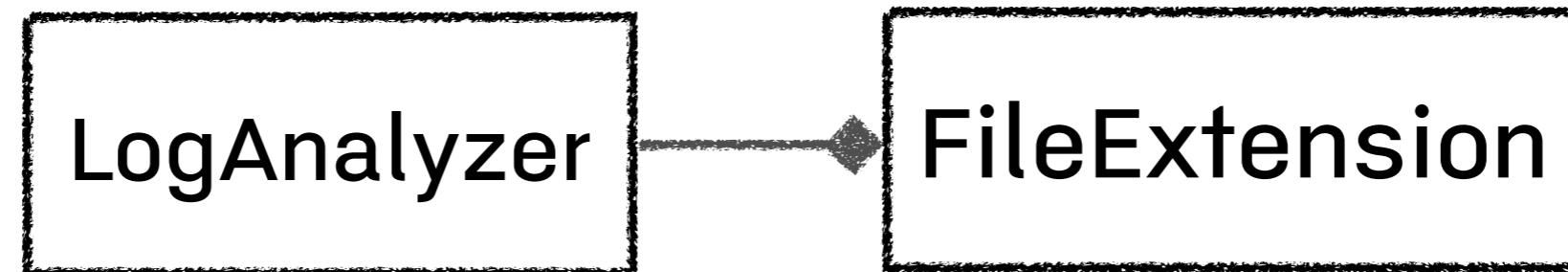
Code tight coupling !!

Discuss

How to solve this problem ?

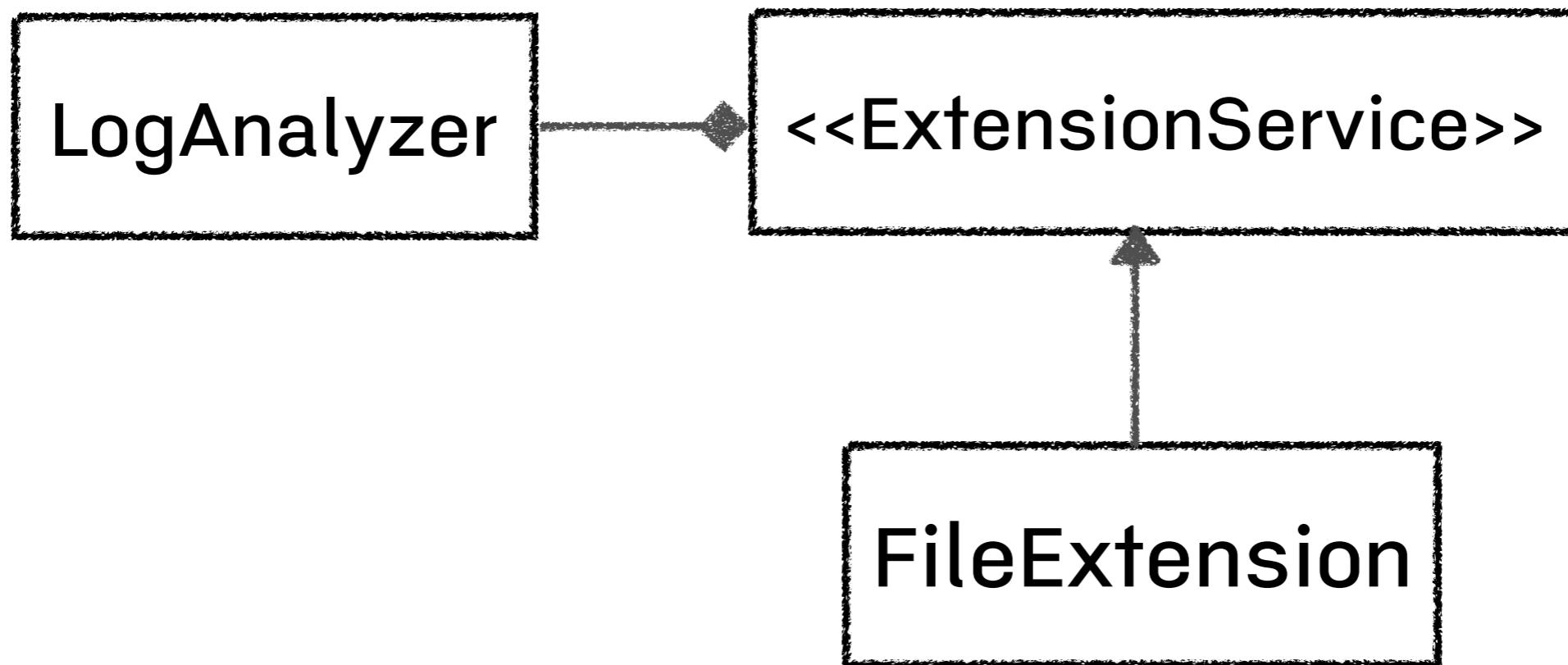
All about Abstraction

Tight coupling !!



All about Abstraction

Loose coupling !!

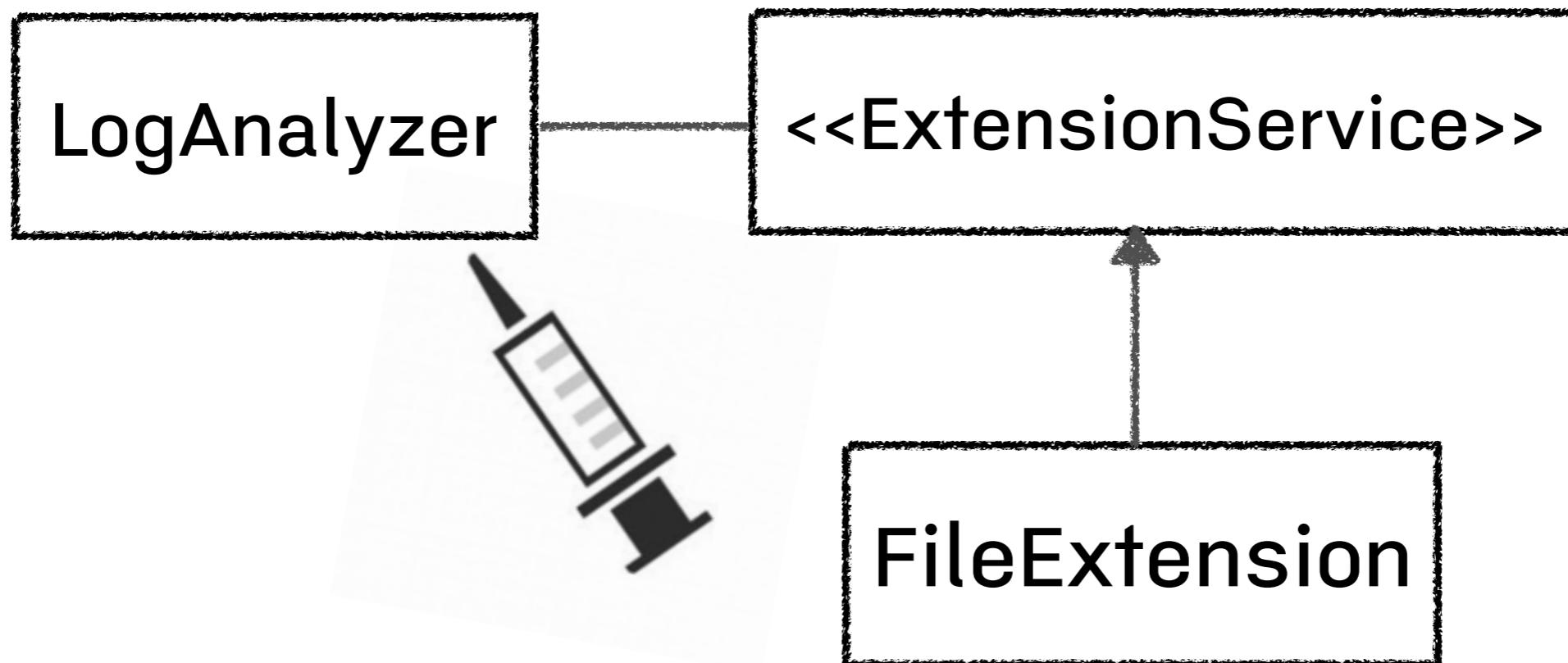


Demo & Discuss

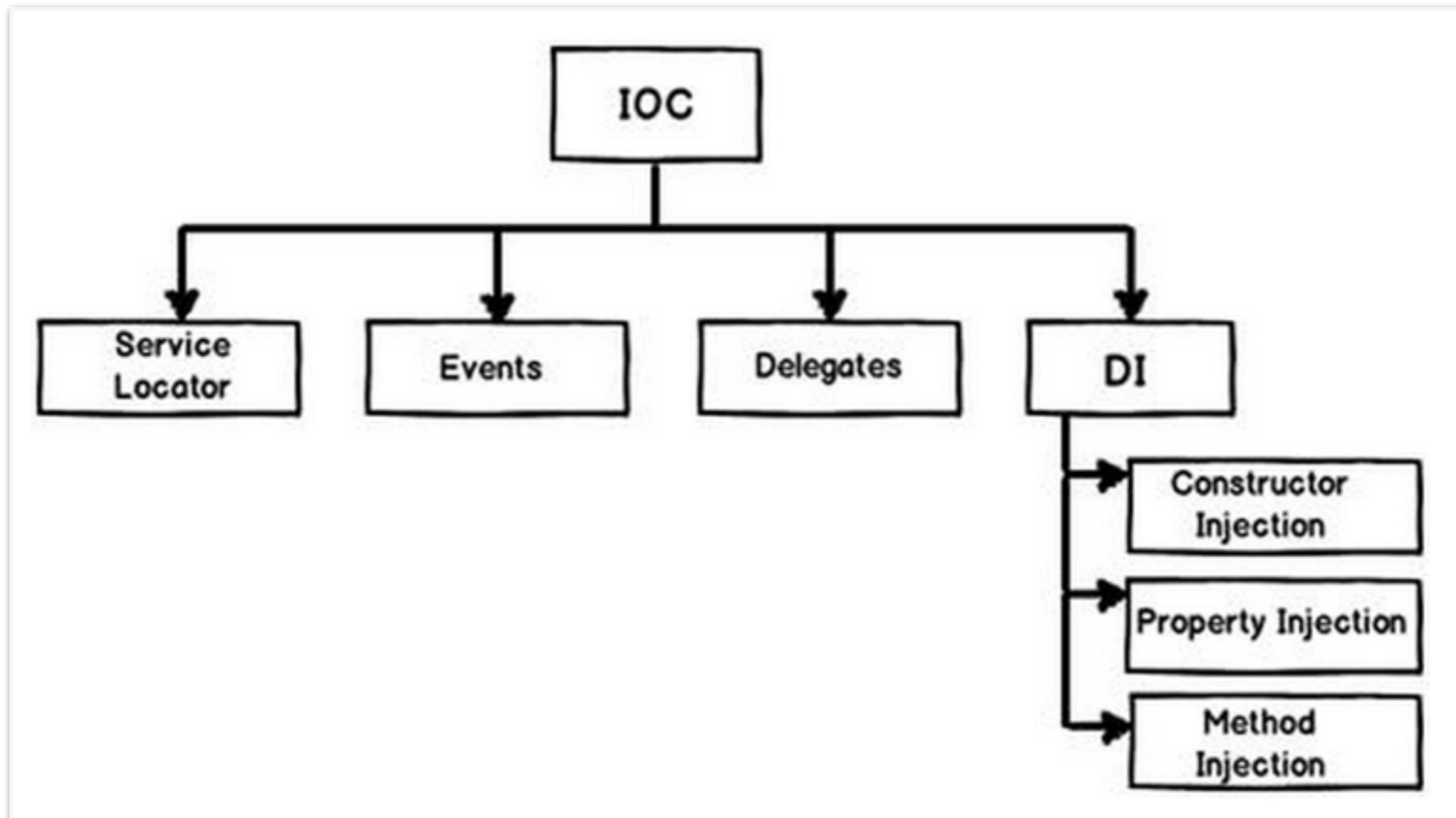
Dependency Inversion Principle (DIP)

How to use FileExtension ?

Loose coupling



Dependency Injection



Dependency Injection

Constructor Injection

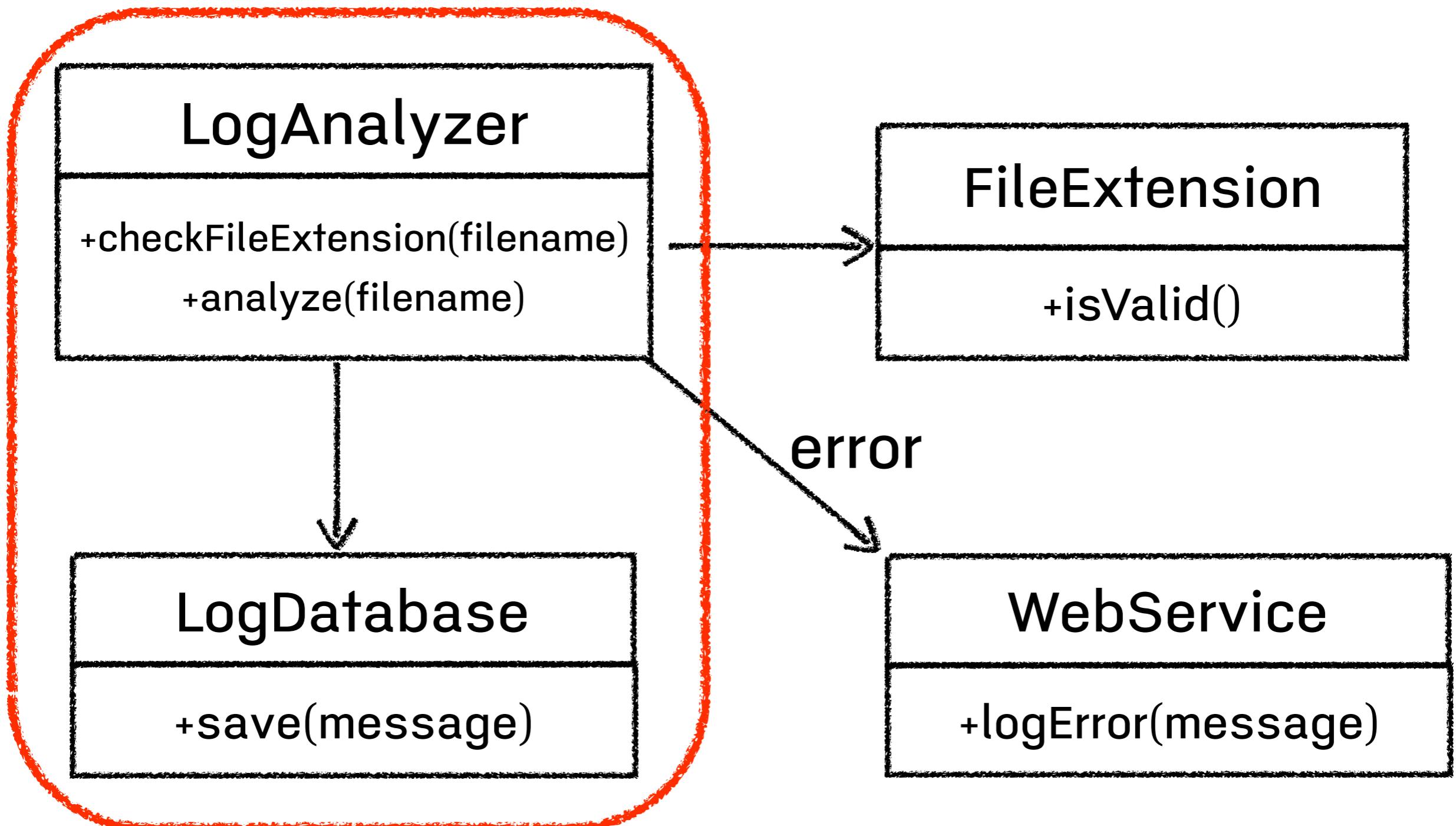
Property Injection

Method Injection

Demo & Discuss

Dependency Injection of LogAnalyzer

Workshop



Workshop #7

Stub with Database

Discuss

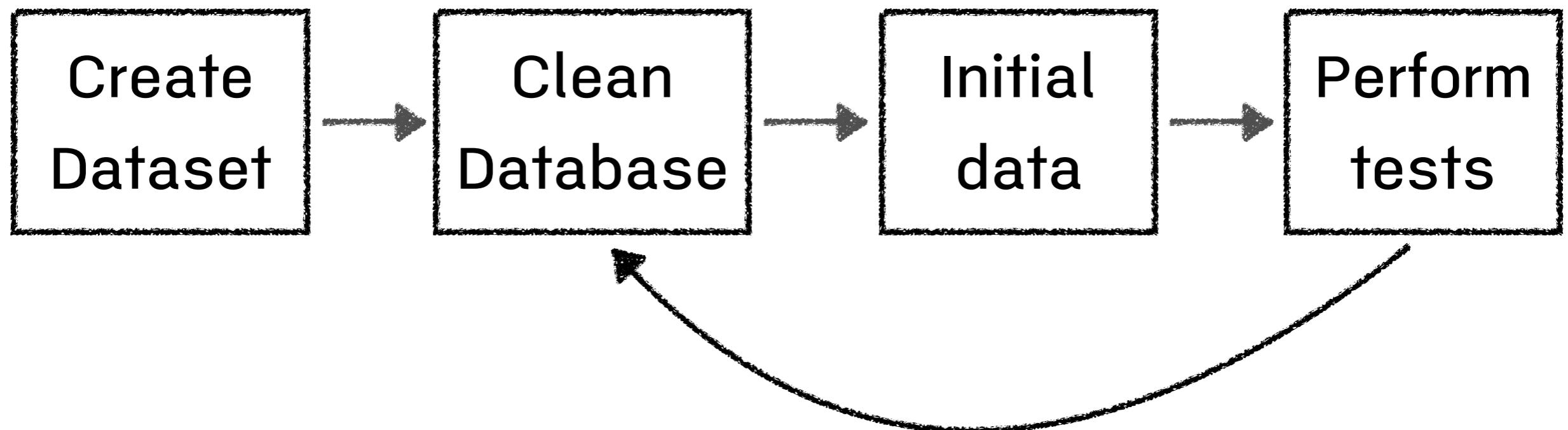
About value of Stub database !!

Problem

เมื่อทำการเปลี่ยน Schema database ?
ใช้ database จะมีคุณค่ากว่า stub ไหม ?

Fake Database with





Supported Database

Oracle

Hypersonic

Microsoft SQL Server

H2

PostgreSQL

Sybase

MySQL

DB2

Informix

Derby

DBUnit Components

Database Connection

DataSet

DatabaseOperation

NONE

INSERT

UPDATE

DELETE

DELETE_ALL

REFRESH

TRUNCATE_TABLE

CLEAN_INSERT = delete_all + insert

XML and POJO

```
public class Log {  
  
    private Long id;  
    private String code;  
    private String message;
```

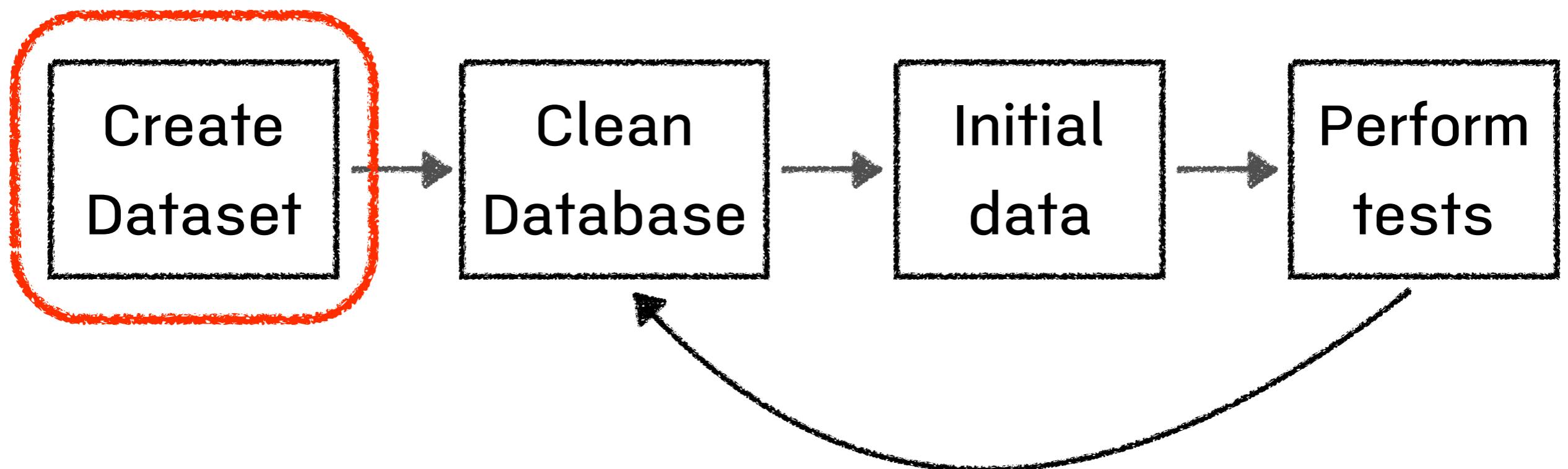
```
<dataset>  
    <LOG ID="1" CODE="200" MESSAGE="success"/>  
    <LOG ID="2" CODE="404" MESSAGE="data not found"/>  
</dataset>
```

Workshop #8

Fake Database
with



Step 1 :: Create dataset



Database schema

schema_log.sql

```
create table if not exists LOG (
    ID int identity primary key,
    CODE varchar,
    MESSAGE  varchar
);
```

Database schema

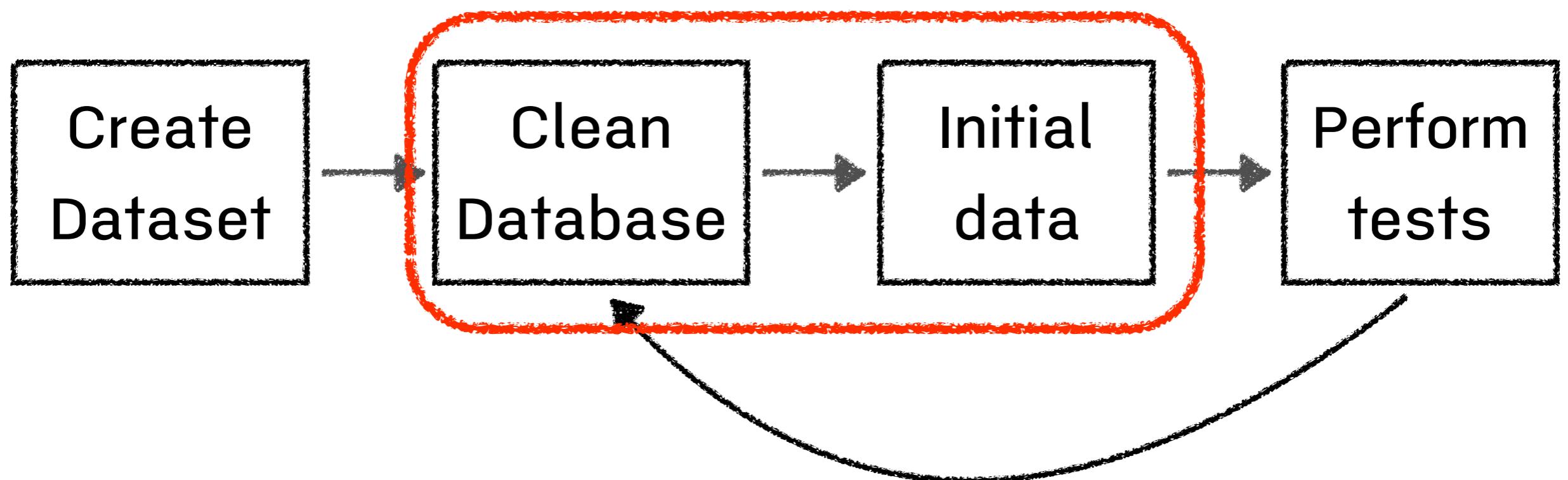
```
public class DBTest {  
  
    private static final String JDBC_DRIVER = org.h2.Driver.class.getName();  
    private static final String JDBC_URL = "jdbc:h2:mem:test;DB_CLOSE_DELAY=-1";  
    private static final String USER = "sa";  
    private static final String PASSWORD = "";  
    private static final String UTF8 = "UTF8";  
  
    @BeforeClass  
    public static void createSchema() throws Exception {  
        RunScript.execute(JDBC_URL, USER, PASSWORD,  
                         "src/com/tdd/db/schema_log.sql",  
                         Charset.forName(UTF8), false);  
    }  
}
```

Dataset as XML

data_log.xml

```
<dataset>
  <LOG ID="1" CODE="200" MESSAGE="success"/>
  <LOG ID="2" CODE="404" MESSAGE="data not found"/>
</dataset>
```

Step 2 :: Clean & Initial database



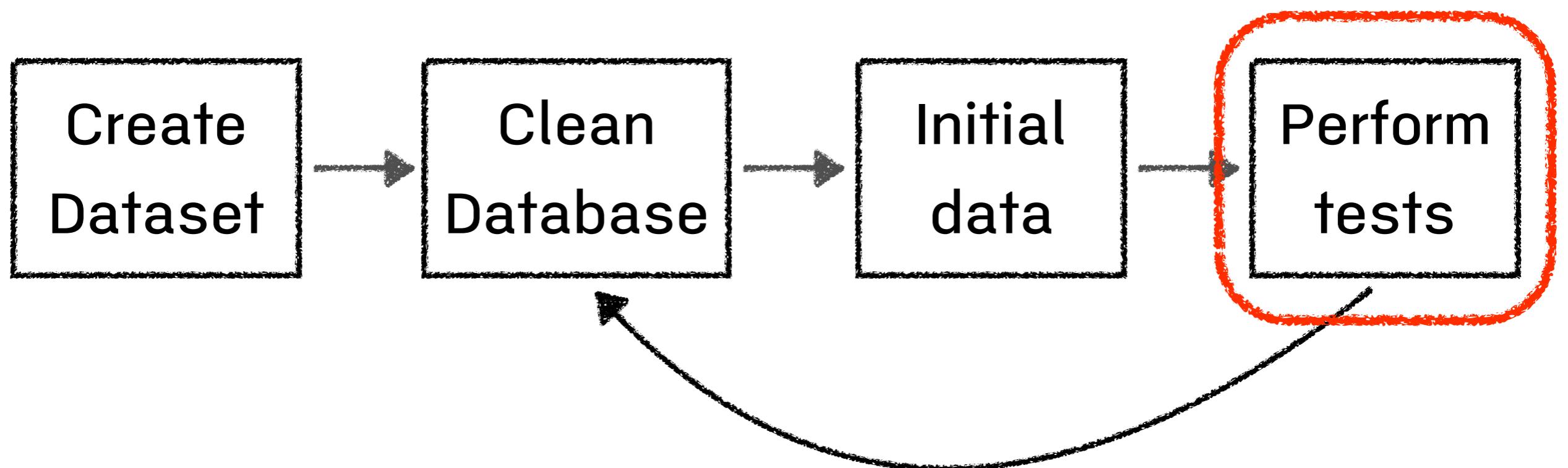
Clean & Initial database

```
@Before
public void importDataSet() throws Exception {
    IDataSet dataSet = readDataSet();
    cleanlyInsert(dataSet);
}

private IDataSet readDataSet() throws Exception {
    return new FlatXmlDataSetBuilder().build(new File("src/com/tdd/db/data_log.xml"));
}

private void cleanlyInsert(IDataSet dataSet) throws Exception {
    IDatabaseTester databaseTester = new JdbcDatabaseTester(JDBC_DRIVER, JDBC_URL,
                                                               USER, PASSWORD);
    databaseTester.setSetUpOperation(DatabaseOperation.CLEAN_INSERT);
    databaseTester.setDataSet(dataSet);
    databaseTester.onSetup();
}
```

Step 3 :: Perform test



How to perform test ?

```
@Test  
public void shouldSuccessWhenAddNewOneLogMessage() throws Exception {  
    LogDAO logDAO = new LogDAO(getDataSource());  
    boolean actualResult = logDAO.insert(new Log("200", "Success"));  
    assertTrue("Add new log message to database is succeed", actualResult);  
}
```

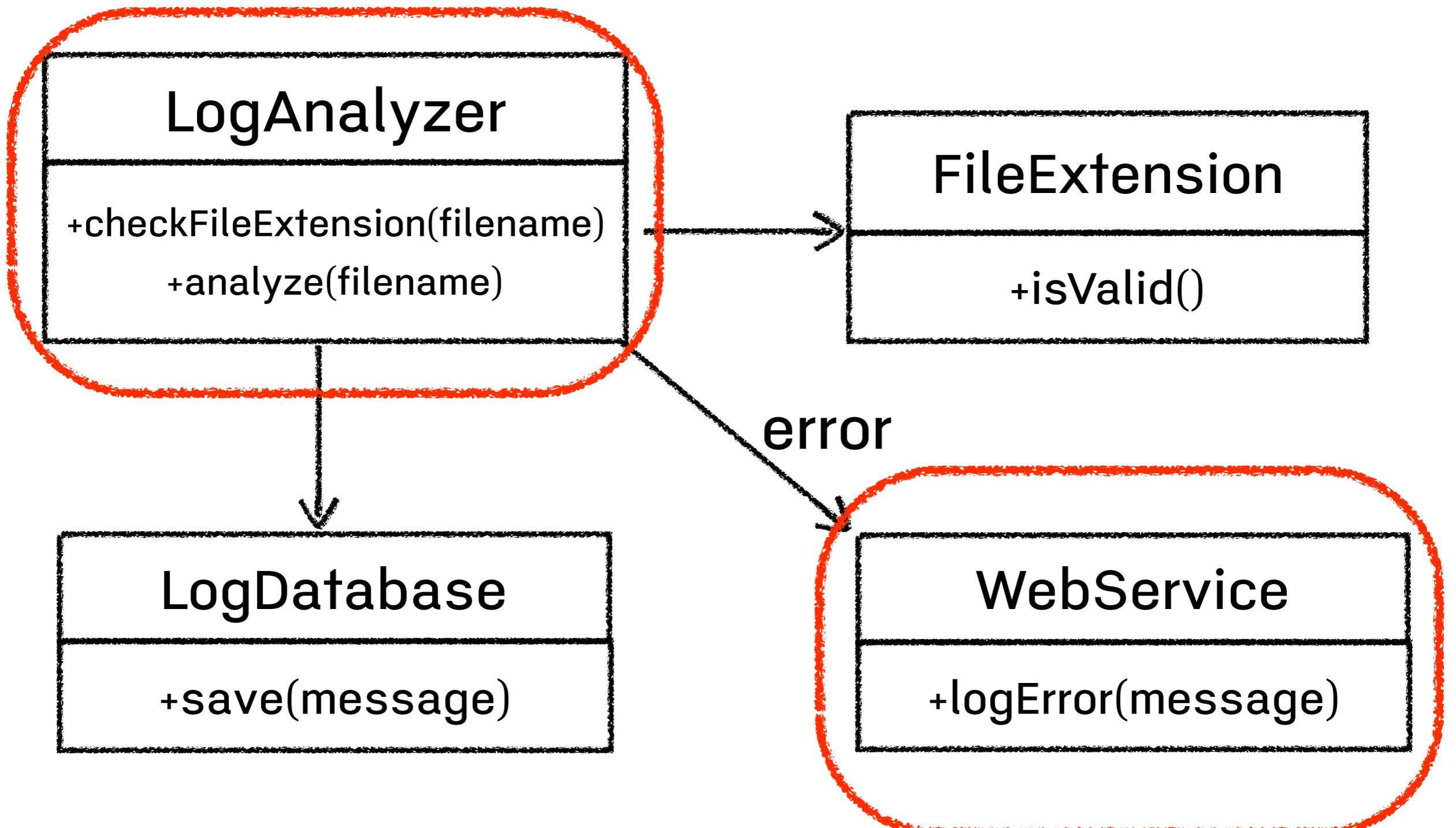
Create database connection

```
private DataSource getDataSource() {  
    JdbcDataSource dataSource = new JdbcDataSource();  
    dataSource.setURL(JDBC_URL);  
    dataSource.setUser(USER);  
    dataSource.setPassword(PASSWORD);  
    return dataSource;  
}
```

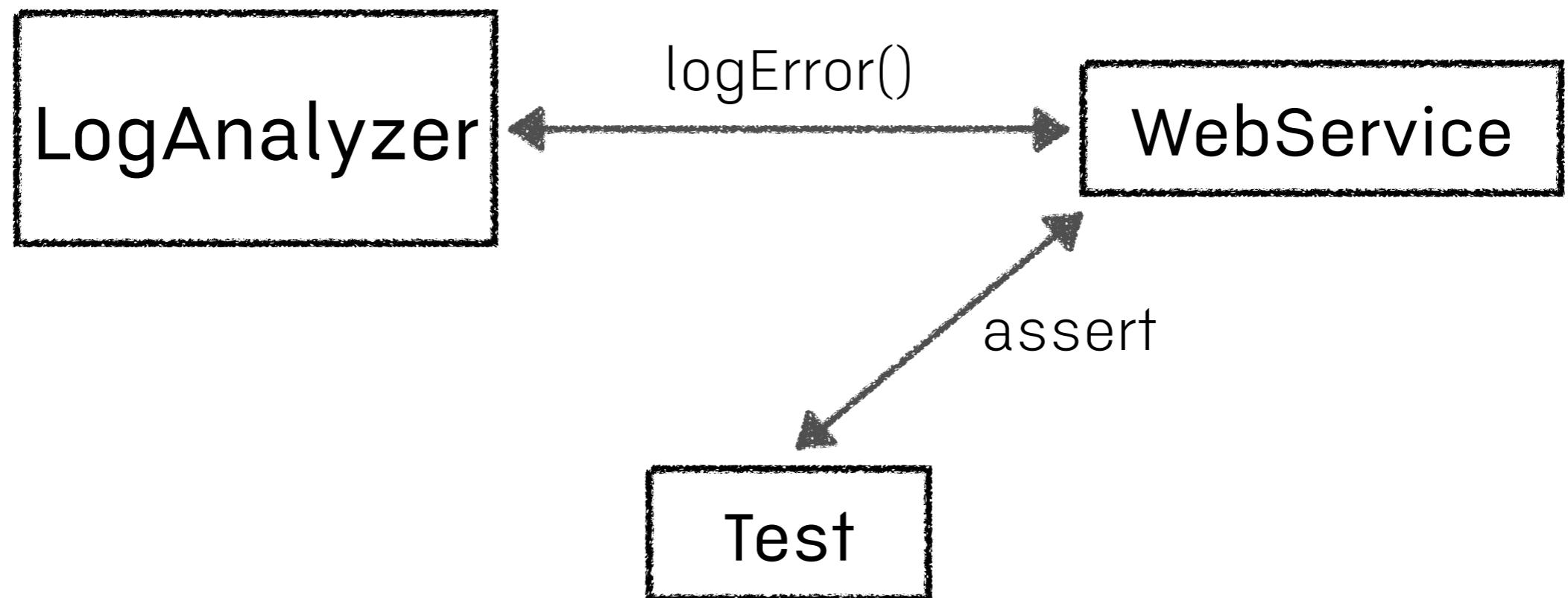
Implement LogDAO

```
public class LogDAO {  
  
    private DataSource dataSource;  
  
    public LogDAO(DataSource dataSource) {  
        this.dataSource = dataSource;  
    }  
  
    public boolean insert(Log log) throws Exception{  
        String sql = "INSERT INTO LOG(CODE, MESSAGE) VALUES(?, ?)";  
        PreparedStatement preparedStatement =  
            dataSource.getConnection().prepareStatement(sql);  
        preparedStatement.setString(1, log.getCode());  
        preparedStatement.setString(2, log.getMessage());  
        return preparedStatement.executeUpdate() == 1 ? true : false;  
    }  
}
```

Workshop



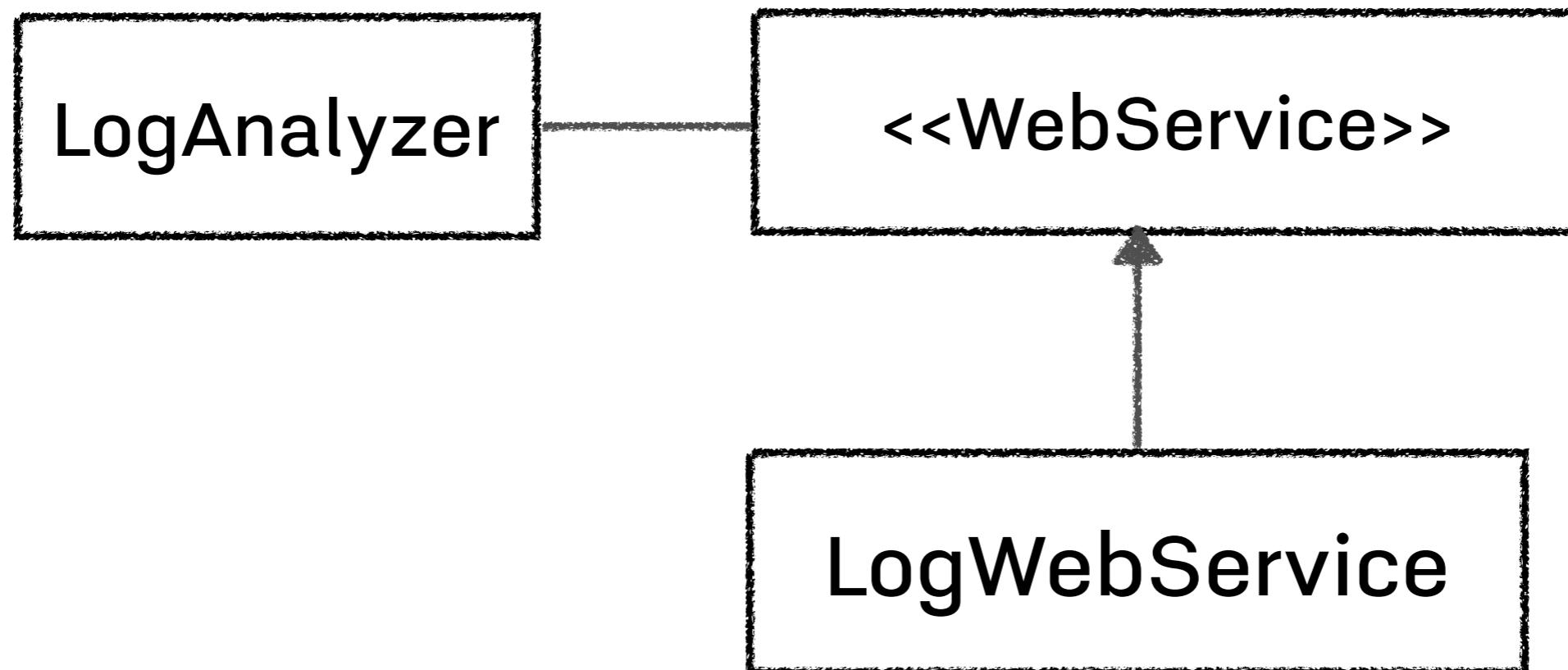
Mock object



Workshop #9

Mock object with WebService

Step 1 :: Break dependency



Step 2 :: Write test

```
@Test  
public void analyzeTooShortFilenameThenCallWebService(){  
    MockService mockService = new MockService();  
    LogAnalyzer logAnalyzer = new LogAnalyzer(mockService);  
    String tooShortFilename = "abc.txt";  
  
    logAnalyzer.analyze(tooShortFilename);  
  
    assertEquals("File name is too short:abc.txt", mockService.getLastErrorMessage());  
}
```

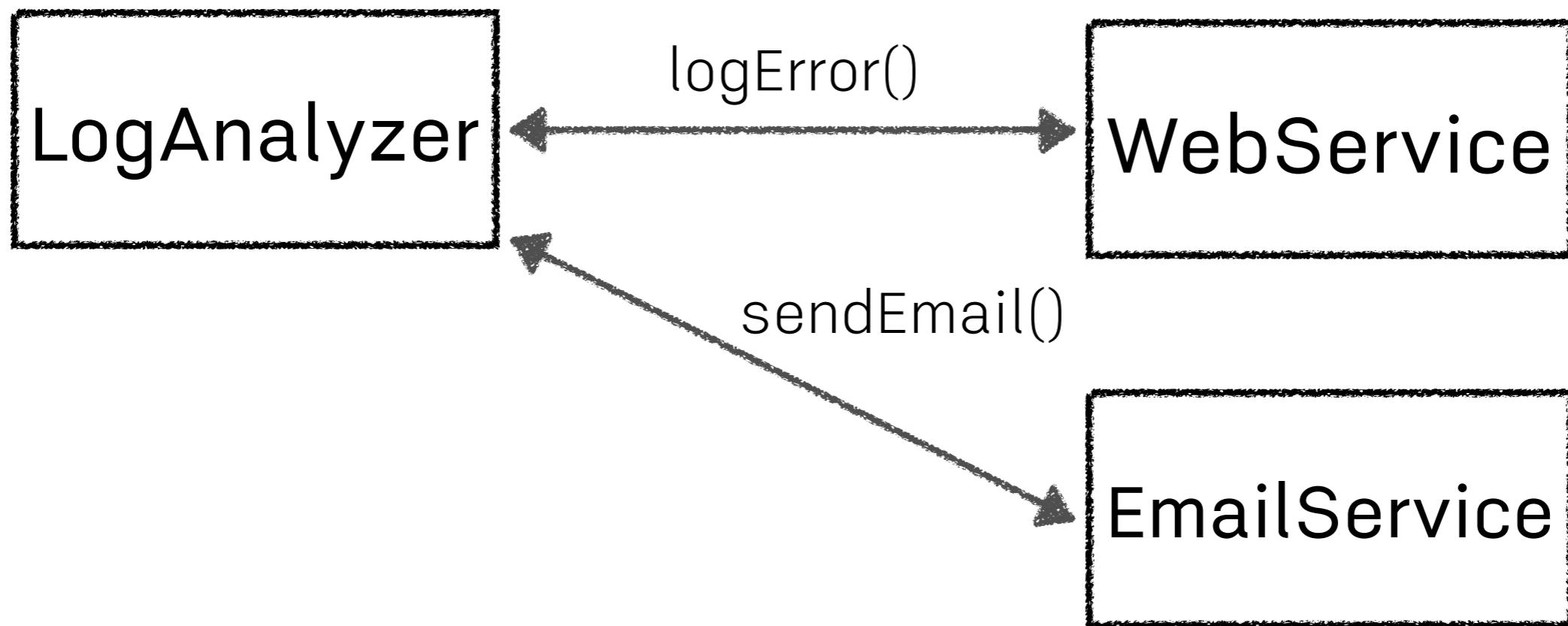
Step 3 :: Write code

Start coding by your pair

Workshop #10

Using Mock object and stub together

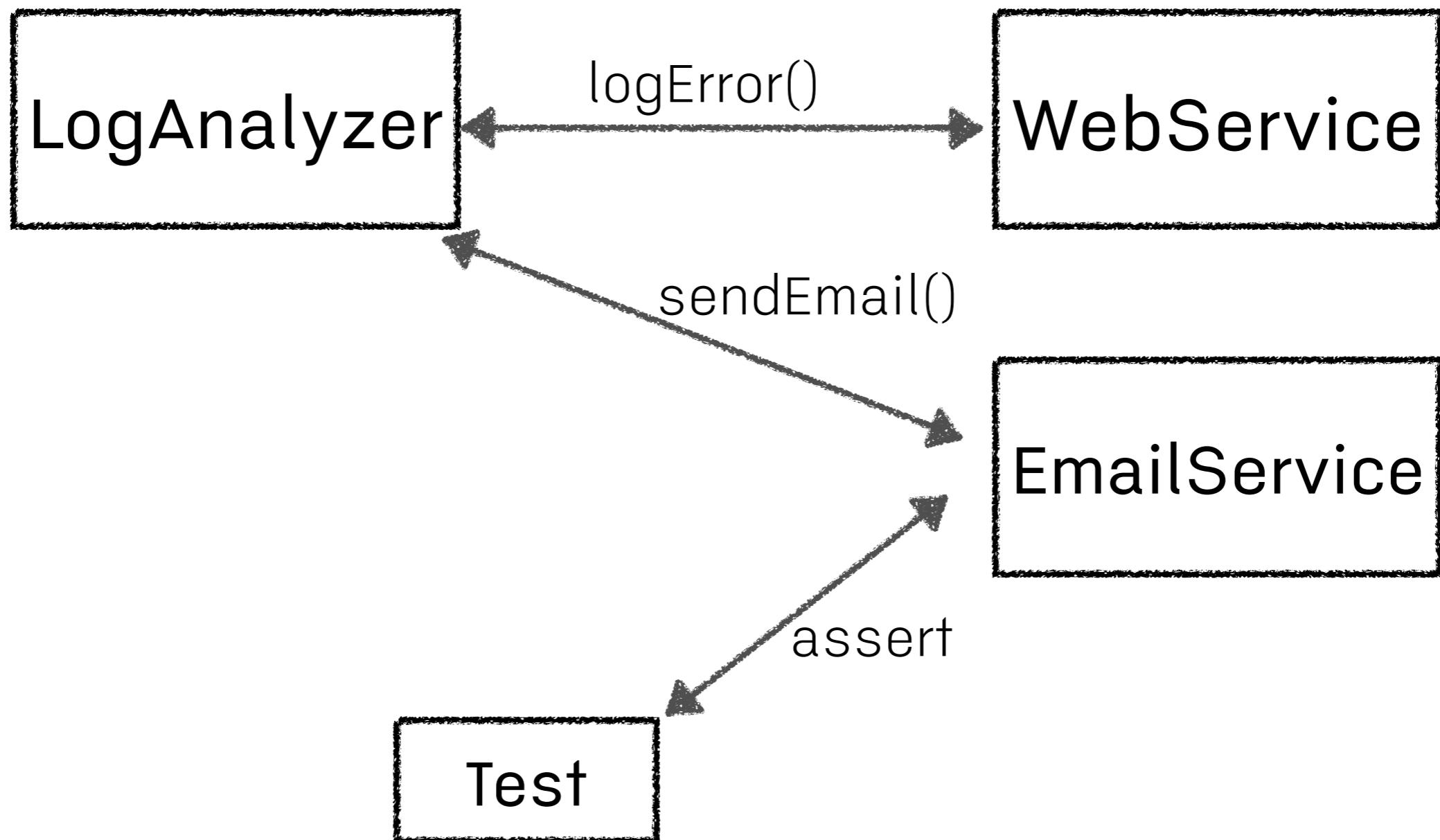
Problem



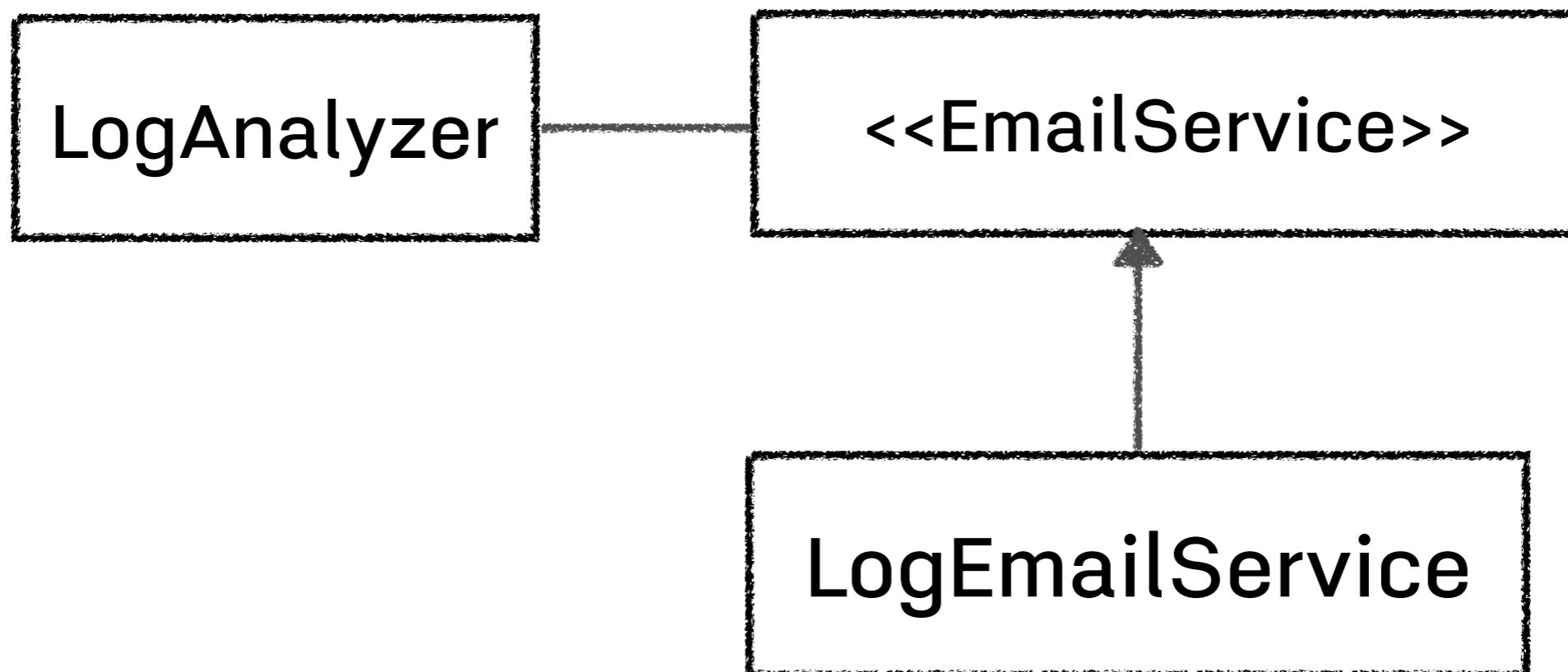
Send Email with expected input ?

```
public void analyze(String filename) {  
    if (filename.length() < 8) {  
        try {  
            this.webService.logError("File name is too short:" + filename);  
        } catch (Exception exception) {  
            this.emailService.sendEmail("TO", "SUBJECT", exception.getMessage());  
        }  
    }  
}
```

Problem



Step 1 :: Break dependency



Step 2 :: Write test

```
@Test
public void analyzeWebServiceThrowsErrorThenSendEmail() throws Exception {
    StubWebService stubWebService = new StubWebService();
    MockEmailService mockEmailService = new MockEmailService();

    LogAnalyzer logAnalyzer = new LogAnalyzer();
    logAnalyzer.setWebService(stubWebService);
    logAnalyzer.setEmailService(mockEmailService);

    logAnalyzer.analyze("abc.txt");

    assertEquals("TO", mockEmailService.getTo());
    assertEquals("SUBJECT", mockEmailService.getSubject());
    assertEquals("Fake exception", mockEmailService.getBody());
}
```

Step 3 :: Write code

Start coding by your pair

Mock Framework



Mockito

Lightweight Java mocking

- ★ Enable mock creation
- ★ Enable verification
- ★ Enable stubbing

<https://github.com/mockito/mockito>

ฝึก ฝึก ฝึก และ ฝึก



Resources

<https://github.com/github-sprint3r/tdd-java-workshop>