



Lesson 05

Test Management 2/2



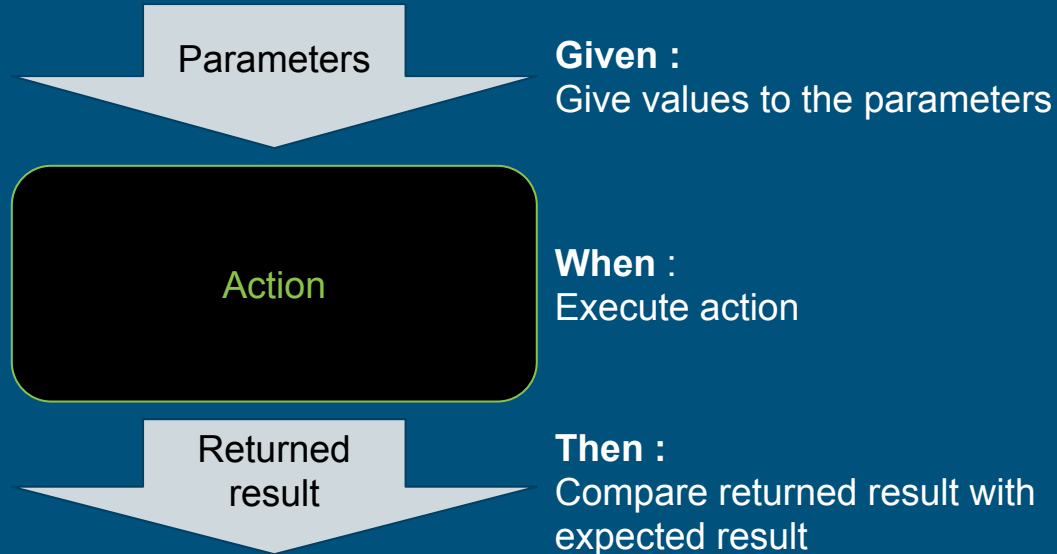
DEVOPS - ITI 4
HEI 2021-2022



Previously in Lesson 04 : Unit Testing (1/2)

- Role of Unit Testing : Check each component individually
- Check :
 - gives expected outputs when inputs are valid
 - does not give expected outputs when inputs are invalid
 - should never crash

Previously in Lesson 04 : Unit Testing (2/2)



Previously in Lesson 04 : JUnit

With JUnit, we can :

- Use *Assert* class to check the result
- Use *expected* attribute in *@Test* annotation to catch an expected exception
- Use annotation to prepare and purge data

AssertJ

Why use another framework ?



Assertion in JUnit

- `assertEquals(E expected, E actual)`
- `assertTrue(boolean condition)` or `assertFalse(boolean condition)`
- `assertNull(E actual)` or `assertNotNull(E actual)`

Problematic : How to check ?

```
public List<Card> draw(List<Card> cards) {  
    List<Card> result = new ArrayList<Card>(cards);  
    Collections.shuffle(result);  
    return result;  
}
```


With JUnit

```
@Test
public void shouldDrawCard() {
    //GIVEN
    List<Card> cards = Arrays.asList(Card.KING, Card.AS, Card.JAKE);

    //WHEN
    List<Card> result = cardService.draw(cards);

    //THEN
    assertEquals(cards.size(), result.size());
    for(Card card : cards) {
        assertTrue(result.contains(card));
    }
    for(Card card : result) {
        assertTrue(cards.contains(card));
    }
}
```

Opinion

- Not easy to read
- Complex to implement



What we need

- More method to check complex situations
- A more verbose syntax
- AssertJ

How to use AssertJ ?



AssertJ

- To write more readable assertions
- Provides a rich set of assertions
- Has really useful error messages

Using AssertJ

```
<dependency>  
  <groupId>org.assertj</groupId>  
  <artifactId>assertj-core</artifactId>  
  <version>3.21.0</version>  
  <scope>test</scope>  
</dependency>
```

Using AssertJ in TestCase

- AssertJ is to be used in addition to JUnit
- The change is only for the assertion

Using AssertJ in TestCase

- Assertion begins by `Assertions.assertThat(objectUnderTest)`
- Call your check after :
 - `isTrue() / isFalse()`
 - `isNull() / isNotNull()`
 - `isEqualTo(expected)`
 - `contains(valuesExpected) / doesNotContain(valuesUnexpected)`
 - ...

Example

```
@Test
public void shouldDrawCard() {
    //GIVEN
    List<Card> cards = Arrays.asList(Card.KING, Card.AS, Card.JAKE);

    //WHEN
    List<Card> result = cardService.draw(cards);

    //THEN
    Assertions.assertThat(result)
        .containsExactlyInAnyOrderElementsOf(cards);
}
```

Yes, but I don't know all methods ...



Use code completion !!!



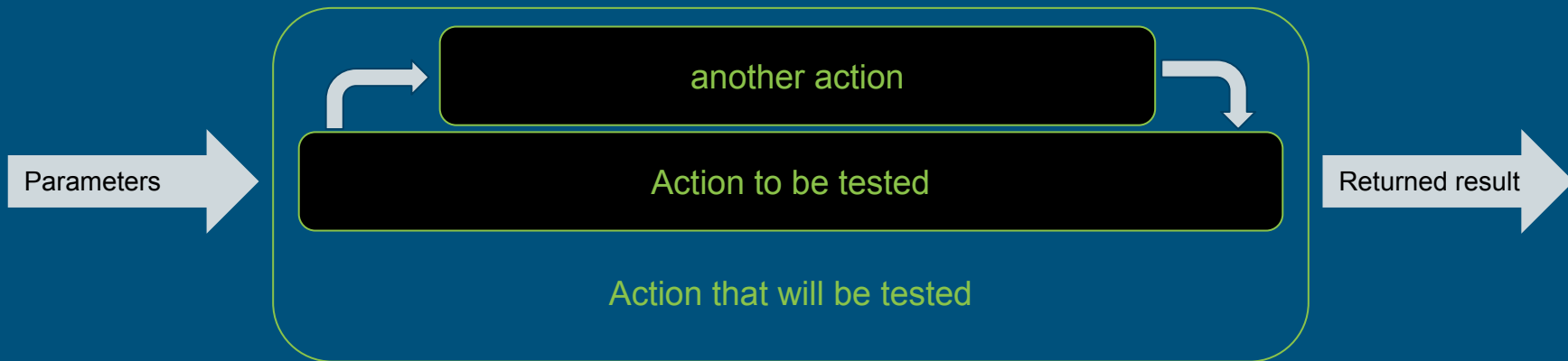
Mockito

Why use
another
framework
again ?



Limit of JUnit

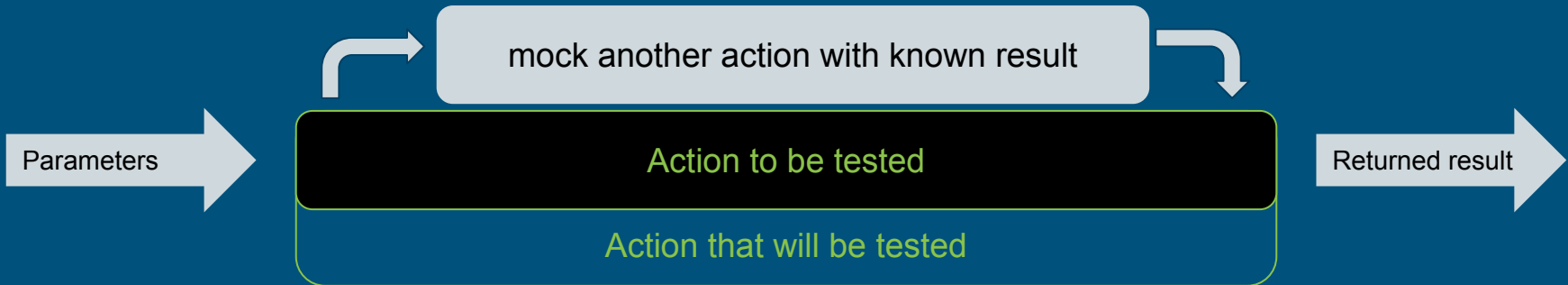
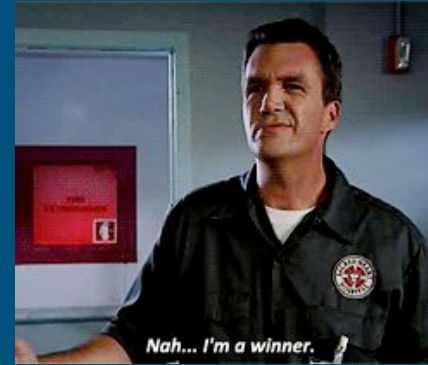
- With JUnit, we cannot check only action if it needs other actions



- Not a real Unit test, because we test more than one component

In Mock we trust

- Solution : Mock others actions !



How to use Mockito ?



Mockito

- Framework to mock java object
- Can change the result (return / throw) of a method
- Can check calls to methods

Using Mockito

```
<dependency>  
  <groupId>org.mockito</groupId>  
  <artifactId>mockito-core</artifactId>  
  <version>3.12.4</version>  
  <scope>test</scope>  
</dependency>
```

Using Mockito in TestCase

- Mockito is to be used in addition to JUnit
- Add `@RunWith(MockitoJUnitRunner.class)` annotation on test case class to use Mockito

Mock

- To simulate the working on a java object
- Add `@Mock` annotation above object to mock
- **WARN** : Not react normally but only according to your instructions !!!

Inject Mock

- To use mocks inside another object like a service
- Add *@InjectMock* annotation above object that needs use mocks

How to test VisioConferenceServiceTestCase ?



	@Mock	@InjectMock
VisioConferenceService		✓
VideoService	✓	
AudioService	✓	

How to modify
the result of a
method ?



When

- To indicate the method to simulate
- `Mockito.when(myMockedObject.mymethod())`
- **WARN** : All methods of mocked object do not work normally !!!

thenReturn

- Return object instead of result of real method
- `Mockito.when(myMockedObject.myMethod()).thenReturn(myValue);`

thenThrow

- Throw Exception instead of result of real method
- `Mockito.when(myMockedObject.myMethod()).thenThrow(myException);`

thenCallRealMethod

- Throw Exception instead of result of real method
- `Mockito.when(myMockedObject.mymethod()).thenCallRealMethod();`

Example : TeamServiceImpl

```
private List<Team> listTeams() {  
    return teamDao.read();  
}  
  
public long countTeams() {  
    LOG.debug("countTeams");  
    long teamCount = this.listTeams().size();  
    LOG.trace("countTeams | return : {}",teamCount);  
    return teamCount;  
}
```

Example : TeamServiceTestCase

```
@Test
public void shouldCountTeams() {
    //GIVEN
    List<Team> teams = Arrays.asList(new Team(1,"new Team"), new Team(2,"Muppet"));
    Mockito.when(teamDao.read()).thenReturn(teams);

    //WHEN
    long result = teamService.countTeams();

    //THEN
    Assertions.assertThat(result).isEqualTo(teams.size());
}
```

How to check the call of a method ?



Verify

- To indicate the call to check
- `Mockito.verify(myMockedObject, VerificationMode).myMethod(params,...);`

Verification Mode

- Set the expected number of calls for the method
- Examples :
 - `Mockito.never();`
 - `Mockito.times(int wantedNumberOfInvocations);`
 - `Mockito.atLeast(int minNumberOfInvocations);`
 - `Mockito.atMost(int maxNumberOfInvocations);`
 - ...
- `Mockito.verify(myMockedObject)` means the same thing as `Mockito.verify(myMockedObject, Mockito.times(1))`

Verify for any parameters

- Verify takes into account the parameters of the method
- To take into account all the values : `Mockito.Any()`

Example : Verify

```
@Test
public void shouldReturnUsersByTeam() throws TeamNotFoundException {

    //GIVEN
    Team newTeam = new Team(1,"new Team");
    List<User> newTeamUsers = Arrays.asList(new User("Ben", "Becker",1),new User("Olivier", "Atton",1));
    Team muppets = new Team(2,"Muppets");
    List<User> muppetUsers = Arrays.asList(new User("Mark", "Landers",2));

    Mockito.when(teamService.getTeam(newTeam.getName())).thenReturn(newTeam);
    Mockito.when(userDao.read()).thenReturn(Stream.concat(newTeamUsers.stream(),
muppetUsers.stream()).collect(Collectors.toList()));

    //WHEN
    List<User> result = userService.listUsersByTeam(newTeam.getName());

    //THEN
    Assertions.assertThat(result).containsExactlyElementsOf(newTeamUsers);
    Mockito.verify(teamService).getTeam(newTeam.getName());
    Mockito.verify(teamService,Mockito.never()).getTeam(muppets.getName());

}
```

Code coverage

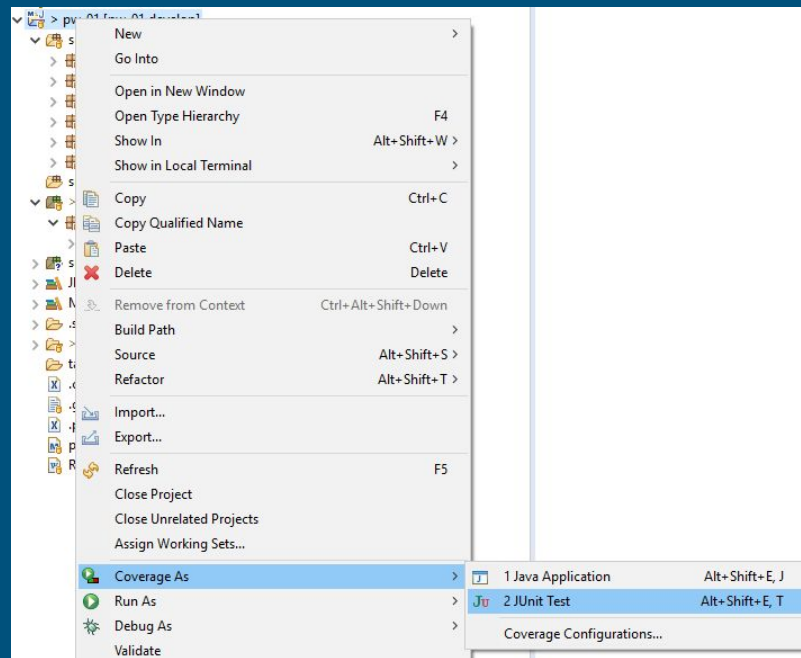
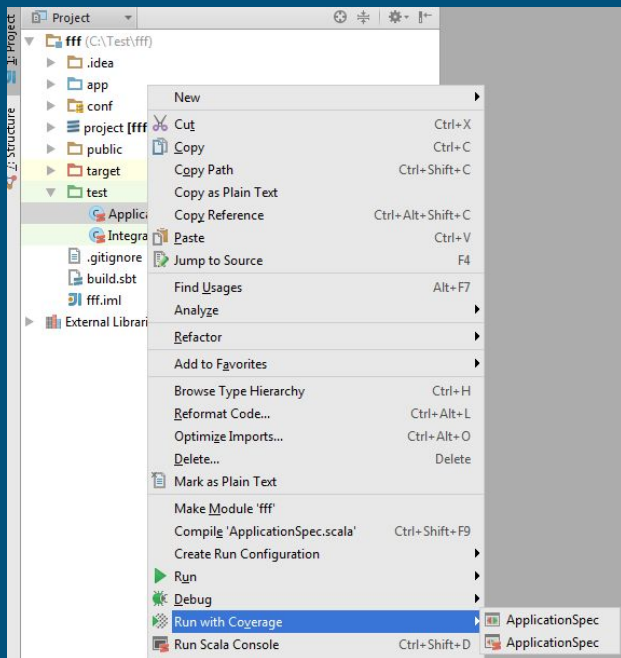
How to make
sure we test
everything ?



Code coverage

- Measure to describe the proportion of code executed during tests
- Higher the percentage, higher the code is safe

In IDE















With Maven



Report

lesson-05

Element ↕	Missed Instructions ↕	Cov. ↕	Missed Branches ↕	Cov. ↕	Missed ↕	Cxty ↕	Missed ↕	Lines ↕	Missed ↕	Methods ↕	Missed ↕	Classes ↕
 hei.devops.lesson05.entity		33 %		7 %	18	25	32	49	11	18	0	2
 hei.devops.lesson05.dao.impl		65 %		n/a	3	9	3	13	3	9	0	4
 hei.devops.lesson05.service.impl		95 %		91 %	3	21	2	42	2	15	1	5
 hei.devops.lesson05.enumeration		100 %		n/a	0	1	0	5	0	1	0	1
 hei.devops.lesson05.exception		100 %		n/a	0	2	0	4	0	2	0	2
Total	112 of 387	71 %	14 of 26	46 %	24	58	37	113	16	45	1	14

Question

How test this method with Mockito ?

```
public User getUser(Long userId) throws UserNotFoundException {  
    LOG.debug("getUser | params : userId = {}",userId);  
    User user = userDao.read(userId);  
    if(user!=null) {  
        return user;  
    }  
    throw new UserNotFoundException(userId);  
}
```

Thank you for your attention !



Links :

- Sources

- <http://www.commitstrip.com/>
- <https://giphy.com/>
- <https://joel-costigliola.github.io/assertj/>
- <https://www.fierdecoder.fr/2016/05/assertj-pour-des-assertions-plus-expressives/>
- <https://www.jmdoudoux.fr/java/dej/chap-objets-mock.htm>
- <https://www.baeldung.com/mockito-annotations>
- https://en.wikipedia.org/wiki/Code_coverage
- <https://www.mkymong.com/maven/maven-jacoco-code-coverage-example/>
- <https://www.jacoco.org/jacoco/trunk/doc/maven.html>

- Examples :

- <https://gitlab.com/hei-devops/lesson/lesson-2020/lesson-05>