Mockito đem lại sự rõ ràng trong việc test các phương thức của java.

* Ví dụ: khi ta muốn test phương thức kiểm tra email có hợp lệ:

**isEmailValid**(email) của class EmailServiceImpl, bên trong PT này có thể hiện của **EmailRepository** gọi **isExist**(email) để truy vấn vào DB để check tồn tại hay chưa, nhưng chúng ta chỉ muốn test logic của **isEmailValid(email)** mà không muốn quan tâm tới **isExist() của EmailRepository**  vì 1 vài lý do như: **EmailRepository**  chưa được implêmnt đúng/ chỉ muốn kiểm tra **isEmailValid()**  của EmailServiceImpl / chúng ta không muốn việc test **isEmailValid**  không thành công do xuất phát từ **isExist. => Lúc này isExist**  là 1 phụ thuộc của **isEmailValid**

Vì vậy chúng ta cần giả lập hành vi của các phụ thuộc (**isExist**) => Mockito sẽ xử lý vấn đè này.

@injectMock: dùng cho đối tượng class cần test

@Mock: dùng cho các thành phần phụ thuộc trong phươngthức cần test / dùng @Mockbean khi kết hợp với spring

Để test trong sprring: mọi class test cần thêm: **@Runwitth(SpringRunner.class)**  trên đầu mỗi class cần test, Khi chạy test, **spring sẽ tạo ra 1 context riêng** để chứa các **Bean** cho phần test.

* @Để test static moethod:Dùng mockitoInline
* Khi dùng mockito-core: cần đảm bảo các thư viện đi kèm được load vào prọect đúng phiên bản vvd:
* <!-- https://mvnrepository.com/artifact/org.mockito/mockito-core -->
* <dependency>
* <groupId>org.mockito</groupId>
* **<artifactId>mockito-core</artifactId>**
* **<version>3.8.0</version>**
* <scope>test</scope>
* </dependency>

+ mockito-core trên đòi hỏi các thư viện theo đúng version sau:

| **Category/License** |  | **Group / Artifact** | **Version** | **Updates** |
| --- | --- | --- | --- | --- |
| [Bytecode](https://mvnrepository.com/open-source/bytecode-libraries) Apache 2.0 |  | [net.bytebuddy](https://mvnrepository.com/artifact/net.bytebuddy) » [byte-buddy](https://mvnrepository.com/artifact/net.bytebuddy/byte-buddy) | [1.10.20](https://mvnrepository.com/artifact/net.bytebuddy/byte-buddy/1.10.20) | [1.12.8](https://mvnrepository.com/artifact/net.bytebuddy/byte-buddy/1.12.8) |
| Apache 2.0 |  | [net.bytebuddy](https://mvnrepository.com/artifact/net.bytebuddy) » [byte-buddy-agent](https://mvnrepository.com/artifact/net.bytebuddy/byte-buddy-agent) | [1.10.20](https://mvnrepository.com/artifact/net.bytebuddy/byte-buddy-agent/1.10.20) | [1.12.8](https://mvnrepository.com/artifact/net.bytebuddy/byte-buddy-agent/1.12.8) |
| [Reflection](https://mvnrepository.com/open-source/reflection-libraries) Apache 2.0 |  | [org.objenesis](https://mvnrepository.com/artifact/org.objenesis) » [objenesis](https://mvnrepository.com/artifact/org.objenesis/objenesis) | [3.1](https://mvnrepository.com/artifact/org.objenesis/objenesis/3.1) | [3.2](https://mvnrepository.com/artifact/org.objenesis/objenesis/3.2) |

Để giả lập 1 đối tượng ta dùng X x =Mockito.mock(X.class);

Khi dùng mock như trên, x không thể dùng tiếp để set các thuộc tính vd: x.setA() – invalid

Mà sẽ phải dùng when(x.getA()).thenReturn(y); để giả lập giá trị A của x

Trường hợp X x = new X(); - x có thẻ dùng được các phương thức get/set, nhưng không thể dùng trong

When, vd: Mockito.when(x.getA).thenReturn(B) - invalid