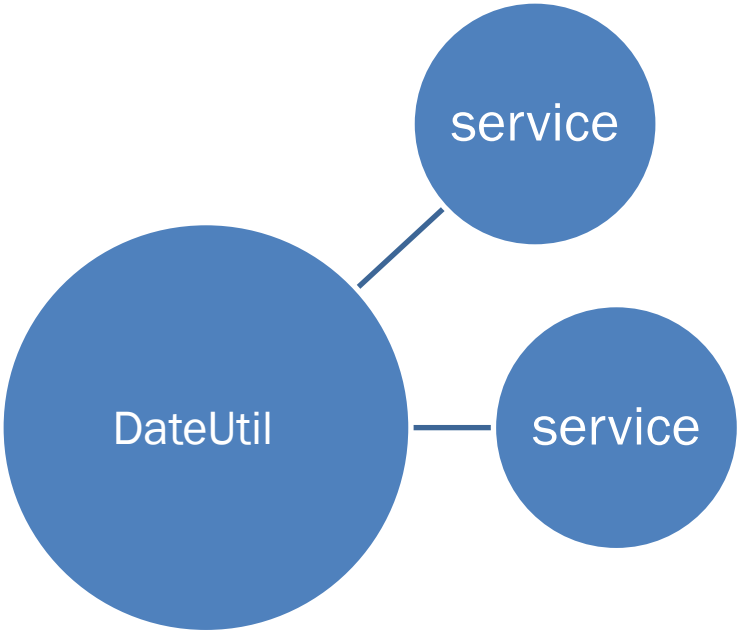
Mock Testing

ITU7 Samuel

Agenda

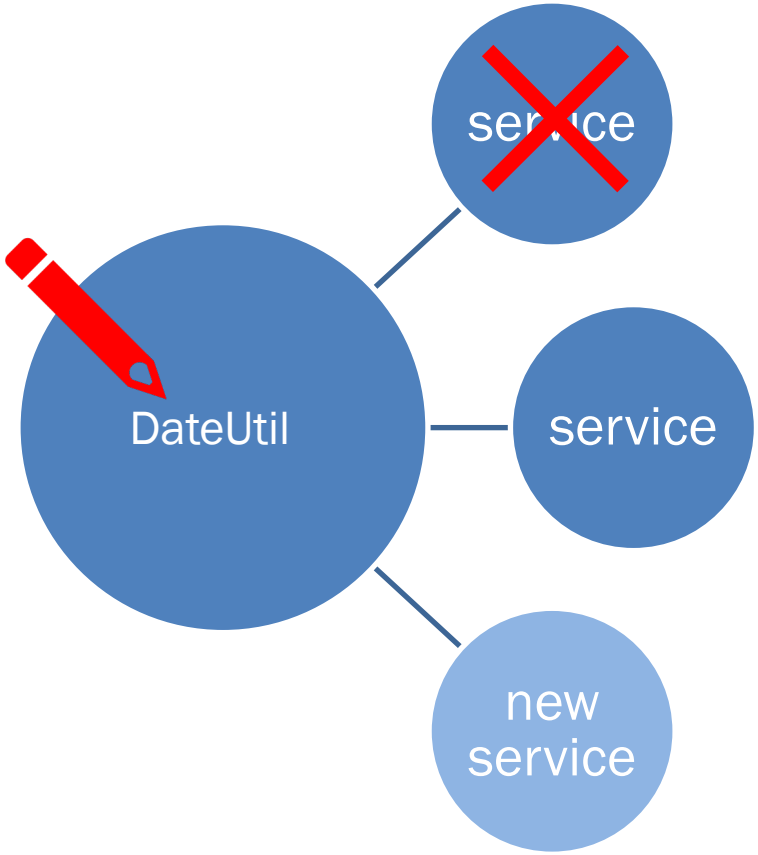
- Why testing ?
- Why mock ?
- Faster & cleaner way

寫程式就是要模組化啊



西元年 -> 民國年

修改

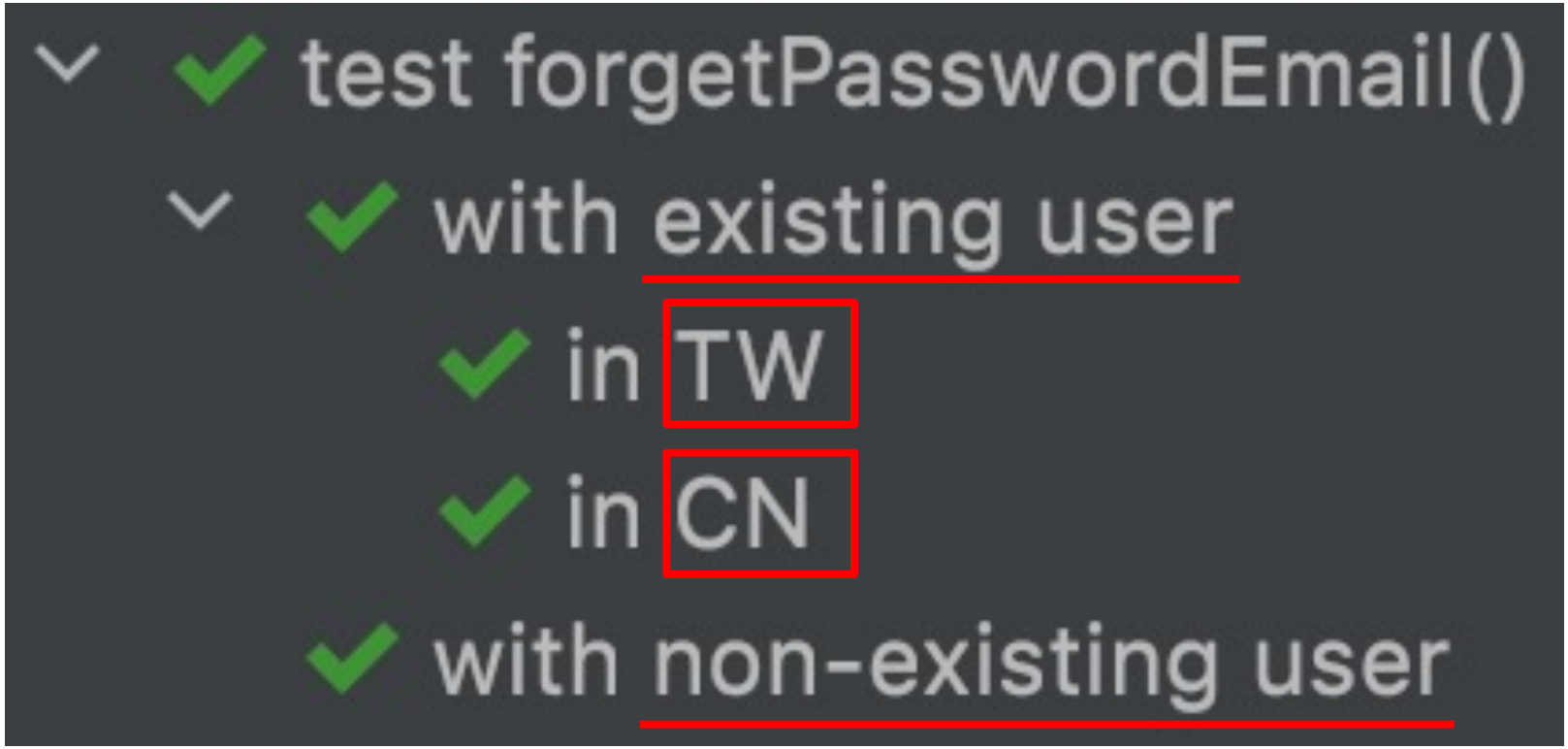


solution

- 用單元測試就對了
- 未來重構程式

more ...

測試即文件


test forgetPasswordEmail()
with existing user
in TW
in CN
with non-existing user

程式有可能...

import 別人的套件

{ REST }

call API

從資料庫拿資料

假如要講求效率的話...

別人自己都測過了

{ REST }

網路連線

資料庫連線

且可能有這些特殊情況...

取得隨機整數

{ REST }

一定要網路錯誤

資料被弄髒

Solution

用 **Mock** 就對了

```
String paasPwd = lookUpValuesRepository.findMeaningByLookUpTypeAndZHTAndDDT(LookUpValuesConstants.PAAS_LOGIN_CHECK_API_PASSWORD);
```



```
doReturn( toBeReturned: "12345678")
        .when(lookUpValuesRepository).findMeaningByLookUpTypeAndZHTAndDDT(
```

小結

單元測試是為了 “未來” 而生

小結

單元測試原則：

- **F**ast : Mock 加快執行
- **I**solated : Mock 解除依賴
- **R**epeated : Mock 可隨時執行
- **S**
- **T**

小結

Mock 需要花時間

Test cases for ...

忘記密碼寄信功能

```
public SendMailResponse forgetPasswordEmail(QueryMap queryMap) {
```

Test cases for ...

```
@ExtendWith(MockitoExtension.class)
public class LoginServiceTest {
    @Test
    void testForgetPasswordEmailWithExistingUserAndTw {...}

    @Test
    void testForgetPasswordEmailWithExistingUserAndCN {...}

    @Test
    void testForgetPasswordEmailWithNonExistingUser {...}
}
```

Problems

```
@ExtendWith(MockitoExtension.class)
public class LoginServiceTest {
    @Test
    void testForgetPasswordEmailWithExistingUserAndTw {...}

    @Test
    void testForgetPasswordEmailWithExistingUserAndCN {...}

    @Test
    void testForgetPasswordEmailWithNonExistingUser {...}
}
```

```
@Test
public void testForgetPasswordEmailWithExistingUserAndTw() throws Exception {
    // Given
    doReturn(FORGET_PWD_EMAIL_URL)
            .when(lookUpValuesRepository).findMeaningByLookUpTypeAndZHTAndDDT(

    JSONObject requestBody = new JSONObject() {{
        put("queryMap", new JSONObject(){{
            put("countryCode", "");
            put("type", "check");
            put("userName", "samuel@samuel.com");
        }});
    }};
    JSONObject responseBody = new JSONObject() {{
        put("response", new JSONObject(){{
            put("CONTACT_NAME", DEFAULT_CONTACT_NAME);
            put("STATUS", "USER EXIST");
            put("USER_ID", DEFAULT_USER_ID);
        }});
    }};
    when(RestClientUtils.post(eq( value: PAAS_URL + FORGET_PWD_EMAIL_URL), jsonE
            .thenReturn(responseBody);

    doNothing().when(paasClientUtils).syncByPaas(any());

    // When
    QueryMap queryMap = QueryMap.builder()
            .type("email")
            .countryCode("")
            .userName(DEFAULT_EMAIL)
            .build();
    SendMailResponse actual = loginService.forgetPasswordEmail(queryMap);

    // Then
    SendMailResponse expected = new SendMailResponse();
    assertEquals(expected, actual);

    verify(paasClientUtils).syncByPaas(any());
    verify(mailService).sendMailVerify(
            eq(MAIL_REFERENCE),
            eq(String.valueOf(DEFAULT_USER_ID)),
            eq(MailTemplateConstants.FORGET_PWD_TITLE_TW),
            eq(MailTemplateConstants.FORGET_PWD_FILENAME_TW),
            any()
    );
}
```

3 x 40 = 120 (lines)

參數化

- test case 合併 (tw、cn)

```
@ExtendWith(MockitoExtension.class)
public class LoginServiceTest {
    @Test
    void testForgetPasswordEmailWithExistingUserAndTw {...}

    @Test
    void testForgetPasswordEmailWithExistingUserAndCN {...}

    @Test
    void testForgetPasswordEmailWithNonExistingUser {...}
}
```

```
@ParameterizedTest(name = "in {0}")
@ValueSource(strings = {"TW", "CN"})
void testForgetPasswordEmailWithExistingUser(String input)
    // Given
```

json 檔

```java
@Test
public void testForgetPasswordEmailWithExistingUserAndTw() throws Exception {
    // Given
    doReturn(FORGET_PWD_EMAIL_URL)
            .when(lookUpValuesRepository).findMeaningByLookUpTypeAndZHTAndDDT(

    JSONObject requestBody = new JSONObject() {{
        put("queryMap", new JSONObject(){{
            put("countryCode", "");
            put("type", "check");
            put("userName", "samuel@samuel.com");
        }});
    }};
    JSONObject responseBody = new JSONObject() {{
        put("response", new JSONObject(){{
            put("CONTACT_NAME", DEFAULT_CONTACT_NAME);
            put("STATUS", "USER EXIST");
            put("USER_ID", DEFAULT_USER_ID);
        }});
    }};
    when(RestClientUtils.post(eq( value: PAAS_URL + FORGET_PWD_EMAIL_URL), jsonE
            .thenReturn(responseBody);

    doNothing().when(paasClientUtils).syncByPaas(any());

    // When
    QueryMap queryMap = QueryMap.builder()
            .type("email")
            .countryCode("")
            .userName(DEFAULT_EMAIL)
            .build();
    SendMailResponse actual = loginService.forgetPasswordEmail(queryMap);

    // Then
    SendMailResponse expected = new SendMailResponse();
    assertEquals(expected, actual);

    verify(paasClientUtils).syncByPaas(any());
    verify(mailService).sendMailVerify(
            eq(MAIL_REFERENCE),
            eq(String.valueOf(DEFAULT_USER_ID)),
            eq(MailTemplateConstants.FORGET_PWD_TITLE_TW),
            eq(MailTemplateConstants.FORGET_PWD_FILENAME_TW),
            any()
    );
}
```

responseBody.json

```json
{
  "response": {
    "CONTACT_NAME":"Samuel",
    "STATUS":"USER EXIST",
    "USER_ID":1
  }
}
```

不必要的流程

檢查 **call API** 所帶的 **body**

- 只關注 **call** 完所回傳的 **response body**
- 之後修改不用動

@Nested

— function 所有 test case

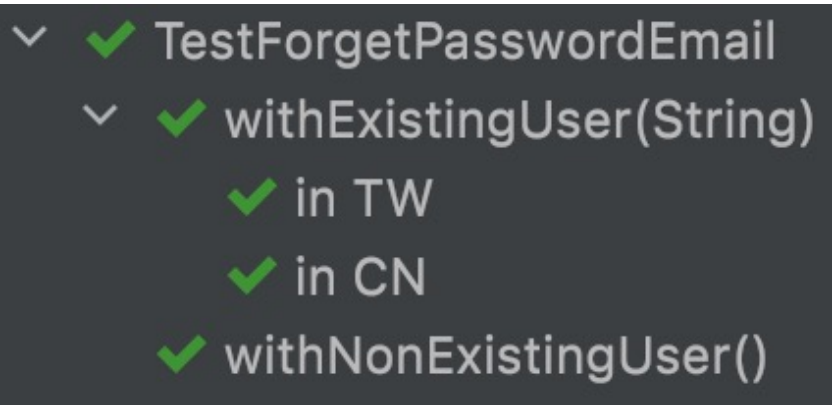
- 分層結構
- 共用邏輯

```
@Nested
@DisplayName("test forgetPasswordEmail()")
class TestForgetPasswordEmail {
    @BeforeEach
    void init() throws Exception {
        doReturn(FORGET_PWD_EMAIL_URL)
                .when(lookUpValuesRepository).
    }
```

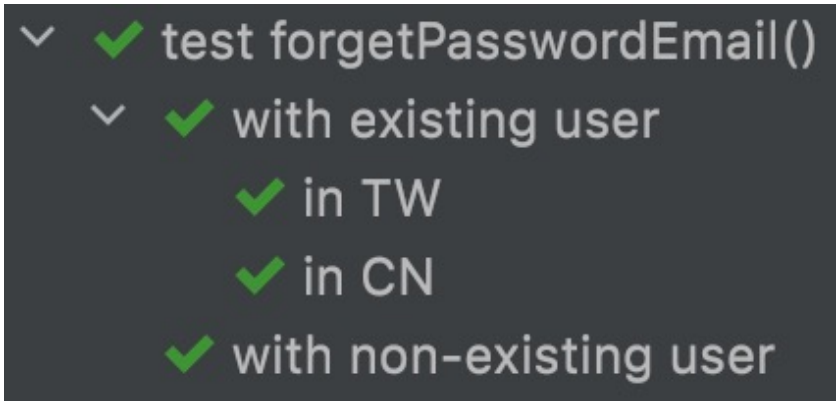
@DisplayName

```
@Nested
@DisplayName("test forgetPasswordEmail()")
class TestForgetPasswordEmail {
    @BeforeEach
    void init() throws Exception {
        doReturn(FORGET_PWD_EMAIL_URL)
                .when(lookUpValuesRepository).findMeaningByLo
    }
```

駝峰



空格



better than before

```java
@Test
public void testForgetPasswordEmailWithExistingUserAndTw() throws Exception {
    // Given
    doReturn(FORGET_PWD_EMAIL_URL)
            .when(lookUpValuesRepository).findMeaningByLookUpTypeAndZHTAndDDT(

    JSONObject requestBody = new JSONObject() {{
        put("queryMap", new JSONObject(){{
            put("countryCode", "");
            put("type", "check");
            put("userName", "samuel@samuel.com");
        }});
    }};
    JSONObject responseBody = new JSONObject() {{
        put("response", new JSONObject(){{
            put("CONTACT_NAME", DEFAULT_CONTACT_NAME);
            put("STATUS", "USER EXIST");
            put("USER_ID", DEFAULT_USER_ID);
        }});
    }};
    when(RestClientUtils.post(eq( value: PAAS_URL + FORGET_PWD_EMAIL_URL), jsonE
            .thenReturn(responseBody);

    doNothing().when(paasClientUtils).syncByPaas(any());

    // When
    QueryMap queryMap = QueryMap.builder()
            .type("email")
            .countryCode("")
            .userName(DEFAULT_EMAIL)
            .build();
    SendMailResponse actual = loginService.forgetPasswordEmail(queryMap);

    // Then
    SendMailResponse expected = new SendMailResponse();
    assertEquals(expected, actual);

    verify(paasClientUtils).syncByPaas(any());
    verify(mailService).sendMailVerify(
            eq(MAIL_REFERENCE),
            eq(String.valueOf(DEFAULT_USER_ID)),
            eq(MailTemplateConstants.FORGET_PWD_TITLE_TW),
            eq(MailTemplateConstants.FORGET_PWD_FILENAME_TW),
            any()
    );
}
```



```java
@ParameterizedTest(name = "in {0}")
@ValueSource(strings = {"TW", "CN"})
@DisplayName("with existing user")
void withExistingUser(String input) throws Exception {
    // Given
    String path = RESOURCE_PATH + "testForgetPasswordEmailWithExistingUser/";
    String text = ResourceHelper.getResourceAsString(getClass(),  name: path + "responseBody.json");
    when(RestClientUtils.post(eq( value: PAAS_URL + FORGET_PWD_EMAIL_URL), any(), any(), any()))
            .thenReturn(new JSONObject(text));

    doNothing().when(paasClientUtils).syncByPaas(any());

    // When
    text = ResourceHelper.getResourceAsString(getClass(),  name: path + "queryMapWith" + input + ".json");
    SendMailResponse actual = loginService.forgetPasswordEmail(gson.fromJson(text, QueryMap.class));

    // Then
    SendMailResponse expected = new SendMailResponse();
    assertEquals(expected, actual);

    verify(paasClientUtils).syncByPaas(any());
    verify(mailService).sendMailVerify(
            eq(MAIL_REFERENCE),
            eq(String.valueOf(DEFAULT_USER_ID)),
            eq(FieldUtils.readStaticField(MailTemplateConstants.class,  fieldName: "FORGET_PWD_TITLE_" + inpu
            eq(FieldUtils.readStaticField(MailTemplateConstants.class,  fieldName: "FORGET_PWD_FILENAME_" + i
            any()
    );
}
```

Principle

1. 以參數化合併邏輯近似的 **test case**
2. 要模擬的 **object** 屬性數 **> 1** ，寫進 **json** 檔裡
3. 不用檢查或模擬不影響結果的物件
4. 一 **function** 所有 **test case**，都以 **@Nested** 包進 **InnerClass**
5. 每個 **test case** 都運用 **@DisplayName** 優化顯示名稱

Q & A

大聯大控股
WPG Holdings

Thank you

產業首選 · 通路標竿