Владимир Афанасьев

Как замокать Singleton

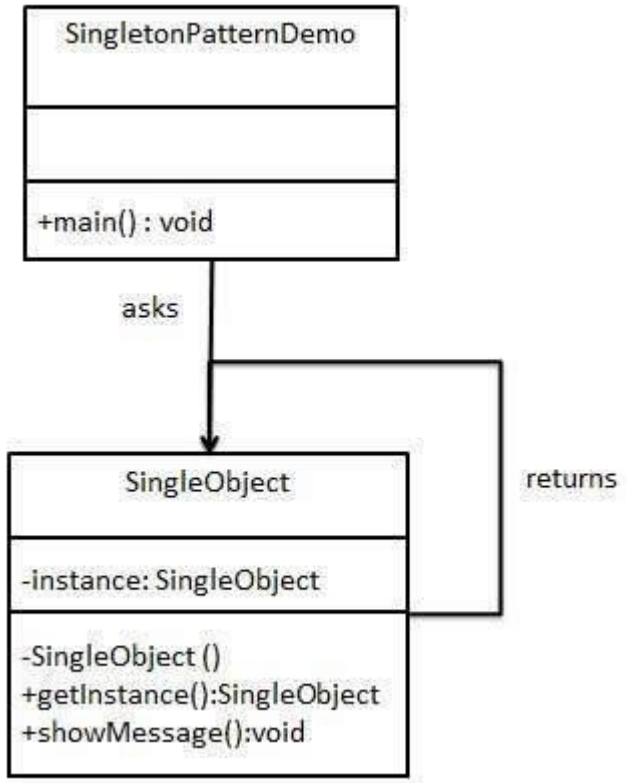
«…Оттого, что в кузнице не было гвоздя.»

Мир наизнанку.

- Потренируемся
- Замокаем в дот.нете
- Пройдемся энтерпрайзными сапогами по фронт-энду
- И, наконец, разузнаем как эта тема могла развиться из проектного опыта.
- А в конце – вопросы

Нагуглим немного Singleton-а

- … трудности с тестированием…
- … вместо синглтона нельзя подпихнуть Mock-объект…
- … нарушает Single Responseability….
- … не используйте…
- … антипаттерн…



Java (страшный синглтон)

```
1.  public abstract class Singleton<T extends Object>  {
2.      private static volatile Object _instance;
3.      private static Object lock = new Object();
4.      protected static <T extends Object> T get_instance_internal(Class<T> clazz)
5.              throws IllegalAccessException, InstantiationException {
6.          if(_instance == null) {
7.              synchronized (lock) {
8.                  if (_instance == null) {
9.                      _instance = clazz.newInstance();
10.                 }
11.             }
12.         }
13.         return (T) _instance;
14.     }
15.     public static  <T extends Object> T get_instance() throws Exception {
16.         throw new NotImplementedException(); //Instancing is not defined in child
17.     };
18. }
```

Java (имплементация)

```
1.  public class ClassForMock extends Singleton<ClassForMock> {
2.
3.      // Singleton workarround
4.      public static ClassForMock get_instance() throws Exception {
5.          return (ClassForMock) get_instance_internal(ClassForMock.class);
6.      }
7.
8.      public String getValue(){
9.          return "This is non-mocked value";
10.     }
11. }
```

Java (используется как зависимость)

```
1.  public class Container {
2.      public static String getValue() throws Exception {
3.          return ClassForMock.get_instance().getValue();
4.      }
5.  }
```

Java (что попытаемся мокать)

….
_instance = clazz.newInstance();
…

…
public static <T extends Object> T get_instance()
….

То есть конструктор или статический метод класса.

Java. PowerMock

По документации умеет мокать:

- И конструкторы
- И статические методы

- https://github.com/powermock/powermock/wiki

"Using PowerMock, it becomes possible to mock static methods, remove static initializers, allow mocking without dependency injection, and more."

Java. PowerMock. Статические методы

```
1.  @RunWith(PowerMockRunner.class)
2.  @PrepareForTest({ClassForMock.class})
3.  public class MockStaticTest {
4.      private String mockedValue = "This value is mocked !!!!";
5.      @Mock
6.      private ClassForMock classForMock;
7.      @Before
8.      public void setup() throws Exception{
9.          PowerMockito.mockStatic(ClassForMock.class);
10.
11.         classForMock = Mockito.mock(ClassForMock.class);
12.         Mockito.when(classForMock.getValue()).thenReturn(mockedValue);
13.         PowerMockito.when(ClassForMock.get_instance()).thenReturn(classForMock);
14.     }
15.     @Test
16.     public void checkStaticInstanceMock() throws Exception{
17.         Assert.assertEquals(mockedValue, ClassForMock.get_instance().getValue());
18.     }
19.     @Test
20.     public void checkStaticInstanceMockThoughContainer() throws Exception{
21.         Assert.assertEquals(mockedValue, Container.getValue());
22.     }
23. }
```

Java. PowerMock. Конструктор. Проблема

До компиляции:

```
public abstract class Singleton<T extends Object>
{
private static volatile Object _instance;
protected static <T extends Object> T
get_instance_internal(Class<T> clazz){
      return (T) _instance;
}
```

В JVM:

```
public abstract class Singleton
{
private static volatile Object _instance;
protected static Object
get_instance_internal(Class clazz){
      return  _instance;
}
```

Конструктор, как выяснилось мокается тем, что подменяется оператор new

Java. Синглтон по рекомендациям Java

```
1.  public class ClassForMockSelfSingleton {
2.      private static ClassForMockSelfSingleton _instance;
3.      private static Object sync = new Object();
4.      public static ClassForMockSelfSingleton getInstance() {
5.          if(_instance == null) {
6.              synchronized (sync) {
7.                  if(_instance == null) {
8.                      _instance = new ClassForMockSelfSingleton();
9.                  }
10.             }
11.         }
12.         return  _instance;
13.     }
14.     public String getValue(){
15.         return "this is non-mocked value";
16.     }
17. }
```

Java. PowerMock. Конструктор

```
1.  @PrepareForTest (ClassForMockSelfSingleton.class)
2.  public class MockConstructorTest {
3.      private String mockedValue = "This value is mocked !!!!";
4.      @Mock
5.      private ClassForMockSelfSingleton classForMock;
6.      @Before
7.      public void setup() throws Exception{
8.
9.          classForMock = Mockito.mock(ClassForMockSelfSingleton.class);
10.         Mockito.when(classForMock.getValue()).thenReturn(mockedValue);
11.
12.         PowerMockito.mock(ClassForMockSelfSingleton.class);
13.         PowerMockito.whenNew(ClassForMockSelfSingleton.class)
14.                 .withAnyArguments()
15.                 .thenReturn(classForMock);
16.     }
17.     @Test
18.     public void checkConstructorMock() throws Exception{
19.         Assert.assertEquals(mockedValue, ClassForMockSelfSingleton.getInstance().getValue(
    ));
20.     }
21.     @Test
22.     public void checkConstructorMockThoughContainer() throws Exception{
23.         Assert.assertEquals(mockedValue, ContainerForSelfSingleton.getValue());
24.     }
25. }
```

Java. Краткие выводы

Добавляем эти зависимости в pom.xml, и…

Ваши тесты могут мокать :

- Статические методы
- И конструкторы классов через оператор new

```
<!-- Mock static methods library -->
    <dependency>
        <groupId>org.powermock</groupId>
        <artifactId>powermock-core</artifactId>
        <version>${powermock.version}</version>
        <scope>test</scope>
    </dependency>

    <!-- Mock static methods library for junit -->
    <dependency>
        <groupId>org.powermock</groupId>
        <artifactId>powermock-module-junit4</artifactId>
        <version>1.${junit.version}</version>
        <scope>test</scope>
    </dependency>

    <!-- Mock static methods library for mockito -->
    <dependency>
        <groupId>org.powermock</groupId>
        <artifactId>powermock-api-mockito</artifactId>
        <version>${powermock.version}</version>
        <scope>test</scope>
    </dependency>
```

DotNet.

- Строгий контроль типов и после компиляции.
- Лаконичный синтаксис
- Делегаты
- «Вменяемые» лямбды
- e.t.c.

Ну и вообще… пора вспомнить что я же дотнетчик ☺

DotNet. Синглтон

```
1.  public class Singleton<T> where T: class
2.   {
3.       private static volatile T _instance;
4.       private static object sync = new object();
5.       protected Singleton(){}
6.       public static T Instance
7.       {
8.           get
9.           {
10.              if ( _instance == null )
11.              {
12.                  lock (sync)
13.                  {
14.                      if (_instance == null)
15.                      {
16.                          _instance = Activator.CreateInstance<T>();
17.                      }
18.                  }
19.              }
20.              return _instance;
21.          }
22.      }
23.  }
```

DotNet. Синглтон. Использование

```
1.  public class ClassForMock: Singleton<ClassForMock>
2.      {
3.          public virtual string GetValue()
4.          {
5.              return "ThisIs not mocked class";
6.          }
7.      }
8.
9.  public class Container
10.     {
11.         public static string GetValue()
12.         {
13.             return ClassForMock.Instance.GetValue();
14.         }
15.     }
```

DotNet. Typemock **Isolator/Isolate**

“You can fake :

- ***statics*,**
- *private*,
- ***constructors*,**
- *events*,
- *linq*,
- *ref args*,
- *live*,
- *future*,
- ***static constructors*.”**

Isolator – коммерческое расширение. Есть триальный период. Ставит с собой свой Test runner в студию.
Isolate – framework, который использует этот самый Isolator
Подробнее https://www.typemock.com/isolator

DotNet. Typemock **Isolate**. Конструктор

```
1.  [Test, Isolated]
2.  public void MockConstructor()  // а что у нас с конструктором? вроде работает.
3.  {
4.      var fakeClassForMock =  Isolate.Fake.NextInstance<ClassForMock>();
5.      Isolate.WhenCalled(()=> fakeClassForMock.GetValue()).WillReturn(mockedValue);
6.      ClassForMock.Instance.GetValue().ShouldBeEquivalentTo(mockedValue);
7.  }
```

Работает ???
А вот и нет ☹

```
1.  [Test, Isolated]
2.  public void MockConstructorAndPassedThruContainer()  // Не работает в этой библиотеке.
3.  {
4.      var fakeClassForMock = Isolate.Fake.NextInstance<ClassForMock>();
5.      Isolate.WhenCalled(() => fakeClassForMock.GetValue()).WillReturn(mockedValue);
6.      Container.GetValue().ShouldBeEquivalentTo(mockedValue);
7.  }
```

var fakeClassForMock = Isolate.Fake.Instance<ClassForMock>();
// так тоже не работает.

DotNet. Typemock Isolate. Статика

```
1.  [Test, Isolated]
2.  public void MockStaticInstance()  // таки, мокаем класс и проверяем.
3.  {
4.      var mock = new Mock<ClassForMock>();
5.      mock.Setup(m => m.GetValue()).Returns(mockedValue);
6.      Isolate.WhenCalled(() => ClassForMock.Instance).WillReturn(mock.Object);
7.      ClassForMock.Instance.GetValue().ShouldBeEquivalentTo(mockedValue);
8.  }
```

Работает ???

```
1.  [Test, Isolated]
2.  public void MockStaticInstanceAndPassedThruContainer()  // Точно рабочий способ
3.  {
4.      var mock = new Mock<ClassForMock>();
5.      mock.Setup(m => m.GetValue()).Returns(mockedValue);
6.      Isolate.WhenCalled(() => ClassForMock.Instance).WillReturn(mock.Object);
7.      Container.GetValue().ShouldBeEquivalentTo(mockedValue);
8.  }
```

DotNet. Избавляемся от платности

Smocks
“Smocks is an experimental framework for "static mocking" for .NET 4 and .NET 4.5. It is not a full-featured mocking framework, but rather a supplement to existing frameworks such as moq. These frameworks typically do not support mocking of ***static or non-virtual methods and properties****.* ***Smocks fills the gap****.”*

Лицензия MIT

Сорцы https://github.com/vanderkleij/Smocks

DotNet. Smocks. Как не работает

Попытка замокать статический Instance провалилась

```
1.  [Test]
2.  public void MockStaticInstance()  // Не работает в этой библиотеке.
3.  {
4.      Smock.Run(context =>
5.      {
6.          var mock = new Mock<ClassForMock>();
7.          mock.Setup(m => m.GetValue()).Returns(mockedValue);
8.          context.Setup(() =>
9.              ClassForMock.Instance).Returns(mock.Object);
10.                 ClassForMock.Instance.GetValue().ShouldBeEquivalentTo(mockedValue);
11.             });
12.     }
13. }
```

DotNet. Smocks. Как работает

```
1.  [Test]
2.  public void SmokeMockValue()
3.  {
4.      Smock.Run(context =>
5.      {
6.          context.Setup(() =>
7.              ClassForMock.Instance.GetValue()).Returns(mockedValue);
8.
9.          ClassForMock.Instance.GetValue().ShouldBeEquivalentTo(mockedValue);
10.
11.     });
12. }
13.
14. [Test]
15. public void SmokeMockValueAndPassedThruContainer()
16. {
17.     Smock.Run(context =>
18.     {
19.         context.Setup(() =>
20.             ClassForMock.Instance.GetValue()).Returns(mockedValue);
21.
22.         Container.GetValue().ShouldBeEquivalentTo(mockedValue);
23.
24.     });
25. }
```

DotNet. Что еще ?

Moles – Isolation framework for .NET
https://www.microsoft.com/en-us/research/project/moles-isolation-framework-for-net/
“Moles allows to replace any .NET method with a delegate. Moles supports static or non-virtual methods.”

Обертка статической зависимости, и передача как зависимости этой обертки.

Для самых смелых и умелых – инжекция IL кода.

Let's talk about JavaScript

UI приложения – на ES6 + ReactJS + Flux
Требования: тестовое покрытие …

Пример компонента (ES6):

```
1.  class DataParserHelper {
2.  /*
3.  ...
4.  */
5.  }
6.
7.  export default new DataParserHelper;
```

ES6. Singleton и его использование

```
1.  class Singleton {
2.      getValue(){
3.          return "This value is not mocked";
4.      }
5.  }
6.
7.  export default new Singleton();
```

```
1.  import  Singleton from './Singleton.js';
2.
3.  class Container{
4.      getValue(){
5.          return Singleton.getValue();
6.      }
7.  }
8.
9.  export default Container;
```

ES6. Sinon

```
1.  import sinon from "sinon"; import chai from "chai";
2.  import Singleton from "../src/Singleton.js"; import Container from "../src/Container.js";

3.  const expect = chai.expect;
4.  describe('Demo', ()=> {
5.      const mockedValue = "!!! This value is mocked";  let sandbox;
6.      beforeEach(()=> { sandbox = sinon.sandbox.create();});
7.      afterEach(()=> { sandbox.restore();});
8.
9.     it('mock singlton', ()=>{
10.        const mock = sandbox.mock(Singleton);
11.        mock.expects("getValue").once().returns(mockedValue);
12.        expect(Singleton.getValue()).to.be.equal(mockedValue);
13.     });
14.
15.    it('mock singlton and call it from container', ()=>{
16.        const mock = sandbox.mock(Singleton);
17.        mock.expects("getValue").once().returns(mockedValue);
18.        const container = new Container();
19.        expect(container.getValue()).to.be.equal(mockedValue);
20.    });
21. });
```

ES6. KISS или не KISS? Вот в чем вопрос.

```
1.                          import chai from "chai";
2.  import Singleton from "../src/Singleton.js"; import Container from "../src/Container.js";

3.  const expect = chai.expect;
4.  describe('KISS', ()=> {
5.      const mockedValue = "!!! This value is mocked";
6.
7.      it('mock singlton', ()=> {
8.
9.          Singleton.getValue = ()=> {  return mockedValue;  };
10.
11.         expect(Singleton.getValue()).to.be.equal(mockedValue);
12.     });
13.
14.     it('mock singlton and call it from container', ()=> {
15.
16.         Singleton.getValue = ()=> { return mockedValue; };
17.
18.         const container = new Container();
19.         expect(container.getValue()).to.be.equal(mockedValue);
20.     });
21. });
```

Как дойти до жизни такой ?

Legacy…

Инжекция зависимостей:

Вариант 1: **`IRestClient restClient = new RestClient(request);`**

Вариант 2: **`OurMegaService.GetMegaValue();`** `// Статический разумеется.`

Состояние тестов:

```
@Test
public void thisMegaTest_ShouldWork_WhenPigsFly(){
/*
…
*/
}
```

Несколько слов в защиту синглтона

Отправим старичка на покой

IOC. (Гарантия единственного экземпляра)
Spring, например, создает объекты в своем контексте по умолчанию… Как синглтоны.
Пример: <object id="Container" type="My.Awesome.Container" />

```
Autofac. RegisterType<Container>().SingleInstance();
```

Lazy initialization. (Гарантия инициализации по требованию)

```
Lazy<Worker> worker = new Lazy<Worker>();
…
worker.Value.DoWork();
```

Краткое содержание предыдущих серий

Не было гвоздя — подкова пропала,
Не было подковы — лошадь захромала,
Лошадь захромала — командир убит,
Конница разбита, армия бежит,
Враг вступает в город, пленных не щадя,
Оттого что в кузнице не было гвоздя…

С. Я. Маршак

Уффффф*

* Это «ВСЕ!» по тилимиллитрямски.

А теперь: Вопросики !!!

Место ссылки…

Java

https://github.com/powermock/powermock/wiki

.Net

https://www.typemock.com/isolator

https://github.com/vanderkleij/Smocks

https://www.microsoft.com/en-us/research/project/moles-isolation-framework-for-net/

Если кому интересно копнуть глубже:

https://github.com/vladimir-afanasiev/singleton

vladimir.vladimir.afanasiev@gmail.com