СПб АУ НОЦНТ РАН

Java 08

28.04.2017

JUnit3

- MyClassTest extends Assert
- public void testMyMethod()

JUnit3

- MyClassTest extends Assert
- public void testMyMethod()

- MyClassTest extends TestCase
- @Override void setUp() *(перед тестом)*
- @Override void tearDown() *(после теста)*
- Каждый тест — на отдельном экземпляре класса *MyClassTest*

JUnit4

- Необязательно “extends Assert”
- import static org.junit.Assert.*

JUnit4

- Необязательно “extends Assert”
- import static org.junit.Assert.*
- @Test public void testMyMethod()
- @Before/@After public void beforeTest()

JUnit4. Параметры

```java
@Test(timeout = 100)
public void testMyMethod1() {
  Thread.sleep(50);
}

@Test(expected = IOException.class)
public void testMyMethod2() {
  new FileInputStream("abc.txt");
}

@Ignore
@Test
public void testMyMethod3() {
  throw new NotImplementedExceptoion();
}
```

JUnit4. RunWith

```
@RunWith(NestedRunner.class)
public class OuterTest {
    public class NestedClassTest {
        @Test
        public void testMyMethod() {
            assertTrue(true);
        }
    }
}
```

JUnit4. Category

```
interface FastTests {}

class MyClassTest {
  @Test
  public void testSlow() {}
  @Test
  @Category(FastTests.class)
  public void testFast() {}
}

@RunWith(Categories.class)
@Categories.IncludeCategory(FastTests.class)
@Suite.SuiteClasses( { MyClassTest.class, ...class
public class FastTestsTestSuite {}
```

JUnit4. Parametrized / Theories

```
@RunWith(Parameterized.class)
class MyClassTest {
  private final int x;
  private final String y;

  @Parametrized.Parameters
  public static Iterable<Object[]> data() {
    return Arrays.asList(
      new Object[][] { {1, "1"}, {2, "2"} );
  }

  public MyClassTest(int x, String y) {...}

  @Test
  public void testToString() {
    assertEquals(y, Integer.valueOf(x).toString())
  }
```

JUnit4. Parametrized / Theories

- Подобная идея также реализуется с помощью ‘@Theories‘/‘@DataPoints‘
- Сложно запустить отдельный тест с выбранными параметрами

JUnit4. Rules

```
class MyClassTest {
  @Rule
  public final TemporaryFolder folder =
      new TemporaryFolder();

  @Rule
  public final Timeout timeout =
      new Timeout(1000);

  @Rule
  public final ExpectedException thrown =
      ExpectedException.none();
}
```

JUnit4. Rules

```
public interface TestRule {
  Statement apply(
    Statement base,
    Description description);
}
public abstract class Statement {
  public abstract void evaluate() throws Throwable;
}
// ----- Each test run ------
Statement statement = new Statement() {
  // run test method
};
for (TestRule rule : allRuleFieldsInClass) {
  statement = rule.apply(statement, definition);
}
statement.evaluate();
```

Unit-тесты. Dependency Injection

```
class Container {
  // Component1,2 --- interfaces
  Component1 getComponent1() { ... }
  Component2 getComponent2() { ... }
}
```

Unit-тесты

Основная идея: когда тестируем “ComponentN”, остальные должны быть максимально простыми

Unit-тесты

Основная идея: когда тестируем “ComponentN”, остальные должны быть максимально простыми
Реализация: нормально это сделать очень сложно. Есть *mockito*, *easymock*

Тесты. DSL

```
// Hamcrest
assertThat(
  responseString,
  either(
    containsString("color"))
    .or(containsString("colour")));

// Scala
"abbcc" should fullyMatch
  regex ("a(b*)(c*)"
    withGroups ("bb", "cc"))
```

Задание 1

- Создать реализацию класса “TestRule” — “MemoryLeakLimit”
- метод “limit(long mb)”

Для каждого теста проверяет, что после его исполнения размер области памяти, которую не может освободить GC, увеличился не более чем на ‘mb‘ мегабайт
NB: Создание PR с результатами все еще обязательно
Дедлайн: 04.05.2017 23:59

Задание 2

- Создать реализацию класса “TestRule” — “ThreadExpectedException”
- метод “expect(Class<? extends Throwable> e)”
- метод “registerThread(Thread t);”

Для каждого теста проверяет, что после его исполнения:

- Зарегистрированные потоки завершены
- Если “expect” последний раз был вызван не с null, то все потоки должны завершиться с исключением класса-аргумента
- Поток, в котором запущен тест, может завершиться как угодно: с исключением и без

NB: Создание PR с результатами все еще обязательно
Дедлайн: 04.05.2017 23:59