Лекции

Поиск

Карта квестов Лекции CS50 Android

## Популярные аннотации в JUnit

JSP & Servlets 3 уровень, 6 лекция

ОТКРЫТА

### 7.1 Аннотация @Suite

И еще несколько часто используемых аннотаций. Даже если ты не будешь их использовать, то точно увидишь тесты с ними. Поэтому нужно хотя бы в общих чертах понимать, что там написано.

Тесты можно объединять в группы. Для этого есть специальная аннотация @Suite . Пример:

```
@Suite
@Suite
@SuiteDisplayName("JUnit Platform Suite Demo")
@SelectPackages("com.javarush.test")
@IncludeClassNamePatterns(".*Tests")
class SuiteDemo {
}
```

В данном случае аннотация @Suite работает в паре с другими аннотациями.

- SuiteDisplayName задает имя группы тестов в логе;
- | SelectPackages | задает список пакетов, где искать классы-тесты;
- IncludeClassNamePatterns задает шаблон имен классов-тестов.

Зачем такие сложности? Ну представь, что все тесты проекта выполняются, скажем, 50 часов. И выполнять их после каждого коммита очень накладно. В таком случае полезно создать некоторые отдельные сценарии тестирования и настроить тестирование гораздо гибче.

### 7.2 Аннотация @Order

Еще одна часто встречающаяся аннотация – @TestMethodOrder . Она позволяет задать порядок вызова тестовых методов в тестовом классе. Бывает очень полезна, когда вы знаете, что вызовы методов влияют друг на друга, но при определенном порядке все должно работать, как нужно. Используется довольно часто.

Во-первых, можно задать вызов методов в алфавитном порядке:

```
1  @TestMethodOrder(MethodOrderer.MethodName.class)
2  public class AlphanumericOrderUnitTest {
3
4 }
```

Во-вторых, у каждого метода можно расставить специальную аннотацию с его порядковым номером.

```
1  @TestMethodOrder(OrderAnnotation.class)
2  public class OrderAnnotationUnitTest {
3
4     @Test
5     @Order(1)
6     public void firstTest() {
7     }
8
```

```
9     @Test
10     @Order(2)
11     public void secondTest() {
12     }
13 }
```

Или даже создать отдельный класс, который будет указывать порядок вызова тестов:

```
1
     @TestMethodOrder(CustomOrder.class)
     public class CustomOrderUnitTest {
2
3
4
     }
5
     //сортируем имена методов по алфавитному порядку, но игнорируя регистр
6
7
     public class CustomOrder implements MethodOrderer {
         public void orderMethods(MethodOrdererContext context) {
8
9
             context.getMethodDescriptors().sort(
              (MethodDescriptor m1, MethodDescriptor m2)->
10
                m1.getMethod().getName().compareToIgnoreCase(m2.getMethod().getName()));
11
         }
12
13
     }
```

# 7.3 Аннотация @DisplayName

И наконец, каждому тесту можно задавать его имя. Бывает удобно, если тестов очень много и вы создаете специальные сценарии (подмножества) тестов. Для этого есть специальная аннотация @DisplayName.

Пример:

```
1
     @DisplayName("Понятное имя для теста")
2
     public class DisplayNameCustomTest {
3
4
         @Test
         @DisplayName("Проверка входных данных")
5
         void inputData() {
6
7
         }
8
9
         @DisplayName("Проверка критических ситуаций")
10
         @Test
         void criticalCases() {
11
12
         }
```

Как и в случае с заданием порядка тестов, тут можно создать специальный метод, который будет генерировать имена тестов и тестовых методов. Пример:

```
1
     @DisplayNameGeneration(DisplayNameGeneratorUnitTest.ReplaceCamelCase.class)
2
     class DisplayNameGeneratorUnitTest {
3
4
         @Test
         void camelCaseName() {
5
         }
6
7
8
         static class ReplaceCamelCase extends DisplayNameGenerator.Standard {
9
             @Override
             public String generateDisplayNameForClass(Class testClass) {
10
```

```
return super.generateDisplayNameForClass(testClass).toUpperCase();
}

}

}

}

}

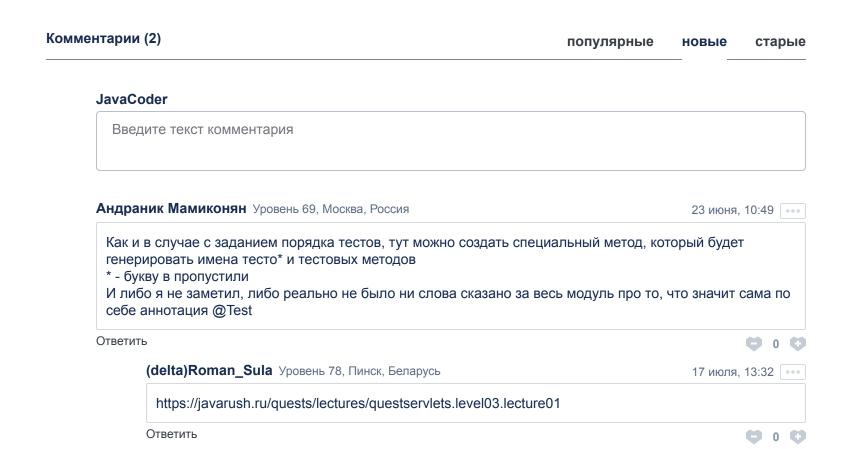
}
```

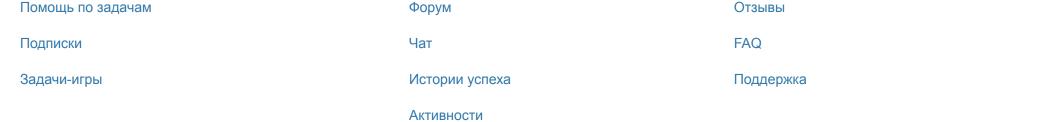
Видите, как все просто:)

< Предыдущая лекция

Следующая лекция >









### RUSH

JavaRush — это интерактивный онлайн-курс по изучению Java-программирования с нуля. Он содержит 1200 практических задач с проверкой решения в один клик, необходимый минимум теории по основам Java и мотивирующие фишки, которые помогут пройти курс до конца: игры, опросы, интересные проекты и статьи об эффективном обучении и карьере Java-девелопера.

#### ПОДПИСЫВАЙТЕСЬ

#### ЯЗЫК ИНТЕРФЕЙСА



#### СКАЧИВАЙТЕ НАШИ ПРИЛОЖЕНИЯ







"Программистами не рождаются" © 2022 JavaRush