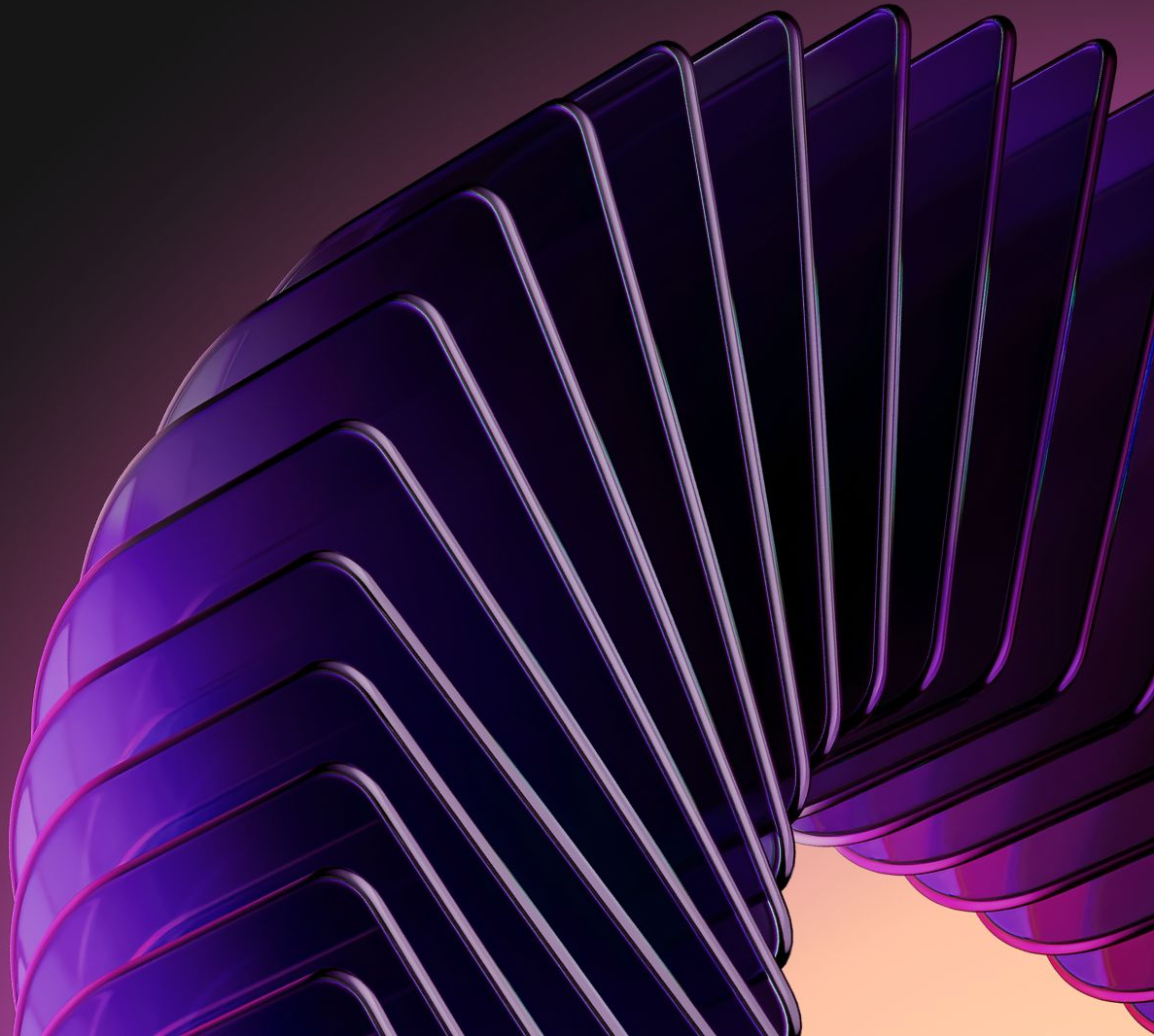


JUnit: прошлое, настоящее и будущее

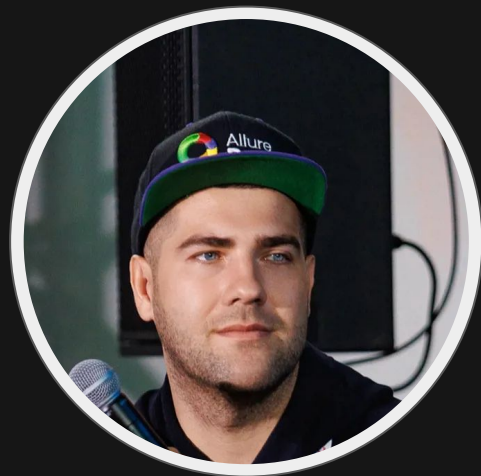
Москва, Heisenbug Spring`26 



Дима Тучс



Head of QA # DODO Engineering



2009 → сегодня



Backend



Project



QA



Manager

Опять про JUnit?!



JUnit, дай пять! Переносим код в JUnit 5 Extensions



Дмитрий Тучс
Dodo Engineering

The art of JUnit extensions



Дмитрий Тучс
Dodo Engineering

JUnit 5 Parallel test execution. Теория и практика



Дмитрий Тучс
Додо Инжиниринг

The art of JUnit extensions 2



Дмитрий Тучс
Dodo Engineering

...НО

JUnit, дай пять! Переносим код в JUnit 5 Extensions

The art of JUnit extensions

The art of JUnit extensions 2

JUnit 5 Parallel test execution. Теория и практика

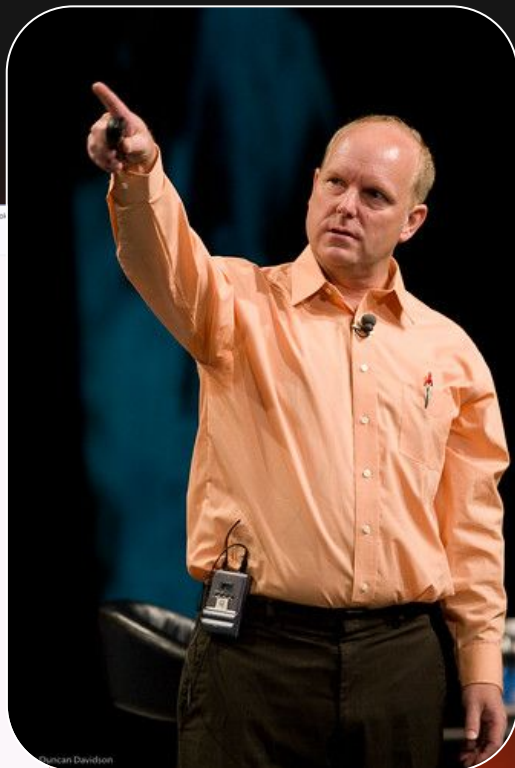
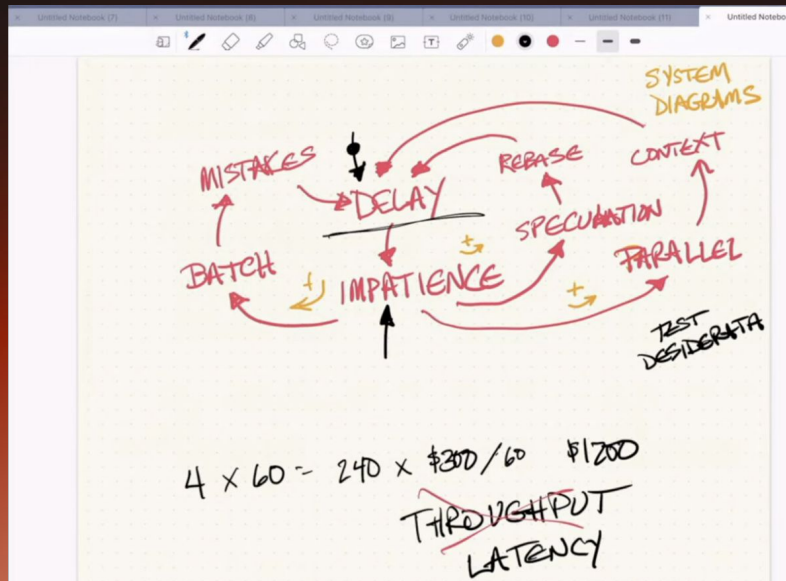


» Как часто вы смотрите на код своего *тестового фреймворка* и не понимаете, почему это написано *так*?

Kent Beck



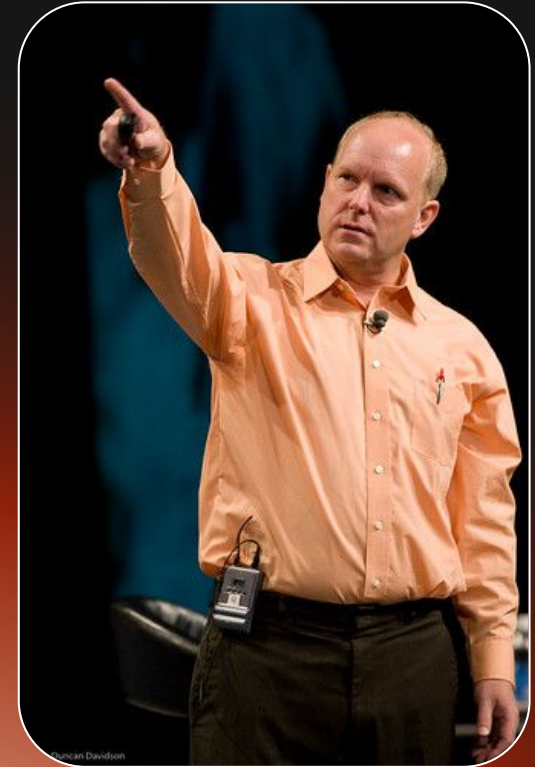
- OOP Patterns
- XP + TDD
- SUnit -> jUnit



Kent Beck — Test Run Latency

https://en.wikipedia.org/wiki/Kent_Beck

Kent Beck





» Каждый тест должен **рассказывать историю**. У него должны быть начало, середина и конец.



» Every test should tell a story. It should have a beginning, a middle, and an end.



Пролог. 28.04.2026 12:33

Пролог



AI Best Practices

AI Tools/Frameworks

AI

AI

AI Tools/Frameworks

AI

AI



AI

AI

Mobile

AI

AI

Tools/Frameworks

AI



Глава 1: Сколько времени нужно на написание тестового фреймворка?



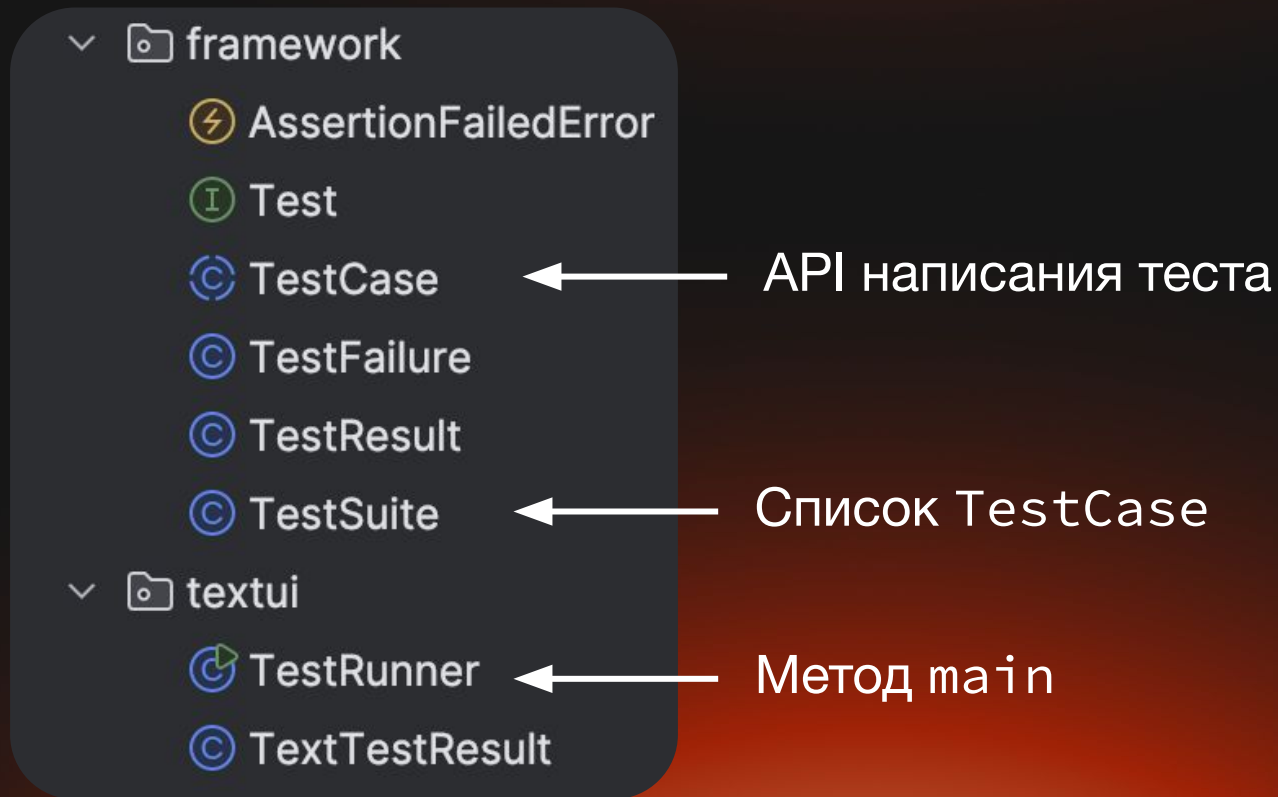


» Я сказал ему: «Ну хорошо, научи меня Java». Мы сели в самолёте, и я спросил: «Какую программу напомним?»»



» I said “Okay, teach me Java.” So we sat down on the airplane, and I said, “Well, what program should we write?”

1997 год: JUnit 1



JUnit 1: API TestCase



```
/**
 * Sets up the fixture, f. e., open a network connection.
 * This method is called before a test is executed.
 */
protected void setUp() {}

/**
 * Tears down the fixture, f. e., close a network connection.
 * This method is called after a test is executed.
 */
protected void tearDown() {}
```

JUnit 1: API TestCase



```
protected void assert(String message, boolean condition) {  
    if (!condition)  
        throw new AssertionError(message);  
}
```

```
protected void assertEquals(Object expected, Object actual) {  
    if (!expected.equals(actual))  
        assert(notEqualsMessage("", expected, actual), false);  
}
```





assertTrue,
как много в этом слове



Владимир Ситников
Netcracker



Assert, как много в этом слове...



Владимир
Ситников
JMeter Committer

Heisenbug
2022 Spring

Владимир Ситников — assertTrue, как много в этом слове...

Владимир Ситников — Assert, как много в этом слове...

JUnit 1: Test



```
public class VectorTest extends TestCase {  
    protected Vector fEmpty = new Vector();  
  
    public VectorTest(String name) {  
        super(name);  
    }  
  
    // @Override  
    protected void setUp() {  
        ...  
    }  
  
    public void testContains() {  
        assert(!fEmpty.contains(new Integer(1)));  
    }  
}
```

// @Override



JUnit 1: Test



```
public class VectorTest extends TestCase {  
    protected Vector fEmpty = new Vector();  
  
    public VectorTest(String name) {  
        super(name);  
    }  
  
    protected void setUp() {  
        ...  
    }  
  
    public void testContains() {  
        assert(!fEmpty.contains(new Integer(1)));  
    }  
}
```

//@Test



JUnit 1: Список тестов для запуска



```
public static Test suite() {  
    TestSuite suite= new TestSuite();  
  
    suite.addTest(new VectorTest("testContains"));  
    suite.addTest(new VectorTest("testClone"));  
    suite.addTest(new VectorTest("testRemoveAll"));  
    return suite;  
}
```



//Главное, не опечататься

JUnit 1: Поиск сюта...



```
public static void main(String argv[]) {  
    String testCase= argv[i];  
    Class testClass= Class.forName(testCase);
```

Method suiteMethod=



```
testClass.getMethod("suite", new Class[0]);
```

```
Test suite= (Test)suiteMethod.invoke(  
                                                    null, new Class[0]);  
    run(suite);  
}
```

JUnit 1: ... и запуск



```
public void run(TestResult result) {  
    result.startTest(this);  
    setUp();  
    try {  
        runTest();  
    }  
    catch (AssertionFailedError e) {  
        result.addFailure(this, e);  
    }  
    catch (Throwable e) {  
        result.addError(this, e);  
    }  
    tearDown();  
    result.endTest(this);  
}
```

A white arrow pointing from the right towards the `runTest();` line in the code block.

JUnit 1: ... с разными результатами



```
public void run(TestResult result) {  
    result.startTest(this);  
    setUp();  
    try {  
        runTest();  
    }  
    catch (AssertionFailedError e) {  
        result.addFailure(this, e);  
    }  
    catch (Throwable e) {  
        result.addError(this, e);  
    }  
    tearDown();  
    result.endTest(this);  
}
```



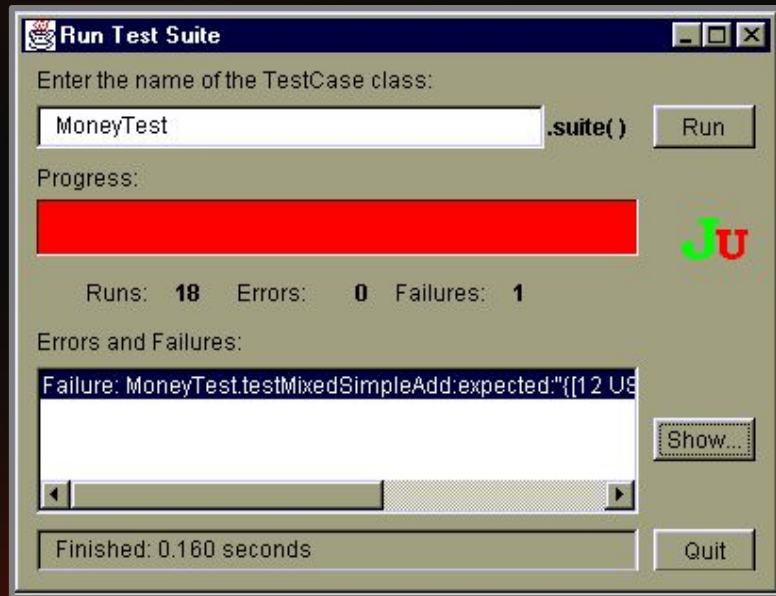
JUnit 1: Отчет



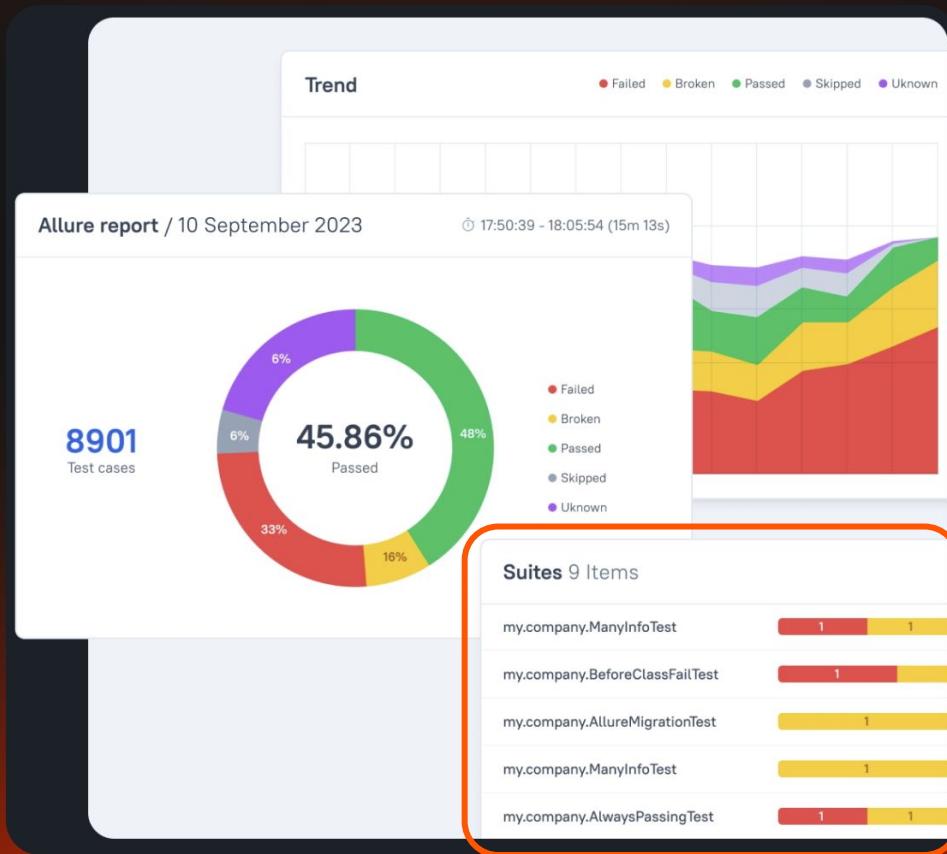
```
public synchronized void print() {  
    printHeader();  
    printErrors();  
    printFailures();  
}
```



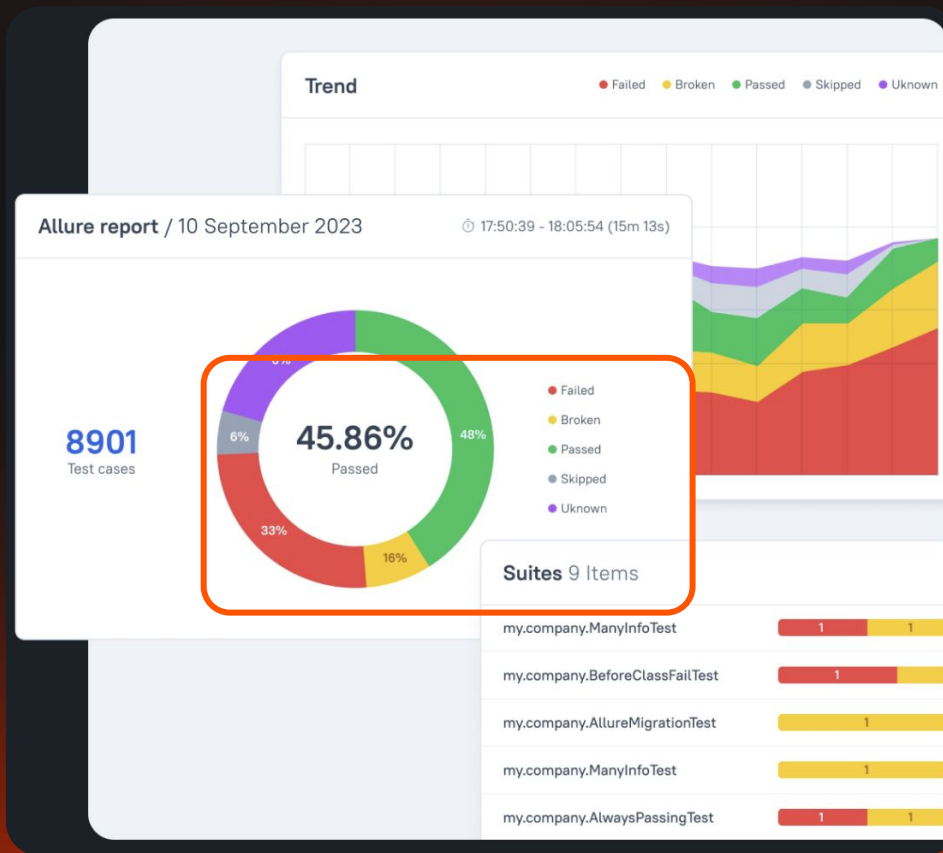
```
public void printFailures() {  
    if (testFailures() != 0) {  
        System.out.println(  
            "There were "+testFailures()+" failures:"  
        );  
    }  
}
```



Отчет сейчас



Отчет сейчас





» Мы дали её Мартину Фаулеру в Атланте, он сразу попробовал и сказал: «Это круто»

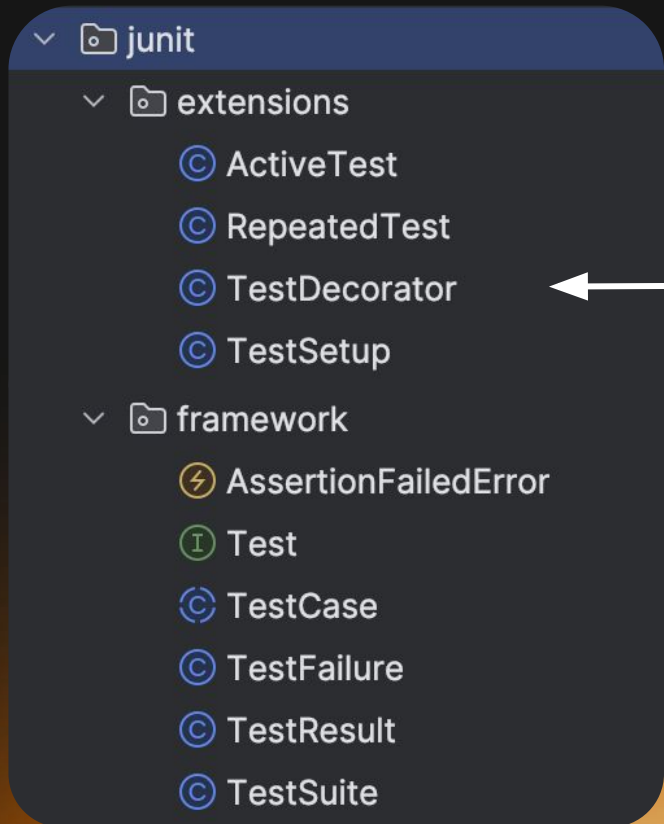


» We gave it to Martin Fowler. He tried it out immediately and said, this is cool

Глава 2, в которой перезапуск flaku понадобился зadолго до изобретения Selenium



JUnit 2: Маленькое обновление...



Не совсем

Почти то же самое?



» Конечно, мы не удержались и добавили несколько функций, но постарались аккуратно вынести их в отдельный пакет расширений (`test.extensions`)



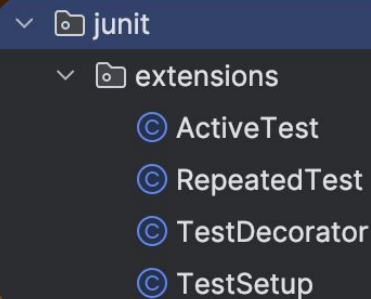
» Sure, we couldn't resist adding some features but we were careful to put them into their own extensions package (`test.extensions`)

JUnit 2: Первая версия extensions API - декораторы



```
/**
 * A Decorator for Tests. Use TestDecorator as the base class
 * for defining new test decorators. Test decorator subclasses
 * can be introduced to add behaviour before or after a test
 * is run.
 *
 */
public class TestDecorator implements Test {

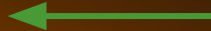
    protected Test fTest;
```



JUnit 2: Первая версия extensions API - декораторы



```
public class RepeatedTest extends TestDecorator {  
    private int fTimesRepeat;  
  
    public RepeatedTest(Test test, int repeat) {  
        super(test);  
        fTimesRepeat= repeat;  
    }  
  
    public void run(TestResult result) {  
        for (int i= 0; i < fTimesRepeat; i++) {  
            super.run(result);  
        }  
    }  
}
```



JUnit 2: Первая версия extensions API - декораторы



```
@ApiLogin
@User(income = 5, outcome = 3)
@Test
void paginationTest() { .. }
```

→ new TestSuite(PaginationTest.class)

JUnit 2: Первая версия extensions API - декораторы

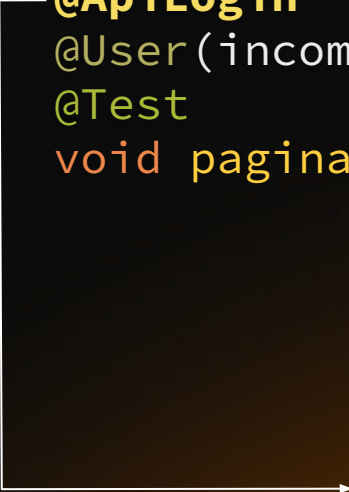


```
@ApiLogin
```

```
@User(income = 5, outcome = 3)
```

```
@Test
```

```
void paginationTest() { .. }
```



```
new LoggedIn(  
    new TestSuite(PaginationTest.class)  
)
```

JUnit 2: Первая версия extensions API - декораторы



```
@ApiLogin
@User(income = 5, outcome = 3)
@Test
void paginationTest() { .. }
```

→

```
new WithIncome(
    new LoggedIn(
        new TestSuite(PaginationTest.class)
    ), 1)
```

JUnit 2: Первая версия extensions API - декораторы

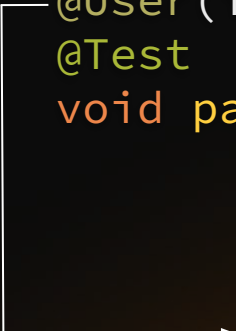


```
@ApiLogin
```

```
@User(income = 5, outcome = 3)
```

```
@Test
```

```
void paginationTest() { .. }
```



```
new WithOutcome(  
    new WithIncome(  
        new LoggedIn(  
            new TestSuite(PaginationTest.class)  
        ), 1), 1)
```

JUnit 2: Первая версия extensions API - декораторы

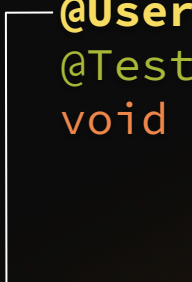


```
@ApiLogin
```

```
@User(income = 5, outcome = 3)
```

```
@Test
```

```
void paginationTest() { .. }
```



```
new WithUser(  
    new WithOutcome(  
        new WithIncome(  
            new LoggedIn(  
                new TestSuite(PaginationTest.class)  
            ), 1), 1));
```

JUnit 2: Первая версия extensions API - декораторы



```
@ApiLogin
@User(income = 5, outcome = 3)
@RepeatedTest(3)
void paginationTest() { .. }
```

→ **new RepeatedTest(**
 new WithUser(
 new WithOutcome(
 new WithIncome(
 new LoggedIn(
 new TestSuite(PaginationTest.class)
), 1), 1)), 3);



JUnit 2: ООП





Новосибирск / 4 апреля 2017

@yegor256

11/13

```
new Farewell(  
    new Attempts(  
        new VerboseDiff(  
            new Diff(  
                new Secret() as secret,  
                new Guess()  
            )  
        ), 5  
    ),  
    secret  
).say();
```



СБЕРБАНК ТЕХНОЛОГИИ





» Тем не менее, проблема не только в ЭТОМ:

) , 1) , 1)) , 3) ;

Глава 3, в которой тестовых фреймворков становится много



The best thing about being me...There are so many me's!

1998: TEST-INFECTED: PROGRAMMERS LOVE WRITING TESTS

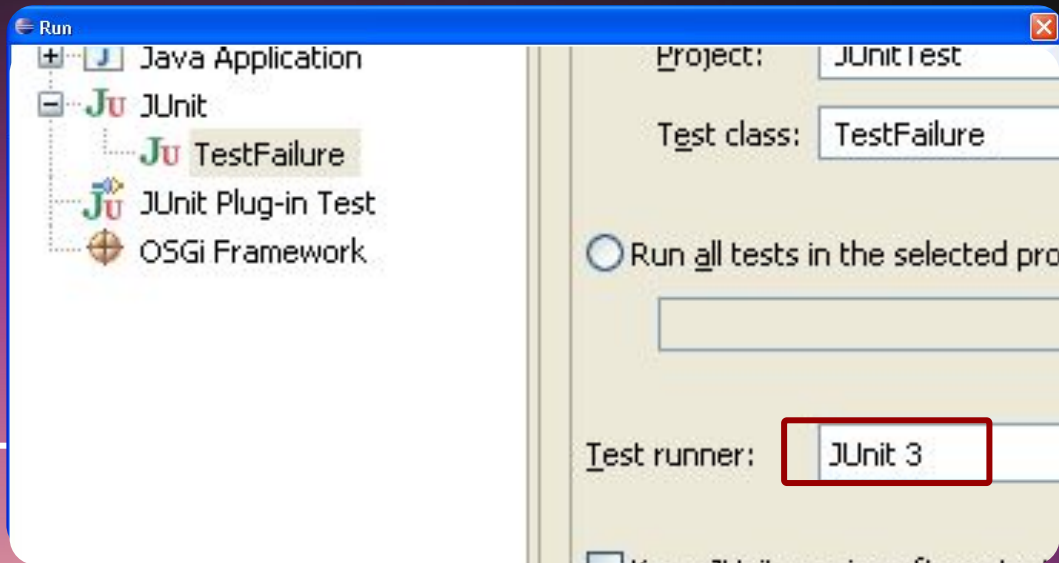
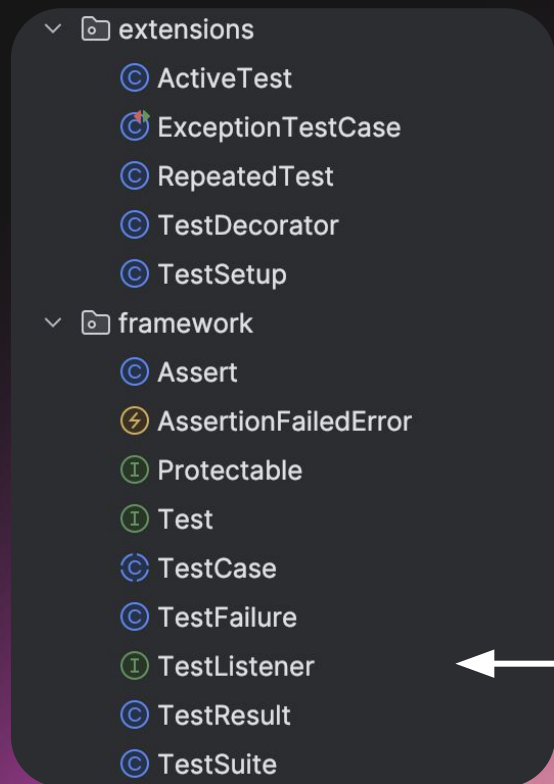


» Тестирование **не тесно**
интегрировано с разработкой.



» **Testing is not** closely integrated with development.

JUnit 3: История успеха





» Чтобы писать тесты, вам нужна последняя версия JUnit (или написать **собственный эквивалент** - это не так уж сложно)



» To write tests you need to get the latest copy JUnit (or write **your own equivalent** - it's not so much work).

JUnit 3: Влияние

PyUnit / unittest

1999

CppUnit

2000

NUnit / CsUnit

2001

PHPUnit

2001

» То, насколько широко
распространилась сама архитектура,
стало для меня большим сюрпризом



» The fact that the architecture spread so far has been a big surprise to me...

[Kent Beck on creating JUnit](#)

JUnit 3: Стандарт на многие годы



- Стандартная модель Runner + Listener, открывшая путь к IDE
- Интеграция с Ant и Maven
- Ant же создал стандарт xml-отчета (JUnit report) на 25 лет вперед



Глава 4, в которой уже хотелось больше Extension-ов



2006: JavaOne



» Тесты - заманчивая цель для расширений



» Test is a tempting target for extensions.

We thought we were just programming on an airplane



- ▼ junit
 - > extensions
 - > framework
 - > runner
 - > textui
- ▼ org.junit
 - > experimental
 - > function
 - > internal
 - > matchers
 - > rules
 - > runner
 - > runners
 - > validator
 - @ After
 - @ AfterClass
 - @ Assert
 - @ Assume
 - ⚡ AssumptionViolatedException
 - @ Before
 - @ BeforeClass
 - @ ClassRule
 - ⚡ ComparisonFailure
 - @ FixMethodOrder
 - @ Ignore
 - 📄 package-info.java
 - @ Rule
 - @ Test
 - ⚡ TestCouldNotBeSkippedException

» Я очень рано увидел
JUnit... увидел
аннотации и подумал:
это классная идея!



» I had seen JUnit very early ... and seen the annotations and
thought, this is a cool idea

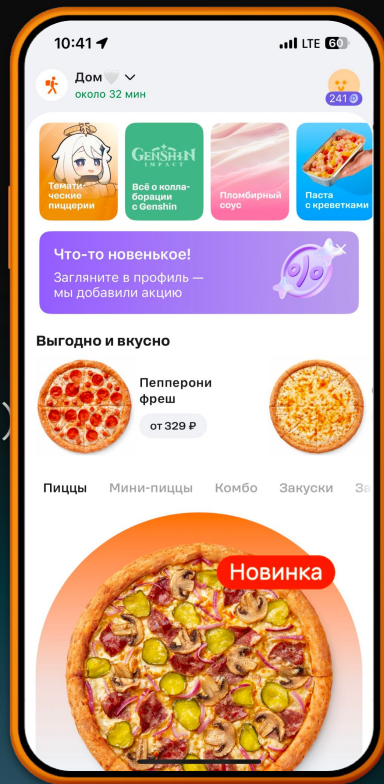
[SE Radio 167: The History of JUnit and the Future of Testing with Kent Beck](#)

JUnit 4: Вполне современно?



```
@RunWith(AndroidJUnit4::class)
class LoyaltyCartTest {

    @Test
    @AllureId("737278")
    @WithUser(cart = [PIZZA_PEPPERONI])
    fun deliveryRewardSum() = test(startOn = Screen.Cart) {
        ShoppingCartScreen {
            verifyLoyaltyRewardSum(expected = "+90")
        }
    }
}
```



JUnit 4: С нами и сегодня



mannodermaus on Oct 1, 2025



JUnit 6 for Android instrumentation tests is probably not going to happen for a long time. We are only slowly arriving at minSdk 26 (the requirement for JUnit 5) being reasonable for applications. JUnit 6 will require minSdk 35, a pretty steep ask in 2025. You'll be better off with 5 if you want to use it for on-device tests.



1. Library & Tool Requirements



- **Jetpack/AndroidX Libraries:** Since June 2025, many libraries require **API Level 23 (Android 6.0)** or higher.
- **Google Play Services:** Version 24.28+ requires **API Level 23** or higher. Older versions (v23.30.99+) previously required API Level 21.
- **Flutter:** As of the 3.22 stable release, Flutter no longer supports Android KitKat (API 19) and requires at least **API Level 21 (Android 5.0)**. Newer Flutter updates (starting June 2025) are moving toward a minimum of **API Level 24**.  Android API Levels +4



JUnit 4: Rules



» Нужно придумать простой способ объяснить: Когда запускается тест, создается цепочка объектов... И с помощью rules ты можешь вставлять объекты в эту цепочку.



» When your test runs this chain of objects is created ... Well, you can insert, using rules, you can insert objects into this chain.

JUnit 4: Rules



```
public interface TestRule {  
    Statement apply(Statement base, Description description);  
}
```



//Тест или сьют, запуск которого будет
передан далее через вызов метода
evaluate();

JUnit 4: Rules



```
public abstract class TestWatcher implements TestRule {
    public Statement apply(Statement base, Description description) {
        return new Statement() {
            @Override public void evaluate() throws Throwable {
                starting(description, errors);
                try {
                    base.evaluate();

                    succeeded(description, errors);
                } catch (AssumptionViolatedException e) {
                    skipped(e, description, errors);
                } catch (Throwable e) {
                    failed(e, description, errors);
                } finally {
                    finished(description, errors);
                }
            }
        };
    }
}
```

JUnit 4: Rules



 <code>ClearAppCacheRule (ru.d</code>	← Очищаем кэш приложения
 <code>DefectTestCaseRule (ru.</code>	← Ищем дефект в TestOps и фэйлим тест
 <code>DodoActivityRule (ru.dod</code>	← Решаем, с какого экрана стартовать
 <code>EnvironmentRule (ru.dod</code>	← На каком окружении тестируем
 <code>FeaturesRule (ru.dodopi</code>	← На каких тоглах
 <code>HealthCheckBackendRule</code>	← Проверяем, жив ли бэкенд перед тестами
 <code>LaunchAppConfigRule (ru</code>	← Кастомный DSL конфигурации запуска
 <code>LocaleRule (ru.dodopizza</code>	← Язык и локация в приложении
 <code>LocalityRule (ru.dodopi</code>	←
 <code>MuteTestCaseRule (ru.dod</code>	← Скипаем тест, если он в карантине
 <code>PermissionsRule (ru.dod</code>	← Доступ к системным API Android
 <code>RequiredAllureIdRule (r</code>	← Просто проверка, есть ли аннотация
 <code>RetrofitMocksRule (ru.d</code>	← Замокать конкретный ответ
 <code>WithUserRule (ru.dodopi</code>	← Создать пользователя...

JUnit 4: Rules



» Я всегда стремлюсь к декларативной форме тестов.



» I always strive for a kind of declarative expression in my tests

JUnit 4: Rules



```
@WithMock(GET_CART_SHARING_PREVIEW, code = 404)
fun openOutdatedSharedCart() = test(startOn = Cart) {
```

```
@WithMock(
    endpoint = POST_ACTUALIZE,
    paths = [Path(FILE, path = "$.delivery", value =
"delivery.json")]
)
class DeliveryIncreaseCartTest
```

JUnit 4: Rules



```
class WithUserRule : BaseRule() {  
  
    private companion object {  
        private const val WAIT_AFTER_RESET_MAPI_CACHE_MILLIS = 2_000L  
    }  
  
    private val user = UserStorage  
    private val settings = SettingsStorage  
}
```

TEST-INFECTED: PROGRAMMERS LOVE WRITING TESTS



» Тестирование ~~не~~ тесно
интегрировано с разработкой.



» Testing is not closely integrated with development.



Глава 5: вы находитесь здесь



2015 год: сбор средств на разработку JUnit 5



JUnit 5: Предпосылки



» Правила созданы для того,
чтобы их нарушать



» Rules... are meant to be broken

JUnit Extensions



» Фреймворк должен быть
пересечением возможностей, а
не объединением всех
ВОЗМОЖНОСТЕЙ



» The framework should be the intersection of features, not the union of features

JUnit 4 - объединения не получилось



```
@ContextConfiguration(locations =  
{"classpath:context-test.xml"})  
@RunWith(SpringJUnit4ClassRunner.class)  
public class SpringJUnit4Test {  
    @Resource(name="testBean")  
    private TestBean bean;  
  
    @Test  
    void beanTest() {...}  
}
```

```
@RunWith(Parameterized::class)  
class CancelOrderTest(  
    private val canBeCancelled: Boolean,  
    ) : BaseTest() {  
  
    companion object {  
        @JvmStatic @Parameters(name = "{0}")  
        fun data() = listOf(true, false)  
    }  
}
```

JUnit 4 - объединения не получилось



```
@ContextConfiguration(locations =  
{"classpath:context-test.xml"})
```

```
@RunWith(SpringJUnit4ClassRunner.class)
```

```
public class SpringJUnit4Test {  
    @Resource(name="testBean")  
    private TestBean bean;
```

```
    @Test  
    void beanTest() {...}  
}
```



OR



```
@RunWith(Parameterized::class)
```

```
class CancelOrderTest(  
    private val canBeCancelled: Boolean,  
    ) : BaseTest() {
```

```
    companion object {  
        @JvmStatic @Parameters(name = "{0}")  
        fun data() = listOf(true, false)  
    }
```

2016 год: первые RC JUnit 5



```
class SimpleTest {  
  
    @Test  
    @DisplayName("1 + 1 = 2")  
    void myFirstTest() {  
        assertEquals(2, 1 + 1);  
    }  
  
    @Test  
    @Disabled("for some reason")  
    void anotherTest() {  
        assertEquals("", 1 + 1);  
    }  
}
```


2016 год: первые RC JUnit 5



```
@TestFactory
@DisplayName("all Fibonacci numbers are odd")
Stream<DynamicTest> testFactory() {
    return IntStream.range(1, 20)
        .map(this::fibonacci)
        .mapToObj(aFibonacci ->
            dynamicTest("Fibonacci = " + aFibonacci, () ->
                isOdd(aFibonacci)));
}
```



Extension Points

- **Conditional Test Execution**

- ContainerExecutionCondition
- TestExecutionCondition

- **General Purpose**

- TestInstancePostProcessor
- TestExecutionExceptionHandler
- ParameterResolver

- **Test Lifecycle Callbacks**

- BeforeAllCallback
- BeforeEachCallback
- BeforeTestExecutionCallback
- AfterTestExecutionCallback
- AfterEachCallback
- AfterAllCallback

2016 год: первые RC JUnit 5



```
public class DisabledOnConference implements TestExecutionCondition {  
    @Override  
    public ConditionEvaluationResult evaluate(TestExtensionContext context){  
        return LocalDate.now().getDayOfWeek() == THURSDAY  
            ? disabled("You're at a conference")  
            : enabled("Get some work done!!!");  
    }  
}
```

2016 год: первые RC JUnit 5



```
public class TmpDirEx implements ParameterResolver, AfterEachCallback {
    @Override
    public boolean supports(ParameterContext parameterCtx,
                           ExtensionContext extensionCtx) {
        return parameterCtx.getParameter().getType()
            .equals(Path.class);
    }

    @Override
    public Object resolve(ParameterContext parameterCtx,
                          ExtensionContext extensionCtx) {
        return extensionCtx.getStore()
            .getOrComputeIfAbsent("root", key -> createTmpDir());
    }
}
```

2017 год: релиз и быстрый успех



ГЕЙЗЕНБАГ Санкт-Петербург 4 июня 2017



```
ParameterizedDemo-encodingReturnsExpectedOutput()
package com.example.usage;

import ...

@ParameterizedTest
public class ParameterizedDemo {

    @Parameter(name = "base64(0) = {1}")
    public static Iterable<Object> parameters() {
        return asList(
            new Object[]{"foo", "2abv"},
            new Object[]{"bar", "pofy"}
        );
    }

    @Parameter(0)
    public String input;

    @Parameter(1)
    public String base64Output;

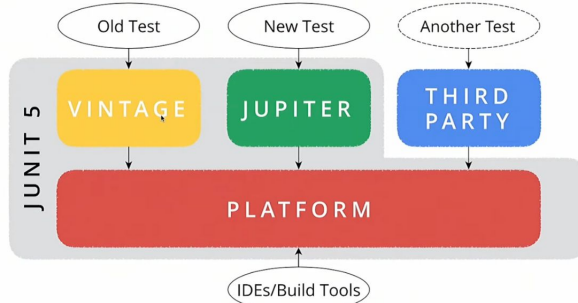
    @Test
    public void encodingReturnsExpectedOutput() {
        Base64.Encoder encoder = Base64.getEncoder();
        assertEquals(base64Output, encoder.encodeToString(input.getBytes()));
    }

    @ParameterizedTest
    void encodingReturnsExpectedOutput() {
    }
}
```

ГЕЙЗЕНБАГ 2017 Питер

Grid Dynamics Тинькофф T-Systems СБЕРБАНК Т

Joker<?> Санкт-Петербург 3-4 ноября



Old Test New Test Another Test

VINTAGE JUPITER THIRD PARTY

PLATFORM

IDEs/Build Tools

5

СБЕРБАНК ТЕХНОЛОГИИ T-Systems ПТК Софт Лабс CROSS OVER SEMRUSH Deutsche Bank GridGain Ignite

JUnit 5 — The New Testing Framework for Java and Platform for the JVM

JUnit 5 extensions: from conditional test execution to test templates

2017 год: в этой истории появляюсь я



» В этом что-то есть



JUnit 5 теперь с нами надолго



- 2017 - первая версия spring-test-junit5;
- **Mockito** - MockitoExtension;
- **Testcontainers** - testcontainers-junit-jupiter;
- **WireMock** - WireMockExtension;
- **Playwright** - @UsePlaywright;
- **Selenide**, **Allure**, и т.д.

JUnit 5: open-test-reporting



`junit.platform.reporting.open.xml.enabled=true`

The screenshot displays the JUnit 5 open-test-reporting interface. At the top, a red status bar indicates "20 tests/containers, 1 failed". The left sidebar shows a tree view of the test suite, with the failed test "sampleHtmlReportIsRenderedCorrectly(Page, TestReporter)" highlighted. The main panel shows the details of this failed test, including its name, sources, JUnit metadata, and the result. The result is an `org.opentest4j.AssertionFailedError` with a message: "Locator expected to contain text: FAILED Received: SUCCESSFUL". The call log shows the test execution steps, including the assertion failure.

20 tests/containers, 1 failed

JUnit Jupiter > DefaultHtmlReportWriterTests >

sampleHtmlReportIsRenderedCorrectly(Page, TestReporter)

FAILED 5 s 396 ms

Sources

Method

className	org.opentest4j.reporting.tooling.core.htmlreport.DefaultHtmlReportWriterTests
methodName	sampleHtmlReportIsRenderedCorrectly
methodParameterTypes	com.microsoft.playwright.Page, org.junit.jupiter.api.TestReporter

JUnit metadata

Type	TEST
Unique ID	[engine:junit-jupiter]/[class:org.opentest4j.reporting.tooling.core.htmlreport.DefaultHtmlReportWriterTests]/[method:sampleHtmlReportIsRenderedCorrectly(com.microsoft.playwright.Page, org.junit.jupiter.api.TestReporter)]
Legacy reporting name	sampleHtmlReportIsRenderedCorrectly(Page, TestReporter)

Result

org.opentest4j.AssertionFailedError

```
org.opentest4j.AssertionFailedError: Locator expected to contain text: FAILED
Received: SUCCESSFUL

Call log:
- Locator.expect with timeout 5000ms
- waiting for getByRole(AriaRole.STATUS)
  9 x locator resolved to <span role="status" aria-label="SUCCESSFUL" class="ml-1 tracking-wide text-sm text-
    - unexpected value "SUCCESSFUL"
at com.microsoft.playwright.impl.AssertionsBase.expectImpl(AssertionsBase.java:74)
at com.microsoft.playwright.impl.AssertionsBase.expectImpl(AssertionsBase.java:51)
at com.microsoft.playwright.impl.AssertionsBase.expectImpl(AssertionsBase.java:42)
at com.microsoft.playwright.impl.LocatorAssertionsImpl.containsText(LocatorAssertionsImpl.java:47)
at com.microsoft.playwright.assertions.LocatorAssertions.containsText(LocatorAssertions.java:853)
at org.opentest4j.reporting.tooling.core.htmlreport.DefaultHtmlReportWriterTests.sampleHtmlReportIsRendered
at java.base/java.lang.reflect.Method.invoke(Method.java:580)
at java.base/java.util.Arraylist.forEach(Arraylist.java:1596)
```

JUnit 5 сегодня: Discovery issues



```
fail("Method \""+fName+"\" should be public");  
warning("No tests found in "+theClass.getName())
```



JUnit 5 сегодня: Discovery issues



Run junit-demo-2025 [test] x

Console Timeline

Test Results 2 ms 1 test failed 1 test total, 2 ms

JUnit Jupiter 2 ms

initializationError 2 ms

TestEngine with ID 'junit-jupiter' encountered 4 critical issues during test discovery:

- (1) [WARNING] @Nested class 'discovery.MalformedJavaTests\$InnerTests' must not be static. It will only be executed if discovered as a standalone test class. You should remove the annotation or make it non-static to resolve this warning.
Source: ClassSource [className = 'discovery.MalformedJavaTests\$InnerTests', filePosition = null]
at discovery.MalformedJavaTests\$InnerTests.<no-method>(SourceFile:0)
- (2) [WARNING] @Test method 'int discovery.MalformedJavaTests.test()' must not return a value. It will not be executed.
Source: MethodSource [className = 'discovery.MalformedJavaTests', methodName = 'test', methodParameterTypes = '']
at discovery.MalformedJavaTests.test(SourceFile:0)
- (3) [WARNING] @Test method 'public final java.lang.Void discovery.MalformedKotlinTests.test1()' must not return a value. It will not be executed.
Source: MethodSource [className = 'discovery.MalformedKotlinTests', methodName = 'test1', methodParameterTypes = '']
at discovery.MalformedKotlinTests.test1(SourceFile:0)
- (4) [ERROR] The test class 'discovery.MalformedKotlinTests' is not a valid Kotlin class. It must be a Kotlin class.

JUnit 5 сегодня: Suite классы



```
@Suite
@SuiteDisplayName("Smoke сценарии")
@SelectClasses({
    LoginTest.class,
    OrderTest.class,
    MenuTest.class
})
class SmokeSuite {

    @SelectPackages("com.example.tests")
    @IncludeTags("smoke")
    @IncludeClassNamePatterns(".*IT|.*SmokeTest")
}
```

JUnit 5 сегодня: Suite методы



```
@Suite
@SelectPackages("com.example.tests")
class RegressSuite {

    @BeforeSuite
    static void beforeSuite() {
    }

    @AfterSuite
    static void afterSuite() {
    }
}
```

JUnit 5 сегодня: Suite методы



```
package guru.qa.rococo.jupiter;

import org.junit.jupiter.api.extension.BeforeAllCallback;
import org.junit.jupiter.api.extension.ExtensionContext;

public interface SuiteExtension extends BeforeAllCallback {

    @Override
    default void beforeAll(ExtensionContext extensionContext) throws Exception {
        extensionContext.getRoot().getExtensionContext()
            .getStore(ExtensionContext.Namespace.GLOBAL).Store
            .getOrComputeIfAbsent(
                this.getClass(),
                k -> {
                    |
                }
            )
    }

    default void beforeAllTests(ExtensionContext context) {

    }

    default void afterAllTests(ExtensionContext context) {

    }
}
```



Дмитрий Тучс

dtuchs

WEISENBUG

ВКонтакте

Т1 | СФ=ра

РоссельхозБанк

X5 Tech

SBER AUTOTECH

JUnit 5 сегодня: ClassTemplate



```
@ClassTemplate
@ExtendWith(UserRepositoryInvocationProvider.class)
class UserRepositoryTest {

    UserRepository repository;

    @Test
    void shouldSaveUser() { }

    @Test
    void shouldUpdateUser() { }
}
```


JUnit 5 сегодня: ParametrizedClass



```
@ParameterizedTest
@ValueSource(strings = { "racecar", "radar" })
void palindromes(String candidate) {}
```

```
@ParameterizedTest
@ValueSource(strings = { "racecar", "radar" })
void reversePalindrome(String candidate) {}
```

JUnit 5 сегодня: ParametrizedClass



```
@ParameterizedClass
@ValueSource(strings = { "racecar", "radar" })
class PalindromeTests {

    @Parameter
    String candidate;

    @Test
    void palindrome() { }

    @Test
    void reversePalindrome() { }
}
```



JUNIT JUPITER API

- programming mode for test authors
- extension model for extension authors



JUnit 6



- Представлен в июне 2025;
- Java 17+, Kotlin 2.2+;
- API не претерпело изменений;
- Единые версии всех артефактов;
- Как и ранее с 5, Spring быстро объявил о переходе на JUnit 6





» Поскольку изменение интерфейсов ломает клиентский код, после публикации следует считать их **неизменяемыми**.



» Since changing interfaces breaks clients you should consider them as immutable once you've published them.

JUnit - Будущее



Новый ThreadPool для параллельного выполнения тестов;

 Introduce support for retrying failed flaky tests

component: Jupiter

status: waiting-for-interest

theme: programming model

type: new feature

Feature #1558 · arcuri82 opened on Aug 20, 2018

 78



Новый отчет станет стандартом (не скоро);

JUnit MCP

JUnit - Будущее





JUnit Jupiter API

JUnit Jupiter is the API for writing tests using JUnit 5.

[Overview](#) [Versions \(114\)](#) [Used By \(22.5k\)](#) [BOMs \(816\)](#)

LICENSE	EPL 2.0
CATEGORIES	Testing
TAGS	quality junit testing api
RANKING	#2 in MvnRepository #1 in Testing
HOMEPAGE	https://junit.org/ Inspect URL
LINKS	  



#7 in MvnRepository

#2 in Testing

TestNG

#44 in MvnRepository

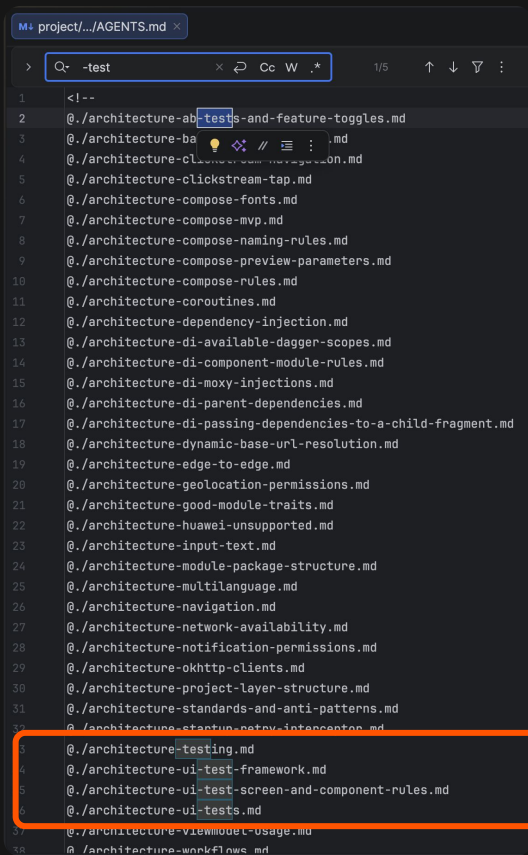
#8 in Testing

Эпилог: любая история может иметь два

финала



Тесты сегодня



```
### Доступ к автоматически созданным мокам
`FixtureRule` неявно создает моки с помощью `MockitoRule`, который также применяется к классу теста.

Для того чтобы получить доступ к этим автоматически созданным мокам (чтобы задать им поведение), необходимо использовать `@Mockito` аннотацию.

```kotlin
@get:Rule
val fixtureRule by fixtureRule()

// Автоматически созданный моки, который будет использован в конструкторе SUT
@Mock
private lateinit var api: SomeApi

// При создании sut фикстура подставит моки api, объявленные выше
private val sut: SomeInteractor by fixtureRule.creator()

@Test
fun test() {
 whenever(api.getData()).thenReturn(...)
 // ...
}
...

Переопределение инъекций (Scoping)
Иногда бывает необходимо подменить конкретную зависимость на реальный объект (например, `CoroutineDispatcher`).

Вы можете вызывать `inject` в двух местах:

1. **При создании правила:** Применяется ко всем тестам в классе.
   ```kotlin
   @get:Rule
   val fixtureRule by fixtureRule {
       inject<CoroutineDispatcher>(testCoroutineRule.dispatcher)
   }
   ...
   ```

2. **Внутри тестового метода:** Применяется только в рамках конкретного теста. Важно отметить, что это работает только для моков, созданных `FixtureRule`.
   ```kotlin
   @Test
   fun test() {
       inject<CoroutineDispatcher>(testCoroutineRule.dispatcher)
       // ...
   }
   ```
```

# Тесты сегодня



**Find in Files** 100+ matches in 18+ files

☐ File mask: \*.java

Q- Rule

In Project Module **Directory** Scope

s/dodo-mobile-android/project/wiki

...

```
Работа с FixtureRule architecture-testing.md 83
`FixtureRule` используется для упрощения создания тестовых данных architecture-testing.md 85
1. **`fixtureRule.create<T>()`: Создает экземпляр класса `T`, запол architecture-testing.md 90
val dto = fixtureRule.create<BonusActionDto>() architecture-testing.md 92
2. **`fixtureRule.build<T>()`: Открывает Builder, который позволяет architecture-testing.md 95
val cartItem = fixtureRule.build<CartItem>() architecture-testing.md 97
`FixtureRule` неявно создает моки с помощью `MockitoRule`, который architecture-testing.md 103
@get:Rule architecture-testing.md 108
val fixtureRule by fixtureRule() architecture-testing.md 109
```



» Написание тестов для меня связано с ответственностью. Это способ сказать: «Вот о чем я думал». Каждый тест - это запись о том, о чем я подумал, пока программировал.



» Testing, to me, is about saying, “Here’s what I was thinking.” Each test is a record of something I thought about while I was programming



## История про то, о чем думал автор теста

```
27 + @SetFeatures(
28 + enabled = [ALWAYS_SHOW_PROMOCODE_IN_CART, ALWAYS_SHOW_PROMOCODE_IN_CART_ENABLED],
29 + disabled = [PROMO_FOR_ROOKIE_WITHOUT_EXPERIMENT, SHARE_CART, UPSSELL_REDESIGN],
30 +)
31 + @SetFirebaseSegments(
32 + segments = [
33 + TestFirebaseSegment(
34 + segmentKey = ALWAYS_SHOW_PROMOCODE_IN_CART_SEGMENT,
35 + segment = VARIANT_2,
36 +)
37 +]
38 +)
39 + class AlwaysShowPromocodeInCartTest : BaseTest() {
40 +
41 + @Test
42 + @AllureId("887887")
43 + @WithUser(needTestActions = true, cart = [PIZZA_HAM_AND_CHEESE_MEDIUM])
44 + @DisplayName("Поле промокода отображается в корзине при примененной акции")
45 + fun actionAndPromocode() = test(startOn = Screen.Profile) {
46 + ProfileScreen {
47 + applyOffer(offer = PIZZA_ARRIVA_AND_COLA_PROMOTION)
48 + clickOnExitButton()
49 + }
50 +
51 + FoodMenuScreen { clickOnCartButton() }
52 + ShoppingCartScreen {
53 + verifyOfferHasBeenApplied(expected = PIZZA_ARRIVA_AND_COLA_PROMOTION)
54 + verifyPromocodeButtonIsDisplayed()
55 + clickOnPromocodeButton()
56 + promocode { verifyIsLoaded() }
```

# Эпилог



История про то, о  
чем думал  
создатель  
Extensions

История про то, о  
чем думала LLM

```
27 + @SetFeatures(
28 + enabled = [ALWAYS_SHOW_PROMOCODE_IN_CART, ALWAYS_SHOW_PROMOCODE_IN_CART_ENABLED],
29 + disabled = [PROMO_FOR_ROOKIE_WITHOUT_EXPERIMENT, SHARE_CART, UPSSELL_REDESIGN],
30 +)
31 + @SetFirebaseSegments(
32 + segments = [
33 + TestFirebaseSegment(
34 + segmentKey = ALWAYS_SHOW_PROMOCODE_IN_CART_SEGMENT,
35 + segment = VARIANT_2,
36 +)
37 +]
38 +)
39 + class AlwaysShowPromocodeInCartTest : BaseTest() {
40 +
41 + @Test
42 + @AllureId("887887")
43 + @WithUser(needTestActions = true, cart = [PIZZA_HAM_AND_CHEESE_MEDIUM])
44 + @DisplayName("Поле промокода отображается в корзине при примененной акции")
45 + fun actionAndPromocode() = test(startOn = Screen.Profile) {
46 + ProfileScreen {
47 + applyOffer(offer = PIZZA_ARRIVA_AND_COLA_PROMOTION)
48 + clickOnExitButton()
49 + }
50 +
51 + FoodMenuScreen { clickOnCartButton() }
52 + ShoppingCartScreen {
53 + verifyOfferHasBeenApplied(expected = PIZZA_ARRIVA_AND_COLA_PROMOTION)
54 + verifyPromocodeButtonIsDisplayed()
55 + clickOnPromocodeButton()
56 + promocode { verifyIsLoaded() }
```



**» Какую историю  
рассказывает ваш  
автотест?**



Спасибо

