
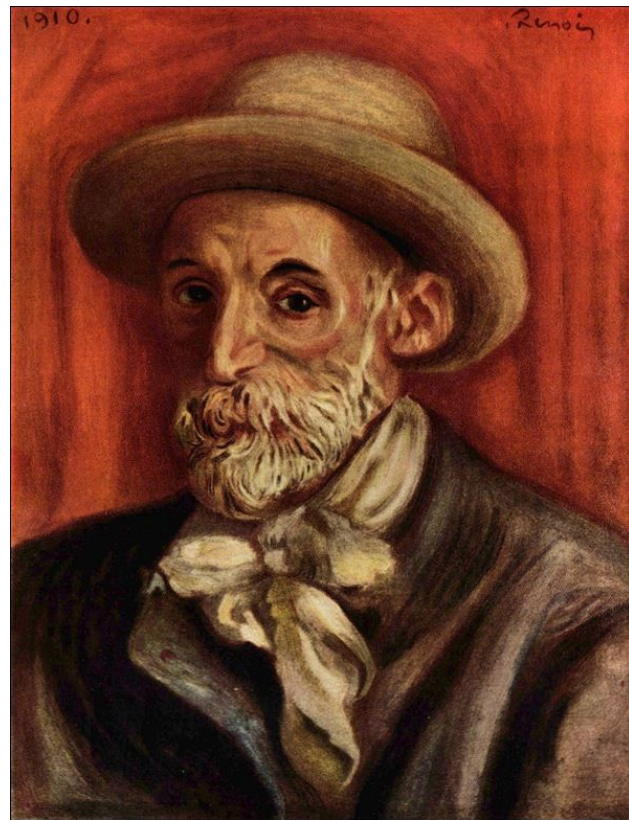


The art of JUnit Extensions

Дмитрий Тучс

 Heisenbug 2023 Autumn online





Head of QA @ Dodo Engineering



dtuchs (и <https://t.me/likeaduck>)



Преподаю в QA.GURU

Опыт в разработке,
аналитике, управлении
проектами более 14 лет



Для кого?



Для всех, кто использует JUnit



Для тех, кому интересно про OAuth 2



... и про экстеншены





Какие такие экстеншены?

Избавиться от
наследования в
тестах

Дать тестам
параметры, не делая
их
`@ParametrizedTest`

Дать гибкость
(ввести понятные
условия, когда
выполнять, когда
нет)

Ускорить тесты и
сделать их проще,
независимее и
повторяемее



Про наследование

```
public abstract class BaseTest {  
  
    String username = UsersQueue.user().username();  
    String password = UsersQueue.user().password();  
    String oauthToken;  
  
    1 override  
    @BeforeEach  
    void apiLogin() {  
        oauthToken = doLogin(username, password);  
    }  
}
```

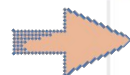


Про наследование

```
public abstract class BaseWebTest extends BaseTest {
```

```
    @BeforeEach
```

```
    void apiLogin() {
```



```
        super.apiLogin(); // BaseTest
```

```
        Selenide.open( relativeOrAbsoluteUrl: "http://127.0.0.1:3000");
```

```
        Selenide.sessionStorage()
```

```
            .setItem("id_token", oauthToken); // from superclass
```

```
        Selenide.refresh();
```

```
    }
```

```
}
```



Про наследование

```
public abstract class BaseRestTest extends BaseTest {
```

```
    AuthClient authClient = new AuthClient();
```

```
    @BeforeEach
```

```
    void apiLogin() {
```

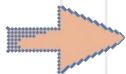


```
        super.apiLogin(); // BaseTest
        authClient.addInterceptor(
            new OAuthTokenInterceptor(oauthToken) // from superclass
        );
    }
}
```



Про наследование

```
public class RestTest extends BaseRestTest {  
  
    @Test  
    void simpleTest() {  
        UserJson userForTest = new UserJson(  
            randomUsername(),  
            randomPassword()  
        );  
        String token = apiLogin( // not @BeforeEach  
            userForTest.username(),  
            userForTest.password()  
        );  
        //  
    }  
}
```





Решение

```
@ApiLogin
public class RestTest {
    @Test @RandomUser
    void randomUserTest() {
    }

    @Test @UserFromQueue
    void userFromQueueTest() {
    }

    @Test @RandomUser(expiredSession = true)
    void incorrectUserTest() {
    }
}
```



Extension – не аналог @BeforeEach/All



Вы знаете **все о потребностях теста**



Результат работы – **часть контекста теста**



Поток управления – в ваших руках, а не в спецификации Java





Extension – не аналог @BeforeEach/All

```
public class ContextHolderExtension
    implements BeforeEachCallback, AfterEachCallback {
    @Override
    public void beforeEach(ExtensionContext extensionContext) {

    }

    @Override
    public void afterEach(ExtensionContext extensionContext) {

    }
```





Интерфейсы – туда

① BeforeAllCallback

@ BeforeAll

* ① LifecycleMethodExecutionExceptionHandler

① BeforeEachCallback

@ BeforeEach

* ① LifecycleMethodExecutionExceptionHandler

① BeforeTestExecutionCallback

@ Test

контекст
тестового
метода



Интерфейсы – обратно

контекст

тестового

метода

@ Test

*

I TestExecutionExceptionHandler

I AfterTestExecutionCallback

@ AfterEach

*

I LifecycleMethodExecutionExceptionHandler

I AfterEachCallback

@ AfterAll

*

I LifecycleMethodExecutionExceptionHandler

I AfterAllCallback



Контекст

```
public interface ExtensionContext
```

```
— final class JupiterEngineExtensionContext  
— final class ClassExtensionContext  
— final class MethodExtensionContext
```



Интерфейсы – и другие

① ExecutionCondition

① ParameterResolver

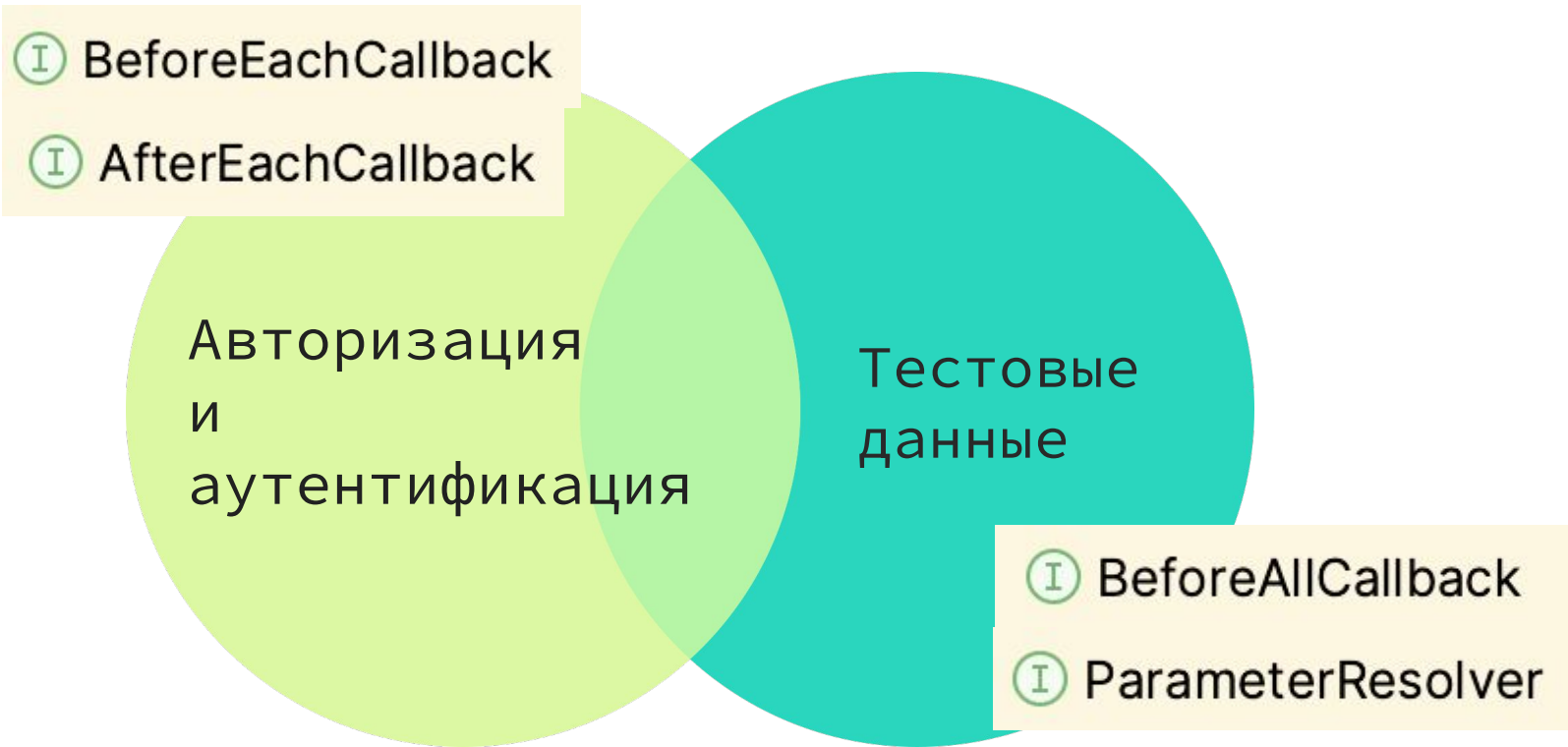
① TestWatcher

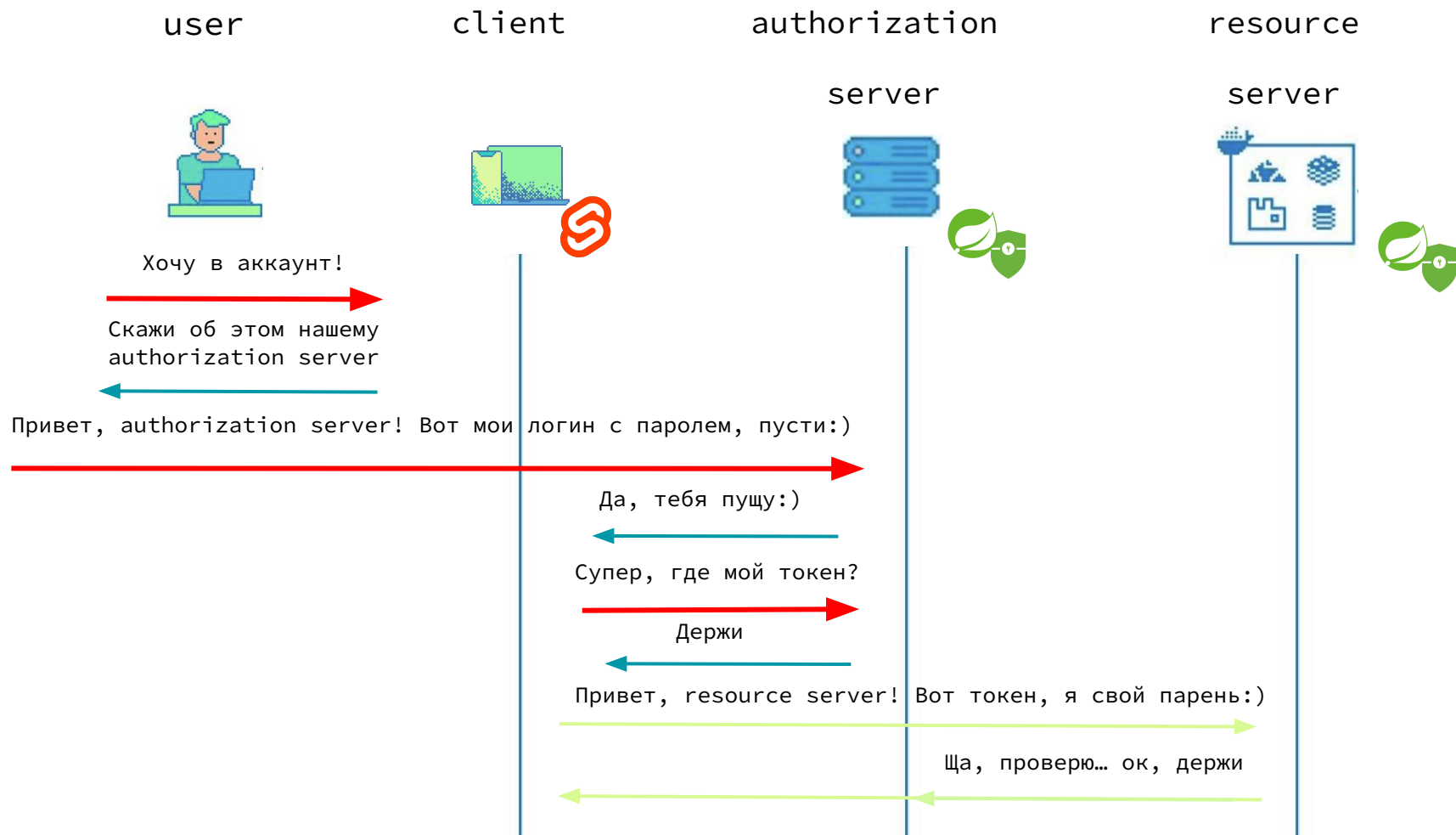
① TestInstancePostProcessor

① TestInstancePreDestroyCallback

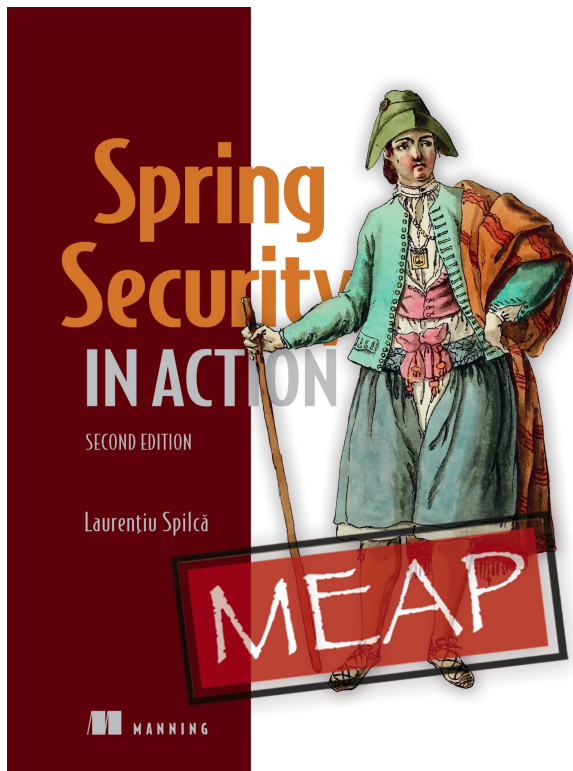


Две стандартные задачи





Почитать и посмотреть



<https://youtu.be/DaUGKnA7aro?si=-BpgNR8WzxsI4BE1>





OAuth 2: Запросы

	Name	Status	Type	Initiator
➡	 authorize?response_type=code&client_id=client&scop...qD8Ge0KI-Jsh89RDGzHJAM&code_chall...	302	docu...	
	 login	200	docu...	:9000/oauth2/authorize?res
➡	 login	302	docu...	Other
	 authorize?response_type=code&client_id=client&scop...Jsh89RDGzHJAM&code_challenge_meth...	302	docu...	<u>login</u>
	 <u>authorized?code=vTeBITsFHMEYqYRUZThTn5HOF61s17mBVg...RGpqT8DQHctXdgjxRRXe1loo7...</u>	200	docu...	:9000/oauth2/authorize?res
➡	 token?client_id=client&redirect_uri=http://127.0.0...ifier=xUsO2iXS1Kxr4x9HLLNj4vxJa5Yf-oASPY...	200	fetch	<u>fetcher.js?v=baf483fe:54</u>



И потестировать



Demo

<https://github.com/dtuchs/heisenbug-2023-autumn>





Прежде чем начнем писать код



ExtensionContext – как использовать для сквозного хранения данных?

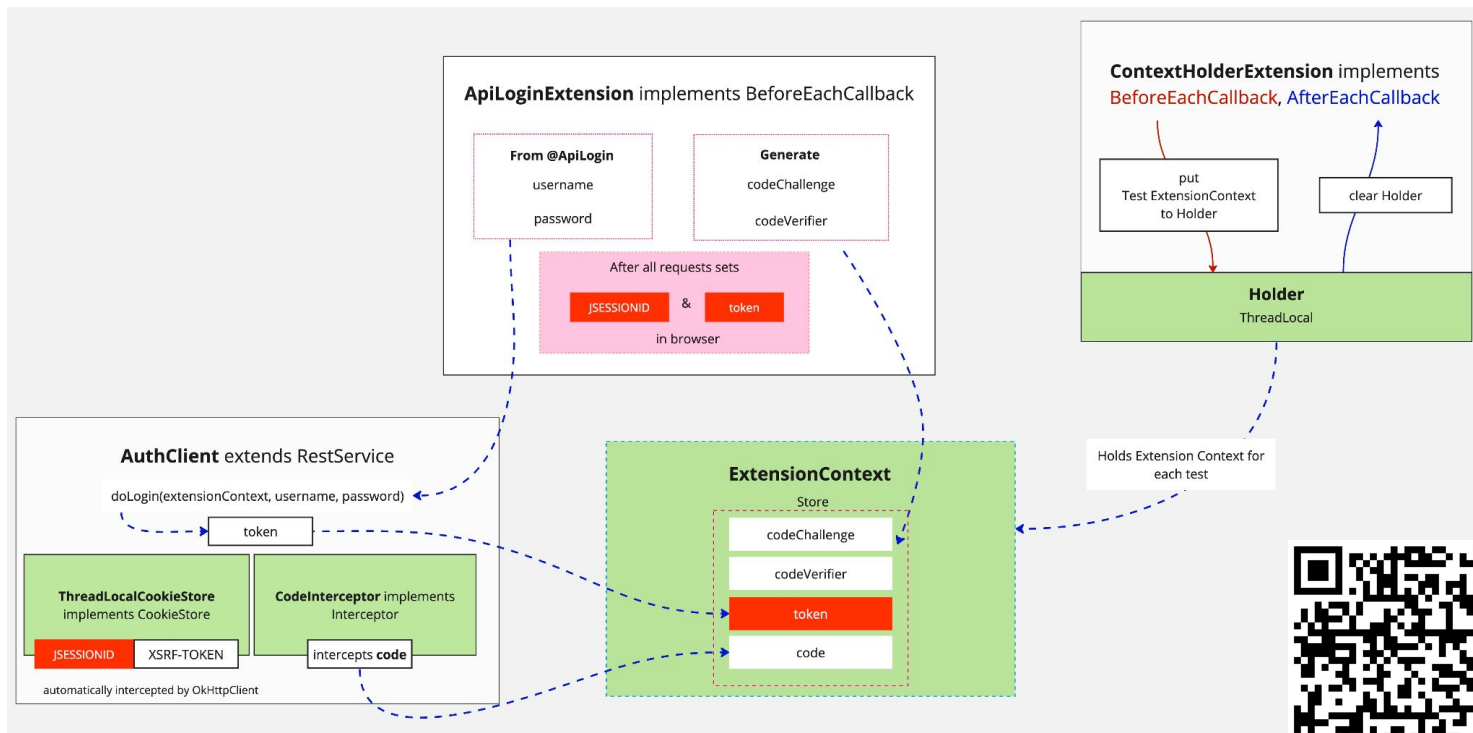


Как дать максимально простой API (getters / setters)

<https://github.com/dtuchs/heisenbug-2023-autumn>



Ничего не понятно (но очень интересно)



<https://github.com/dtuchs/heisenbug-2023-autumn>





Выводы



Вы можете сделать подобное послезавтра на своем проекте



Extensions API не идеален, но лучше него – нет (в Java)



Пишите *НЕ ТОЛЬКО* тесты, это интереснее, чем
arrange -> act -> assert



Мой TG канал <https://t.me/likeaduck>