

kotlenide (0.0.1)

Maksim Kostromin

Version 0.0.1, 2018-09-10 02:22:49 UTC

Table of Contents

1. Introduction	2
2. How to (work in progress)	4
3. Links	7
4. License	8

KOTLENIDE



Idiomatic UI tests

Chapter 1. Introduction

TODO: kotlinide is available in both —maven central and bintray jcenter repositories



Kotlinide itself is not included any selenide library, so we assume you already have one in your project dependencies. Otherwise — just add some together with kotlinide...

gradle

```
dependencies {
    testImplementation 'com.github.daggerok:kotlinide:0.0.1'
    // Kotlinide requires Selenide dependency
    testImplementation 'com.codeborne:selenide:4.14.0'
}
```

maven

```
<dependencies>
  <dependency>
    <groupId>com.github.daggerok</groupId>
    <artifactId>kotlinide</artifactId>
    <version>0.0.1</version>
    <scope>test</scope>
  </dependency>
  <!-- Kotlinide requires Selenide dependency -->
  <dependency>
    <groupId>com.codeborne</groupId>
    <artifactId>selenide</artifactId>
    <version>4.14.0</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

idiomatic kotlin DSL for UI ent-to-end testing, which is using famous good known [selenide](#) under the hood:

```
class AwesomeKotlenideTests {
    @Test fun `selenide using idiomatic kotlin should be awesome`() {
        "https://google.com".open {
            find("body") {
                this contains "G"
                this has "mail"
            }
            close()
        }
    }

    @Test fun `should open given URL in chrome and assert conditions using infix
extension functions`() {
        "https://google.com".openIn("chrome") {
            find("body") {
                this.find(".ololo") shouldNotBe exist
                find(".trololo") shouldBeNot visible
                this shouldHaveText "Gmail"
            }
            close()
        }
    }
}
```

Chapter 2. How to (work in progress)

open url (String)

java

```
@Test
@DisplayName("should open method with String argument")
void test() {
    open("https://google.com")
    // ...
    close()
}
```

open(String relativeOrAbsoluteUrl)

kotlin

```
@Test fun `should open String extension function`() {
    "https://google.com".open {
        // ...
        close()
    }
}

@Test fun `in addition can open given URL in given browser`() {
    "https://google.com".open("firefox") {
        // ...
        close()
    }
}

@Test fun `semantically has openIn function also to open given URL in given browser`() {
    "https://google.com".openIn("chrome") {
        // ...
        close()
    }
}
```

API:

- `String.open { ... }`
- `String.open("browserName") { ... }`
- `String.open(browser = "...") { ... }`
- `String.openIn("browserName") { ... }`
- `String.openIn(browser = "...") { ... }`

kotlin

```
@Test fun `should open Pair extension function with free first key name contract`()
{
    ("url" to "https://google.com").open {
        // ...
        close()
    }
}
```

Pair.open { ... }

open absolute url (URL)

java

```
@Test
@DisplayName("should open method with URL argument")
void test() {
    open(new URL("https://ya.ru"))
    // ...
    close()
}
```

open(URL absoluteUrl)

kotlin

```
@Test fun `should open URL extension function`() {
    URL("https://ya.ru").open {
        // ...
        close()
    }
}
```

URL.open { ... }

open by ...

java

```
@Test
@DisplayName("should open using string arguments by some contract")
void test() {
    open("https://blog.example.com?q=java", "example.com", "login", "password")
    // ...
    close()
}
```

open(String relativeOrAbsoluteUrl, String domain, String login, String password)

```
@Test fun `should open Map extension function minimal contract`() {
    mapOf("relativeOrAbsoluteUrl" to "https://google.com").open {
        // ...
        close()
    }
}

// or:

@Test fun `should open Map extension function full contract`() {
    mapOf(
        "relativeOrAbsoluteUrl" to "https://blog.example.com?q=java",
        "domain" to "example.com",
        "login" to "login",
        "password" to "password"
    ).open {
        // ...
        close()
    }
}
```

Map<String, String>.open { ... }

Chapter 3. Links

- [Kotlin](#)
- [Kotlin DSL guide](#)
- [Kotlin infix functions](#)
- [Selenide](#)
- [Awesome free online images editor LunaPic](#)
- [GitHub repo](#)
- [GitHub pages](#)

Chapter 4. License

MIT