kotlenide (0.0.2)

Maksim Kostromin

Version 0.0.2, 2018-09-18 01:47:23 UTC

Table of Contents

KOTLENIDE
Idiomatic UI tests

Chapter 1. Introduction

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

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

[Maven] | *https://www.bintray.com/docs/images/bintray_badge_color.png*

gradle

```
dependencies {
  testImplementation 'com.github.daggerok:kotlenide:0.0.2'
  // Kotlenide also requires Selenide dependency:
  testImplementation 'com.codeborne:selenide:4.14.1'
}
```

[Maven] | *http://maven.apache.org/images/maven-logo-black-on-white.png*

maven

```
<project>
  <dependencies>
    <dependency>
      <groupId>com.github.daggerok</groupId>
      <artifactId>kotlenide</artifactId>
      <version>0.0.2</version>
      <scope>test</scope>
    </dependency>
    <!-- Kotlenide also requires Selenide dependency: -->
    <dependency>
      <groupId>com.codeborne</groupId>
      <artifactId>selenide</artifactId>
      <version>4.14.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

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

```
class `Awesome Kotlenide Tests` {
  @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 using infix extension
functions`() {
    "https://google.com".open("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
  @DispayName("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
  @DispayName("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
  @DispayName("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)` *kotlin*

```kotlin
@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