|  |  |
| --- | --- |
| Welcome to the new flow of Automation for everyone!  Presentation on how to create, edit, share, and execute automation tests. |  |

Typically, automating the testing process involves using a Java command with Selenium, which can be quite complex. Our project aims to simplify the usage of the Java with Selenium combination and make test automation accessible to users with basic knowledge of web application testing

|  |  |
| --- | --- |
|  |  |

# 

# Simple and clear code

Based on this flow, it is possible to create a full-fledged application with a graphical interface.

Previously, in order to organize test automation, it was necessary to create several classes:

1. A class for setup (driver initialization, browser support, window size configuration, etc.).
2. A class for shared variables.
3. A class for utility methods.
4. A class specifically for writing tests.

Depending on the complexity of the project, the number of classes could increase.

At the moment, we have created a kind of framework based on Java and Selenium. It already has all the initial setup, classes, and methods inside. The operator only needs to write step-by-step instructions for executing the test. The only professional skill required for writing these tests is the ability to write XPath selectors. However, a comprehensive course with practice for an unprepared operator would take only about 1.5 hours.

This framework will be particularly suitable for checking Pixel perfect, and in the future, when new commands are added, it should not become inconvenient or cumbersome.

**Application operation principle**

A picture containing text, font, line, screenshot

Description automatically generated

The main goal of the program is to compare a recently taken screenshot with a previously verified screenshot during manual testing.

As a result, any discrepancies are outputted to the "difference" file, allowing the operator to visually identify the differences.

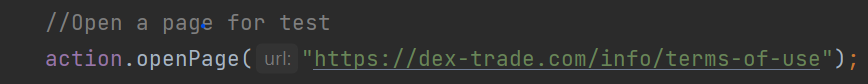
**Writing Instructions**

To write instructions created using methods in auxiliary classes, user-friendly commands in a conversational language are used.

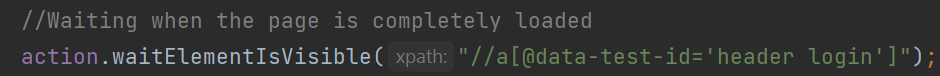
At the beginning of writing a test or a test set, it is necessary to set the platform and the browser name.



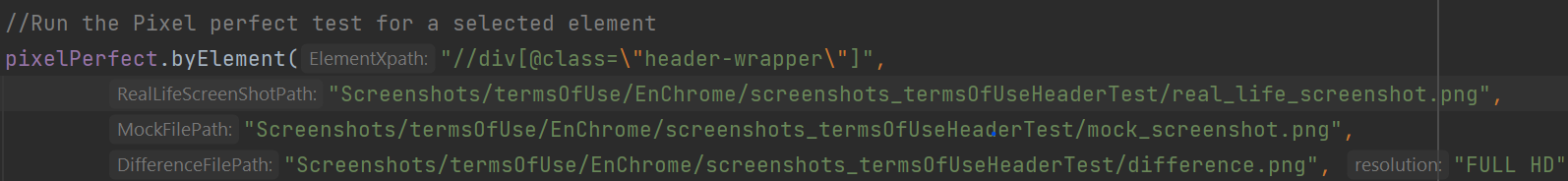
**The next step is to write the commands/instructions directly.**



The command to open the page where the testing will take place.



The command used to ensure that the page has fully loaded, specifically waiting for the last element on the page to be loaded.

**

And the last command is used to execute the method for comparing the similarity of screenshots. In this method, data about the screen size (screen resolution) at which the screenshot should be taken is specified, and paths to the screenshots are provided.

**Running tests and automation of test execution.**

After the tests are written and debugged, the operator adds them to a remote repository, which allows integrating these tests into an automated execution system using a resource such as Jenkins. Jenkins provides the ability to use the written tests, modify the execution, and run tests in different environments at specified times.

Different environments in terms of WPT operation include specific stages (Stage 1, Stage 2, Stage 3, etc.).

**The next steps**.

In the "Automation for Everyone" project, the following steps are clearly outlined:

1. Development and implementation of functionality related to stage changes (there are several approaches). This functionality should be made as clear and user-friendly as possible for the operator. The operator can be either a tester or a developer who needs to verify the changes made in the code without involving testers. This will reduce the time required for testing the beta stage.
2. Refinement of tests for testing the mobile versions of websites.
3. The next step will be to implement testing for pages where data changes rapidly. An example of such a page is a trading platform (https://dex-trade.com/spot/trading/BTCUSDT). It will be necessary to create the ability to substitute highly dynamic data with static data.

A screenshot of a computer

Description automatically generated

It is understood that during the usage of our framework, there may be moments associated with the inconvenience of working with functionality and even possible failures and inaccuracies in code execution. We will strive to address all these issues at the earliest stages. As they say, "You won't know until you try."

[](https://go.microsoft.com/fwlink/?linkid=854192)