|  |  |
| --- | --- |
| Ласкаво просимо до нового підходу у Автоматичному тестуванні для всіх!  Презентація про те як створювати, редагувати, поширювати та виконувати автоматичне тестування. |  |

Зазвичай, автоматизація процесу тестування передбачає використання команд на Java з Selenium, що може бути досить складним. Наш проект має на меті спростити використання поєднання Java з Selenium та зробити автоматизацію тестування доступною для користувачів з базовими знаннями тестування веб-додатків.

# 

# Простий та зрозумілий код.

На базі цього флоу у подальшому можливо створити повноцінний додаток/застосунок з Графічним інтерфейсом.

Раніше для того щоб організувати написання та запуск автотесту, необхідно було створити декілька класів:

1 Клас для підтримки запуску (ініціювання драйверів підтримка браузера, налаштування розмірів вікна та інше)

2 Клас для загальних змінних

3 Клас для робочих методів

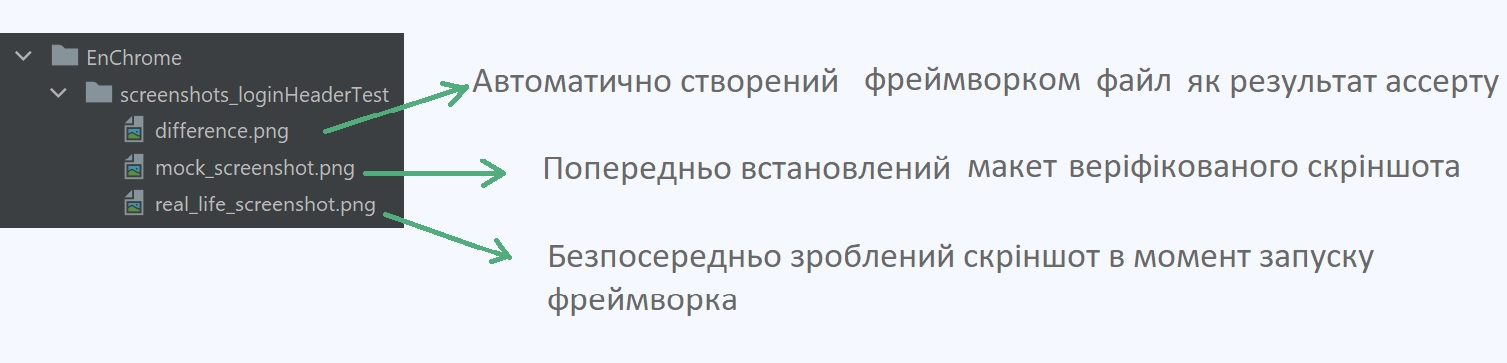
4 Та клас безпосередньо для написання тестів

В залежності від складності проекту кількість класів може збільшуватись

На даний момент ми створили свого роду Фреймворк на базі Java and Selenium. Який вже має в середині всі початкові налаштування, класи, методи. І оператору потрібно тільки написати покроковий запит/інструкцію для виконання тесту. Єдиними професійними навичками в створенні цих тестів будуть навички написання Xpath селекторів. Але повноцінний курс з практикою у непідготовленого оператора займе всього біля 1,5 години.

Саме для перевірки Pixel perfect цей фреймворк буде максимально підходящій. Та у подальшому при доповнені новими командами не повинен стати незручним та громіздким.

Принцип роботи застосунку



Основною метою програми є порівняння тільки що зробленого скріншоту з зразком вже верефікованого раніше скріншота при мануальному тестуванні.

В результаті чого не співпадіння виводиться у файл “difference”, що дає змогу оператору побачити візуальну різницю.

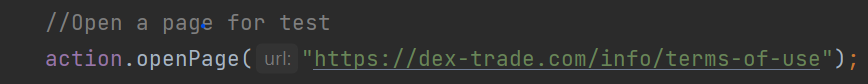
**Написання інструкцій**

Для написання інструкцій створених за допомогою методів в допоміжних класах використовуються зрозуміли команди на звичній для людини мові, тобто розмовній.

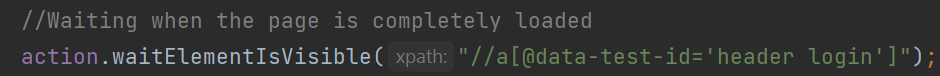
На початку написання тесту чи сету тестів, необхідно встановити платформу та назву браузера.



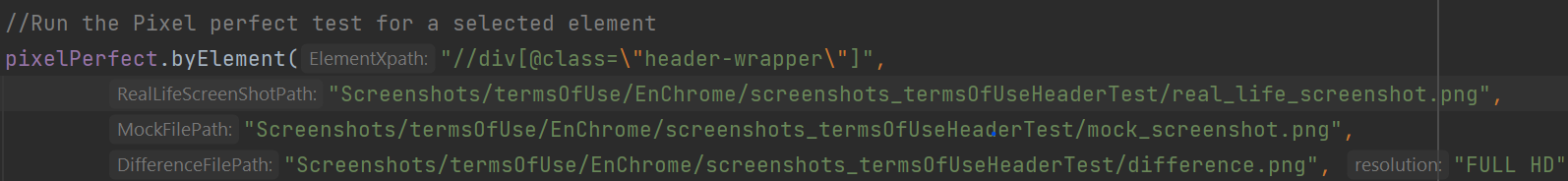
Наступним кроком є безпосередньо написання команд/інструкцій.



*Команда на відкриття сторінки на якій буде проходити тестування*



*Команда за допомогою якої переконуємось що сторінка* повністю завантажилась*, а саме очікуємо поки останній елемент сторінки завантажений.*

**

*Та остання команда за допомогою якої запускається* метод перевірки схожості скріншотів.

*В цьому методі вносяться данні про* розмір екрану *(розширення екрану) при якому необхідно зробити скріншот. Та прописуються шляхи до скріншотів.*

**Запуск тестів та автоматизація запуску.**

Після того як тести написані та налагоджені, оператор додає їх до віддаленого репозиторію що дасть змогу доєднати ці тести до системи автоматичного запуску, використовуючи такий ресурс як Jenkins. Який надає можливість використовувати написані тести, модифікувати запуск, тобто запускати тести в різних середовищах та у вказаний час.

Різні середовищами в розрізі роботи **WPT** є безпосередньо стейджі (St1, St2, St3… та інші).

**Наступні кроки.**

На цей день в проекті «Автоматизація для кожного» вже чітко вимальовуються слідуючи кроки, а саме:

Розробка та втілення функціоналу що до зміни стейджів (є декілька підходів). Цей функціонал необхідно зробити максимально зрозумілий та зручним у користуванні для оператора. Оператором може бути як тестувальник так і девелопер, якому потрібно перевірити створені зміни у коді, та не долучати до цього тестувальників. Що в результаті скоротить час при тестуванні бета стейджа.

Наступним кроком буде втілення тестування сторінок де данні змінюються дуже швидко. Як приклад це сторінка біржової торгівлі (<https://dex-trade.com/spot/trading/BTCUSDT>). Необхідно буде створити можливість підміни високо динамічних даних на статичні.

Зрозуміло що в процесі використання Нашого фреймворку можуть відкриватися моменти пов‘язані з недостатньою зручністю при роботі з функціоналом та навіть можливий збій та не точне відпрацювання коду. Всі ці несправності ми будемо намагатися виправляти на самих ранніх стадіях. Як то кажуть «Не спробуєш, не зрозумієш».