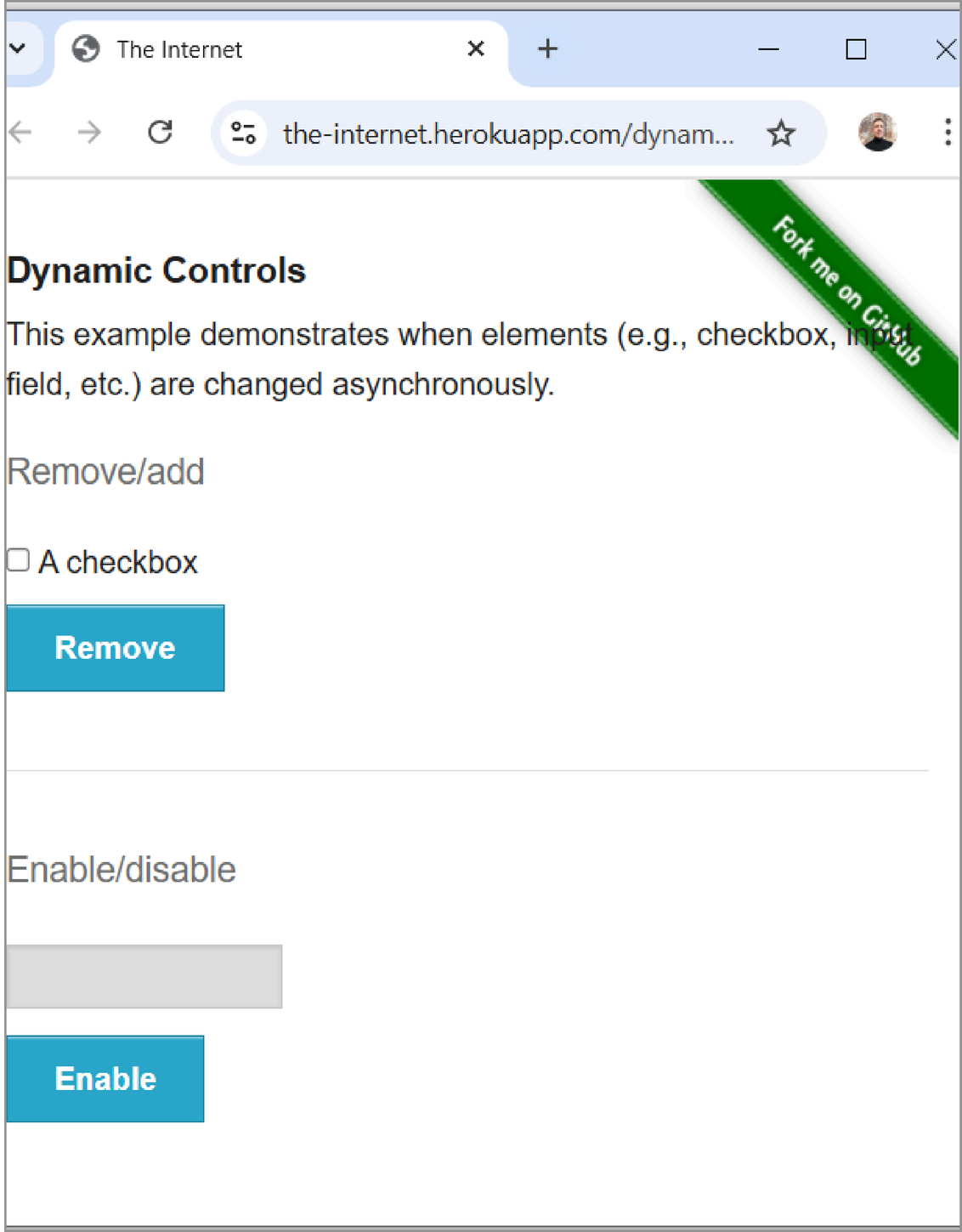
СТРАТЕГИИ ОЖИДАНИЯ
WAITING STRATEGIES

Стратегии ожидания
Waiting Strategies

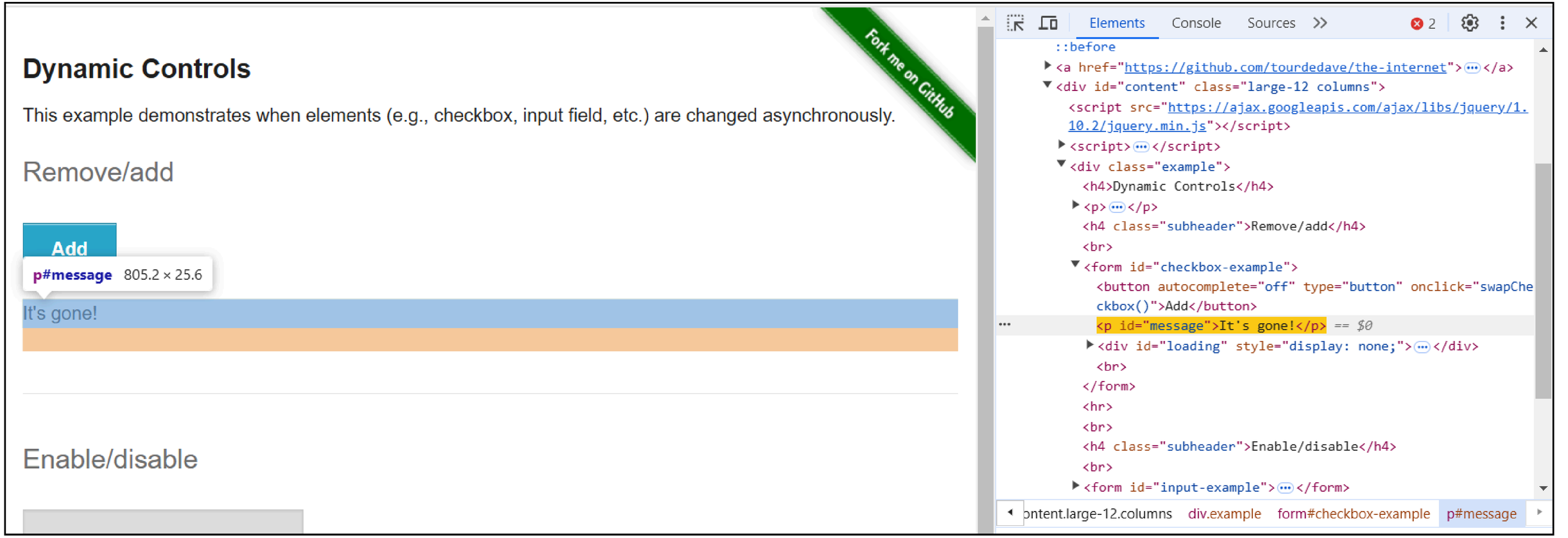
Наиболее распространенной проблемой для автоматизации браузера является обеспечение того, чтобы веб-приложение находилось в требуемом состоянии для выполнения определенной команды Selenium.

Процессы часто оказываются в состоянии гонки (**race condition**). Иногда браузер первым переходит в правильное состояние и все работает так, как задумано. А иногда первым выполняется код Selenium и все работает не так, как задумано.

Таким образом, одна из основных причин нестабильных тестов (**flaky tests**) – это когда веб-элементы с которыми необходимо взаимодействовать отсутствуют на веб-странице, в то время когда код готов выполнить следующую команду Selenium.



Стратегии ожидания
Waiting Strategies



Веб-элемент должен присутствовать и отображаться на странице, чтобы Selenium мог с ним взаимодействовать (https://the-internet.herokuapp.com/dynamic_controls).

Стратегии ожидания
Waiting Strategies

1. Неявное ожидание
   (Implicit waits)

2. Явное ожидание
   (Explicit waits)

НЕЯВНОЕ ОЖИДАНИЕ
IMPLICIT WAITS

Неявное ожидание
Implicit waits

Selenium имеет встроенный способ автоматического ожидания элементов, называемый неявным ожиданием. Неявное значение ожидания можно задать:
- с помощью возможности тайм-аутов в параметрах браузера (the timeouts capability in the browser options),
- с помощью метода драйвера.

Неявное ожидание – это глобальная настройка, которая применяется к каждому вызову поиска веб-элемента для всего сеанса. Значение по умолчанию равно 0, что означает, что если элемент не найден, он немедленно вернет ошибку.

Неявное ожидание
Implicit waits

Если задано неявное ожидание, драйвер будет ждать в течение указанного значения, прежде чем вернет ошибку.

Обратите внимание, что как только элемент будет найден, драйвер вернет ссылку на веб-элемент, и код продолжит выполнение.

Неявное ожидание (Implicit waits) с помощью метода веб-драйвера

Фрагмент кода для настройки неявного ожидания с помощью метода драйвера:

```
WebDriver driver = new ChromeDriver();
driver
    .manage()
    .timeouts()
    .implicitlyWait(Duration.ofSeconds(2));
```

Локаторы веб-элементов:

```
String CHECKBOX_EXAMPLE_BUTTON_LOCATOR =
"//form[@id='checkbox-example']/button";

String MESSAGE_LOCATOR = "//p[@id='message']";
```

ControlsWithImplicitWaitsTest.java × DynamicControlsWithoutWaitsTest.java

```
ort org.openqa.selenium.chrome.ChromeOptions;

ort java.time.Duration;

lic class DynamicControlsWithImplicitWaitsTest {  Sergei Tsarik

 private final String URL = "https://the-internet.herokuapp.com/dynamic_contro
 private WebDriver driver;  12 usages

@Test  Sergei Tsarik
public void testDynamicLoadedMessageWithDriverMethodImplicitWait() {
    driver = new ChromeDriver();
    driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(4));
    driver.manage().window().maximize();

    driver.get(URL);

    String CHECKBOX_EXAMPLE_BUTTON_LOCATOR = "//form[@id='checkbox-example']/
    driver.findElement(By.xpath(CHECKBOX_EXAMPLE_BUTTON_LOCATOR)).click();

    String MESSAGE_LOCATOR = "//p[@id='message']";
    String actual = driver.findElement(By.xpath(MESSAGE_LOCATOR)).getText();

    Assertions.assertEquals( expected: "It's gone!", actual);
}

@Test  Sergei Tsarik
public void testDynamicLoadedMessageWithChromeOptionsImplicitWait() {
    ChromeOptions chromeOptions = new ChromeOptions();
    Duration duration = Duration.ofSeconds(4);
    chromeOptions.setImplicitWaitTimeout(duration);

    driver = new ChromeDriver(chromeOptions);
```

Неявное ожидание (Implicit waits) с помощью возможности тайм-аутов в параметрах браузера (the timeouts capability in the browser options)

Фрагмент кода для настройки неявного ожидания с помощью возможности тайм-аутов в параметрах браузера:

ChromeOptions **chromeOptions** = getDefaultChromeOptions();
Duration duration = Duration.of(5, ChronoUnit.SECONDS);
chromeOptions.setImplicitWaitTimeout(duration);
WebDriver driver = new ChromeDriver(**chromeOptions**);

Локаторы веб-элементов:

String CHECKBOX_EXAMPLE_BUTTON_LOCATOR = "//form[@id='checkbox-example']/button";

String MESSAGE_LOCATOR = "//p[@id='message']";

```
nicControlsWithImplicitWaitsTest.java  ×   DynamicControlsWithoutWaitsTest.java
ublic class DynamicControlsWithImplicitWaitsTest {   Sergei Tsarik
    public void testDynamicLoadedMessageWithDriverMethodImplicitWait() {
        String MESSAGE_LOCATOR = "//p[@id='message']";
        Extract  Surround  //
        (By.xpath(MESSAGE_LOCATOR)).getText()
        Assertions.assertEquals( expected: "It's gone!", actual);
    }

    @Test   Sergei Tsarik
    public void testDynamicLoadedMessageWithChromeOptionsImplicitWait() {
        ChromeOptions chromeOptions = new ChromeOptions();
        Duration duration = Duration.ofSeconds(4);
        chromeOptions.setImplicitWaitTimeout(duration);

        driver = new ChromeDriver(chromeOptions);
        driver.manage().window().maximize();

        driver.get(URL);

        String CHECKBOX_EXAMPLE_BUTTON_LOCATOR = "//form[@id='checkbox-example'
        driver.findElement(By.xpath(CHECKBOX_EXAMPLE_BUTTON_LOCATOR)).click();

        String MESSAGE_LOCATOR = "//p[@id='message']";
        String actual = driver.findElement(By.xpath(MESSAGE_LOCATOR)).getText()

        Assertions.assertEquals( expected: "It's gone!", actual);
    }

    @AfterEach   Sergei Tsarik
    public void tearDown() {
        driver.quit();
    }
}
```

ЯВНОЕ ОЖИДАНИЕ
EXPLICIT WAITS

Явные ожидания
Explicit waits

Явные ожидания – это циклы, добавленные в код, которые опрашивают веб-страницу на предмет определенного условия, чтобы оценить его как истинное, прежде чем оно выйдет из цикла и перейдет к следующей команде в коде.

Если условие не будет выполнено до указанного значения тайм-аута, код выдаст ошибку тайм-аута.

Поскольку существует множество способов, при которых веб-страница не может находиться в желаемом состоянии, то явные ожидания – отличный выбор для указания точного условия ожидания веб-элемента, где это необходимо.

Явные ожидания
Explicit waits

Фрагмент кода для настройки явного ожидания веб-элемента с помощью метода класса ExpectedConditions:

Wait<WebDriver> wait = new WebDriverWait(driver, Duration.ofSeconds(DEFAULT_DURATION_OF_SECONDS));

WebElement message = wait.**until**(**ExpectedConditions**
.visibilityOfElementLocated(By.xpath(MESSAGE_LOCATOR)));

String actual = message.getText();

amicControlsWithoutWaitsTest.java | DynamicControlsWithExplicitWaitsTest.java

```
import org.openqa.selenium.support.ui.WebDriverWait;

import java.time.Duration;

public class DynamicControlsWithExplicitWaitsTest {  Sergei Tsarik *

    private final String URL = "https://the-internet.herokuapp.com/dynamic_contr
    private WebDriver driver;  11 usages

    @Test  Sergei Tsarik *
    public void testDynamicLoadedMessageWithExplicitWait() {
        driver = new ChromeDriver();
        driver.manage().window().maximize();

        driver.get(URL);

        String CHECKBOX_EXAMPLE_BUTTON_LOCATOR =
                "//form[@id='checkbox-example']/button";
        driver.findElement(By.xpath(CHECKBOX_EXAMPLE_BUTTON_LOCATOR)).click();

        String MESSAGE_LOCATOR = "//p[@id='message']";
        Wait<WebDriver> wait = new WebDriverWait(driver, Duration.ofSeconds(4));
        WebElement message = wait.until(ExpectedConditions
                .visibilityOfElementLocated(By.xpath(MESSAGE_LOCATOR)));
        String actual = message.getText();

        Assertions.assertEquals( expected: "It's gone!", actual);
    }

    @Test  Sergei Tsarik
    public void testDynamicLoadedMessageWithCustomizedExplicitWait() {
        driver = new ChromeDriver();
        driver.manage().window().maximize();
```

aitsTest > testDynamicLoadedMessageWithExplicitWait 28:17 CRLF UTF-8 4

Настраиваемые явные ожидания

Экземпляр класса Wait может быть создан с различными параметрами, которые изменят способ оценки условий:
- задание общей продолжительности тайм-аута для данного ожидания,
- задание интервала запуска кода для оценки условия проверки веб-элемента,
- указание исключений, которые должны обрабатываться автоматически при возникновении исключительной ситуации при запуске кода для оценки условия проверки веб-элемента

Настраиваемые явные ожидания

Фрагмент кода для настройки явного ожидания веб-элемента с пользовательскими на стройками явного ожидания:

```
Wait<WebDriver> wait = new FluentWait<>(driver)
.withTimeout(Duration.ofSeconds(4))
.pollingEvery(Duration.ofMillis(500))
.ignoring(NoSuchElementException.class);

WebElement message = wait.until(ExpectedConditions
.visibilityOfElementLocated(By.xpath(MESSAGE_LOCATOR)));

 String actual = message.getText();
```

```
namicControlsWithExplicitWaitsTest.java

public class DynamicControlsWithExplicitWaitsTest {  Sergei Tsarik *
    public void testDynamicLoadedMessageWithExplicitWait() {
        Assertions.assertEquals( expected: "It's gone!", actual);
    }

    @Test  Sergei Tsarik *
    public void testDynamicLoadedMessageWithCustomizedExplicitWait() {
        driver = new ChromeDriver();
        driver.manage().window().maximize();

        driver.get(URL);

        String CHECKBOX_EXAMPLE_BUTTON_LOCATOR =
                "//form[@id='checkbox-example']/button";
        driver.findElement(By.xpath(CHECKBOX_EXAMPLE_BUTTON_LOCATOR)).click();

        String MESSAGE_LOCATOR = "//p[@id='message']";
        Wait<WebDriver> wait =
                new FluentWait<>(driver)
                        .withTimeout(Duration.ofSeconds(4))
                        .pollingEvery(Duration.ofMillis(500))
                        .ignoring(NoSuchElementException.class);
        WebElement message = wait.until(ExpectedConditions
                .visibilityOfElementLocated(By.xpath(MESSAGE_LOCATOR)));
        String actual = message.getText();

        Assertions.assertEquals( expected: "It's gone!", actual);
    }

    @AfterEach  Sergei Tsarik
    public void tearDown() {
        driver.quit();
    }
}
```

СТРАТЕГИИ ОЖИДАНИЯ
WAITING STRATEGIES

Стратегии ожидания
Waiting Strategies

Таким образом, использование механизмов
синхронизации Selenium
- неявное ожидание (Implicit waits),
- явное ожидание (Explicit waits)
позволяют дождаться присутствия и отображения
на веб-странице веб-элементов с которыми
необходимо взаимодействовать коду Selenium.

Документация:
- https://www.selenium.dev/documentation/webdriver/waits/
- https://www.selenium.dev/documentation/webdriver/drivers/options/#timeouts
- https://www.selenium.dev/documentation/webdriver/support_features/expected_conditions/

Код:
- https://github.com/github4ta/the-internet-herokuapp-com/tree/dynamic-controls