## **Протоколирование действий Selenium**

### EventFiringWebDriver: C#

#### **Примеры кода**

EventFiringWebDriver driver = new EventFiringWebDriver(new ChromeDriver());  
driver.FindingElement += (sender, e) => Console.WriteLine(e.FindMethod);  
driver.FindElementCompleted += (sender, e) => Console.WriteLine(e.FindMethod + " found");  
driver.ExceptionThrown += (sender, e) => Console.WriteLine(e.ThrownException);

### EventFiringWebDriver: Python

#### **Примеры кода**

from selenium.webdriver.support.events import EventFiringWebDriver, AbstractEventListener

class MyListener(AbstractEventListener):  
 def before\_find(self, by, value, driver):  
 print(by, value)  
 def after\_find(self, by, value, driver):  
 print(by, value, "found")  
 def on\_exception(self, exception, driver):  
 print(exception)

wd = EventFiringWebDriver(webdriver.Chrome(), MyListener())

### EventFiringWebDriver: Ruby

#### **Примеры кода**

require 'logger'

class MyListener < Selenium::WebDriver::Support::AbstractEventListener  
 def initialize(log)  
 @log = log  
 end  
  
 def before\_find(by, what, driver)  
 @log.info "#{by} #{what}"  
 end  
  
 def after\_find(by, what, driver)  
 @log.info "#{by} #{what} found"  
 end  
end

@driver = Selenium::WebDriver.for(:chrome, :listener => MyListener.new(Logger.new(STDOUT)))

### EventFiringWebDriver: Java

#### **Примеры кода**

public static class MyListener extends AbstractWebDriverEventListener {  
 @Override  
 public void beforeFindBy(By by, WebElement element, WebDriver driver) {  
 System.out.println(by);  
 }  
  
 @Override  
 public void afterFindBy(By by, WebElement element, WebDriver driver) {  
 System.out.println(by + " found");  
 }  
  
 @Override  
 public void onException(Throwable throwable, WebDriver driver) {  
 System.out.println(throwable);  
 }  
}

driver = new EventFiringWebDriver(new ChromeDriver());

### Снятие скриншотов

#### **Примеры кода**

##### **Java**

File tempFile = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);  
try {  
 Files.copy(tempFile, new File("screen.png"));  
} catch (IOException e) {  
 e.printStackTrace();  
}

##### **C#**

driver.GetScreenshot().SaveAsFile("screen.png", ImageFormat.Png);

##### **Python**

driver.get\_screenshot\_as\_file('screen.png')

##### **Ruby**

@driverdriver.save\_screenshot('screen.png')

##### **JavaScript**

var fs = require('fs');  
  
driver.takeScreenshot().then(

function(image) {

fs.writeFile('screen.png', image, 'base64', function(err) {

console.log(err);

});

}

);

#### **Ссылки**

Видеорекордер (Java): <https://github.com/SergeyPirogov/video-recorder-java>

### Доступ к логам браузера

#### **Примеры кода**

##### **Java**

for (LogEntry l : driver.manage().logs().get("browser").getAll()) {  
 System.out.println(l);  
}

##### **C#**

foreach (LogEntry l in driver.Manage().Logs.GetLog("browser"))  
{  
 Console.WriteLine(l);  
}

##### **Python**

for l in driver.get\_log("browser"):  
 print(l)

##### **Ruby**

@driver.manage.logs.get("browser").each do |l|  
 puts l  
end

##### **JavaScript**

driver.manage().logs().get("browser").then(function(logsEntries) {  
 logsEntries.forEach(function(l) {  
 console.log(l)  
 });  
});

### Перехват трафика

#### **Примеры кода**

##### **Java**

Proxy proxy = new Proxy();  
proxy.setHttpProxy("localhost:8888");  
DesiredCapabilities caps = new DesiredCapabilities();  
caps.setCapability("proxy", proxy);  
WebDriver driver = new ChromeDriver(caps);

##### **C#**

Proxy proxy = new Proxy();  
proxy.Kind = ProxyKind.Manual;  
proxy.HttpProxy = "localhost:8888";  
ChromeOptions options = new ChromeOptions();  
chromeOptions.Proxy = proxy;  
IWebDriver driver = new ChromeDriver(options);

##### **Python**

driver = webdriver.Chrome(desired\_capabilities={"proxy": {"proxyType": "MANUAL", "httpProxy": "localhost:8888"}})

##### **Ruby**

p = Selenium::WebDriver::Proxy.new(:http => "localhost:8888")  
caps = Selenium::WebDriver::Remote::Capabilities.new(:proxy => p)  
@driver = Selenium::WebDriver.for :chrome, desired\_capabilities: caps

##### **JavaScript**

var webdriver = require('selenium-webdriver'),  
 proxy = require('selenium-webdriver/proxy');  
  
var driver = new webdriver.Builder()  
 .withCapabilities(webdriver.Capabilities.chrome())  
 .setProxy(proxy.manual({http: 'localhost:8888'}))  
 .build();

#### **Ссылки**

BrowserMobProxy (Java):  
<https://github.com/lightbody/browsermob-proxy>

BrowserMobProxy, обёртка для C#:  
<https://github.com/AutomatedTester/AutomatedTester.BrowserMob>Пример использования:  
<http://automatedtester.science/load-testing-with-selenium-and-browsermob-proxy/>Ещё одна обёртка для C#:  
<https://github.com/tmulkern/RemoteBrowserMobProxy>

BrowserMobProxy, обёртка для Python:  
<https://github.com/AutomatedTester/browsermob-proxy-py>

BrowserMobProxy, обёртка для Ruby:  
<https://github.com/jarib/browsermob-proxy-rb>

BrowserMobProxy, обёртка для JavaScript:  
<https://github.com/zzo/browsermob-node>

Что можно делать при помощи прокси:  
<http://selenium2.ru/articles/106-selenium-i-browsermobproxy-vmeste-veselee.html>

Titanium: встраиваемый прокси на .Net:  
<https://github.com/justcoding121/Titanium-Web-Proxy>

FiddlerCore: встраиваемый прокси на .Net:  
<http://www.telerik.com/fiddler/fiddlercore>

mitmproxy: встраиваемый прокси на Python:  
<https://mitmproxy.org/>

## **Задание 17. Проверьте отсутствие сообщений в логе браузера**

Сделайте сценарий, который проверяет, не появляются ли в логе браузера сообщения при открытии страниц в учебном приложении, а именно -- страниц товаров в каталоге в административной панели.

Сценарий должен состоять из следующих частей:

1) зайти в админку  
2) открыть каталог, категорию, которая содержит товары (страница http://localhost/litecart/admin/?app=catalog&doc=catalog&category\_id=1)  
3) последовательно открывать страницы товаров и проверять, не появляются ли в логе браузера сообщения (любого уровня)

## **Задание 18. Перенаправьте трафик в прокси-сервер**

Установите какой-нибудь прокси-сервер, который умеет протоколировать запросы и ответы.

На выбор прокси-сервера для разных платформ:  
<http://www.telerik.com/fiddler> (Windows)  
<https://www.charlesproxy.com/> (Windows, Linux, MacOS, платный, но есть пробная версия)  
<https://mitmproxy.org/> (Linux, MacOS)  
<https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project> (Windows, LInux, MacOS)

Инициализируйте драйвер так, чтобы запросы из браузера отправлялись через этот прокси-сервер, убедитесь, что они там видны.