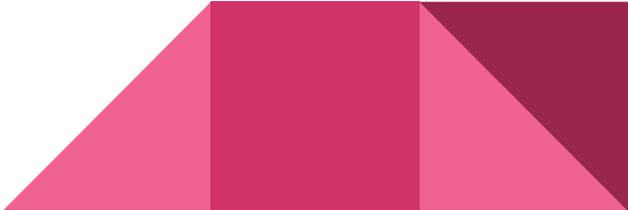


Автоматическое тестирование

JavaScript приложений

О чем поговорим

- Что такое автотестирование
 - Автоматические тесты при помощи chai и mocha
 - Автозапуск тестов через Karma
 - Headless браузеры
 - Интеграция с travis
 - WebPack для автотестирования
 - Тонкости тестирования в JavaScript
- 

Автоматизированное тестирование – это процесс верификации ПО, при котором основные функции и шаги теста (запуск, инициализация, выполнение, анализ и выдача результата) выполняются автоматически.

JavaScript ("JS" для краткости) — это полноценный динамический язык программирования, который применяется к HTML документу, и может обеспечить динамическую интерактивность на веб-сайтах



Преимущества автоматизации тестирования:

- Повторяемость – все тесты будут выполняться однообразно, полностью исключен «человеческий фактор».
- Экономия времени
 - автоматизированному скрипту не нужно сверяться с инструкциями и документациями.
 - в разы быстрее тестирования вручную.
 - не требуют контроля (могут выполняться даже ночью)
- Отчеты – автоматически рассылаются и сохраняются



Где нужно применять автоматизацию?

1. Труднодоступные места в системе (бэкенд процессы, логирование файлов, запись в БД)
 2. Часто используемая функциональность, риски от ошибок в которой достаточно высоки.
 3. Рутинные операции, такие как переборы данных (формы с большим количеством вводимых полей).
 4. Длинные end-to-end сценарии
 5. Проверка данных, требующих точных математических расчетов
 6. Проверка правильности поиска данных
- 

Виды тестирования:

1. Модульное тестирование — проверка работы программы на уровне отдельных модулей (классов, методов)
 2. Интеграционное тестирование — проверка совместной работы нескольких модулей
 3. Системное тестирование — проверка работы системы в целом
- 

БИБЛИОТЕКИ

1. Selenium Web Driver <https://www.seleniumhq.org/projects/webdriver/>
2. Puppeteer <https://github.com/GoogleChrome/puppeteer>
3. Запуск кода в Headless браузерах (FF или Chrome, Phantom)



```
// Скрипт работает с интерфейсом,  
// а не с реализацией.  
WebDriver driver = new FirefoxDriver();  
  
// Открываем гугл, используя драйвер  
driver.get("http://www.google.com");  
  
// Находим элемент по атрибуту name  
WebElement element = driver.findElement(By.name("q"));  
  
// Вводим текст  
element.sendKeys("Selenium");  
// Отправляем форму  
element.submit();  
driver.quit();
```

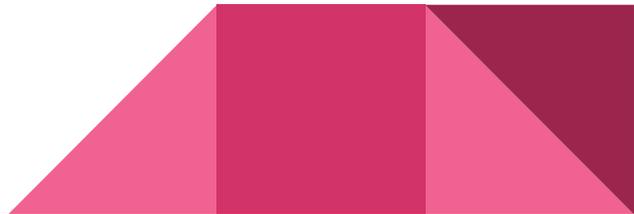


Puppeteer

```
const puppeteer = require('puppeteer');

(async () => {
  const browser = await puppeteer.launch();
  const page = await browser.newPage();
  await page.goto('https://example.com');
  await page.screenshot({path: 'example.png'});

  await browser.close();
})();
```



Установка пакетов

```
mkdir autotest && cd autotest && npm init -y
```

```
npm i webpack webpack-cli mocha chai karma karma-firefox-launcher karma-chai  
karma-mocha -D
```

```
./node_modules/.bin/webpack init (добавит еще 129 пакетов)
```



karma.config.js

```
module.exports = function(config) {  
  
  config.set({  
  
    basePath: "",  
  
    frameworks: ['mocha', 'chai'],  
  
    files: [  
  
      'jquery.periodpicker.css', 'jquery.timepicker.css', 'jquery.min.js',  
'node_modules/moment/min/moment-with-locales.min.js', 'jquery.periodpicker.js', 'jquery.timepicker.js', 'test/bootstrap.js',  
'test/tests/*.js'  
  
    ],  
  
    reporters: ['progress'],
```



```
port: 9876,  
  
colors: true,  
  
browsers: ['Firefox'],  
  
autoWatch: true,  
  
singleRun: false, // Karma captures browsers, runs the tests and exits  
  
concurrency: Infinity,  
  
plugins: [  
  'karma-firefox-launcher', 'karma-mocha', 'karma-chai'  
]  
})  
};
```



Спасибо за внимание