|  |
| --- |
| РГПУ им. Герцена, ИКНиТО, ИВТ1 |
| Хочу стать веб-разработчиком |
| Подробный план по изучению JavaScript |

|  |
| --- |
| Храмов Сергей Анатольевич  05.11.2017 |

Оглавление

[Хочу стать веб-разработчиком: подробный план по изучению JavaScript 2](#_Toc498366154)

[Бэкенд 4](#_Toc498366155)

[Фронтенд 6](#_Toc498366156)

[Продвинутые темы 8](#_Toc498366157)

**Хочу стать веб-разработчиком: подробный план по изучению JavaScript**

Веб-разработка — одно из самых простых и оттого популярных направлений среди начинающих программистов. Для работы достаточно любого текстового редактора и браузера, не нужно изучать алгоритмы на продвинутом уровне, результат каждого этапа написания программы нагляден — в общем, преимуществ много. Ключевым навыком в контексте веб-разработки считается знание JavaScript.

Сейчас JavaScript очень быстро развивается, а потому при изучении языка легко запутаться. Мы предлагаем вашему вниманию учебный план с удобной структурой, который охватывает все необходимые аспекты JavaScript и смежных технологий.

Почему JavaScript?

В первую очередь нужно оценить уровень своих знаний и определить, действительно ли вам подходит JavaScript — для этого предлагаем пройти [специальный тест](https://geekbrains.ru/welcome_test?utm_medium=adv_post&utm_source=tproger.ru&utm_campaign=19.06.17&utm_term=izuchenie_javascript&utm_content=welcome_test).

Стоит отметить открытость языка — компании, обычно соперничающие друг с другом, сотрудничают с целью развития JavaScript. Язык очень гибок и подойдёт сторонникам как объектно-ориентированного, так и функционального подхода. Огромное количество библиотек и фреймворков позволяет с лёгкостью решать любые типы задач, а серверная платформа Node.js даёт возможность использовать язык не только в браузере, но и в консоли. Вы даже можете писать настольные и мобильные приложения: первые — при помощи фреймворка Electron, а вторые — на NativeScript или React Native.

Основы

Сперва необходимо изучить основные понятия JavaScript, веб-разработки и программирования в целом:

Git[[1]](#footnote-2)

Git — необходимый разработчикам инструмент, поэтому [познакомиться](https://geekbrains.ru/courses/66?utm_medium=adv_post&utm_source=tproger.ru&utm_campaign=19.06.17&utm_term=izuchenie_javascript&utm_content=courses_66) с ним нужно как можно раньше. Вот основные навыки, которыми вы должны обладать:

* создание и перемещение файлов в каталогах;
* инициализация и коммиты в Git;
* настройка репозиториев в GitHub.

Алгоритмы и структуры данных

Затем стоит изучить алгоритмы (в частности, понятие сложности алгоритмов), а также базовые структуры данных:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| связные списки | очереди | стеки | двоичные деревья поиска | хэш-таблицы |

В этом вам поможет [серия статей](https://tproger.ru/tag/algos-and-data-structs/).

**Бэкенд**

Node.js

10 лет назад JavaScript можно было использовать только для фронтенд-разработки. Теперь благодаря Node.js одним «фронтом» дело не ограничивается. Node — это просто среда для выполнения JS-кода на стороне сервера, поэтому вам не придётся изучать новый синтаксис, но понадобится импортировать и экспортировать файлы, разбивать код на модули и использовать менеджер пакетов npm.

Серверы, HTTP, Express.js

После изучения Node стоит продолжить знакомство с бэкенд-разработкой и разобраться в серверах и маршрутизации. Можно начать с портов и протоколов с акцентом на HTTP, а потом заняться Express — Node-библиотекой для обработки запросов.

Асинхронный JavaScript

[Асинхронность](https://tproger.ru/translations/asynchronous-javascript/) является одной из отличительных фич JavaScript, хотя и создаёт некий раскол среди разработчиков: некоторые её любят, а некоторые — ненавидят. Вам стоит понимать плюсы и минусы этой технологии. Начните со стека вызовов, цикла событий и коллбеков, а затем перейдите к изучению [промисов](https://tproger.ru/translations/meet-the-promises/" \t "_blank).

Базы данных, схемы, модели и ORM

Базы данных — один из важнейших элементов веб-разработки. Если вашему приложению нужно загружать или хранить какие-либо данные, не теряющиеся при обновлении страницы, придётся использовать БД. Нужно научиться различать реляционные и нереляционные базы данных и разобраться в типах связей. Затем [изучить SQL](https://tproger.ru/video/sql-introduction/) и познакомиться с разными [системами управления баз данных](https://tproger.ru/translations/sqlite-mysql-postgresql-comparison/). Умение работать с ORM тоже не будет лишним.

**Фронтенд**

HTML и CSS

HTML и CSS — это основа основ для любого веб-разработчика. Вам не обязательно знать их в совершенстве, но разбираться в них вы должны. Также можно изучить какую-нибудь популярную библиотеку (например, Bootstrap) и CSS-препроцессор вроде Sass — он поможет сделать CSS похожим на обычный код. Для упрощения работы с HTML можно выбрать один из популярных шаблонизаторов, например, pug.

jQuery и манипуляция DOM

Создав внешний облик страницы при помощи HTML и CSS, вы будете использовать трансляторы событий и библиотеку jQuery для управления DOM. Многие считают, что jQuery бесполезна и её скоро заменят Angular и React, но она безумно популярна, и поэтому её стоит знать. Кроме того, однажды вы попадёте в ситуацию, когда забивать гвозди React-микроскопом вам будет неудобно, и тогда на помощь вам придёт лёгкая jQuery.

Инструменты разработчика Chrome



Было бы непростительно обделить вниманием инструменты Chrome, которые дают огромное количество возможностей. С ними вы сможете изучать элементы DOM, производить отладку через консоль, отслеживать маршруты и многое другое. В [статье](https://tproger.ru/translations/chrome-command-line-api-cheatsheet/) описаны несколько удобных возможностей консоли Chrome, которые упростят выполнение рутинных задач.



AJAX

Если вы хотите, чтобы приложение не перезагружало страницы после каждой операции с базой данных, вам точно понадобится AJAX — он отправляет фоновые асинхронные HTTP-запросы, ответы на которые обновляют лишь часть отображения. Работать с AJAX можно через jQuery при помощи метода.

**Продвинутые темы**

Разработка через тестирование

[Test-Driven Development](https://tproger.ru/articles/why-test-driven-development/), или TDD — это такая техника разработки, при которой создание ПО разбивается на множество небольших циклов: сначала пишутся тесты, которые покрывают желаемое изменение, затем пишется код, который эти тесты проходит. После этого производится рефакторинг кода, при необходимости пишутся новые тесты. Если какие-то тесты участок кода не проходит, это исправляется.

В случае с JavaScript советуем обратить внимание на фреймворки Jasmine, Chai и Mocha. Кстати, на нашем сайте есть [цикл статей](https://tproger.ru/translations/frontend-testing-1/), посвящённый тестированию фронтенда — может пригодиться.

Веб-сокеты

Этой теме стоит уделить особое внимание, поскольку веб-сокеты очень полезны. Протокол WebSocket, в отличие от HTTP, позволяет работать с двунаправленным потоком данных, что делает эту технологию уникальной. Самой распространённой реализацией является библиотека socket.io — разобраться с ней и применить полученные навыки на практике поможет [наше руководство](https://tproger.ru/translations/multiplayer-browser-game-1/) по созданию многопользовательской браузерной игры.

ES6, Babel, Webpack

Сейчас основным стандартом является ES6 (ES2015), но уже принят ES2016, а ES2017 находится в разработке, поэтому нужно всегда быть в курсе обновлений и использовать их. Проблемы совместимости решаются всего парой инструментов:

1. Babel — компилирует ES6-код (скоро появится поддержка ES2016) в ES5, который поддерживается всеми браузерами. Он даже умеет компилировать JSX/React-компоненты, что делает его незаменимым для любого веб-разработчика;
2. Webpack — собирает все ваши исходные файлы (картинки, шрифты, таблицы стилей, JS-файлы и т.д.) в единый граф зависимостей. Он не пригодится при создании маленьких приложений, но при работе с React его помощь неоценима.

React и Redux

React — библиотека для создания пользовательских интерфейсов. Она была создана компанией Facebook в 2013 году и достаточно быстро стала популярной среди разработчиков. Вам стоит прочитать [эти 9 советов](https://tproger.ru/translations/9-useful-tips-react/), а затем разобрать написание простого приложения, чтобы понять, как работает React и для чего он нужен. React пригодится не только для фронтенд-разработки: Facebook выпустила варианты фреймворка для мобильной (React Native) и VR-разработки (React VR).

Redux — контейнер предсказуемых состояний, обычно используемый в связке с React. Его можно использовать для сокращения кода благодаря модульности. Особенно полезен в многопользовательских приложениях, работающих в режиме реального времени, например, в играх

Аутентификация, сессии, cookies

Вам также стоит разобраться, как приложения взаимодействуют с пользователями, обрабатывают вход в учётную запись и выход из неё, выдают привилегии. Для отслеживания личности пользователя в течение сессии используются cookies — небольшие текстовые файлы, передаваемые сервером браузеру в ответе на HTTP-запрос. А для обеспечения связи базы данных и страничек авторизации можно использовать библиотеку express-session.

Веб-безопасность[[2]](#footnote-3)

И, наконец, стоит обратить внимание на безопасность. Важно изучить как методы атак, так и способы защиты от них.

1. GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки [↑](#footnote-ref-2)
2. Обеспечение того, чтобы ваш веб-сайт или открытое веб-приложение было безопасным [↑](#footnote-ref-3)