

**Николай Прохоренко  
Владимир Дронов**

**PRO**

**ПРОФЕССИОНАЛЬНОЕ  
ПРОГРАММИРОВАНИЕ**

# JavaScript и Node.js

**для веб-  
разработчиков**

**Итераторы и генераторы  
Модули**

**Класс Promise**

**Работа с файловой системой**

**Программирование веб-серверов**

**Объектная модель документа**

**AJAX и Fetch API**

**Webpack**

**Visual Studio Code**



Материалы  
на [www.bhv.ru](http://www.bhv.ru)

Николай Прохоренок  
Владимир Дронов

# JavaScript и Node.js

**для веб-разработчиков**

Санкт-Петербург  
«БХВ-Петербург»  
2022

УДК 004.438JavaScript

ББК 32.973.26-018.1

П84

**Прохоренок, Н. А.**

П84 JavaScript и Node.js для веб-разработчиков / Н. А. Прохоренок, В. А. Дронов. — СПб.: БХВ-Петербург, 2022. — 768 с.: ил. — (Профессиональное программирование)

ISBN 978-5-9775-6847-0

Книга рассказывает о языке программирования JavaScript, разработке на нем как программ общего назначения, выполняющихся в среде Node.js, так и скриптов для веб-страниц. Даны основы JavaScript: типы данных, операторы, работа с числами, строками, датой и временем, массивами, функции, классы (как старого, так и нового синтаксиса), итераторы, генераторы и класс Promise. Объяснена работа с отладчиком, встроенным в редактор Visual Studio Code. Рассказано о модулях, средствах для работы с файловой системой и программирования веб-серверов. Описана объектная модель документа. Рассмотрены средства для работы с элементами веб-страницы, самой страницей и браузером и технология AJAX (в том числе Fetch API), а также готовые программные пакеты для разработки веб-сайтов, в частности Webpack.

Электронный архив на сайте издательства содержит коды всех пронумерованных листингов.

*Для веб-разработчиков*

УДК 004.438JavaScript

ББК 32.973.26-018.1

**Группа подготовки издания:**

Руководитель проекта	<i>Евгений Рыбаков</i>
Зав. редакцией	<i>Людмила Гауль</i>
Компьютерная верстка	<i>Ольги Сергиенко</i>
Дизайн серии	<i>Инны Тачиной</i>
Оформление обложки	<i>Карины Соловьевой</i>

"БХВ-Петербург", 191036, Санкт-Петербург, Гончарная ул., 20.

ISBN 978-5-9775-6847-0

© ООО "БХВ", 2022

© Оформление. ООО "БХВ-Петербург", 2022

# Оглавление

Lituz.com

<b>Введение .....</b>	<b>15</b>
Структура книги.....	15
Типографские соглашения.....	17
<b>Глава 1. Редактор Visual Studio Code .....</b>	<b>19</b>
1.1. Установка VS Code.....	19
1.2. Настройка редактора .....	23
1.3. Смена цветовой темы, тем значков файлов и продукта .....	25
1.4. Структура окна редактора.....	27
1.4.1. Главное меню .....	28
1.4.2. Боковая панель и панель действий .....	32
1.4.3. Строка состояния .....	33
1.4.4. Нижняя панель .....	34
1.4.5. Палитра команд.....	34
1.5. Работа с файлами и каталогами.....	35
1.5.1. Создание и сохранение файла.....	35
1.5.2. Закрытие вкладки с файлом .....	36
1.5.3. Открытие файла .....	37
1.5.4. Открытие каталога .....	37
1.6. Отображение содержимого нескольких файлов одновременно .....	39
1.7. Live Server: автоматическое обновление веб-страницы .....	41
1.8. Emmet: ускорение набора HTML-кода .....	42
1.8.1. Вставка структуры HTML-документа .....	42
1.8.2. Вставка HTML-комментария .....	43
1.8.3. Теги из секции заголовка ( <i>&lt;head&gt;</i> ).....	43
1.8.4. Основные теги из секции тела ( <i>&lt;body&gt;</i> ).....	44
1.8.5. Добавление текста.....	46
1.8.6. Добавление параметров тегов.....	47
1.8.7. Формы и элементы управления .....	48
1.8.8. Таблицы .....	49
1.8.9. Списки.....	50
1.8.10. Вставка сразу нескольких тегов.....	51

1.9. Emmet: ускорение набора CSS-кода.....	52
1.9.1. Форматирование шрифта .....	52
1.9.2. Форматирование текста.....	55
1.9.3. Отступы .....	57
1.9.4. Рамки.....	58
1.9.5. Фон элемента.....	60
1.9.6. Списки.....	61
1.9.7. Таблицы .....	62
1.9.8. Вид курсора .....	62
1.9.9. Форматирование блоков.....	62
1.9.10. Flex-контейнеры.....	64
1.9.11. CSS Grid.....	66
1.9.12. Многоколоночный текст .....	68
1.9.13. Эффекты .....	68
1.9.14. Анимация с двумя состояниями .....	69
1.9.15. Анимация с несколькими состояниями.....	69
1.9.16. Двумерные трансформации .....	70
1.9.17. Трехмерные трансформации.....	71
1.9.18. Прочие атрибуты и правила.....	71
1.9.19. Ввод числового значения атрибута .....	73
1.9.20. Добавление вендорных префиксов.....	73
1.10. Всплывающие подсказки при вводе.....	74
1.11. Пользовательские фрагменты кода .....	75
1.11.1. В Emmet.....	75
1.11.2. В редакторе VS Code .....	77
1.12. Работа с курсорами и выделениями .....	82
1.13. Дублирование, перемещение и удаление строк .....	83
1.14. Сворачивание и разворачивание блоков кода.....	84
1.15. Изменение регистра символов.....	84
1.16. Изменение масштаба .....	85
1.17. Полноэкранный режим.....	85
1.18. Форматирование кода .....	85
1.19. Полезные расширения.....	86
<b>Глава 2. Node.js: первые шаги .....</b>	<b>88</b>
2.1. Установка Node.js .....	88
2.2. Работа с командной строкой.....	92
2.3. Первая программа на JavaScript .....	94
2.4. Вкладка <i>Терминал</i> в редакторе VS Code.....	95
2.5. NPM (Node Package Manager).....	97
2.5.1. Создание пакета и добавление файла package.json .....	98
2.5.2. Установка пакетов .....	101
2.5.3. Обновление и удаление пакетов .....	103
2.5.4. Добавление и запуск скриптов.....	104
2.6. Структура программы .....	108
2.7. Комментарии в JavaScript .....	109
2.8. Интерактивный режим .....	111
2.9. Вывод результатов работы программы .....	112

2.10. Получение данных из командной строки .....	115
2.11. Завершение выполнения программы .....	117
2.12. Получение версии Node.js.....	117
<b>Глава 3. Переменные и типы данных .....</b>	<b>119</b>
3.1. Именованые переменных .....	119
3.2. Объявление переменной .....	120
3.3. Инициализация переменной и присваивание значения.....	120
3.4. Ключевые слова <i>var</i> и <i>let</i> .....	121
3.5. Типы данных в языке JavaScript .....	123
3.6. Проверка существования переменной .....	124
3.7. Преобразование типов данных .....	125
3.8. Классы-"обертки" над элементарными типами .....	127
3.9. Константы .....	128
3.10. Области видимости переменных .....	129
3.11. Вывод значений переменных.....	132
<b>Глава 4. Операторы и циклы.....</b>	<b>133</b>
4.1. Математические операторы.....	133
4.2. Побитовые операторы .....	135
4.3. Оператор запятая .....	136
4.4. Операторы сравнения.....	137
4.5. Оператор нулевого слияния.....	139
4.6. Операторы присваивания.....	139
4.7. Приоритет выполнения операторов .....	140
4.8. Оператор ветвления <i>if...else</i> .....	141
4.9. Оператор <i>?:</i> .....	147
4.10. Оператор выбора <i>switch</i> .....	148
4.11. Цикл <i>for</i> .....	151
4.12. Цикл <i>while</i> .....	153
4.13. Цикл <i>do...while</i> .....	154
4.14. Цикл <i>for...in</i> .....	155
4.15. Цикл <i>for...of</i> .....	157
4.16. Оператор <i>continue</i> : переход на следующую итерацию цикла .....	157
4.17. Оператор <i>break</i> : прерывание цикла.....	158
<b>Глава 5. Числа.....</b>	<b>159</b>
5.1. Указание значений.....	159
5.2. Класс <i>Number</i> .....	160
5.3. Математические константы .....	164
5.4. Основные методы для работы с числами .....	165
5.5. Округление чисел .....	166
5.6. Тригонометрические функции .....	166
5.7. Преобразование строки в число .....	167
5.8. Преобразование числа в строку.....	168
5.9. Генерация псевдослучайных чисел.....	171
5.10. Бесконечность и значение <i>NaN</i> .....	171
5.11. Тип данных <i>bigint</i> .....	172

<b>Глава 6. Массивы и множества .....</b>	<b>175</b>
6.1. Инициализация массива .....	175
6.2. Получение и изменение элемента массива .....	177
6.3. Определение размера массива .....	179
6.4. Многомерные массивы .....	179
6.5. Создание копии массива .....	181
6.6. Объединение массивов .....	182
6.7. Перебор элементов массива .....	182
6.8. Добавление и удаление элементов массива .....	183
6.9. "Переворачивание" массива .....	185
6.10. Сортировка массива .....	185
6.11. Получение части массива .....	186
6.12. Преобразование массива .....	187
6.13. Поиск элемента в массиве .....	189
6.14. Фильтрация массива .....	190
6.15. Методы, возвращающие итераторы .....	191
6.16. Ассоциативные массивы .....	192
6.17. Класс <i>Map</i> : словарь .....	193
6.17.1. Создание объекта .....	193
6.17.2. Добавление элементов и изменение значения .....	194
6.17.3. Получение значения по ключу .....	194
6.17.4. Проверка наличия ключа .....	194
6.17.5. Определение размера массива .....	194
6.17.6. Удаление элементов .....	195
6.17.7. Методы класса <i>Map</i> , возвращающие итераторы .....	195
6.17.8. Перебор элементов словаря .....	196
6.18. Класс <i>Set</i> : множество .....	197
6.18.1. Создание объекта .....	197
6.18.2. Добавление элементов .....	197
6.18.3. Проверка наличия элемента .....	198
6.18.4. Определение размера множества .....	198
6.18.5. Удаление элементов .....	198
6.18.6. Методы класса <i>Set</i> , возвращающие итераторы .....	199
6.18.7. Перебор элементов множества .....	199
6.18.8. Преобразование множества в массив .....	200
6.18.9. Операции с двумя множествами .....	200
<b>Глава 7. Строки .....</b>	<b>202</b>
7.1. Инициализация строк .....	202
7.2. Специальные символы в строке .....	204
7.3. Строки в обратных кавычках (шаблоны строк) .....	205
7.4. Неформатированные строки .....	206
7.5. Конкатенация строк .....	206
7.6. Определение длины строки .....	207
7.7. Обращение к отдельному символу в строке .....	207
7.8. Изменение регистра символов .....	208
7.9. Получение фрагмента строки .....	209

7.10. Сравнение строк .....	209
7.11. Поиск и замена в строке.....	210
7.12. Преобразование строки в массив .....	212
7.13. URL-кодирование строк.....	213
7.14. Повтор строки.....	213
7.15. Выполнение команд, содержащихся в строке.....	214
<b>Глава 8. Регулярные выражения .....</b>	<b>215</b>
8.1. Создание регулярного выражения .....	215
8.2. Методы класса <i>String</i> .....	216
8.3. Методы класса <i>RegExp</i> .....	222
8.4. Свойства класса <i>RegExp</i> .....	224
8.5. Синтаксис регулярных выражений .....	226
8.5.1. Экранирование специальных символов .....	226
8.5.2. Метасимволы.....	228
8.5.3. Стандартные классы .....	230
8.5.4. Квантификаторы.....	231
8.5.5. "Жадность" квантификаторов .....	231
8.5.6. Группы .....	232
8.5.7. Обратные ссылки .....	235
8.5.8. Просмотр вперед и назад.....	235
<b>Глава 9. Работа с датой и временем.....</b>	<b>238</b>
9.1. Получение текущей даты и времени .....	238
9.2. Указание произвольных значений даты и времени .....	238
9.3. Разбор строки с датой и временем .....	239
9.4. Преобразование даты в строку .....	240
9.5. Получение и изменение значений компонентов даты и времени.....	241
9.6. Форматированный вывод текущей даты и времени .....	243
9.7. Таймеры.....	244
9.8. Измерение времени выполнения кода .....	247
<b>Глава 10. Пользовательские функции .....</b>	<b>249</b>
10.1. Создание функции и ее вызов.....	249
10.2. Расположение определений функций .....	251
10.3. Вложенные функции .....	252
10.4. Замыкание .....	252
10.5. Класс <i>Function</i> .....	253
10.6. Функции с произвольным числом параметров .....	255
10.6.1. Объект <i>arguments</i> .....	255
10.6.2. Оператор <i>REST</i> .....	256
10.7. Способы передачи параметров в функцию .....	256
10.8. Необязательные параметры .....	257
10.9. Анонимные функции .....	258
10.10. Стрелочные функции (лямбда-выражения).....	260
10.11. Функции-генераторы.....	261
10.12. Рекурсия .....	263



<b>Глава 11. Классы и объекты .....</b>	<b>264</b>
11.1. Объекты .....	264
11.1.1. Создание объекта с помощью фигурных скобок.....	264
11.1.2. Свойства объекта.....	265
11.1.3. Распаковка объекта .....	268
11.1.4. Методы объекта.....	270
11.1.5. Указатель <i>this</i> .....	272
11.1.6. Setter и getter .....	274
11.1.7. Свойство <i>__proto__</i> .....	276
11.1.8. Атрибуты свойств и метод <i>Object.create()</i> .....	278
11.1.9. Получение значений атрибутов свойств .....	280
11.1.10. Методы <i>defineProperty()</i> и <i>defineProperties()</i> .....	282
11.1.11. Ограничение доступа к объекту.....	284
11.1.12. Получение списка имен свойств .....	287
11.1.13. Перебор свойств объекта.....	289
11.1.14. Проверка существования свойств и методов.....	290
11.1.15. Свойства с типом <i>symbol</i> .....	291
11.1.16. Сравнение объектов .....	292
11.1.17. Создание копии объекта .....	293
11.1.18. Объединение объектов.....	295
11.1.19. Удаление свойства .....	296
11.2. Создание объекта с помощью класса <i>Object</i> .....	298
11.3. Создание класса (старый стиль).....	299
11.3.1. Функция в качестве конструктора класса .....	300
11.3.2. Прототипы .....	301
11.3.3. Свойство <i>constructor</i> .....	303
11.3.4. Наследование.....	303
11.3.5. Переопределение методов.....	305
11.3.6. Статические свойства и методы.....	307
11.4. Создание класса (новый стиль) .....	307
11.4.1. Инструкция <i>class</i> .....	308
11.4.2. Конструктор класса.....	308
11.4.3. Наследование.....	309
11.4.4. Переопределение методов.....	310
11.4.5. Публичные свойства .....	311
11.4.6. Приватные свойства и методы .....	312
11.4.7. Статические свойства и методы.....	313
11.5. Оператор <i>instanceof</i> .....	315
11.6. Массивоподобные объекты (псевдомассивы).....	316
11.7. Итераторы .....	317
11.8. Методы-генераторы.....	319
11.9. Пространства имен .....	320
11.10. Класс <i>Proxy</i> .....	322
11.10.1. Создание проху-объекта.....	322
11.10.2. Ограничение доступа к свойствам объекта.....	324
11.10.3. Ограничение проверки существования свойства .....	327
11.10.4. Ограничение получения списка свойств и их атрибутов .....	328
11.10.5. Ограничение добавления свойства и изменения атрибутов .....	330

11.10.6. Ограничение удаления свойства .....	331
11.10.7. Ограничение нерасширяемости объекта .....	332
11.10.8. Ограничение доступа к прототипу.....	333
11.10.9. Перехват создания объекта.....	334
11.10.10. Перехват вызова функции .....	335
11.10.11. Отключаемый проху-объект .....	335
11.11. Класс <i>Reflect</i> .....	336
11.12. Формат JSON: преобразование объекта в строку и обратно .....	340
<b>Глава 12. Модули.....</b>	<b>345</b>
12.1. Модули CommonJS .....	345
12.1.1. Подключение встроенных модулей .....	345
12.1.2. Получение списка встроенных модулей.....	346
12.1.3. Подключение пользовательских модулей.....	347
12.1.4. Кэширование модулей .....	350
12.1.5. Подключение каталогов.....	352
12.1.6. Подключение модулей и пакетов из каталога <i>node_modules</i> .....	353
12.1.7. Переменная окружения <i>NODE_PATH</i> .....	354
12.1.8. Экспорт идентификаторов из модуля.....	355
12.1.9. Объект <i>module</i> .....	357
12.2. Модули ECMAScript (ESM).....	358
12.2.1. Основные отличия модулей ECMAScript от модулей CommonJS .....	358
12.2.2. Подключение встроенных модулей .....	359
12.2.3. Получение списка встроенных модулей.....	360
12.2.4. Подключение пользовательских модулей.....	361
12.2.5. Кэширование модулей .....	363
12.2.6. Подключение модулей и пакетов из каталога <i>node_modules</i> .....	364
12.2.7. Экспорт идентификаторов из модуля.....	364
12.2.8. Экспорт идентификатора по умолчанию .....	367
12.2.9. Реэкспорт .....	369
12.2.10. Динамический импорт .....	371
12.2.11. Подключение модулей CommonJS.....	373
12.2.12. Подключение модуля ECMAScript внутри модуля CommonJS.....	374
12.2.13. Подключение модулей в формате <i>JSON</i> .....	375
12.3. Свойства <i>main</i> , <i>exports</i> и <i>imports</i> в файле <i>package.json</i> .....	376
12.4. Переменные уровня приложения .....	385
<b>Глава 13. Обработка ошибок .....</b>	<b>387</b>
13.1. Типы ошибок.....	387
13.2. Исключения и их обработка. Инструкция <i>try...catch...finally</i> .....	389
13.3. Класс <i>Error</i> : объекты исключения .....	392
13.4. Оператор <i>throw</i> : генерирование исключений.....	393
13.5. Поиск ошибок в программе .....	394
13.6. Метод <i>assert()</i> .....	396
13.7. Отладка программы в редакторе VS Code.....	396
13.8. Строгий режим.....	402
13.9. Установка и настройка <i>Prettier</i> .....	407
13.10. Установка и настройка <i>ESLint</i> .....	415

<b>Глава 14. Асинхронность .....</b>	<b>421</b>
14.1. Объект класса <i>Promise</i> .....	421
14.1.1. Создание объекта класса <i>Promise</i> .....	421
14.1.2. Обработка изменения статуса .....	423
14.1.3. Массовая обработка объектов класса <i>Promise</i> .....	425
14.2. Ключевые слова <i>async</i> и <i>await</i> .....	429
14.3. Цикл <i>for await...of</i> .....	431
14.4. Асинхронные итераторы.....	433
14.5. Асинхронные методы-генераторы .....	434
14.6. Обработка событий.....	435
14.6.1. Генерирование событий.....	436
14.6.2. Назначение обработчиков событий .....	437
14.6.3. Удаление обработчиков событий.....	440
14.6.4. Асинхронная обработка событий.....	441
<b>Глава 15. Класс <i>Buffer</i>: массив байтов фиксированного размера .....</b>	<b>443</b>
15.1. Создание массива байтов .....	443
15.2. Определение размера массива.....	447
15.3. Получение и изменение значения по индексу.....	448
15.4. Запись и чтение данных .....	448
15.5. Создание копии массива .....	451
15.6. Получение части массива.....	452
15.7. Объединение массивов.....	453
15.8. Изменение порядка следования байтов .....	453
15.9. Перебор элементов массива.....	454
15.10. Методы, возвращающие итераторы.....	455
15.11. Сравнение массивов .....	456
15.12. Проверка наличия значения в массиве .....	457
15.13. Преобразование массива в строку или в другой объект.....	459
15.14. Преобразование кодировок.....	460
<b>Глава 16. Чтение и запись файлов .....</b>	<b>462</b>
16.1. Указание пути к файлу или каталогу .....	462
16.2. Модуль <i>path</i> : преобразование путей .....	466
16.3. Запись в файл с указанием пути к файлу .....	470
16.4. Чтение из файла с указанием пути к файлу.....	480
16.5. Открытие и закрытие файла .....	485
16.6. Режимы открытия файла.....	489
16.7. Запись в файл с указанием дескриптора.....	491
16.8. Чтение из файла с указанием дескриптора.....	500
16.9. Изменение размера файла.....	510
16.10. Дескрипторы стандартных потоков ввода/вывода .....	514
<b>Глава 17. Файловые потоки ввода/вывода .....</b>	<b>516</b>
17.1. Класс <i>WriteStream</i> : поток вывода в файл.....	516
17.1.1. Создание потока вывода .....	516
17.1.2. Запись данных .....	518
17.1.3. События потока вывода .....	519

17.2. Класс <i>ReadStream</i> : поток ввода из файла.....	520
17.2.1. Создание потока ввода.....	520
17.2.2. Чтение данных.....	521
17.2.3. События потока ввода.....	522
17.2.4. Метод <i>pipe()</i> .....	523
17.2.5. Чтение файла, сохраненного в русской кодировке.....	524
17.3. Перенаправление стандартных потоков ввода/вывода.....	524
<b>Глава 18. Работа с файловой системой.....</b>	<b>526</b>
18.1. Переименование и перемещение файлов.....	526
18.2. Копирование файлов.....	529
18.3. Удаление файлов.....	532
18.4. Получение сведений о файлах и каталогах.....	535
18.5. Права доступа к файлу и каталогу.....	542
18.6. Проверка существования файлов и каталогов.....	544
18.7. Создание каталогов.....	545
18.8. Создание временных каталогов.....	547
18.9. Удаление каталогов.....	548
18.10. Перебор элементов в каталоге.....	551
18.11. Отслеживание изменения файла или каталога.....	556
<b>Глава 19. Веб-сервер на Node.js.....</b>	<b>560</b>
19.1. Создание и запуск веб-сервера.....	560
19.2. Объект запроса.....	562
19.3. Объект ответа.....	563
19.4. Отправка файла в качестве серверного ответа.....	565
19.5. <i>Nodemon</i> : автоматическая перезагрузка веб-сервера.....	566
<b>Глава 20. Работа JavaScript в веб-браузерах.....</b>	<b>568</b>
20.1. Особенности работы JavaScript в веб-браузерах.....	568
20.2. Первая JavaScript-программа.....	569
20.3. Тег <i>&lt;script&gt;</i> .....	570
20.4. Расположение JavaScript-программы.....	571
20.5. Расположение определений функций в HTML-коде.....	574
20.6. Консоль Mozilla Firefox.....	575
20.7. Встроенные диалоговые окна.....	576
20.7.1. Окно с сообщением и кнопкой <i>ОК</i> .....	576
20.7.2. Окно с сообщением и кнопками <i>ОК</i> и <i>Отмена</i> .....	577
20.7.3. Окно с полем ввода и кнопками <i>ОК</i> и <i>Отмена</i> .....	577
20.8. Получение доступа к элементам веб-страницы.....	578
20.9. Создание часов на веб-странице.....	579
20.10. Отладчик Mozilla Firefox.....	580
20.11. JavaScript-библиотеки.....	582
<b>Глава 21. Объектная модель документа.....</b>	<b>584</b>
21.1. Структура объектной модели документа.....	584
21.2. Объект <i>window</i> .....	585
21.3. Работа с фреймами.....	586
21.4. Объект <i>navigator</i> : получение сведений о веб-браузере.....	586

21.5. Объект <i>screen</i> : получение сведений об экране клиентского компьютера.....	587
21.6. Объект <i>location</i> : доступ к интернет-адресу веб-страницы .....	587
21.7. Объект <i>history</i> : история веб-браузера .....	588
21.8. Объект <i>document</i> : работа с веб-страницей.....	588
21.9. Узлы DOM.....	591
21.10. Общие свойства и методы элементов веб-страницы .....	595
21.11. Работа с таблицами стилей .....	596
21.12. Объект <i>selection</i> : работа с выделением .....	599
21.13. Объект <i>Range</i> : работа с фрагментами текста .....	601
21.14. Cookie: хранение данных на компьютере клиента.....	604
21.15. Хранилище .....	607
21.15.1. Сессионное и локальное хранилища .....	608
21.15.2. Работа с хранилищем .....	608
21.15.3. Использование локального хранилища для хранения данных .....	609
21.16. Работа с графическими изображениями .....	610
21.17. Работа с мультимедиа.....	611
21.18. Средства геолокации .....	613
21.18.1. Доступ к средствам геолокации .....	613
21.18.2. Получение данных геолокации .....	613
21.18.3. Обработка нештатных ситуаций .....	614
21.18.4. Задание дополнительных параметров .....	615
21.18.5. Отслеживание местоположения компьютера .....	616
<b>Глава 22. События .....</b>	<b>617</b>
22.1. Назначение обработчиков событий .....	617
22.2. Удаление обработчиков событий.....	619
22.3. Указатель <i>this</i> .....	619
22.4. Объект <i>event</i> .....	620
22.5. Действия по умолчанию и их отмена.....	621
22.6. Всплытие событий.....	623
22.7. Фазы событий .....	625
22.8. События веб-страницы .....	626
22.9. События мыши.....	627
22.10. События клавиатуры .....	630
22.11. События аудио- и видеопроигрывателей.....	631
<b>Глава 23. Взаимодействие с элементами форм .....</b>	<b>633</b>
23.1. Элементы формы .....	633
23.2. Коллекция <i>forms</i> . Доступ к элементу формы из скрипта .....	634
23.3. Работа с формами .....	635
23.4. Работа с элементами формы .....	636
23.4.1. Текстовые поля, поля ввода пароля, подстроки поиска, адреса электронной почты, интернет-адреса и телефона.....	638
23.4.2. Поле ввода числа и регулятор .....	640
23.4.3. Поля ввода даты и времени .....	641
23.4.4. Поле выбора цвета .....	641
23.4.5. Поле выбора файла .....	641
23.4.6. Область редактирования.....	642
23.4.7. Флажок и переключатель .....	644

23.4.8. Список .....	645
23.4.9. Кнопки.....	649
23.5. Расширенная проверка значения, занесенного в поле ввода .....	651

## **Глава 24. Холст: программируемая графика ..... 653**

24.1. Тег <code>&lt;canvas&gt;</code> .....	653
24.2. Создание контекста рисования.....	653
24.3. Заливка.....	654
24.4. Контур.....	654
24.5. Рисование прямоугольников.....	656
24.6. Очистка области холста .....	656
24.7. Вывод текста .....	657
24.8. Вывод графических изображений.....	658
24.9. Рисование сложных фигур.....	659
24.10. Определение вхождения точки в состав контура.....	662
24.11. Задание сложных цветов.....	662
24.11.1. Линейный градиент.....	662
24.11.2. Радиальный градиент.....	663
24.11.3. Заливка текстурой .....	664
24.12. Сохранение и восстановление состояния .....	664
24.13. Преобразования .....	665
24.14. Управление наложением графики.....	665
24.15. Задание уровня прозрачности.....	666
24.16. Создание тени .....	667
24.17. Работа с отдельными пикселями.....	667
24.17.1. Получение массива пикселей.....	667
24.17.2. Создание пустого массива пикселей.....	668
24.17.3. Манипуляция пикселями.....	668
24.17.4. Вывод массива пикселей.....	669

## **Глава 25. AJAX: обмен данными без перезагрузки веб-страницы..... 671**

25.1. Основы технологии AJAX .....	671
25.1.1. Обмен данными с помощью тега <code>&lt;iframe&gt;</code> .....	671
25.1.2. Объект <code>XMLHttpRequest</code> .....	675
25.1.3. Получение данных в текстовом формате.....	680
25.1.4. Получение данных в формате XML.....	685
25.1.5. Получение данных в формате JSON.....	692
25.2. <i>Fetch API</i> .....	698
25.2.1. Функция <code>fetch()</code> .....	698
25.2.2. Классы <code>URL</code> и <code>URLSearchParams</code> .....	701
25.2.3. Объект запроса <code>Request</code> .....	707
25.2.4. Отправка данных при выгрузке веб-страницы .....	711
25.2.5. Прерывание запроса.....	712
25.2.6. Класс <code>FormData</code> .....	712
25.2.7. Отправка файлов .....	715
25.2.8. Объект ответа <code>Response</code> .....	718
25.2.9. Класс <code>Headers</code> .....	721
25.2.10. Кросс-доменные запросы .....	724

<b>Глава 26. Сборка веб-сайтов .....</b>	<b>730</b>
26.1. Файл <i>browserslistrc</i> .....	730
26.2. Сборка CSS- и SCSS-файлов.....	731
26.2.1. Добавление вендорных префиксов .....	731
26.2.2. Сжатие CSS-файлов .....	732
26.2.3. SCSS-транслятор <i>node-sass</i> .....	733
26.3. Пакет <i>npm-run-all</i> : запуск нескольких скриптов.....	736
26.4. Сборщик проектов Webpack .....	737
26.4.1. Создание файла конфигурации .....	737
26.4.2. Сборка JavaScript-файлов .....	738
26.4.3. Несколько точек входа.....	739
26.4.4. Очистка содержимого каталога .....	741
26.4.5. Подключение файлов в формате JSON .....	742
26.4.6. Подключение библиотек из каталога <i>node_modules</i> .....	742
26.4.7. Пакет <i>babel-loader</i> .....	743
26.4.8. Сборка SCSS-файлов .....	744
26.4.9. Подключение изображений .....	746
26.4.10. Копирование файлов.....	747
<b>Заключение.....</b>	<b>749</b>
<b>Приложение. Описание электронного архива.....</b>	<b>750</b>
<b>Предметный указатель .....</b>	<b>751</b>

# Введение

*JavaScript* (также известен как *ECMAScript*) — язык программирования, созданный для написания скриптов — программ, которые встраиваются в веб-страницы и выполняются непосредственно веб-браузером. Самый простой скрипт может в ответ на возникновение какого-либо события (например, наведения мыши на определенный элемент страницы) выводить на странице заданное сообщение (скажем, поясняющий текст). Более сложный скрипт способен прочесть значения, занесенные пользователем в форму, чтобы проверить их на корректность перед отправкой на сервер или даже обработать самостоятельно и тут же вывести результат. И наконец, высший пилотаж JavaScript-программирования — программная подгрузка данных с сервера для формирования на их основе содержимого целой страницы.

JavaScript оказался настолько удачным языком, что остальные программисты стали завидовать разработчикам веб-страниц. И вот в 2009 году появилась первая версия программной платформы Node.js, которая была создана на основе интерпретатора (исполняющей среды) языка JavaScript, встроенного в популярный браузер Google Chrome, и позволяла исполнять JavaScript-программы без веб-браузера. С появлением Node.js появилась возможность писать на JavaScript программы практически любого назначения: инструментальные и системные утилиты, серверы и даже приложения с графическим интерфейсом (правда, для этого понадобятся дополнительные библиотеки).

Предлагаемая вниманию читателей книга рассказывает о языке JavaScript, его применении для написания как программ, работающих под управлением Node.js, так и скриптов, выполняемых веб-браузерами. По сути, это всеобъемлющее руководство для JavaScript-программистов, желающих знать об этом языке все.

## Структура книги

- В *главе 1* описывается редактор программного кода Visual Studio Code, который предназначен именно для программистов и которым мы будем пользоваться на протяжении всего обучения. Редактор корректно работает с кодировкой UTF-8, имеет подсветку синтаксиса языков HTML, CSS, JavaScript и др., выводит различные полезные подсказки и, вообще, всячески помогает разработчику.



- *Глава 2* является вводной. Мы установим платформу Node.js, создадим и запустим в ней первую программу — из командной строки и из редактора Visual Studio Code. Кроме того, вкратце разберемся со структурой программы, научимся выводить результаты ее работы и получать данные от пользователя.
- В *главе 3* мы познакомимся с переменными и типами данных, поддерживаемыми JavaScript.
- В *главе 4* мы рассмотрим операторы, выполняющие различные действия с данными, в частности изучим операторы ветвления и циклы, позволяющие изменять порядок выполнения инструкций в программе.
- *Глава 5* полностью посвящена работе с числами. Мы узнаем, какие числовые типы данных поддерживает JavaScript, научимся применять различные функции, генерировать случайные числа и др.
- *Глава 6* познакомит с массивами и множествами JavaScript. Мы научимся создавать массивы, одно- и многомерные, перебирать элементы массива, сортировать, выполнять поиск значений в массивах и др.
- *Глава 7* полностью посвящена работе со строками.
- В *главе 8* мы рассмотрим регулярные выражения, которые очень пригодятся при выполнении сложного поиска или замены в строках.
- *Глава 9* рассказывает о работе со значениями даты и времени (их еще называют временными метками).
- В *главе 10* мы научимся создавать пользовательские функции — фрагменты произвольного программного кода, которые можно вызывать на выполнение в любом месте программы.
- *Глава 11* познакомит нас с *объектно-ориентированным программированием (ООП)* — еще одним способом многократного использования кода. В отличие от функций, ООП позволяет описать предметы реального мира в виде объектов и организовать связи между этими объектами.
- В *главе 12* мы научимся размещать код в отдельных файлах — *модулях*, выполнять экспорт идентификаторов из одного модуля и импортировать эти идентификаторы для использования в других модулях.
- В *главе 13* мы рассмотрим способы поиска ошибок в программе и научимся отлаживать код в отладчике, встроенном в Visual Studio Code.
- *Глава 14* познакомит нас с асинхронным выполнением программного кода и событиями Node.js.
- В *главе 15* мы рассмотрим класс `Buffer`, который позволит работать с отдельными байтами и выполнять преобразование кодировок.
- *Главы 16, 17 и 18* научат работать с файлами и каталогами, читать и записывать файлы в различных форматах.
- В *главе 19* мы напишем и запустим собственный веб-сервер.

- В *главе 20* мы оставим в покое Node.js и познакомимся с особенностями исполнения JavaScript-скриптов в веб-браузерах.
- *Глава 21* рассматривает объектную модель документа (DOM) веб-браузера, благодаря которой мы сможем получить из скрипта доступ к странице и ее элементам и управлять ими: менять содержимое элементов, добавлять, изменять и удалять элементы страницы, работать со стилями CSS и др.
- *Глава 22* познакомит нас с многочисленными событиями, возникающими в веб-страницах и их элементах, и обработкой этих событий.
- В *главе 23* мы научимся программно обрабатывать данные, занесенные пользователем в элементы формы.
- *Глава 24* познакомит нас с программным рисованием на холсте, реализуемым тегом `<canvas>`.
- В *главе 25* рассматривается технология AJAX, предназначенная для фоновой загрузки произвольных данных с сервера, и два реализующих ее программных инструмента.
- И наконец, в *главе 26* мы рассмотрим готовые программные пакеты для Node.js, которые могут помочь при разработке веб-сайтов. Например, пакет sass транслирует таблицы стилей, написанные на языках SCSS и SASS, в поддерживаемый браузерами CSS. А пакет Webpack может оказаться очень полезным при сборке сложных веб-проектов из разнородных исходных файлов.

Для полного понимания материала книги от читателя потребуются знания HTML и CSS в объеме первых двух глав книги "HTML, JavaScript, PHP и MySQL. Джентльменский набор Web-мастера".

Все листинги из этой книги вы найдете в файле Listings.doc, электронный архив с которым можно загрузить с FTP-сервера издательства "БХВ" по ссылке: <ftp://ftp.bhv.ru/9785977568470.zip> или со страницы книги на сайте <https://bhv.ru> (см. приложение).

## Типографские соглашения

В книге будут часто приводиться различные языковые конструкции, применяемые в программировании. Для наглядности при их написании приняты следующие типографские соглашения (в реальном коде они недействительны):

- HTML-, CSS-, JavaScript- и PHP-код набран моноширинным шрифтом:

```
for (var i = 0; i < 5; i++) {  
    console.log(i);  
}  
console.log('i =', i);
```

- в угловые скобки `<>` заключены наименования различных языковых конструкций. В код программы, разумеется, должны быть подставлены реальные конструкции. Например:

```
const <Имя константы> = <Значение константы>
```

Здесь вместо фраз <Имя константы> и <Значение константы> должны быть подставлены реальные имя и значение;

- в квадратные скобки [] заключены необязательные фрагменты кода:

```
process.exit([<Код завершения>]);
```

Здесь параметр <Код завершения> может указываться, а может и отсутствовать;

- вертикальной чертой | разделены доступные для выбора языковые конструкции, из которых в код можно подставить лишь одну:

```
var|let <Переменная>
```

Здесь можно подставить либо ключевое слово var, либо слово let;

- слишком длинные, не помещающиеся на одной строке книги фрагменты кода разделены на несколько строк, и в местах разрывов поставлены знаки ↵:

```
node-sass -o dist/css/ --include-path node_modules/ ↵  
scss/main.scss
```

Приведенный здесь код хотя и разбит на две строки, но должен быть набран в одну. Символ ↵ при этом следует удалить;

- троеточие . . . помечены фрагменты кода, пропущенные ради сокращения объема текста книги:

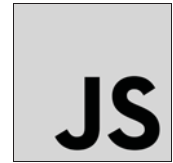
```
class SomeClass {  
    *#method1() { . . . }  
    static *staticMethod2() { . . . }  
    static *#staticMethod3() { . . . }  
}
```

Здесь весь код внутри фигурных скобок пропущен.

### ***НЕ ЗАБЫВАЙТЕ!***

Все приведенные здесь типографские соглашения имеют смысл только в примерах написания языковых конструкций. В реальном программном коде они приведут к возникновению ошибки.

Желаем приятного изучения и надеемся, что книга поможет вам написать сначала простые, а потом и более сложные веб-приложения.



## ГЛАВА 1

# Редактор Visual Studio Code

[https://t.me/it\\_books/2](https://t.me/it_books/2)

Для создания файлов с кодом из книги можно воспользоваться любым текстовым редактором, например Блокнотом. Однако мы будем работать с многобайтовой кодировкой UTF-8, а Блокнот при сохранении в этой кодировке добавляет *метку порядка байтов* (сокращенно *BOM*), которая может стать причиной программных ошибок. Чтобы избежать этого, в качестве редактора кода на протяжении всего обучения воспользуемся программой *Visual Studio Code* (далее *VS Code*). Она также позволяет корректно работать с однобайтовой кодировкой Windows-1251 и имеет подсветку синтаксиса языков HTML, CSS, JavaScript и др.

Для быстрого редактирования файлов советуем дополнительно установить редактор Notepad++. Скачать Notepad++ можно абсолютно бесплатно со страницы <https://notepad-plus-plus.org/>. Из двух вариантов (архив и инсталлятор) советуем выбрать именно инсталлятор, т. к. при установке можно будет указать язык интерфейса программы. Установка Notepad++ предельно проста и в пояснениях не нуждается.

## 1.1. Установка VS Code

Для установки VS Code переходим на сайт <https://code.visualstudio.com/>, прокручиваем страницу в самый низ, в раздел загрузок, и скачиваем установщик **User Installer 64 bit** (для 64-разрядной редакции Windows) или **User Installer 32 bit** (для ее 32-разрядной редакции). Файл установщика будет иметь имя VSCODEUSERSETUP-X64-1.55.2.EXE или VSCodeUserSetup-ia32-1.55.2.exe соответственно.

Запускаем программу установки. На первом шаге принимаем лицензионное соглашение и нажимаем кнопку **Далее** (рис. 1.1). На втором шаге при желании можно изменить каталог, в который будет установлена программа, или оставить каталог по умолчанию (рис. 1.2). Нажимаем кнопку **Далее**. На следующем шаге имеется возможность изменить имя группы, которая будет создана в меню **Пуск** после установки программы, также можно оставить имя группы по умолчанию (рис. 1.3). Нажимаем кнопку **Далее**. На следующем шаге устанавливаем все флажки, кроме **Зарегистрировать Code в качестве редактора для поддерживаемых типов файлов** (рис. 1.4). Для этой цели лучше использовать более простой редактор,

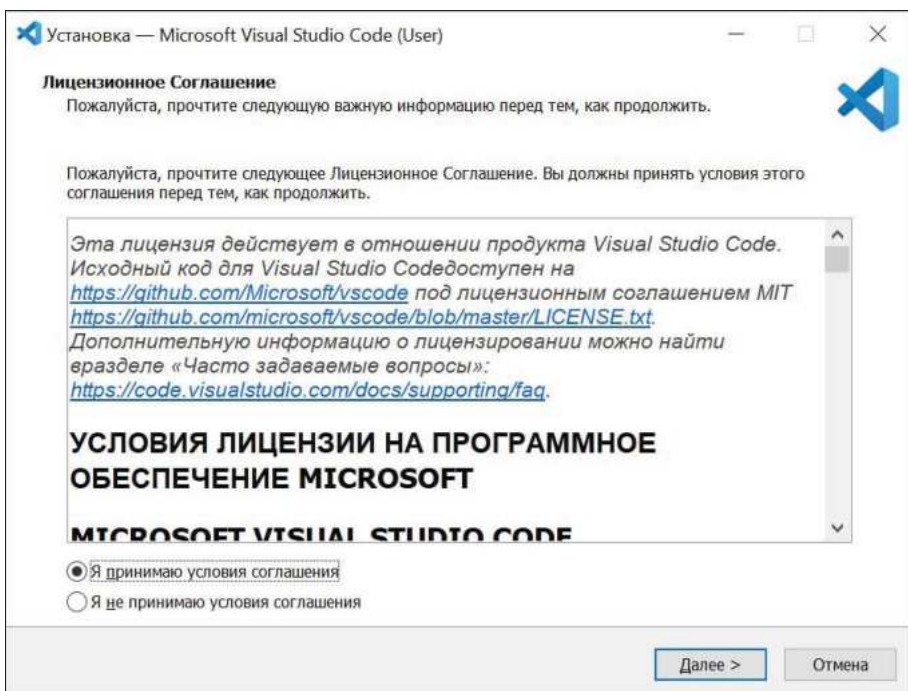


Рис. 1.1. Установка VS Code: шаг 1

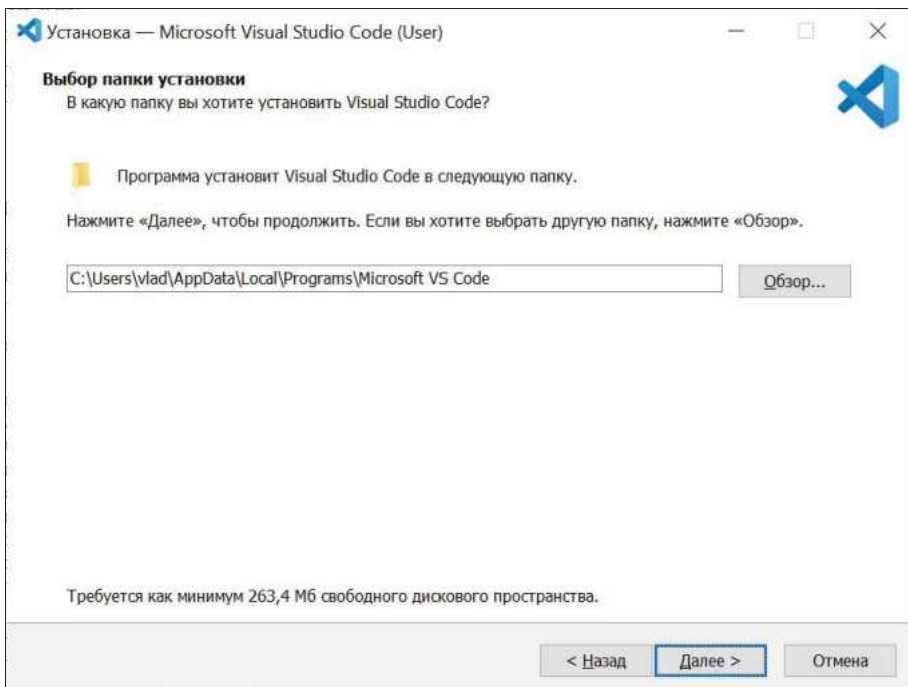


Рис. 1.2. Установка VS Code: шаг 2

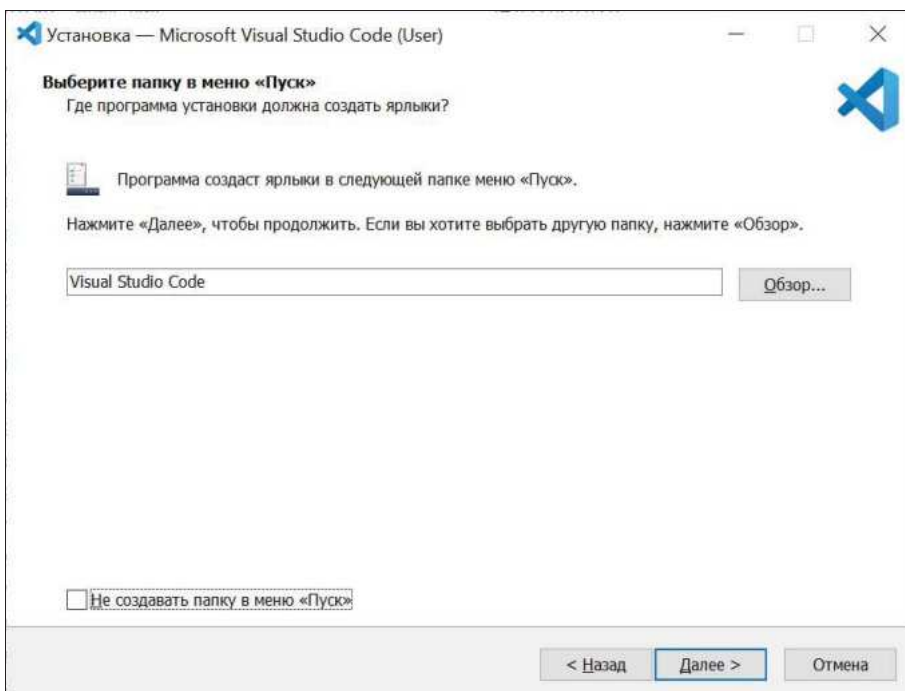


Рис. 1.3. Установка VS Code: шаг 3

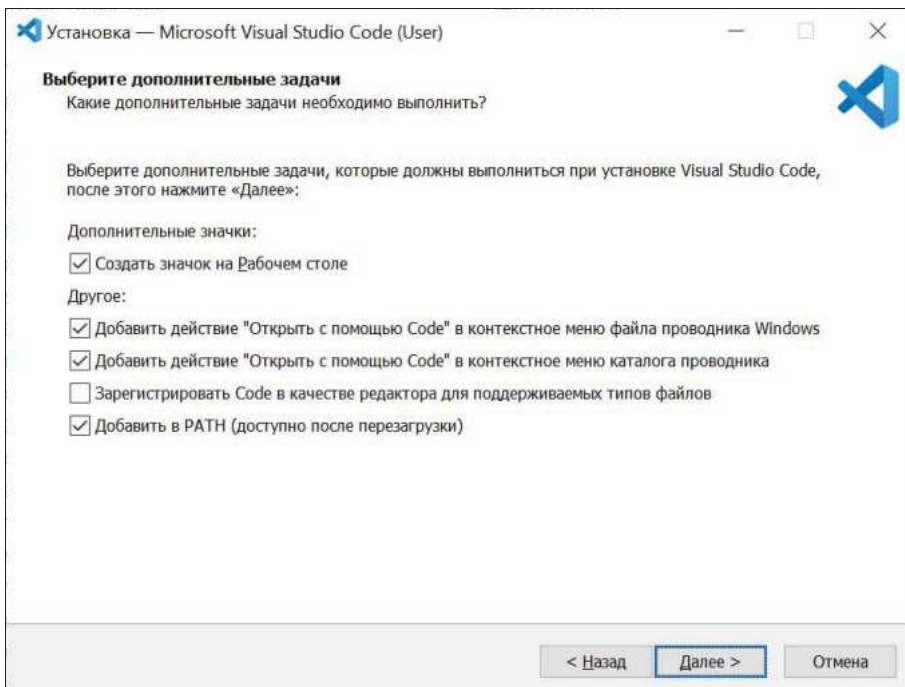


Рис. 1.4. Установка VS Code: шаг 4

наподобие Notepad++. Нажимаем кнопку **Далее**. На очередном шаге проверяем настройки и нажимаем кнопку **Установить** (рис. 1.5) для запуска процесса установки. На последнем шаге, по окончании установки, нажимаем кнопку **Завершить** (рис. 1.6).

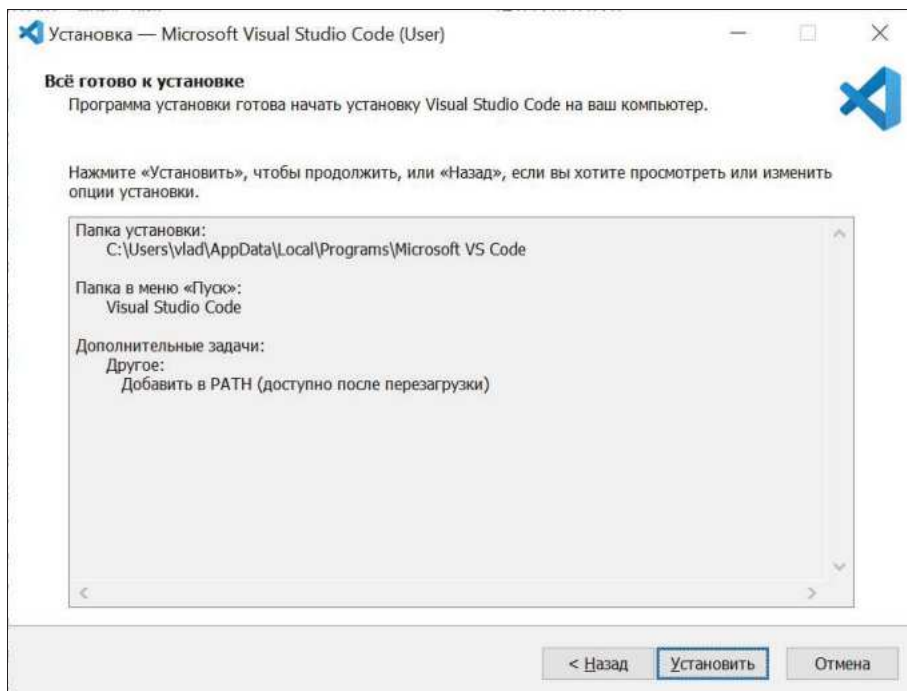


Рис. 1.5. Установка VS Code: шаг 5

По умолчанию, если на шаге 2 (см. рис. 1.2) не был выбран другой путь, редактор устанавливается в каталог `C:\Users\<Имя пользователя>\AppData\Local\Programs\Microsoft VS Code`. Путь до вложенного каталога `bin` записывается в переменную окружения `PATH` для текущего пользователя. Кроме того, на Рабочем столе создается ярлык для запуска редактора. В меню **Пуск** добавляется группа **Visual Studio Code** (имя по умолчанию, может быть изменено на шаге 3, см. рис. 1.3) с одноименным пунктом. В контекстное меню Проводника Windows добавляются пункты для открытия в редакторе файла или целого каталога. Чтобы открыть файл, щелкаем правой кнопкой мыши на значке файла и из контекстного меню выбираем пункт **Открыть с помощью Code**. Аналогичный пункт будет при щелчке правой кнопкой мыши на значке каталога или внутри свободной области каталога в Проводнике Windows.

Если на последнем шаге был установлен флажок **Запустить Visual Studio Code** (см. рис. 1.6), то редактор автоматически запустится. Если после запуска отобразилось черное окно, и долго ничего не меняется, нужно выполнить дополнительную настройку. Закрываем редактор. На Рабочем столе находим ярлык для запуска редактора и щелкаем на нем правой кнопкой мыши. Из контекстного меню выбираем

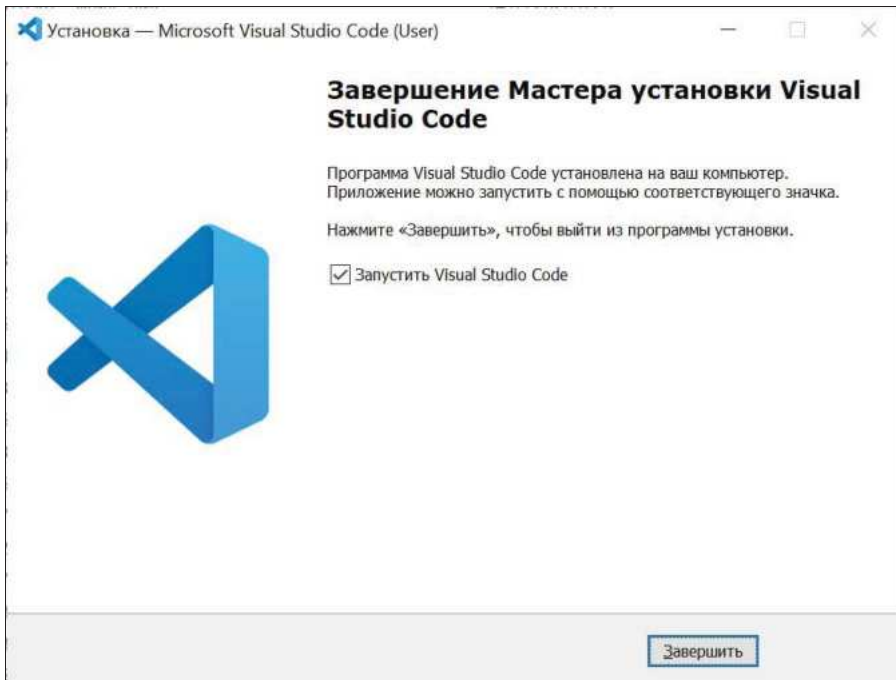


Рис. 1.6. Установка VS Code: шаг 6

пункт **Свойства**. В открывшемся окне на вкладке **Ярлык** в конец содержимого поля **Объект** добавляем пробел и флаг `--disable-gpu`:

```
"C:\Users\<Имя пользователя>\AppData\Local\Programs\Microsoft VS Code\Code.exe" --disable-gpu
```

Вместо `<Имя пользователя>` должно быть указано имя пользователя вашего компьютера. Сохраняем изменения и запускаем редактор двойным щелчком на ярлыке. Если редактор успешно запустился, то мы можем продолжить изучение, но предварительно произведем его настройку.

## 1.2. Настройка редактора

По умолчанию редактор не поддерживает русский язык. Добавить поддержку русского языка можно, установив расширение **Russian Language Pack for Visual Studio Code**. Для этого в меню **View** выбираем пункт **Extensions** или нажимаем комбинацию клавиш `<Shift>+<Ctrl>+<X>`. В результате в левой части окна редактора отобразится панель **Extensions**. В строке поиска вводим имя искомого расширения: `"Russian Language Pack for Visual Studio Code"`. Как только найденное расширение появится в списке, устанавливаем его, нажав небольшую кнопку **Install**, которая находится в правом нижнем углу пункта списка, представляющего это расширение. После установки расширения перезапускаем редактор.

Дополнительно устанавливаем расширение **Auto Complete Tag**, которое автоматически установит еще два расширения: **Auto Rename Tag** и **Auto Close Tag**. Эти рас-



ширения упростят работу с HTML-тегами (например, при переименовании открывающего тега будет автоматически изменяться содержимое закрывающего тега). Поскольку редактор теперь "говорит" по-русски, для вывода панели **Расширения** следует выбрать одноименный пункт меню **Вид**, а для установки расширения — щелкнуть кнопку **Установить** в пункте списка.

### **ВНИМАНИЕ!**

В дальнейшем будет описываться редакция Visual Studio Code, русифицированная с помощью расширения Russian Language Pack for Visual Studio Code.

### **НА ЗАМЕТКУ**

Расширения сохраняются в каталоге `C:\Users\<Имя пользователя>\vscode\extensions`.

Теперь переходим к настройкам. В меню **Файл** выбираем пункт **Настройки | Параметры** или нажимаем комбинацию клавиш `<Ctrl>+<запятая>`. Выполняем настройку следующих параметров в разделе **Пользователь** (вводим их названия в поле для поиска):

- Editor: Font Family** — названия используемых шрифтов через запятую. Рекомендуем значение "Consolas, 'Courier New', monospace".
- Editor: Font Size** — размер шрифта. Можете подобрать удобное значение, например 16 пунктов.
- Editor: Tab Size** — число пробелов в табуляции. Вводим значение 3.
- Editor: Detect Indentation** — сбрасываем флажок, чтобы всегда использовать пробелы вместо символов табуляции.
- Editor: Mouse Wheel Zoom** — если флажок установлен, изменять масштаб можно с помощью колесика мыши, удерживая нажатой клавишу `<Ctrl>`.
- Editor: Word Wrap** — стиль переноса длинных строк. Из списка выбираем пункт **bounded**, включающий автоматический перенос длинных строк.
- Editor: Word Wrap Column** — количество символов в строке. Должно быть указано значение 80 (задано по умолчанию).
- Files: Auto Save** — управляет автоматическим сохранением файла. Рекомендуем из списка выбрать пункт **onFocusChange**, при котором файл будет автоматически сохраняться при потере фокуса ввода.
- Editor: Links** — снимите флажок, если не хотите, чтобы внутри файла выделялись ссылки. С одной стороны, удобно открывать файл переходом по ссылке, с другой — подчеркнутые ссылки мешают.
- Files: Encoding** — из списка выбираем пункт **utf8**, который обозначает кодировку UTF-8 без BOM (метки порядка байтов).
- Files: Eol** — задает символ перевода строк. Из списка выбираем пункт `\n`. Чтобы сменить символ перевода строк для текущего файла, в строке состояния щелкните на значке **CRLF** или **LF**, а затем из списка выбираем нужный пункт.

Точно таким же способом можно задать и другие настройки.

Все изменения настроек в разделе **Пользователь** сохраняются в файле settings.json, который расположен в каталоге C:\Users\<Имя пользователя>\AppData\Roaming\Code\User. Этот файл можно открыть в VS Code, щелкнув расположенную в правой части строки заголовков вкладок кнопку **Открыть параметры**, и изменить вручную.

Изменения настроек в разделе **Рабочая область** записываются в файле settings.json, созданном в каталоге .vscode, вложенном в каталог проекта. Таким образом, можно указывать какие-либо настройки на уровне отдельного проекта.

### 1.3. Смена цветовой темы, тем значков файлов и продукта

По умолчанию используется темная тема оформления. Для смены темы в меню **Файл** выбираем пункт **Настройки | Цветовая тема** или нажимаем комбинацию клавиш <Ctrl>+<K>, а затем комбинацию <Ctrl>+<T>. В итоге отобразится список установленных тем (рис. 1.7). Доступны темы светлые, темные и с высокой контрастностью. Существует также возможность установить тему как расширение или создать свою собственную тему. При выборе пункта из списка внешний вид редактора сразу изменится. Советуем попробовать выбрать темную тему Monokai.

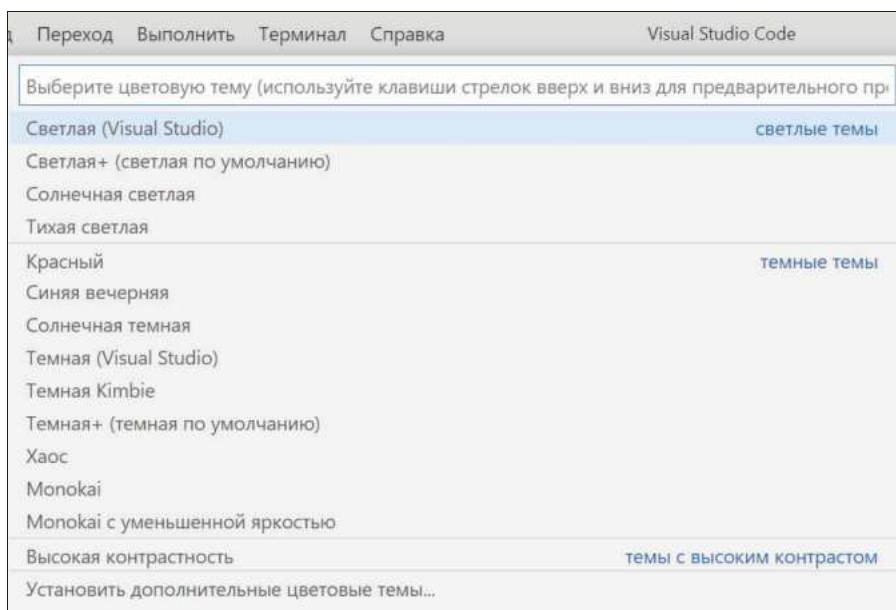


Рис. 1.7. Выбор цветовой темы

Все установленные темы доступны для выбора в настройках редактора. В строке поиска настроек введите **Workbench: Color Theme**, затем выберите тему из раскрывающегося списка (рис. 1.8). Так, при выборе темы "Светлая (Visual Studio)" в файл settings.json будет добавлена следующая строка:

```
"workbench.colorTheme": "Visual Studio Light"
```

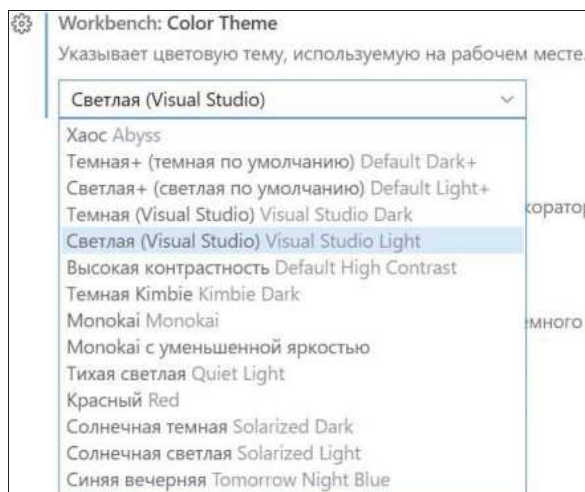


Рис. 1.8. Выбор цветовой темы в настройках редактора

Также имеется возможность изменить значки файлов, выводимые в панели **Проводник** (будет описана далее), задав другую тему значков. Для этого в меню **Файл** выбираем пункт **Настройки | Тема значков файлов**. На экране отобразится список доступных тем значков файлов, аналогичный списку цветовых тем (см. рис. 1.7), в котором можно выбрать нужную тему.

Тему значков файлов можно указать и в настройках редактора. В строке поиска настроек введите **Workbench: Icon Theme** и выберите тему из раскрывающегося списка, аналогичного списку цветовых тем (см. рис. 1.8). Так, при выборе темы "Минимальная (Visual Studio Code)" в файл settings.json будет добавлена следующая строка:

```
"workbench.iconTheme": "vs-minimal"
```

Наконец, можно изменить вид значков, выводящихся на панели действий редактора (будет описана далее), выбрав другую тему. В меню **Файл** выбираем пункт **Настройки | Тема значков продукта**. На экране появится список доступных тем значков продукта, аналогичный списку цветовых тем (см. рис. 1.7), в котором можно выбрать нужную тему. Правда, в этом списке изначально присутствует лишь одна тема, используемая по умолчанию, так что дополнительные темы придется устанавливать в панели **Расширения**.

Тему значков продукта можно задать и в настройках редактора, выполнив поиск по фразе **Workbench: Product Icon Theme** и выбрав тему из раскрывающегося списка. Так, при выборе темы "Fluent Icons" в файл settings.json будет добавлена следующая строка:

```
"workbench.productIconTheme": "fluent-icons"
```

### **ВНИМАНИЕ!**

В книге будут приведены скриншоты, сделанные с использованием цветовой темы "Светлая (Visual Studio)", стандартной темы значков файлов "Seti (Visual Studio Code)" и стандартной темы значков продукта.

## 1.4. Структура окна редактора

При запуске редактора в центральной части окна отображается вкладка с приветствием (рис. 1.9). На ней расположены ссылки, с помощью которых можно создать файл, открыть каталог или последний проект, получить справочную информацию и т. д. Доступны также советы по установке расширений и настройке редактора. В нижней части вкладки расположен флажок **Отображать страницу приветствия при запуске**. Нам эта вкладка не понадобится, поэтому советуем сбросить флажок. В файл `settings.json` будет добавлена следующая строка:

```
"workbench.startupEditor": "newUntitledFile"
```

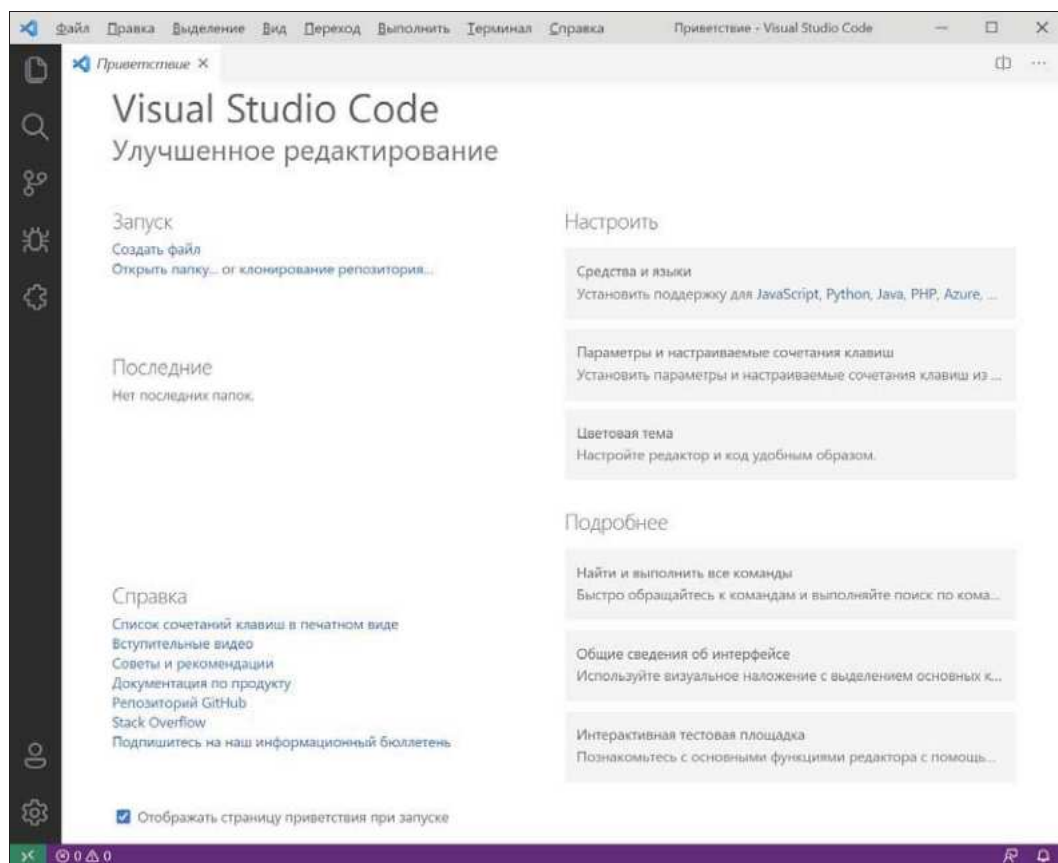


Рис. 1.9. Редактор Visual Studio Code

Значение `newUntitledFile` означает, что при запуске будет создаваться новый файл. Нас это также не устраивает. Переходим в настройки редактора и в строке поиска настроек вводим **Workbench: Startup Editor**. Из списка выбираем пункт **none**. В файл `settings.json` будет добавлена следующая строка вместо предыдущей:

```
"workbench.startupEditor": "none"
```

В результате при запуске редактор не будет содержать никаких вкладок. Чтобы вывести вкладку с приветствием, в меню **Справка** выбираем пункт **Приветствие** или удаляем из файла settings.json строку с упомянутой ранее настройкой.

### 1.4.1. Главное меню

В верхней части окна расположена строка меню, имя файла, открытого в активной вкладке, и название редактора, а также стандартные для окон кнопки **Свернуть**, **Развернуть** и **Закрывать**. Кратко рассмотрим структуру главного меню.

- Меню **Файл** — команды для работы с файлами, каталогами и проектами (рис. 1.10). При выборе пункта **Настройки** отобразится меню с командами, позволяющими выполнить настройку редактора (рис. 1.11).

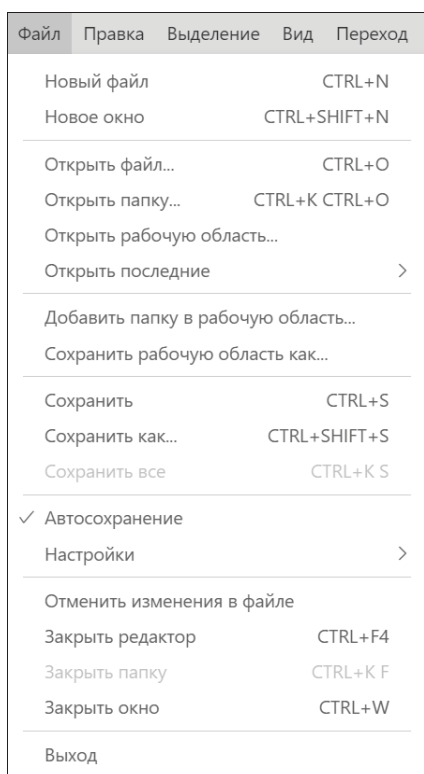


Рис. 1.10. Меню **Файл**

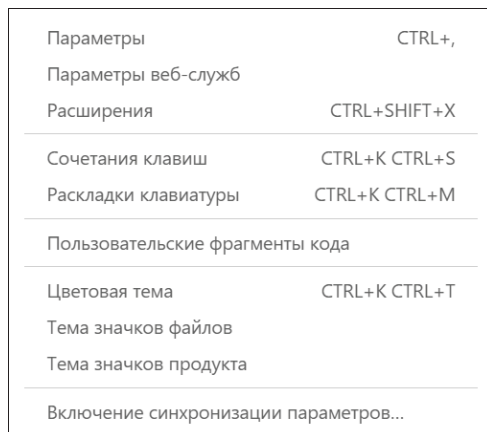


Рис. 1.11. Меню **Файл**, пункт **Настройки**

- Меню **Правка** — команды для редактирования кода (рис. 1.12).
- Меню **Выделение** — команды для работы с выделением и курсорами (рис. 1.13).
- Меню **Вид** — команды для настройки внешнего вида редактора (рис. 1.14). При выборе пункта **Внешний вид** появится меню с командами для управления панелями (рис. 1.15), а при выборе пункта **Макет редактора** — меню с командами, позволяющими разделить содержимое окна редактора на несколько областей (рис. 1.16).



**- Lituz.com**

**Elektron kitoblar**

**To'liq qismini Shu tugmani  
bosish orqali sotib oling!**