

ВТОРОЕ ИЗДАНИЕ

---

# Эффективный TypeScript

*83 способа улучшить код*

*Дэн Вандеркам*

**SPRINT**  
book

2025

---

# Краткое содержание

<b>Глава 1.</b> Знакомство с TypeScript .....	28
<b>Глава 2.</b> Система типов в TypeScript .....	55
<b>Глава 3.</b> Вывод типов и анализ потока управления.....	118
<b>Глава 4.</b> Проектирование типов .....	165
<b>Глава 5.</b> Ненадежность и тип any .....	214
<b>Глава 6.</b> Обобщения (дженерики) и программирование на уровне типов.....	244
<b>Глава 7.</b> Рецепты TypeScript .....	291
<b>Глава 8.</b> Объявления типов и @types.....	314
<b>Глава 9.</b> Написание и запуск кода .....	336
<b>Глава 10.</b> Модернизация и миграция.....	375
<b>Приложение А.</b> Соответствие правил в двух изданиях книги .....	395

---

# Оглавление

<b>Отзывы .....</b>	<b>11</b>
<b>Введение ко второму изданию .....</b>	<b>14</b>
Для кого эта книга .....	15
Почему я написал эту книгу.....	15
Структура книги .....	16
Условные обозначения в примерах кода.....	16
Типографские соглашения .....	18
Использование примеров кода.....	19
<b>Благодарности .....</b>	<b>20</b>
<b>Введение к первому изданию (2019).....</b>	<b>23</b>
<b>Благодарности к первому изданию.....</b>	<b>24</b>
<b>От издательства .....</b>	<b>27</b>
<b>Глава 1. Знакомство с TypeScript .....</b>	<b>28</b>
Правило 1. Разберитесь, как TypeScript связан с JavaScript.....	28
Правило 2. Выбирайте нужные опции в TypeScript .....	34
Правило 3. Помните, что генерация кода не зависит от типов .....	38
Правило 4. Привыкайте к структурной типизации.....	45
Правило 5. Ограничьте применение типа any .....	50
<b>Глава 2. Система типов в TypeScript.....</b>	<b>55</b>
Правило 6. Используйте редактор для получения информации о системе типов.....	55
Правило 7. Воспринимайте типы как множества значений .....	60
Правило 8. Правильно выражайте принадлежность сущностей к пространству типов или пространству значений.....	68

Правило 9. Отдавайте предпочтение аннотациям типов, а не утверждениям .....	73
Правило 10. Избегайте объектных типов-обертток (String, Number, Boolean, Symbol, BigInt).....	78
Правило 11. Отличайте проверку лишних свойств от проверки типов.....	81
Правило 12. По возможности применяйте типы ко всему функциональному выражению .....	85
Правило 13. Знайте различия между type и interface.....	88
Правило 14. Используйте readonly для предотвращения ошибок, связанных с изменяемостью.....	95
Правило 15. Используйте операции типов и обобщения для предотвращения повторов.....	100
Правило 16. Используйте более точные альтернативы для сигнатур индексов.....	110
Правило 17. Избегайте использования числовых сигнатур индексов.....	114
<b>Глава 3. Вывод типов и анализ потока управления.....</b>	<b>118</b>
Правило 18. Не засоряйте код лишними аннотациями типов.....	119
Правило 19. Используйте разные переменные для разных типов .....	126
Правило 20. Разберитесь в процессе расширения типов .....	128
Правило 21. Создавайте объекты целиком.....	133
Правило 22. Освойте сужение типов.....	136
Правило 23. Будьте последовательны с использованием псевдонимов .....	140
Правило 24. Используйте контекст при выводе типов .....	144
Правило 25. Разберитесь с эволюционирующими типами.....	148
Правило 26. Используйте функциональные конструкции и библиотеки для улучшения работы с типами .....	152
Правило 27. Для асинхронного кода используйте аsync-функции вместо обратных вызовов.....	156
Правило 28. Используйте классы и каррирование для создания новых точек вывода типов.....	161
<b>Глава 4. Проектирование типов .....</b>	<b>165</b>
Правило 29. Используйте типы, всегда представляющие допустимые состояния .....	165
Правило 30. Будьте консервативны в том, что отправляете, и либеральны в том, что принимаете .....	171

Правило 31. Не повторяйте информацию типа в документации .....	175
Правило 32: Избегайте включения <code>null</code> или <code>undefined</code> в псевдонимы типов ...	177
Правило 33. Выносите <code>null</code> на периферию типов .....	178
Правило 34. Отдавайте предпочтение объединениям интерфейсов, а не интерфейсам с объединениями.....	182
Правило 35. Вместо <code>string</code> используйте более узкие типы .....	186
Правило 36. Используйте отдельный тип для специальных значений .....	190
Правило 37. Ограничьте использование опциональных свойств.....	193
Правило 38. Избегайте повторяющихся параметров одного типа .....	197
Правило 39. Стремитесь к унификации типов, а не к моделированию различий.....	199
Правило 40. Лучше сделать тип неточным, чем недостоверным .....	201
Правило 41. Назначайте имена типов на языке предметной области задачи .....	206
Правило 42. Генерируйте типы на основе API и спецификаций, а не данных.....	209
<b>Глава 5. Ненадежность и тип <code>any</code> .....</b>	<b>214</b>
Правило 43. Используйте максимально узкий диапазон для типов <code>any</code> .....	214
Правило 44. Используйте более точные варианты <code>any</code> .....	217
Правило 45. Скрывайте небезопасные утверждения типов в хорошо типизированных функциях.....	220
Правило 46. Используйте <code>unknown</code> вместо <code>any</code> для значений с неизвестным типом.....	224
Правило 47. Используйте типобезопасные решения вместо динамической модификации.....	228
Правило 48. Избегайте ловушек надежности.....	232
Правило 49. Отслеживайте зону охвата типов для предотвращения регрессии в типобезопасности.....	241
<b>Глава 6. Обобщения (дженерики) и программирование на уровне типов...244</b>	
Правило 50. Рассматривайте обобщения (дженерики) как функции между типами.....	245
Правило 51. Избегайте лишних параметров типа.....	251
Правило 52. Лучше условные типы, чем перегруженные сигнатуры .....	256
Правило 53. Научитесь управлять распределением объединений с условными типами.....	260

Правило 54. Используйте шаблонные литеральные типы для моделирования DSL и отношений между строками .....	265
Правило 55. Пишите тесты для своих типов .....	270
Правило 56. Обращайте внимание на отображение типов .....	279
Правило 57. Отдавайте предпочтение обобщенным типам с хвостовой рекурсией .....	283
Правило 58. Рассмотрите кодогенерацию в качестве альтернативы для сложных типов .....	287
<b>Глава 7. Рецепты TypeScript.....</b>	<b>291</b>
Правило 59. Используйте типы <code>never</code> для выполнения проверки полноты .....	291
Правило 60. Разберитесь, как итерировать по объектам.....	297
Правило 61. Используйте типы записей для поддержания синхронизации значений.....	301
Правило 62. Используйте остаточные параметры и типы кортежа для моделирования функций с переменным числом аргументов.....	304
Правило 63. Используйте опциональные свойства <code>never</code> для моделирования исключающего ИЛИ .....	307
Правило 64. Рассмотрите применение маркировок для номинальной типизации .....	309
<b>Глава 8. Объявления типов и <code>@types</code> .....</b>	<b>314</b>
Правило 65. Размещайте TypeScript и <code>@types</code> в <code>devDependencies</code> .....	314
Правило 66. Проверяйте совместимость трех версий, задействованных в объявлениях типов .....	317
Правило 67. Экпортируйте все типы, появляющиеся в открытых API .....	321
Правило 68. Используйте TSDoc для комментариев к API.....	322
Правило 69. Определяйте тип <code>this</code> в обратных вызовах, если он является частью API .....	325
Правило 70. Зеркалируйте типы для разрыва зависимостей.....	329
Правило 71. Используйте аугментацию модулей для улучшения типов .....	332
<b>Глава 9. Написание и запуск кода .....</b>	<b>336</b>
Правило 72. Отдавайте предпочтение функциональности ECMAScript перед TypeScript.....	336
Правило 73. Используйте Source Map для отладки TypeScript .....	344
Правило 74. Научитесь реконструировать типы на стадии выполнения .....	350

Правило 75. Разберитесь с иерархией DOM.....	355
Правило 76. Создайте точную модель среды выполнения.....	360
Правило 77. Понимайте связь между проверкой типов и юнит-тестированием.....	363
Правило 78. Уделяйте внимание быстродействию компилятора.....	367
<b>Глава 10. Модернизация и миграция .....</b>	<b>375</b>
Правило 79. Пишите современный JavaScript.....	376
Правило 80. Используйте @ts-check и JSDoc для экспериментов в TypeScript .....	380
Правило 81. Используйте allowJs для совмещения TypeScript и JavaScript.....	385
Правило 82. Конвертируйте модуль за модулем вверх по графу зависимостей .....	387
Правило 83. Не считайте миграцию завершенной, пока не включите noImplicitAny .....	393
<b>Приложение А. Соответствие правил в двух изданиях книги .....</b>	<b>395</b>
<b>Об авторе .....</b>	<b>398</b>
<b>Иллюстрация на обложке.....</b>	<b>399</b>