

# МАГИЯ CSS

**САМОЕ ГЛАВНОЕ**

**В КРАТКОМ  
ИЗЛОЖЕНИИ**

МОСКВА  
ИЗДАТЕЛЬСТВО АСТ

УДК 004.4  
ББК 32.973-018.1  
К49

**Климович, Александр Дмитриевич.**

К49 Магия CSS — самое главное в кратком изложении / А. Д. Климович. — Москва : Издательство АСТ, 2026. — 144 с. : ил. — (Учимся программировать). ISBN 978-5-17-174758-9.

Перед вами практическое руководство по современным техникам веб-стилизации, в котором язык CSS раскрывается как инструмент точной настройки визуальной среды — от фундаментальных принципов до самых продвинутых возможностей.

Книга охватывает весь путь освоения CSS: от блочной модели — основы компоновки элементов на странице — до сложных анимаций, эффектов и приемов, позволяющих создавать выразительные и в то же время производительные интерфейсы. В отдельных главах рассматриваются современные подходы к сеткам и табличной верстке, работа с цветом, типографикой и переходами. Особое внимание уделено архитектурным аспектам: управлению специфичностью и слоями, поддержке адаптивности с помощью контейнерных запросов, применению суб-гридов и логических свойств. Все это сопровождается практическими примерами и стратегиями оптимизации, что делает книгу полезной как для проектирования модульных компонентов, так и для масштабных систем.

«Магия CSS» предназначена для широкого круга специалистов — и для тех, кто только начинает знакомство с веб-стилизацией, и для опытных разработчиков, стремящихся углубить свои знания и использовать CSS на профессиональном уровне.

**УДК 004.4**  
**ББК 32.973-018.1**

**ISBN 978-5-17-174758-9**

© Оформление. ООО «Интеджер», 2025  
© ООО «Издательство АСТ», 2026

# Содержание

Введение.....	9
<b>Глава 1. Блочная модель CSS: основа визуального отображения .....</b>	<b>11</b>
Теория и терминология.....	11
Исторический контекст.....	12
Быстрый пример .....	12
Основные техники и паттерны.....	13
Универсальное применение border-box .....	13
Гибкие формы ввода .....	14
Компонентная система с предсказуемыми размерами .....	15
Современные CSS-возможности.....	16
Логические свойства и блочная модель.....	16
Контейнерные запросы (Container Queries) и блочная модель .....	17
Cascade Layers и управление специфичностью .....	17
Современные CSS-функции для размеров .....	18
Селекторы :has() и условная стилизация.....	18
Производительность .....	19
Оптимизация перерасчета макета .....	19
Использование CSS Containment.....	19
Стратегии для will-change.....	19
Доступность (A11y) .....	20
Адаптивная типографика и блочная модель .....	20
Видимые области фокуса .....	21
Поддержка направления текста.....	21
Отладка .....	22
Инструменты разработчика.....	22
Методы диагностики размеров.....	22
Диагностика специфичности и каскада.....	23
Частые ошибки и антипаттерны.....	23
Специфичность и !important.....	23
Смешивание единиц измерения без border-box.....	24
Неправильное использование margin для внутренних отступов .....	24
Z-index без контекста наложения .....	25
Архитектура и масштабирование.....	26
Совместимость браузеров.....	29
Стратегии прогрессивного улучшения .....	29
Практические упражнения.....	30
Упражнение 1. Адаптивная карточная система .....	30
Упражнение 2. Универсальная система форм .....	31

Упражнение 3. Производительный макет списка .....	31
Заключение .....	32
Углубленное чтение .....	32
<b>Глава 2. Продвинутое техники компоновки.....</b>	<b>34</b>
Теория и терминология.....	34
Layout vs Paint vs Composite.....	35
Быстрый пример.....	35
Основные техники и паттерны.....	36
Типы отображения (Display Types) .....	36
Современные возможности CSS .....	38
Cascade Layers (Экспериментальная функция) .....	38
Container Queries (Экспериментальная функция) .....	38
CSS Subgrid (Экспериментальная функция) .....	38
Селекторы :has(), :is(), :where() .....	39
CSS Nesting (Экспериментальная функция).....	39
Логические свойства.....	39
Производительность .....	40
Стоимость селекторов .....	40
Оптимизация Reflow/Repaint/Composite.....	40
Использование will-change .....	40
Доступность.....	41
Стили фокуса.....	41
Контрастность и пользовательские предпочтения.....	41
Логические свойства для RTL .....	42
Отладка .....	42
Техники DevTools.....	42
Диагностика специфичности .....	43
Типичные ошибки и антипаттерны .....	43
Проблемы специфичности .....	43
Проблемы z-index .....	44
Схлопывание margin.....	44
Архитектура и масштабирование.....	45
Методологии .....	45
Cascade Layers для архитектуры.....	45
Система дизайн-токенов .....	46
Совместимость браузеров.....	47
Стратегии fallback.....	47
Практические упражнения.....	48
Упражнение 1. Адаптивная карточная сетка .....	48
Упражнение 2. Сложная компоновка с Subgrid .....	48
Упражнение 3. Оптимизированная анимация .....	48
Заклучение.....	49
Углубленное чтение .....	49
<b>Глава 3. Мастерство табличной верстки и продвинутое техники .....</b>	<b>51</b>
Теория и терминология.....	51
Быстрый пример .....	52
Основные техники и паттерны.....	53

Контроль пропорций с table-layout: auto .....	53
Точное позиционирование с table-layout: fixed .....	55
Гибридный подход с Container Queries .....	56
Современные возможности CSS .....	58
Каскадные слои для организации табличных стилей .....	58
Селектор :has() для контекстной стилизации .....	59
Логические свойства для интернационализации .....	59
Современные функции для динамических размеров .....	60
Производительность .....	61
Стоимость селекторов в табличном контексте .....	61
Создание слоев компоновки для анимаций .....	62
Оптимизация перерисовки .....	62
Доступность (a11y) .....	62
Семантическая разметка и навигация .....	62
Стили фокуса и навигации .....	63
Адаптивная типографика .....	64
Отладка .....	65
Техники визуальной отладки .....	65
Диагностика специфичности и каскада .....	65
Методология бинарного поиска проблем .....	66
Распространенные ошибки и антипаттерны .....	66
Специфичность и злоупотребление !important .....	66
Утечки z-index и нежелательные контексты наложения .....	67
Схлопывание margin и негативные отступы как костыли .....	68
Переиспользование табличных хаков при доступности современных решений .....	68
Архитектура и масштабирование .....	69
Методологии организации табличных стилей .....	69
Design tokens и кастомные свойства для тематизации .....	71
Паттерны композиции и переиспользования .....	72
Совместимость браузеров .....	73
Стратегии fallback и прогрессивного улучшения .....	74
Практические упражнения .....	75
Упражнение 1. Адаптивная таблица данных (средняя сложность) .....	75
Упражнение 2. Сложная финансовая таблица (высокая сложность) .....	75
Упражнение 3. Табличный дашборд (высокая сложность) .....	76
Заключение .....	77
Углубленное чтение .....	78
<b>Глава 4. Цвет и современные техники стилизации .....</b>	<b>79</b>
Цветовые пространства и рендеринг .....	79
Этапы рендеринга цвета .....	79
Быстрый пример .....	79
Основные техники и паттерны .....	80
Управление прозрачностью без цветовых конфликтов .....	80
Семантическая система цветов .....	81
Контрастные пары и доступность .....	82
Современные возможности CSS .....	83
Каскадные слои для цветовых систем .....	83

Контейнерные запросы для адаптивных цветовых схем .....	84
Новые цветовые функции и пространства .....	85
Логические свойства для интернационализации .....	86
Производительность .....	86
Оптимизация селекторов и инвалидация .....	86
GPU-компоЗИТИНГ и стекинг .....	87
Оптимизация градиентов и фильтров .....	88
Доступность .....	89
Контраст и цветовые предпочтения пользователя .....	89
Цветонезависимая индикация .....	90
Отладка .....	91
DevTools и инструменты диагностики .....	91
Диагностика проблем специфичности .....	92
Распространенные ошибки и антипаттерны .....	93
Проблемы специфичности .....	93
Злоупотребление !important .....	93
Архитектура и масштабирование .....	94
Организация цветовых токенов .....	94
Темизация и области видимости .....	95
Интеграция с методологиями CSS .....	96
Совместимость браузеров .....	97
Стратегии прогрессивного улучшения .....	98
Практические упражнения .....	98
Упражнение 1. Адаптивная цветовая система .....	98
Упражнение 2. Система статусных индикаторов .....	99
Упражнение 3. Производительная анимация цветов .....	99
Заключение .....	100
Углубленное чтение .....	100
<b>Глава 5. CSS-типографика: продвинутые техники и современные возможности .....</b>	<b>102</b>
Теория и терминология .....	102
Этапы рендеринга типографики .....	102
Быстрый пример .....	103
Основные техники и паттерны .....	104
Адаптивное масштабирование типографики .....	104
Оптические корректировки межбуквенных интервалов .....	105
Контрастная типографическая иерархия .....	105
Современные возможности CSS .....	106
Каскадные слои для типографических систем .....	106
Контейнерные запросы для адаптивной типографики .....	107
Продвинутые селекторы и состояния .....	107
Новые единицы измерения области просмотра .....	108
Производительность .....	109
Оптимизация селекторов .....	109
Управление перерисовками .....	109
Оптимизация загрузки шрифтов .....	110
Доступность .....	110
Фокусные состояния .....	110

Адаптивная типографика для различных режимов письма .....	110
Отладка .....	111
Визуализация типографических метрик .....	111
Диагностика каскада и специфичности .....	111
Распространенные ошибки и антипаттерны .....	112
Злоупотребление отрицательными отступами .....	112
Система дизайн-токенов .....	113
Модульная архитектура (BEM + современный CSS) .....	113
Совместимость браузеров .....	114
Практические упражнения .....	114
Упражнение 1. Адаптивная типографическая система .....	114
Упражнение 2. Компонентная типографика .....	115
Упражнение 3. Производительная анимация текста .....	115
Заключение .....	115
Долгосрочная перспектива .....	116
Углубленное чтение .....	116
<b>Глава 6. CSS-переходы и магия анимации .....</b>	<b>118</b>
Теория и терминология .....	118
Быстрый пример .....	119
Основные техники и паттерны .....	120
Базовый синтаксис и свойства .....	120
Техника поэтапной задержки .....	120
Паттерн accordion без JavaScript .....	122
3D-трансформации с переходами .....	123
Современные CSS-возможности .....	124
Интеграция с каскадными слоями .....	124
Контейнерные запросы и адаптивные переходы .....	125
Современные селекторы с переходами .....	126
CSS Nesting с переходами .....	127
Производительность .....	128
Оптимизация селекторов и инвалидация .....	128
Управление перерасчетом макета .....	128
Композитные слои и GPU-ускорение .....	129
Управление производительностью через containment .....	129
Доступность .....	130
Доступные фокусные состояния .....	130
Контрастность и цветовые переходы .....	131
Отладка .....	131
Визуализация слоев композитинга .....	131
Диагностика производительности .....	132
Распространенные ошибки и антипаттерны .....	132
Гонка специфичности .....	132
Утечки z-index и непредвиденные контексты наложения .....	133
Злоупотребление хаками вместо современных решений .....	134
Архитектура и масштабирование .....	135
Методологии и организация переходов .....	135
Дизайн-токены и темизация .....	136

---

Композиция и повторное использование .....	137
Совместимость браузеров.....	138
Стратегии прогрессивного улучшения .....	139
Практические упражнения.....	139
Упражнение 1. Адаптивная карточка товара .....	139
Упражнение 2. Accordion с плавными переходами высоты.....	140
Упражнение 3. Система уведомлений с анимацией .....	140
Заключение.....	141
Стратегии безопасного внедрения в крупных проектах .....	141
Углубленное чтение .....	141
<b>Заключение .....</b>	<b>143</b>

# Введение

Каскадные таблицы стилей (CSS) в современной веб-разработке представляют собой не просто средство декоративного оформления, а сложную декларативную систему, способную решать архитектурные задачи пользовательского интерфейса, которые ранее требовали императивного программирования. За двадцать пять лет эволюции CSS трансформировался из простого языка стилизации в мощную платформу для создания интерактивных, адаптивных и производительных веб-приложений.

Термин «магия CSS» в контексте данной книги относится к техникам, в основе которых лежат глубинные механизмы спецификации для достижения результатов, кажущихся невозможными при поверхностном понимании языка. Эти техники основаны на фундаментальных принципах каскадирования, наследования, контекстов форматирования и алгоритмов компоновки, закрепленных в рекомендациях W3C.

Современная веб-платформа предоставляет разработчикам беспрецедентные возможности: контейнерные запросы (Container Queries) для создания истинно модульных компонентов, каскадные слои (@layer) для управления специфичностью на архитектурном уровне, логические свойства для поддержки многонаправленного письма, субгриды для точного контроля вложенных сеток. Однако использование этих возможностей требует понимания не только синтаксиса, но и внутренних процессов браузерного движка.

Данная книга предназначена для ведущих инженеров среднего и старшего звена, которые стремятся выйти за рамки фреймворк-специфичных решений и овладеть фундаментальными принципами платформы. Материал ориентирован на практическое применение в производственных системах, где критически важны производительность, поддерживаемость кода и предсказуемость поведения интерфейсов.

В каждой главе теоретические основы сочетаются с практическими примерами, анализом производительности и стратегиями отладки. От блочной модели CSS — фундамента всех алгоритмов компоновки — до сложных композиционных техник с переходами и анимациями книга обеспечивает систематическое усвоение возможностей современного CSS.

Понимание этих принципов позволяет создавать решения, которые работают стабильно в различных браузерных средах, масштабируются в крупных кодовых базах и адаптируются к изменяющимся требованиям без архитектурных переработок. Эти знания становятся особенно ценными в эпоху компонентно-ориентированной разработки, где каждый интерфейсный элемент должен функционировать независимо, но согласованно в рамках общей системы дизайна.

Материал книги основан на актуальных спецификациях W3C, тестировании в современных браузерных движках и опыте применения техник в производственных проектах различного масштаба. Каждый пример протестирован на совместимость и производительность, что обеспечивает готовность решений к использованию в реальных условиях.

# Глава 1. Блочная модель CSS: основа визуального отображения

Блочная модель CSS (CSS Box Model) представляет собой фундаментальную концепцию, определяющую, как браузер вычисляет размеры и позиционирование элементов на веб-странице. Каждый элемент в документе генерирует прямоугольный блок, состоящий из областей содержимого, внутренних отступов (padding), границ (border) и внешних отступов (margin). Понимание принципов работы блочной модели критически важно для создания предсказуемых и масштабируемых пользовательских интерфейсов.

В этой главе мы рассмотрим не только базовые концепции блочной модели, но и современные техники управления размерами элементов, включая свойство `box-sizing`, взаимодействие с новыми CSS-возможностями, такими как контейнерные запросы (Container Queries) и логические свойства, а также стратегии оптимизации производительности и отладки. Материал ориентирован на разработчиков, которые стремятся к глубокому пониманию механизмов рендеринга браузера и созданию профессиональных CSS-архитектур.

Техники, описанные в данной главе, применимы в создании адаптивных компонентов, системах дизайна, сложных макетах с использованием CSS Grid и Flexbox, а также при оптимизации производительности рендеринга в крупных веб-приложениях.

## Теория и терминология

**Блочная модель** — спецификация CSS, определяющая, как браузер вычисляет общий размер элемента на основе четырех прямоугольных областей: содержимого (content), внутренних отступов (padding), границ (border) и внешних отступов (margin).

**Каскад (Cascade)** — алгоритм определения приоритета CSS-правил на основе специфичности, порядка объявления и источника стилей.

**Специфичность (Specificity)** — числовое значение, определяющее вес CSS-селектора при разрешении конфликтов стилей.

**Контекст форматирования (Formatting Context)** — область документа, в которой дочерние элементы размещаются согласно определенному набору правил. Основные типы: блочный (Block Formatting Context, BFC), строчный (Inline Formatting Context) и flex/grid контексты.

**Контекст наложения (Stacking Context)** — трехмерное представление HTML-элементов вдоль оси Z, определяющее порядок отрисовки элементов на экране.

**Вычисленное значение (Computed Value)** — значение CSS-свойства после применения каскада и наследования, но до окончательного разрешения в используемое значение.

**Используемое значение (Used Value)** — окончательное значение CSS-свойства после всех вычислений, включая разрешение процентных значений и единиц измерения.

## Исторический контекст

Блочная модель была стандартизирована в CSS1 (1996) и существенно уточнена в CSS 2.1. Введение свойства `box-sizing` в CSS3 решило многие проблемы интуитивности размеров элементов. Современные возможности, такие как логические свойства (CSS Logical Properties) и контейнерные запросы (Container Queries), расширяют блочную модель для поддержки многонаправленного письма и контейнерно-ориентированного дизайна.

## Быстрый пример

Рассмотрим классический пример, демонстрирующий разницу между `content-box` и `border-box`:

```
<div class="container">
  <div class="box content-box">Content Box</div>
  <div class="box border-box">Border Box</div>
</div>

.container {
  display: flex;
  gap: 20px;
  max-width: 600px;
}

.box {
  width: 200px;
  height: 100px;
  padding: 20px;
  border: 5px solid #333;
  background-color: #f0f0f0;
  font-family: system-ui;
}

.content-box {
  box-sizing: content-box; /* значение по умолчанию */
}
```

```
.border-box {
  box-sizing: border-box;
}
.container {
  display: flex;
  gap: 20px;
  max-width: 600px;
}
```

### ***Почему это работает***

При `box-sizing: content-box` (значение по умолчанию) браузер интерпретирует `width` и `height` как размеры области содержимого. Общий размер элемента вычисляется как:

Общая ширина = `width + padding-left + padding-right + border-left-width + border-right-width`.

Для нашего примера:  $200\text{px} + 20\text{px} + 20\text{px} + 5\text{px} + 5\text{px} = 250\text{px}$ .

При `box-sizing: border-box` значения `width` и `height` включают `padding` и `border`:

Общая ширина = `width = 200px`.

Область содержимого = `width - padding - border = 200px - 40px - 10px = 150px`.

Браузер применяет каскадные правила для определения окончательного значения `box-sizing`, затем использует это значение при вычислении размеров в процессе `layout` (макетирования).

## **Основные техники и паттерны**

### **Универсальное применение border-box**

Техника глобального переопределения `box-sizing` для всех элементов, обеспечивающая более интуитивное поведение размеров:

```
*,
*::before,
*::after {
  box-sizing: border-box;
}
```

### ***Расширенная версия с наследованием:***

```
html {
  box-sizing: border-box;
}
```

```
*,
*::before,
```

```
*::after {  
  box-sizing: inherit;  
}
```

### *Почему это работает*

Первый подход использует универсальный селектор ( \* ) со специфичностью 0-0-0-1, что позволяет легко переопределить поведение для конкретных элементов. Селекторы псевдоэлементов ::before и ::after обеспечивают согласованность поведения генерируемого содержимого.

Второй подход создает каскадное наследование: корневой элемент html устанавливает border-box, а все остальные элементы наследуют это значение через inherit. Это позволяет создавать компоненты, которые могут временно переключаться на content-box без глобального влияния.

## Гибкие формы ввода

Техника создания адаптивных полей ввода с фиксированными внутренними отступами и переменной шириной:

```
<form class="search-form">  
  <input  
    type="search"  
    class="search-input"  
    placeholder="Поиск..."  
  >  
  <button type="submit" class="search-button">  
    Найти  
  </button>  
</form>
```

```
.search-form {  
  display: flex;  
  max-width: 500px;  
  margin: 0 auto;  
}
```

```
.search-input {  
  flex: 1;  
  padding: 0.8em 1em;  
  border: 2px solid #ddd;  
  border-radius: 4px 0 0 4px;  
  font-size: 16px;  
  box-sizing: border-box;  
  min-width: 0; /* предотвращает переполнение flex-элемента */  
}
```