**Глава 3. Селекторы CSS: от простого к магии. Всё, что должен знать верстальщик**

**3.1. Почему селекторы — “мозг” CSS**

Селектор — это “указка”, которой ты выбираешь элементы на странице для оформления.  
**Без правильных селекторов CSS не работает:**

* Ошибся — стиль не применяется,
* Переборщил — стиль “сломает” всё вокруг,
* Применил слишком мощный — потом не перебить.

Селекторы — фундамент для масштабируемых, гибких и поддерживаемых сайтов.  
**Это “язык обращения” к разметке — ты говоришь браузеру, что “именно этот” элемент нужно сделать синим, крупным, а этот — спрятать.**

**3.2. Базовые типы селекторов**

**3.2.1. Селектор по тегу (элементу)**

css

КопироватьРедактировать

p { color: #222; }

Выбирает **все** теги <p> на странице.  
**Плюсы:** быстро, понятно.  
**Минусы:** слишком “широко”, легко сломать макет глобально.

**3.2.2. Селектор по классу**

css

.card { border-radius: 12px; }

Выбирает **все** элементы с class="card".  
**Плюсы:** можно применять на любых тегах, повторять сколько угодно.  
**Минусы:** слишком абстрактный класс (.red, .main) — со временем не ясно, для чего он.

**Лайфхак:**

* Используй осмысленные классы: .btn-primary, .card-product, .header-nav.

**3.2.3. Селектор по id**

css

#main { background: #f9f9f9; }

Только один элемент на странице может иметь одинаковый id.  
**Плюсы:** гарантированная уникальность.  
**Минусы:** почти не используют для стилей — перебивает всё, плохо для модульности, ломает переиспользование.

**3.2.4. Селектор по атрибуту**

css

КопироватьРедактировать

input[type="email"] { border-color: #337ab7; }

a[target="\_blank"] { color: orange; }

Огромная гибкость!

* Можно выбирать элементы по любому атрибуту: [attr], [attr="значение"], [attr^="начало"], [attr$="конец"], [attr\*="подстрока"]

**Пример:**

css

a[href^="https://vk.com"] { color: #4680c2; }

img[alt\*="cat"] { border: 2px solid orange; }

**Плюсы:** мощно для “живых” шаблонов (когда классы не всегда удобны).

**3.3. Составные и комбинированные селекторы**

**3.3.1. Потомок (пробел)**

css

КопироватьРедактировать

header nav a { font-weight: bold; }

Оформляет **все ссылки**, которые находятся внутри навигации в шапке.

**3.3.2. Дочерний селектор (>)**

css

КопироватьРедактировать

ul > li { padding: 6px 0; }

Только **первые** потомки (не вложенные li в li).

**3.3.3. Соседний (+) и общий брат (~)**

css

КопироватьРедактировать

h2 + p { margin-top: 0; }

input:checked ~ label { font-weight: bold; }

* — только первый сосед, ~ — все соседи после данного.

**3.3.4. Множественные селекторы (через запятую)**

css

КопироватьРедактировать

h1, h2, h3 { margin-bottom: 20px; }

Все элементы из списка.

**3.3.5. Группировка и каскады**

css

КопироватьРедактировать

.card .title { color: navy; }

.navbar .btn { padding: 10px 20px; }

“Только те .title, что внутри .card”.

**3.4. Псевдоклассы и псевдоэлементы: магия без JS**

**3.4.1. Псевдоклассы**

* **:hover** — при наведении мыши.
* **:active** — при нажатии.
* **:focus** — при фокусе (input, кнопка).
* **:nth-child(n)** — n-й ребенок.
* **:first-child, :last-child** — первый/последний ребенок.
* **:checked, :disabled, :required** — состояния форм.

**Примеры:**

css

КопироватьРедактировать

a:hover { color: #337ab7; }

li:nth-child(even) { background: #f5f5f5; }

input:focus { border-color: #337ab7; }

**3.4.2. Псевдоэлементы**

* **::before, ::after** — вставить контент до/после элемента.
* **::first-line, ::first-letter** — оформление первой строки/буквы.

**Примеры:**

css

КопироватьРедактировать

.card::before { content: "🔥"; }

h2::first-letter { font-size: 200%; color: #f90; }

**3.5. Современные супер-селекторы (CSS4+, 2020+)**

* **:not()** — отрицание.

css

КопироватьРедактировать

button:not(.active) { opacity: .7; }

* **:is(), :where()** — группировка.

css

КопироватьРедактировать

:is(header, main, footer) > h1 { font-size: 36px; }

* **:has()** (ещё не во всех браузерах!) — “если внутри есть что-то”.

css

КопироватьРедактировать

.card:has(img) { border: 2px solid #337ab7; }

* **:nth-child(even/odd), :nth-of-type(n)** — мощные шаблоны для оформления списков, таблиц, галерей.

**3.6. Лайфхаки и паттерны реальных проектов**

* Для адаптивного меню:

css

.nav\_\_item:last-child { margin-left: auto; }

* Для карточек-зебр:

css

.card:nth-child(even) { background: #fafafa; }

* Для выделения внешних ссылок:

css

a[href^="http"]:after { content: "↗"; font-size: 12px; margin-left: 2px; }

**Совет:**  
Пиши коротко, но понятно.  
Используй цепочки классов вместо избыточных тегов.  
Для переиспользования — только классы (никаких id!).

**3.7. Ошибки, анти-паттерны, проблемы**

* **Селектор слишком общий:**

css

\* { box-sizing: border-box; }

— норм для сброса, но опасно для цвета или размеров.

* **Селектор слишком сложный:**  
  .main .header .menu ul li a span b — трудночитаемо, сложно поддерживать.
* **Много селекторов для одного блока:**  
  .card, .main .card, .main .content .card — запутывают и делают каскад неуправляемым.
* **Использование id для оформления** — приводит к “неперебиваемости” и конфликтам.
* **Псевдоклассы только для красоты:**  
  — не забывай об accessibility: добавляй :focus для клавиатуры, не только :hover для мыши!

**3.8. Селекторы vs JS и CSS-in-JS**

* **CSS-in-JS** (например, styled-components в React) создаёт локальные, уникальные классы для компонентов.  
  — Не нужно “бороться” с глобальными селекторами.  
  — Практически нет конфликтов, всё внутри компонента.  
  — Для больших проектов это спасение (но тоже свои минусы — сложнее отлаживать, избыточность кода).
* **Классика (глобальный CSS):**  
  — Классы должны быть уникальными в рамках смысловой области.  
  — Стиль может “утечь” на соседние блоки.  
  — Каскад может привести к неожиданным результатам.

**3.9. Лучшие практики 2024**

* Используй осмысленные классы: .btn-primary, .nav-link, .product-card.
* Не стилизуй по тегам, кроме сброса базовых стилей.
* Избегай глубоких селекторных цепочек.
* Пиши стили компонентно — 1 класс = 1 блок/модуль.
* Проверяй себя: если селектор длиннее 3 уровней — скорее всего, архитектура “плывёт”.

**3.10. Глоссарий**

| **Термин** | **Определение** |
| --- | --- |
| Селектор | Указание, какие элементы HTML стилизовать |
| Класс | class="..." — множественное назначение стиля |
| id | id="..." — уникальное назначение стиля |
| Атрибут | href, type, alt, src и др. |
| Псевдокласс | :hover, :focus, :active, :nth-child и др. |
| Псевдоэлемент | ::before, ::after, ::first-line, ::first-letter и др. |
| Составной селектор | Несколько критериев выбора (например, .card > h3 + p) |
| CSS-in-JS | Технология стилизации прямо в JS/React компонентах |

**3.11. Вопросы для самопроверки**

1. Чем отличается селектор класса от селектора id? Когда какой использовать?
2. Как выбрать только третий элемент в списке и почему это важно?
3. Зачем нужны псевдоклассы и псевдоэлементы? Приведи пример.
4. Какой селектор опаснее — слишком общий или слишком глубокий?
5. Как через селекторы сделать доступный интерфейс для пользователей с клавиатурой?