CSS

Стили и методологии

Занятие 3

Frontend Course 2025

Что такое CSS?

> **Вопрос:** CSS это тоже язык программирования?

Нет! CSS — язык стилей. HTML говорит "что", CSS говорит "как выглядит". Это как наряжать манекена — сначала каркас, потом одежда.

CSS vs inline стили

> **Вопрос:** Почему не писать стили прямо в HTML?

❌ Плохо:

```
<h1 style="color: red; font-size: 24px;">
  Заголовок
</h1>
<p style="color: red; font-size: 16px;">
  Текст
</p>
```

✅ Хорошо:

```
.red-theme h1 { color: red; font-size: 24px; }
.red-theme p { color: red; font-size: 16px; }
```

Один CSS файл управляет всем сайтом!

Селекторы

> **Вопрос:** Как CSS находит нужные элементы?

Через селекторы — это "адреса" элементов:

- `h1` — все заголовки h1
- `.button` — элементы с классом button
- `#header` — элемент с id="header"
- `div p` — все p внутри div

```
h1 { color: blue; }
.button { background: green; }
#header { height: 100px; }
```

Каскадность

> **Вопрос:** Что если несколько правил применяются к одному элементу?

Каскад! CSS выбирает самое специфичное правило:
1. Inline стили (style="...")
2. ID селекторы (#id)
3. Классы и атрибуты (.class, [attr])
4. Теги (div, p, h1)
5. Последний в коде побеждает при равной специфичности

Специфичность в действии

```
p { color: black; }           /* специфичность: 1 */
.text { color: blue; }        /* специфичность: 10 */
#content { color: red; }      /* специфичность: 100 */
```

```
<p id="content" class="text">Какой цвет?</p>
```

Красный! ID (#content) имеет наивысшую специфичность.

Flexbox

> **Вопрос:** Как выровнять элементы в ряд без мучений?

Flexbox! Одномерная раскладка — либо в строку, либо в столбец.

```
.flex-container {
  display: flex;
  justify-content: center;    /* горизонтальное выравнивание */
  align-items: center;        /* вертикальное выравнивание */
  gap: 20px;                  /* отступы между элементами */
}
```

Flexbox: направления

> **Вопрос:** Как сделать элементы в столбец вместо строки?

```
.column {
  display: flex;
  flex-direction: column;     /* в столбец */
}

.row {
  display: flex;
  flex-direction: row;        /* в строку (по умолчанию) */
}
```

`flex-direction` меняет главную ось. Строка или столбец — выбираете вы.

Grid Layout

> **Вопрос:** А если нужна сложная сетка, как в газете?

CSS Grid! Двумерная раскладка — строки И столбцы одновременно.

```
.grid-container {
  display: grid;
  grid-template-columns: 1fr 2fr 1fr;  /* 3 колонки */
  grid-template-rows: auto 1fr auto;    /* 3 строки */
  gap: 20px;
}
```

Grid: области

```css
.layout {
  display: grid;
  grid-template-areas:
    "header header header"
    "sidebar main aside"
    "footer footer footer";
}

.header { grid-area: header; }
.sidebar { grid-area: sidebar; }
.main { grid-area: main; }
```

Можно рисовать макет прямо в CSS! Визуально и понятно.

Flexbox vs Grid

> **Вопрос:** Когда использовать Flexbox, а когда Grid?

Flexbox: навигация, кнопки в ряд, центрирование
Grid: макеты страниц, карточки товаров, сложные сетки

Часто используют вместе — Grid для макета, Flexbox для компонентов внутри.

БЭМ методология

> **Вопрос:** Как называть CSS классы, чтобы не запутаться?

БЭМ! Блок, Элемент, Модификатор — система именования классов.

```
.button { }                    /* Блок */
.button__text { }              /* Элемент блока */
.button--large { }             /* Модификатор блока */
.button__text--bold { }        /* Модификатор элемента */
```

БЭМ на практике

HTML:

```
<div class="card card--featured">
  <h3 class="card__title">
    Заголовок
  </h3>
  <p class="card__text card__text--small">
    Описание
  </p>
  <button class="card__button">
    Читать
  </button>
</div>
```

CSS:

```
.card { padding: 20px; }
.card--featured { border: 2px solid gold; }
.card__title { font-size: 18px; }
.card__text { color: #666; }
.card__text--small { font-size: 14px; }
.card__button { background: blue; }
```

Читается как книга!

CSS переменные

> **Вопрос:** Как не повторять одни и те же значения?

CSS переменные! Объявили один раз, используете везде.

```
:root {
  --primary-color: #007bff;
  --spacing: 16px;
  --border-radius: 8px;
}

.button {
  background: var(--primary-color);
  padding: var(--spacing);
  border-radius: var(--border-radius);
}
```

Медиа-запросы

> **Вопрос:** Как сделать сайт красивым на телефоне?

Медиа-запросы! Разные стили для разных экранов.

```
.container {
  width: 1200px;              /* на больших экранах */
}

@media (max-width: 768px) {
  .container {
    width: 100%;               /* на мобильных */
    padding: 16px;
  }
}
```

Mobile-first подход

> **Вопрос:** С какого экрана начинать дизайн?

С мобильного! Сначала маленький экран, потом расширяем возможности.

```
.nav {
  flex-direction: column;      /* мобильная версия */
}

@media (min-width: 768px) {
  .nav {
    flex-direction: row;       /* десктопная версия */
  }
}
```

Альтернативы CSS

> **Вопрос:** Неужели есть что-то проще CSS?

Не проще, а удобнее для разных задач:

- **Tailwind CSS** — утилитарные классы
- **Bootstrap** — готовые компоненты
- **UnoCSS** — атомарный CSS
- **Styled Components** — CSS в JavaScript

Tailwind CSS

Обычный CSS:

```
.button {
  background: blue;
  color: white;
  padding: 12px 24px;
  border-radius: 8px;
  border: none;
}
```

Tailwind:

```
<button class="bg-blue-500 text-white
               px-6 py-3 rounded-lg
               border-none">
  Кнопка
</button>
```

Стили прямо в HTML, но системно!

Bootstrap

> **Вопрос:** А если хочется готовых красивых компонентов?

Bootstrap! Библиотека готовых стилей и компонентов.

```
<div class="container">
  <div class="row">
    <div class="col-md-6">
      <div class="card">
        <div class="card-body">
          <h5 class="card-title">Заголовок</h5>
          <p class="card-text">Текст карточки</p>
        </div>
      </div>
    </div>
  </div>
</div>
```

CSS-in-JS

> **Вопрос:** Можно ли писать CSS прямо в JavaScript?

Да! Styled Components и Emotion позволяют это делать.

```
const Button = styled.button`
  background: ${props => props.primary ? 'blue' : 'grey'};
  color: white;
  padding: 12px 24px;
  border-radius: 8px;
`;

<Button primary>Главная кнопка</Button>
```

Инструменты разработки

> **Вопрос:** Как отладить CSS в браузере?

DevTools! F12 → Elements → Styles. Можете менять стили прямо в браузере и видеть результат.

Полезные функции:

- Инспектор элементов
- Computed стили
- Box model диаграмма
- Responsive design mode

CSS препроцессоры

> **Вопрос:** Чем SCSS лучше обычного CSS?

- Переменные и вложенности
- Миксины для переиспользования
- Функции и циклы
- Импорты для модульности
- Компилируется в обычный CSS

SCSS пример

SCSS:

```scss
$primary: #007bff;
$spacing: 16px;

.card {
  padding: $spacing;

  &__title {
    color: $primary;

    &--large {
      font-size: 24px;
    }
  }
}
```

Скомпилированный CSS:

```css
.card {
  padding: 16px;
}

.card__title {
  color: #007bff;
}

.card__title--large {
  font-size: 24px;
}
```

Производительность CSS

> **Вопрос:** Может ли CSS тормозить сайт?

Да! Слишком сложные селекторы, неиспользуемый CSS, перерисовки.

Решения:

- Удаляйте неиспользуемый CSS
- Избегайте глубокой вложенности
- Используйте CSS-in-JS для code splitting
- Critical CSS для быстрой загрузки

Темная тема

> **Вопрос:** Как сделать переключение темной/светлой темы?

```
:root {
  --bg-color: white;
  --text-color: black;
}

[data-theme="dark"] {
  --bg-color: #1a1a1a;
  --text-color: white;
}

body {
  background: var(--bg-color);
  color: var(--text-color);
}
```

JavaScript меняет атрибут, CSS делает остальное!

Чек-лист качественного CSS

- Семантичные классы (не .red-text, а .error)
- Консистентное именование (БЭМ или другая система)
- CSS переменные для повторяющихся значений
- Мобильная адаптивность
- Логическая структура файлов
- Отсутствие !important (почти всегда)
- Переиспользуемые компоненты

Домашнее задание

- Создать адаптивную страницу с БЭМ методологией
- Использовать Flexbox и Grid
- Добавить темную/светлую тему через CSS переменные
- Попробовать Tailwind CSS (дополнительно)
- Проверить адаптивность в DevTools

Следующее занятие: SCSS + Vite

Вопросы?

CSS — это искусство.
Flexbox и Grid — ваши кисти.