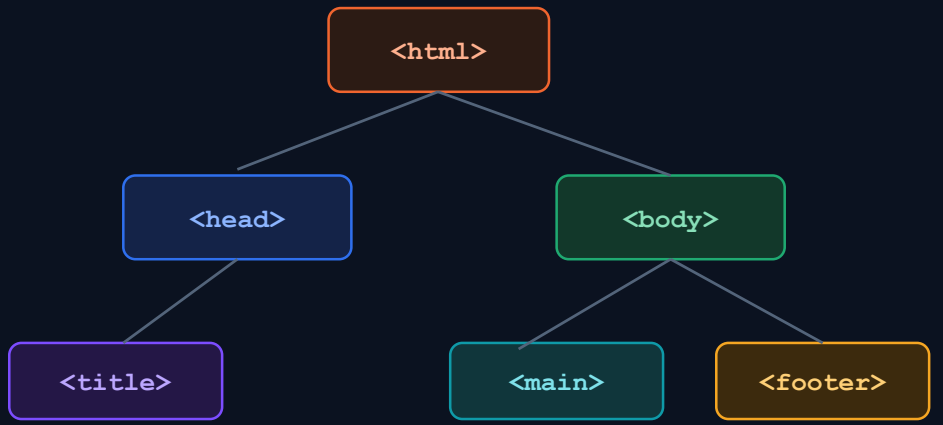
HTML

<HTML>

Структура, теги, атрибуты и основные элементы



Дробышев Максим

старший разработчик · Яндекс Еда

ПЛАН ЗАНЯТИЯ

Что успеем сегодня

01	Как браузер получает и читает HTML	контекст
02	Теги, элементы, вложенность и DOM	синтаксис
03	Атрибуты и пути к ресурсам	навигация
04	Основные теги: текст, медиа, таблицы, формы	контент
05	Семантика, доступность и практика	качество

План рассчитан на одну насыщенную лекцию с короткими мини-разборами по ходу.

КОНТЕКСТ

Три слоя клиентской части

<article>

HTML

Структура и смысл

- что находится на странице
- как части связаны между собой

.card { }

CSS

Внешний вид

- как элементы выглядят
- где и какого они размера

addEventListener()

JavaScript

Поведение

- что происходит при действиях
- как меняются данные и интерфейс

Сегодня разбираем первый слой — HTML. CSS и JavaScript будут опираться на созданную им структуру.

КОНТЕКСТ

От URL до страницы в браузере

Браузер не «показывает файл как есть». Он разбирает HTML, создаёт дерево объектов (DOM), затем применяет CSS и выполняет JavaScript.

DevTools → Elements показывает итоговый DOM

Что такое HTML

HyperText Markup Language

язык гипертекстовой разметки

- описывает структуру документа
- задаёт смысл частей контента
- связывает страницы и ресурсы

HTML не отвечает за всё

✓ структура и семантика

✓ ссылки на изображения, стили и скрипты

× сложная логика и вычисления

× точное визуальное оформление

HTML — декларативный язык разметки, а не язык программирования.

Тег, элемент и DOM-узел — не одно и то же

Тег

Синтаксис в исходном тексте

```
<p>  и  </p>
```

Открывающий и закрывающий тег — две отдельные записи.

Элемент

Конструкция целиком

```
<p>Привет!</p>
```

Элемент включает теги, атрибуты и содержимое.

DOM-узел

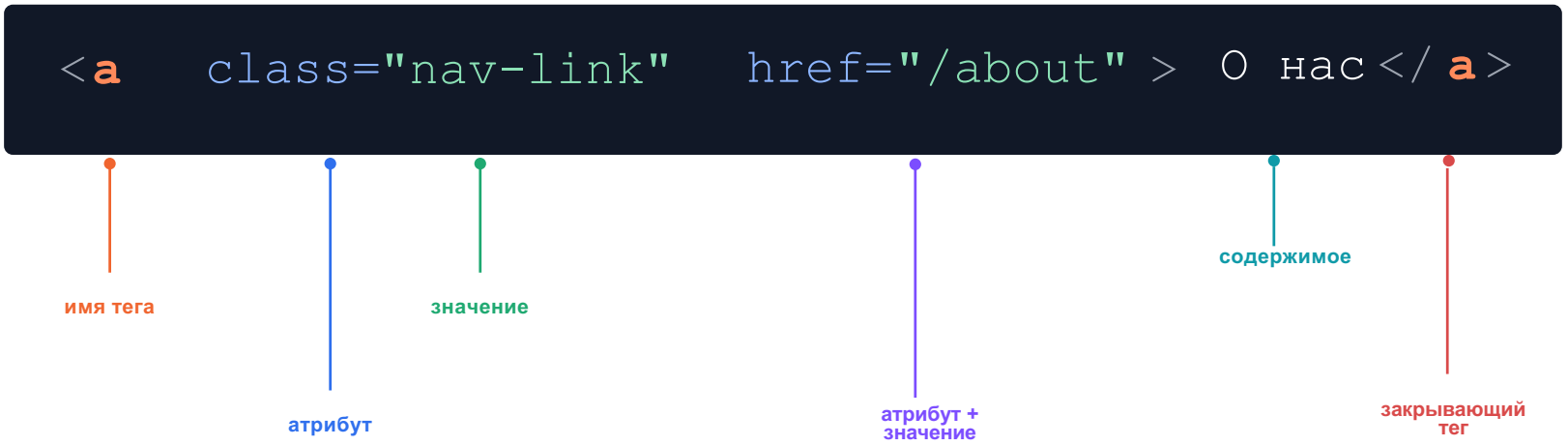
Объект, созданный браузером

```
HTMLParagraphElement
```

JavaScript и DevTools работают с объектами DOM.

Текст внутри элемента тоже становится отдельным текстовым узлом.

Анатомия HTML-элемента



Имя тега пишется без пробелов; значения атрибутов обычно заключают в кавычки

Парные и пустые элементы

Парные

Есть открывающий тег, содержимое и закрывающий тег.

HTML

```html
<p>Абзац</p>
<a href="/">Главная</a>
<button>Отправить</button>
```

Пустые (void)

Не имеют содержимого и закрывающего тега.

HTML

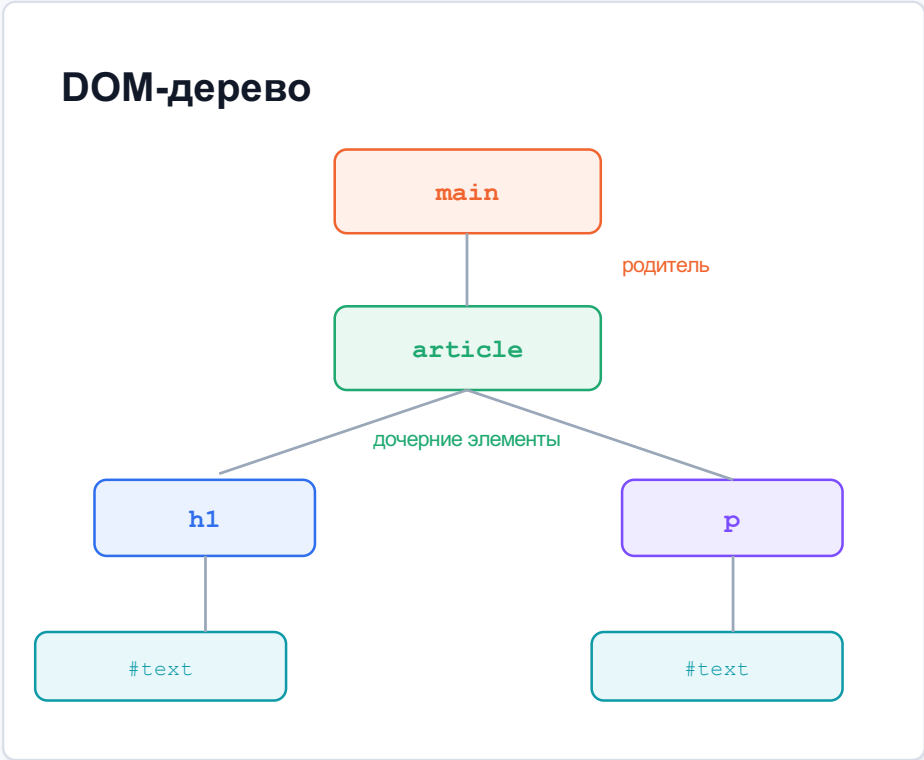
```html
<img src="cat.jpg" alt="Кот">
<br>
<hr>
<input type="email">
<meta charset="UTF-8">
```

В HTML запись <img /> допустима, но слэш не «закрывает» элемент и не обязателен.

Вложенность превращает разметку в дерево

index.html

```
1 <main>
2   <article>
3     <h1>Новости курса</h1>
4     <p>Первая лекция — про HTML.</p>
5   </article>
6 </main>
```



Корректная вложенность и форматирование

Корректно

HTML

```html
<p>
  Текст с <strong>акцентом</strong>.
</p>

<ul>
  <li>Первый пункт</li>
  <li>Второй пункт</li>
</ul>
```

Проблема

HTML

```html
<p>Текст <strong>акцент</p></strong>

<ul>
  <div>Пункт</div>
</ul>

<p><div>Блок внутри p</div></p>
```

Закрывайте в обратном порядке · соблюдайте допустимых родителей · используйте отступы

Минимальный HTML-документ

index.html

```
<!doctype html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width,
initial-scale=1">
    <title>Название страницы</title>
  </head>
  <body>
    <h1>Привет, HTML!</h1>
  </body>
</html>
```

`<!doctype html>`	режим современного HTML
`lang="ru"`	язык содержимого
`charset`	кодировка UTF-8
`viewport`	масштаб на мобильных
`title`	название вкладки
`body`	видимое содержимое

`doctype` — декларация, а не HTML-элемент.

<head> и <body>: служебная и видимая части

<head>

Метаданные и подключения

- <title> — название вкладки
- <meta> — кодировка, viewport, описание
- <link> — CSS, иконки и другие ресурсы
- <script defer> — подключение JavaScript

<body>

Контент страницы

- заголовки и текст
- ссылки и изображения
- семантические разделы
- таблицы, формы и медиа

Пользователь обычно видит только содержимое <body>, но <head> влияет на вкладку, поиск, адаптивность и загрузку ресурсов.

Пробелы, комментарии и специальные символы

Пробелы

```
<p>Один     абзац
с переносом в коде</p>
```

В обычном потоке цепочки пробелов и переносов схлопываются.

Комментарии

```
<!-- TODO: добавить фото
-->
```

Не отображаются, но видны в исходном коде. Не храните в них секреты.

Сущности

```
&lt;  →  <
&amp; →  &
  → неразрывный
пробел
```

Нужны, чтобы вывести символы, имеющие специальный смысл в HTML.

<pre> сохраняет пробелы и переносы — используйте для кода и предварительно форматированного текста

Атрибуты: дополнительные настройки элемента

- пишутся только в открывающем или пустом теге
- несколько атрибутов разделяются пробелами
- порядок атрибутов обычно не влияет на результат
- кавычки вокруг значений — безопасная и читаемая привычка

АТРИБУТЫ

Глобальные атрибуты

id — уникальный идентификатор

`#contacts`

class — одна или несколько групп

`card featured`

lang — язык элемента

`ru / en`

title — дополнительная подсказка

`текст`

hidden — скрывает элемент

`булев`

tabindex — участие в фокусе

`0 / -1`

contenteditable — разрешает редактирование

`true`

style — встроенные CSS-стили

`лучше редко`

Глобальные атрибуты допустимы почти у любого HTML-элемента — но смысл всё равно должен быть оправдан.

Булевы атрибуты, data-* и ARIA

Булевы

```html
<input required>
<input checked>
<button disabled>Сохранить</button>
```

Сам факт присутствия означает true. Значение писать не обязательно.

data-*

```html
<article
  data-id="42"
  data-state="draft">
</article>
```

Пользовательские данные для JavaScript. Имя после data- выбираете вы.

ARIA

```html
<button
  aria-expanded="false"
  aria-controls="menu">
  Меню
</button>
```

Добавляет доступностную информацию, когда нативной семантики недостаточно.

Правило: сначала нативный HTML, затем ARIA — не наоборот.

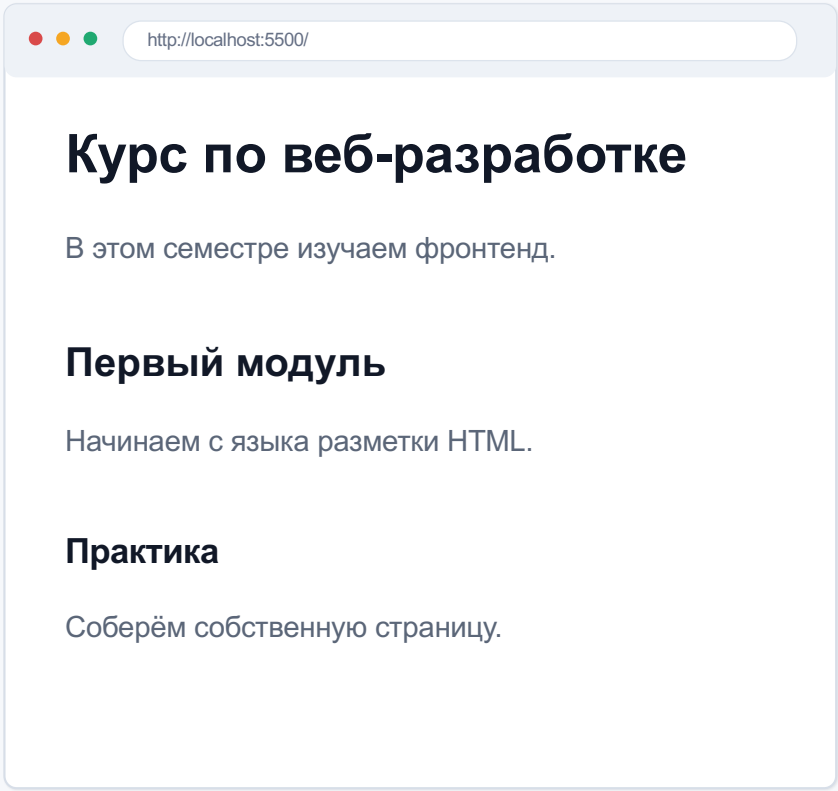
Заголовки и абзацы

HTML

```html
<h1>Курс по веб-разработке</h1>
<p>В этом семестре изучаем фронтенд.</p>

<h2>Первый модуль</h2>
<p>Начинаем с языка разметки HTML.</p>

<h3>Практика</h3>
<p>Соберём собственную страницу.</p>
```



h1–h6 задают иерархию, а не «размер текста». Размер позже настраивает CSS.

Смысловое выделение текста

**** — **важность**

Срок сдачи — завтра

**** — **логическое ударение**

Я именно это имел в виду

<mark> — **актуальное выделение**

Найденное совпадение

<small> — **второстепенный текст**

Правовая информация

<del> / <ins> — **изменения**

старая / новая цена

<b> / <i> — **визуальное или типографическое отличие**

термин, имя, голос

Выбирайте тег по смыслу. Внешний вид любого элемента можно изменить CSS-ом.

Код, цитаты, сокращения и время

HTML

```
<p>Нажмите <kbd>Ctrl</kbd> + <kbd>S</kbd>.</p>
<pre><code>const answer = 42;</code></pre>
<blockquote cite="https://example.com/source">
  Хорошая разметка передаёт смысл.
</blockquote>
<p><abbr title="HyperText Markup Language">HTML</abbr></p>
<time datetime="2026-09-07">7 сентября</time>
```

`<code>`	фрагмент кода
`<pre>`	сохраняет форматирование
`<kbd>`	ввод с клавиатуры
`<blockquote> / <q>`	цитаты
`<abbr>`	сокращение с расшифровкой
`<time>`	машиночитаемая дата или время

Списки: маркированные, нумерованные и описания

HTML

```html
<ul>
  <li>HTML</li>
  <li>CSS</li>
  <li>JavaScript</li>
</ul>

<ol start="4">
  <li>React</li>
  <li>Фреймворки</li>
</ol>

<dl>
  <dt>HTML</dt>
  <dd>Язык разметки</dd>
</dl>
```

http://localhost:5500/lists.html

Технологии

- HTML
- CSS
- JavaScript

Продолжение

4. React
5. Фреймворки

HTML Язык разметки

Внутри ul и ol непосредственными детьми должны быть li. Списки можно вкладывать друг в друга.

Ссылки: элемент <a> и атрибут href

HTML

```html
<a href="/courses/html.html">Курс HTML</a>
<a href="#contacts">К контактам</a>
<a href="mailto:hello@example.com">Написать</a>
<a href="tel:+79990000000">Позвонить</a>
<a href="guide.pdf" download>Скачать PDF</a>
<a href="https://example.com" target="_blank" rel="noopener">
  Внешний сайт
</a>
```

Что может быть ссылкой?

- текст
- изображение
- целая карточка
- иконка или логотип

href	адрес назначения
target	где открыть ресурс

Пути к файлам и якорям

`https://site.ru/img/logo.svg`	абсолютный URL
`/img/logo.svg`	от корня сайта
`img/logo.svg`	от текущего файла
`../img/logo.svg`	на уровень выше
`#contacts`	к элементу id="contacts"

Структура проекта

- ▾ **project/**
  - index.html
  - ▾ **pages/**
    - about.html
  - ▾ **img/**
    - logo.svg
    - team.webp

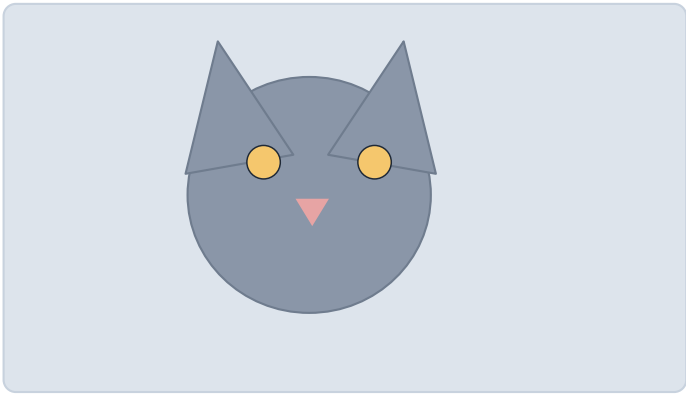
Из pages/about.html до logo.svg: ../img/logo.svg

Изображения: <img>, alt и <figure>

HTML

```
<figure>
  <img
    src="images/cat.webp"
    alt="Серый кот смотрит в камеру"
    width="640"
    height="420"
    loading="lazy">
  <figcaption>Кот — главный тестировщик.</figcaption>
</figure>
```

http://localhost:5500/gallery.html



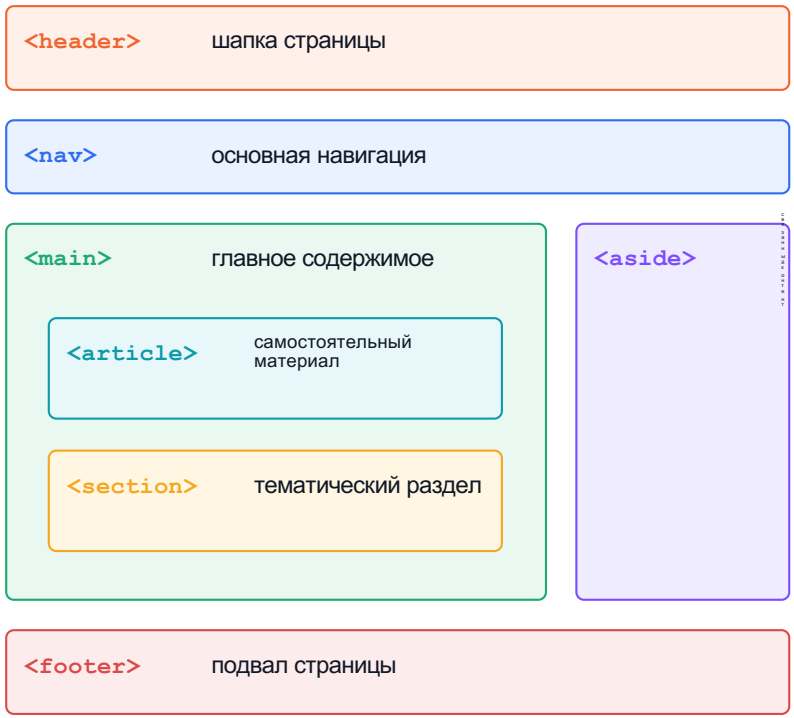
Кот — главный тестировщик.

alt: смысл изображения, а не имя файла

width + height уменьшают скачки макета

Декоративная картинка: alt=""

Семантический каркас страницы



Как выбирать элемент

- <main> — один основной контент документа
- <section> — тематический раздел, обычно с заголовком
- <article> — материал, который имеет смысл сам по себе
- <nav> — значимый набор навигационных ссылок
- семантика описывает назначение, а не внешний вид

и : нейтральные контейнеры

HTML

```html
<div class="price">
  <span class="price__value">1 490</span>
  <span class="price__currency">₽</span>
</div>
```

** ** группирует крупные фрагменты, когда подходящего семантического элемента нет

**** оборачивает короткий фрагмент внутри строки

Важно: «блочный» и «строчный» — это поведение CSS по умолчанию, а не тип тега HTML. display можно изменить.

Сначала ищите семантический тег. div и span — запасной вариант, а не универсальный ответ.

Таблицы — только для табличных данных

HTML

```html
<table>
  <caption>Баллы за лабораторные</caption>
  <thead>
    <tr>
      <th scope="col">Работа</th>
      <th scope="col">Баллы</th>
    </tr>
  </thead>
  <tbody>
    <tr><td>HTML</td><td>9</td></tr>
    <tr><td>CSS</td><td>12</td></tr>
  </tbody>
</table>
```

Баллы за лабораторные

Работа	Баллы
HTML	9
CSS	12

caption — название таблицы

th + scope — заголовки

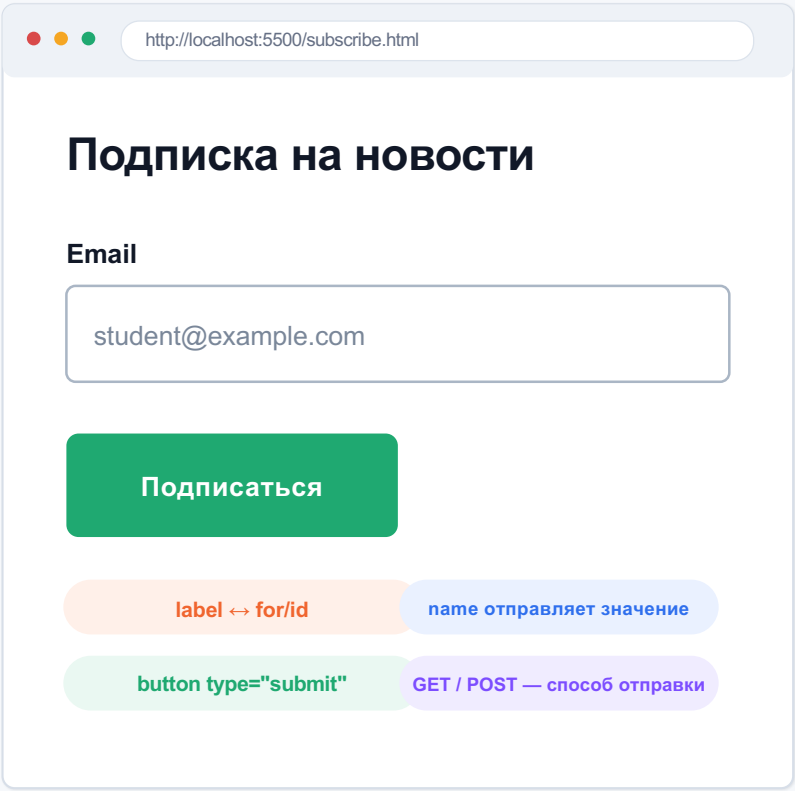
Не используйте table для раскладки страницы

Формы: как собрать данные пользователя

HTML

```html
<form action="/subscribe" method="post">
  <div>
    <label for="email">Email</label>
    <input id="email" name="email" type="email" required>
  </div>

  <button type="submit">Подписаться</button>
</form>
```



Поля формы и встроенная валидация

<input type="…">

text / search	короткий текст
email / url / tel	специализированный ввод
number / range	числа
date / time	дата и время
checkbox / radio	выбор
file	загрузка файла

Другие элементы

<textarea>	многострочный текст
<select> + <option>	выбор из списка
<button>	действие: submit / button / reset

Полезные атрибуты

required · autocomplete · min / max · pattern · multiple · placeholder ≠ label

Аудио, видео и встроенный контент

<video>

```HTML
<video controls width="640"
poster="cover.jpg">
  <source src="lesson.mp4"
type="video/mp4">
  <track kind="captions"
src="captions.vtt" srclang="ru">
</video>
```

controls — управление; track — субтитры.

<audio>

```HTML
<audio controls>
  <source src="podcast.mp3"
type="audio/mpeg">
</audio>
```

Предусмотрите текстовую альтернативу или расшифровку.

<iframe>

```HTML
<iframe
  src="https://example.com/embed"
  title="Карта кампуса"
  loading="lazy"
  sandbox>
</iframe>
```

Всегда добавляйте title. Ограничивайте права через sandbox.

Не запускайте медиа автоматически со звуком — это ухудшает UX и доступность.

КАЧЕСТВО

Доступность начинается с правильного HTML

Нативный HTML уже многое умеет

`<button>`	для действия
`<a href>`	для перехода
`<label>`	для поля формы
`alt`	для изображения
`lang="ru"`	для языка документа
`h1 → h2 → h3`	для структуры

Антипаттерны

- ✕ <div onclick> вместо кнопки
- ✕ placeholder вместо label
- ✕ изображение без alt
- ✕ заголовки только ради размера
- ✕ положительный tabindex

Семантика помогает клавиатуре, скринридерам, поиску и вашей команде.

Частые ошибки и проверка разметки

Частые ошибки

- незакрытые или пересекающиеся теги
- одинаковый id у нескольких элементов
- div внутри p и другие нарушения модели содержимого
-
 и таблицы для раскладки
- img без alt, input без label
- локальные пути C:\Users\…
- button без явного type внутри формы

Рабочий цикл проверки

1. **Открыть страницу**
2. **DevTools → Elements**
3. **Проверить клавиатурой**
4. **Nu HTML Checker**
5. **Исправить предупреждения**

Браузер «показал страницу» ≠ разметка корректна

ЗАКРЕПЛЕНИЕ

Шпаргалка: базовый набор тегов

Документ

`html` · `head` · `body` · `title` · `meta` · `link` · `script`

Структура

`header` · `nav` · `main` · `section` · `article` · `aside` · `footer` · `div` · `span`

Текст

`h1–h6` · `p` · `strong` · `em` · `mark` · `code` · `pre` · `blockquote` · `time`

Списки и ссылки

`ul` · `ol` · `li` · `dl` · `dt` · `dd` · `a`

Медиа

`img` · `figure` · `figcaption` · `picture` · `video` · `audio` · `iframe`

Таблицы

`table` · `caption` · `thead` · `tbody` · `tr` · `th` · `td`

Формы

`form` · `label` · `input` · `textarea` · `select` · `option` · `button`

Не нужно запоминать все теги.
Нужно уметь выбрать подходящий по смыслу и проверить в справочнике.

Итоги занятия

HTML = структура + смысл + связи между ресурсами

1. **Теги формируют элементы** — парные, пустые, вложенные
2. **Атрибуты уточняют поведение** — id, class, href, src, alt, name…
3. **Семантика важнее внешнего вида** — CSS можно поменять, смысл должен остаться
4. **Качество проверяется** — DevTools, клавиатура, валидатор

Спасибо! Вопросы?