АО «АЛМАТИНСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»
https://atu.edu.kz/

Факультет «Информационные технологии»
Кафедра «Компьютерная инженерия»

Название курса: «Разработка клиентских Web-приложений»



Зиятбекова Гулзат Зиятбеккызы – PhD, ассоциированный профессор
g.ziyatbekova@atu.edu.kz

ОСНОВЫ ВЕБ-ТЕХНОЛОГИЙ

ЛЕКЦИЯ 2

Структура HTML5-документа, элементы и атрибуты

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <title>Лекция 2</title>
  </head>
  <body>
    <h1>HTML5</h1>
    <p>Структура,
       элементы,
       атрибуты</p>
  </body>
</html>
```

ПЛАН ЛЕКЦИИ

Что мы рассмотрим сегодня

1. Назначение языка HTML
2. Стандарт HTML5
3. Базовая структура HTML-документа
4. Декларация DOCTYPE
5. HTML-элементы
6. Теги
7. Атрибуты
8. Текстовые элементы
9. Ссылки
10. Списки
11. Правила организации HTML-кода

ВВЕДЕНИЕ

Место HTML в клиентской веб-разработке

Клиентская часть веб-приложения строится на трёх технологиях. Каждая отвечает за свой слой, и подменять один слой другим — ошибка проектирования.

HTML5

СТРУКТУРА

Из чего состоит документ: заголовки, абзацы, списки, изображения, формы, таблицы.

CSS3

ОФОРМЛЕНИЕ

Как документ выглядит: цвет, шрифт, размеры, расположение, адаптивность, анимация.

JavaScript

ПОВЕДЕНИЕ

Что документ делает: реакция на действия пользователя, вычисления, работа с данными.

Важно для студента

Ошибки в HTML-структуре нельзя «вылечить» стилями. Неверная разметка ломает доступность, SEO, работу JavaScript и поведение страницы на мобильных устройствах. Поэтому изучение фронтенда всегда начинается с HTML.

Что такое HTML

HTML (HyperText Markup Language) — язык разметки, предназначенный для описания структуры веб-документов.

С помощью HTML описывают

- заголовок страницы и заголовки разделов
- абзацы текста
- изображения, аудио и видео
- ссылки и навигацию
- списки и таблицы
- формы ввода данных

HTML — не язык программирования

В нём нет переменных, условий, циклов и функций — механизма выполнения алгоритмов в традиционном понимании. HTML только описывает браузеру, из каких элементов состоит документ и как они вложены друг в друга.

Это язык разметки (markup), как и XML или Markdown. Алгоритмы в браузере выполняет JavaScript.

Ключевая мысль: HTML отвечает за структуру и смысловое содержание документа, а внешний вид задаётся с помощью CSS. Разделение содержания и оформления — базовый принцип современного веба.

Как развивался язык разметки

1991

HTML 1.0

Первая версия Тима Бернерса-Ли: около 20 тегов, только текст и ссылки.

1997—1999

HTML 4.01

Появились таблицы, фреймы, скрипты. Оформление всё ещё внутри разметки.

2000

XHTML 1.0

Строгий синтаксис по правилам XML. Любая ошибка разметки — отказ отображать.

2014

HTML5

Семантика, встроенное аудио и видео, новые типы полей форм, Canvas и API.

Сегодня

Living Standard

Единый развивающийся стандарт WHATWG: обновляется постоянно, без номеров версий.

Что это значит на практике

Сегодня не существует «HTML6»: стандарт развивается непрерывно, а браузеры внедряют новые возможности постепенно. Актуальную спецификацию публикуют WHATWG (html.spec.whatwg.org), справочник для разработчика — MDN Web Docs.

Ключевые возможности HTML5

1 Семантические элементы

<header>, <nav>, <main>, <section>, <article>, <aside>, <footer> — описывают смысл блока, а не его вид.

2 Мультимедиа без плагинов

<audio> и <video> воспроизводятся браузером напрямую, без Flash и сторонних проигрывателей.

3 Улучшенные формы

Типы email, date, number, range, color и встроенная клиентская валидация (required, pattern).

4 Графика и API

Canvas и SVG, геолокация, Drag & Drop, localStorage, Web Workers, Service Worker.

Главный результат

Использование современных элементов HTML5 позволяет создавать более понятную, структурированную и доступную разметку: её лучше «читают» поисковые системы, программы экранного доступа и другие разработчики.

Скелет любого HTML5-документа

```
<!DOCTYPE html>
<html lang="ru">

<head>
    <meta charset="UTF-8">
    <title>Мой сайт</title>
</head>

<body>

    <h1>Привет, мир!</h1>

    <p>Это HTML5-документ.</p>

</body>

</html>
```

Иерархия документа

```
HTML-документ
│
├── DOCTYPE
│
└── HTML
      ├── HEAD   (служебное)
      └── BODY   (видимое)
            ├── Заголовок
            ├── Текст
            └── Изображения
```

Запомнить

Всё содержимое документа лежит внутри <html>. Внутри него ровно два раздела: <head> — информация о документе, <body> — то, что видит пользователь.

Разбор четырёх обязательных частей

`<!DOCTYPE html>`	Декларация типа документа. Сообщает браузеру, что документ обрабатывается по современным правилам HTML.	*Стоит первой строкой, тегом не является.*
`<html>`	Корневой элемент всего документа. Содержит <head> и <body>.	*Атрибут lang="ru" указывает язык содержимого.*
`<head>`	Служебная информация: кодировка, название вкладки, подключение CSS и JS, метаданные.	*Содержимое <head> пользователь на странице не видит.*
`<body>`	Основное содержимое, которое отображается пользователю: тексты, изображения, кнопки, формы.	*Всё видимое содержимое пишется только здесь.*

4. ДЕКЛАРАЦИЯ DOCTYPE

<!DOCTYPE html> и режимы отображения

Декларация типа документа располагается в самом начале файла — до всех остальных элементов — и сообщает браузеру, по каким правилам интерпретировать разметку.

```
<!DOCTYPE html>
```

В HTML5 используется короткая форма — в отличие от HTML 4 и XHTML, где требовалось длинное определение типа документа со ссылкой на DTD.

Правило: <!DOCTYPE html> — всегда первая строка файла, перед ней не должно быть ни пробелов, ни комментариев.

Что будет, если DOCTYPE забыть

Браузер переключится в режим совместимости (quirks mode) и начнёт эмулировать поведение старых браузеров 1990-х годов.

- иначе считаются размеры блоков (box model);
- «плывёт» вёрстка и отступы;
- часть свойств CSS работает непредсказуемо;
- поведение отличается в разных браузерах.

Проверить режим можно в DevTools: вкладка Console → команда document.compatMode. Значение CSS1Compat — стандартный режим, BackCompat — режим совместимости.

Раздел <head>: метаданные документа

```
<head>
    <meta charset="UTF-8">
    <meta name="viewport"
          content="width=device-width,
                   initial-scale=1.0">
    <meta name="description" content="...">
    <title>Мой сайт</title>
    <link rel="stylesheet" href="style.css">
</head>
```

Типичная ошибка новичка

Забыть <meta charset="UTF-8"> и сохранить файл в другой кодировке. Русский текст отобразится нечитаемыми символами, хотя в редакторе всё было верно. Кодировка файла и объявленная кодировка должны совпадать.

<meta charset>

Кодировка. UTF-8 — иначе кириллица превратится в «кракозябры».

<meta viewport>

Обязателен для корректного отображения на смартфонах.

<meta description>

Описание страницы для поисковой выдачи.

<title>

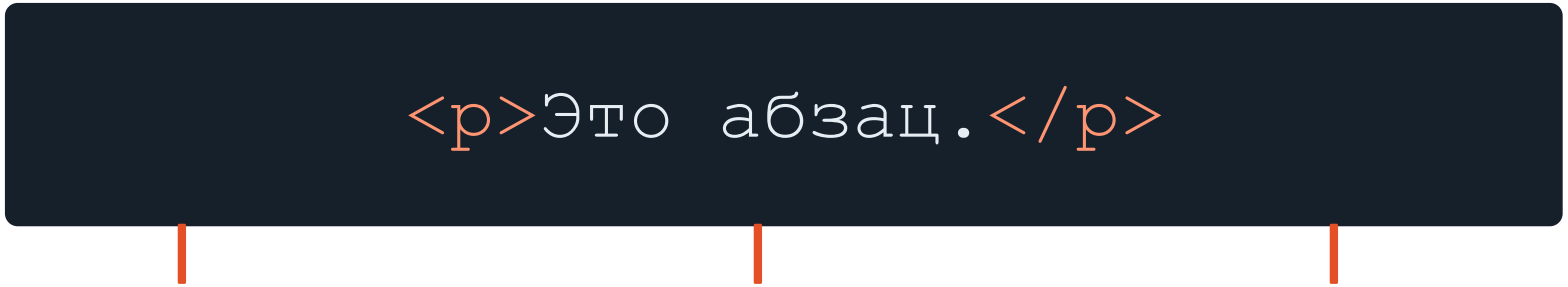
Название вкладки браузера и заголовок в результатах поиска.

<link>

Подключение внешних файлов: CSS, иконки сайта (favicon), шрифтов.

Анатомия HTML-элемента

HTML-элемент — структурная единица документа, представляющая определённое содержимое или функцию.



Открывающий тег

начало элемента

Содержимое

то, что видит пользователь

Закрывающий тег

конец элемента, символ /

Элемент ≠ тег

Тег — это только запись в угловых скобках. Элемент — тег вместе с содержимым и закрывающим тегом. Браузер работает с элементами, а не с текстом тегов.

Вложенность (родитель и потомок)

Элементы образуют дерево. В Важно элемент — родитель, — потомок. Из этого дерева браузер строит DOM.

Виды элементов: две классификации

Парные элементы

Имеют открывающий и закрывающий тег, между ними — содержимое.

```
<p>Текст абзаца</p>
<h1>Заголовок</h1>
```

Одиночные (void)

Не имеют содержимого и закрывающего тега — всё задаётся атрибутами.

```
<img src="photo.jpg" alt="Фото">
<br>    <hr>    <input>
```

Блочные (block)

Занимают всю доступную ширину и начинаются с новой строки.

```
<p>  <div>  <h1>–<h6>  <ul>
<section>  <header>  <table>
```

Строчные (inline)

Занимают ширину содержимого и располагаются в потоке текста.

```
<a>  <span>  <strong>  <em>
<img>  <code>  <small>  <sup>
```

Правила записи тегов

Тег — конструкция HTML, обозначающая начало или конец элемента. Записывается в угловых скобках, закрывающий тег содержит символ /.

Правильно

```
<p>Текст абзаца</p>

<img src="photo.jpg" alt="Фото">

<a href="index.html">Главная</a>

<br>
```

Ошибки

```
<p>Текст абзаца          ← не закрыт

<P>Текст</P>             ← верхний регистр

<a href=index.html>      ← нет кавычек

</img>                   ← лишний тег
```

Регистр

Теги и атрибуты пишутся строчными буквами: <h1>, а не <H1>.

Парность

Каждый парный тег должен быть закрыт — иначе структура «поедет».

Порядок

Открыт последним — закрыт первым: … .

Дополнительные свойства элементов

Атрибут — дополнительное свойство элемента, дающее браузеру уточняющую информацию. Указывается только внутри открывающего тега.

```
<a href="https://example.com">Перейти</a>
```

href	**=**	**"..."**
имя атрибута	знак присваивания	значение в кавычках

Ещё пример

```
<img src="photo.jpg" alt="Фотография аудитории">
```

src указывает путь к изображению, alt — альтернативное текстовое описание. У одного элемента может быть несколько атрибутов; порядок их записи значения не имеет, разделяются пробелом.

Часто используемые и глобальные атрибуты

`id`	Уникальный идентификатор элемента на странице
`class`	Имя класса для стилизации и выбора группы элементов
`href`	Адрес перехода для ссылки <a>
`src`	Путь к внешнему файлу: изображению, видео, скрипту
`alt`	Альтернативный текст изображения
`title`	Всплывающая подсказка при наведении курсора
`type / name / value`	Тип, имя и значение поля формы
`lang`	Язык содержимого элемента
`style`	Инлайновые стили (в учебных целях; в проектах — CSS-файл)
`data-*`	Собственные данные разработчика для JavaScript

Глобальные атрибуты (id, class, title, style, lang, hidden, data-*) применимы почти к любому элементу. Остальные — только к «своим»: href — к <a>, src — к <img>.

id и class: в чём разница

Два самых востребованных атрибута: через них CSS находит элемент для оформления, а JavaScript — для управления.

`id` Идентификатор

- уникален — только один такой элемент на странице;
- используется для якорной ссылки #contacts;
- в CSS обращение через #;
- в JS: document.getElementById("...")

```
<div id="header">…</div>
#header { height: 100px; }
```

`class` Класс

- может повторяться у множества элементов;
- у одного элемента может быть несколько классов;
- в CSS обращение через точку;
- основной инструмент вёрстки

```
<div class="card">…</div>
.card { border: 1px solid gray; }
```

Заголовки h1—h6 и иерархия документа

```
<h1>Главный заголовок</h1>
<h2>Раздел</h2>
<h3>Подраздел</h3>
<h4>Подраздел 4</h4>
<h5>Подраздел 5</h5>
<h6>Подраздел 6</h6>
```

Правила использования

- один <h1> на страницу — главная тема документа;
- уровни не пропускают: после <h1> идёт <h2>, а не <h4>;
- заголовок выбирают по смыслу, а не по размеру шрифта;
- размер настраивается через CSS, а не выбором тега.

В HTML шесть уровней заголовков. <h1> — самый высокий уровень, <h6> — самый низкий.

Зачем нужна правильная иерархия: структурирует документ · улучшает читаемость · упрощает навигацию с клавиатуры и через программы экранного доступа · напрямую влияет на SEO — поисковые системы строят по заголовкам «карту» смысла страницы.

Абзацы и смысловое выделение текста

Абзац <p>

```
<p>HTML отвечает за структуру.</p>

<p>CSS используется для оформления.</p>
```

Каждый <p> — отдельный абзац. Переносы строк и лишние пробелы внутри кода браузер игнорирует: новый абзац создаётся тегом, а не клавишей Enter.

Выделение и

```
<p><strong>Важно:</strong> сохраните данные.</p>

<p>Это <em>очень важная</em> информация.</p>
```

 — текст особой важности, — логическое ударение. Это семантика, а не только оформление: программа экранного доступа произнесёт такой текст с другой интонацией.

Частый вопрос: чем отличается от <b>, а от <i>?

Визуально в браузере — ничем: <b> и дают полужирный, <i> и — курсив. Разница в смысле.

- / — несут смысл: важность и акцент. Их «понимают» вспомогательные технологии и поисковые системы.
- <b> / <i> — только визуальное выделение без смысловой нагрузки (например, термин или название вида на латыни).
- Рекомендация: в учебных и рабочих проектах по умолчанию используйте и .

Другие элементы для работы с текстом

` `	Принудительный перенос строки внутри абзаца
`<hr>`	Смысловой разрыв — смена темы в тексте
`<blockquote>`	Развёрнутая цитата (блочная)
`<q>`	Короткая цитата внутри строки
`<abbr>`	Аббревиатура с расшифровкой в title
`<code>`	Фрагмент программного кода
`<pre>`	Текст с сохранением пробелов и переносов
`<mark>`	Подсветка найденного или значимого фрагмента
`<sub> / <sup>`	Нижний и верхний индекс: H_2O, $м^2$
`<time>`	Дата и время в машиночитаемом формате

Осторожно с <br>: он предназначен для переноса строки внутри одного смыслового блока (адрес, стихотворение). Разделять абзацы несколькими
 вместо <p> — распространённая ошибка.

Элемент <a> и атрибут href

Ссылка обеспечивает переход к другой странице, документу, ресурсу или к определённой части текущей страницы. Гипертекст — основа веба.

```
<a href="https://example.com">Подробнее</a>

<a href="about.html">О компании</a>

<a href="#contacts">Контакты</a>

<h2 id="contacts">Контакты</h2>
```

Как это работает

- href задаёт адрес перехода;
- текст между тегами — то, что видит и нажимает пользователь;
- ссылка #contacts прокручивает страницу к элементу с id="contacts";
- ссылка без href перестаёт быть ссылкой.

Текст ссылки тоже важен

Пользователи программ экранного доступа часто просматривают страницу списком ссылок. Ссылка «Нажмите здесь» вне контекста бессмысленна — пишите осмысленный текст: «Скачать программу курса» или «Перейти к расписанию занятий».

Виды путей и специальные ссылки

Абсолютный путь	`https://example.com/page.html`	Полный адрес внешнего ресурса, включая протокол
Относительный путь	`about.html` · `img/logo.png`	Путь относительно текущего файла — внутри своего сайта
На уровень выше	`../index.html`	Две точки — переход в родительскую папку
Якорь на странице	`#contacts`	Переход к элементу с соответствующим id
Почта и телефон	`mailto:info@edu.kz` · `tel:+77010000000`	Открывает почтовый клиент или набор номера

`target="_blank"`

Открывает ссылку в новой вкладке. Для внешних ссылок добавляют rel="noopener" — из соображений безопасности.

Совет по структуре проекта

Держите файлы в папках: css/, js/, img/. Относительные пути тогда короткие и предсказуемые.

Маркированные и нумерованные списки

Маркированный список

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

Результат в браузере

- HTML
- CSS
- JavaScript

Порядок пунктов не важен — используется для перечислений, меню, тегов.

Нумерованный список

```
<ol>
    <li>Первый шаг</li>
    <li>Второй шаг</li>
    <li>Третий шаг</li>
</ol>
```

Результат в браузере

1. Первый шаг
2. Второй шаг
3. Третий шаг

Порядок важен — инструкции, алгоритмы, этапы выполнения работы.

Внутри <ul> и <ol> допустимы только элементы <li>. Любое другое содержимое (текст,) помещайте внутрь <li>.

Вложенные списки и список определений

Вложенный список

```
<ul>
    <li>
        Веб-разработка
        <ul>
            <li>HTML</li>
            <li>CSS</li>
            <li>JavaScript</li>
        </ul>
    </li>
</ul>
```

Вложенный <ul> помещается внутрь <li> родительского списка, а не между пунктами. Так строят многоуровневые меню и оглавления.

Список определений <dl>

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

    <dt>CSS</dt>
    <dd>Язык описания стилей</dd>
</dl>
```

<dl> — список «термин — определение»: <dt> задаёт термин, <dd> — его описание. Подходит для глоссариев, характеристик товара, FAQ.

Списки — это тоже семантика

Программа экранного доступа объявляет: «список, 3 элемента». Набор абзацев с дефисами такой информации не даёт. Навигационное меню — это по смыслу список ссылок внутри <nav>.

Правила оформления HTML-кода

Код читают чаще, чем пишут. Он должен быть структурированным, читаемым, логичным и понятным другому разработчику.

1 Отступы

Вложенные элементы оформляются с отступом в 2 или 4 пробела — видна структура документа.

2 Нижний регистр

Теги и атрибуты пишутся строчными буквами: <h1>, а не <H1>.

3 Кавычки

Значения атрибутов заключаются в двойные кавычки: alt="Фотография".

4 Закрытие тегов

Все парные элементы закрываются. Незакрытый тег ломает отображение страницы.

5 Вложенность

Элементы вкладываются корректно: Это важный текст.

6 Комментарии

<!-- Навигационное меню --> поясняет назначение блока и не отображается на странице.

Совет: включите в редакторе (VS Code) автоформатирование — Prettier или горячие клавиши Shift + Alt + F.

ПРАКТИКА

Типичные ошибки начинающих

Незакрытый тег

Страница «расползается»: последующее содержимое попадает внутрь незакрытого элемента.

Неверная вложенность

Это важный текст. — браузер «исправит» по-своему, результат непредсказуем.

Отсутствие alt

Изображение без alt недоступно программам экранного доступа и не индексируется поиском.

Заголовок ради размера

<h4> выбран потому, что «мельче». Размер задаётся CSS, тег — по уровню в иерархии.

Абзацы через

Вместо используются переносы: теряется смысловая структура текста.

Кириллица в именах файлов

Пути вида img/фото.jpg часто ломаются на сервере. Только латиница, без пробелов.

Большинство этих ошибок находит валидатор за несколько секунд — привыкайте проверять работу перед сдачей.

ПРАКТИКА

Инструменты проверки и отладки

1

W3C Validator

`validator.w3.org`

Официальная проверка разметки: незакрытые теги, неверная вложенность, недопустимые атрибуты. Принимает файл, URL или вставленный код.

2

DevTools браузера

`клавиша F12`

Вкладка Elements показывает итоговое DOM-дерево — то, как браузер понял вашу разметку. Ошибки вложенности видны сразу.

3

VS Code + Emmet

`редактор кода`

Сокращение ! + Tab создаёт скелет HTML5-документа. Подсветка парных тегов и автозакрытие снижают число ошибок.

4

MDN Web Docs

`developer.mozilla.org`

Справочник по каждому элементу и атрибуту: описание, примеры, поддержка браузерами. Основной источник при работе.

Порядок работы: написать разметку → открыть в браузере → проверить валидатором → исправить ошибки → посмотреть DOM-дерево в DevTools.

ИТОГ РАЗДЕЛА

Пример полного HTML5-документа

```
<!DOCTYPE html>
<html lang="ru">

<head>
    <meta charset="UTF-8">
    <title>Учебная веб-страница</title>
</head>

<body>

    <h1>Клиентская веб-разработка</h1>

    <p>HTML5 используется для создания
       структуры веб-страницы.</p>

    <h2>Основные технологии</h2>

    <ul>
        <li>HTML5</li>
        <li>CSS3</li>
        <li>JavaScript</li>
    </ul>

    <p><a href="https://example.com">Подробнее</a></p>

</body>
</html>
```

Что здесь использовано

Декларация типа
<!DOCTYPE html>

Корневой элемент и язык
<html lang="ru">

Служебная часть
<head>, <meta>, <title>

Видимое содержимое
<body>

Иерархия заголовков
<h1> → <h2>

Текст и список
**<p>, <ul>, **

Ссылка с атрибутом
<a href="...">

САМОПРОВЕРКА

Контрольные вопросы

1. Что такое HTML и для чего он используется?
2. Является ли HTML языком программирования? Обоснуйте ответ.
3. Каково назначение HTML5 в клиентской веб-разработке?
4. Какова базовая структура HTML5-документа?
5. Для чего используется <!DOCTYPE html>?
6. Какая информация размещается в разделе <head>?
7. Чем HTML-элемент отличается от тега?
8. Что такое атрибут? Приведите примеры.
9. В чём разница между атрибутами id и class?
10. Чем отличается от <b>?
11. Как создать ссылку на раздел текущей страницы?
12. В чём разница между <ul> и ?

ЗАДАНИЕ

Практическая работа к лекции

Создайте страницу «Моя специальность» — один файл index.html, оформление CSS пока не требуется.

Обязательные требования

- корректная структура: DOCTYPE, html с lang, head, body;
- <meta charset="UTF-8"> и осмысленный <title>;
- один <h1> и не менее двух <h2>;
- не менее трёх абзацев <p>;
- маркированный и нумерованный списки;
- внутренняя ссылка-якорь и внешняя ссылка;
- изображение с заполненным атрибутом alt;
- минимум один комментарий в коде.

Критерии оценивания

Корректность структуры	**30%**
Полнота требований	**30%**
Отсутствие ошибок валидатора	**20%**
Оформление и читаемость кода	**20%**

Формат сдачи

Архив с папкой проекта (index.html и img/) + скриншот результата проверки на validator.w3.org. Срок — до следующего практического занятия.

ЗАКЛЮЧЕНИЕ

Основные выводы

1 HTML описывает структуру

Это язык разметки, а не программирования: он определяет, из каких элементов состоит документ.

2 Документ имеет строгий скелет

DOCTYPE → <html> → <head> (служебное) и <body> (видимое пользователю).

3 Элементы, теги, атрибуты

Элемент = открывающий тег + содержимое + закрывающий тег. Атрибуты уточняют его свойства.

4 Смысл важнее внешнего вида

Заголовки, списки, и alt выбираются по значению, а оформление задаёт CSS.

Следующая лекция: Семантический HTML5 и доступный веб-документ