Практическая работа №1 «Введение в HTML. Заголовки, форматирование текста, якоря и гиперссылки, списки, изображения, фавикон»

HTML (**H**yper**T**ext **M**arkup **L**anguage — «язык гипертекстовой разметки») – «скелет», каркас веб-сайта. HTML обеспечивает структуру контента, появляющегося на веб-сайте, такого как изображения, текст или видео. Щелкните правой кнопкой мыши на любой странице в Интернете, выберите «Просмотреть код» (или нажмите комбинацию клавиш *Ctrl+U* – для Windows, *Cmd+Option+U* – для macOS*)*, и вы увидите разметку этой страницы.

- Язык разметки (*markup language*) — это компьютерный язык, который определяет структуру и представление необработанного текста с помощью тегов и атрибутов.
- Гипертекст (*HyperText*) — это текст, отображаемый на компьютере или устройстве, который обеспечивает доступ к другому тексту через ссылки, также известные как гиперссылки.

Слово «язык» в HTML стоит воспринимать как правила. Сам по себе HTML только размечает данные, но никак с ними не взаимодействует и визуально с ними ничего не делает. Всю работу по выводу текстовых данных и разметки берёт на себя браузер.

1 Структура языка HTML

HTML состоит из элементов. Эти элементы структурируют веб-страницу и определяют ее содержимое (рисунок 1.1).

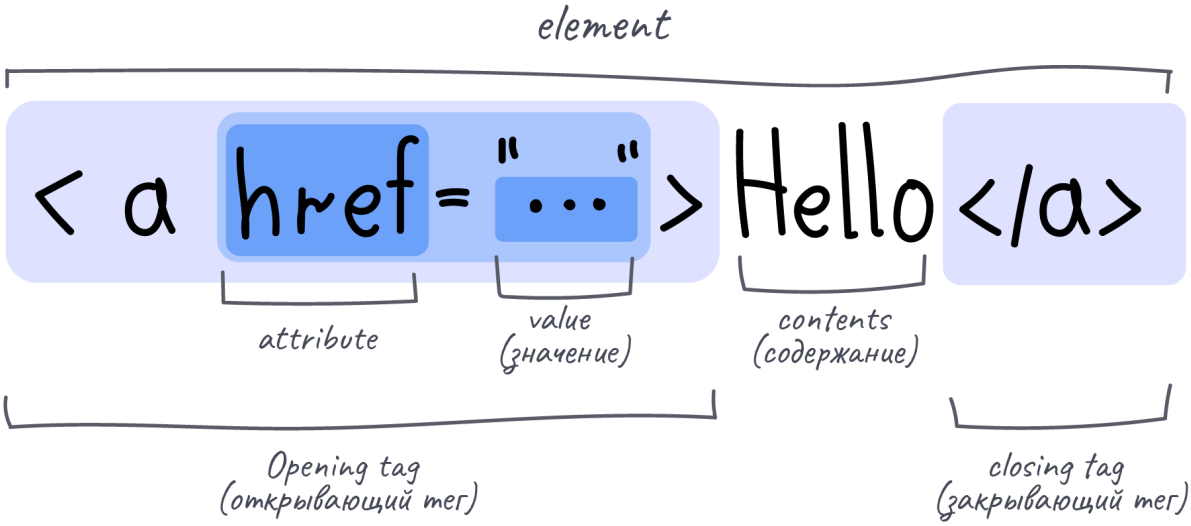

Рисунок 1.1 - Общая схема элементов HTML

1.2 Что такое «тег» (Tag) и «элемент» (Element)?

Теги HTML обеспечивают семантическое значение и машиночитаемость содержимого на странице. Элемент обычно состоит из открывающего тега (**<element_name>**) и закрывающего тега (**</element_name>**), которые содержат имя элемента, заключенное в угловые скобки, и содержимое между ними:

```
<element_name>...содержимое (content)...</element_name>
```

Теги, состоящие из двух частей – открывающего и закрывающего тегов – **называются парными тегами.**

Пример использования парного тега абзаца <p>:

```
<p>Это простой абзац.</p>
```

При **использовании нескольких тегов** элементы должны правильно закрываться с учетом вложенности одного элемента в другой. Пример:

```
<p><em>Тег <em> придает особый акцент тексту.</em></p>
```

Некоторые элементы HTML **не имеют закрывающего тега** или **какого-либо содержимого**. Они называются **пустыми элементами**. Например, пустые элементы включают теги <img>, <meta>, <link>, <input>, <hr> и
, которые **называются одиночными тегами**.

Имена элементов можно рассматривать как описательные ключевые слова для контента, который они содержат, например, video, audio, table, footer.

HTML-страница может состоять потенциально из сотен элементов, которые затем считываются веб-браузером, интерпретируются и преобразуются в удобочитаемый или слышимый контент на экране.

1.3 Что такое «атрибут» (Attribute)?

Атрибуты содержат дополнительную информацию для браузера. Атрибуты имеют форму открывающего тега, внутри которого размещается дополнительная информация. У тега может быть как один атрибут, так и несколько – в этом случае они пишутся через пробел.

Пример использования тега <img> с атрибутами src, alt и title:

```
<img src="moon.jpg" alt="Фотография луны на небе" title="Луна">
```

В данном случае источник изображения *source* (`src`), замещающий (*alternative*) изображение текст (`alt`) и всплывающая подсказка с текстом, которая появляется при наведении курсора на элемент – изображение, – являются атрибутами тега `<img>`.

2 Базовая структура HTML-документа

Базовая структура любого HTML-документа представлена на листинге 1.1 и рисунке 1.2:

Листинг 1.1 – Базовая структура HTML-документа

```
<!DOCTYPE html>
<html lang="ru">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Моя первая страница</title>
  </head>
  <body>
  </body>
</html>
```

Рисунок 1.2 – Визуальное представление базовой структуры веб-страницы

Эти теги должны располагаться друг под другом в верхней части каждой HTML-страницы, которую вы создаете (таблица 1.1).

Таблица 1.1 – Значение тегов из базовой структуры HTML-документа

Тег (Tag)	Значение
`<!DOCTYPE html>`	Определяет версию HTML, используемую в документе. В данном случае это HTML5.
`<html>`	Открывает страницу и обозначает то, что с этого момента мы будем писать HTML-код. После закрывающего тега `</html>` не должно быть никакой разметки. Атрибут `lang` объявляет основной язык страницы с использованием языковых кодов ISO.
`<head>`	Открывает головной раздел, который не отображается в главном окне браузера. Здесь находятся все метаданные страницы – в основном они предназначены для поисковых систем и других компьютерных программ. Раздел также может содержать импорт из внешних таблиц стилей и скриптов. Закрывающий тег `</head>`.
`<meta>`	Предоставляет браузеру некоторые метаданные о документе. Здесь хранится информация о документе: кодировка символов, название (контекст страницы), описание. Атрибут `charset` объявляет кодировку символов. Современные HTML-документы всегда должны использовать кодировку UTF-8, даже если это не является обязательным требованием. Атрибут `name` со значением `viewport` задает настройки области просмотра, т.е. контролирует масштаб отображения страницы на устройствах. Атрибут `content` со значениями `width=device-width` и `initial-scale=1.0` задает ширину видимой области равной ширине устройств при отсутствии изначального масштабирования. Мета-теги с вышеупомянутыми атрибутами должны стоять первыми в `<head>` для последовательного обеспечения правильного отображения документа. Любой другой элемент заголовка должен идти после этих тегов. В HTML тег `<meta>` не требует закрывающего тега.
`<title>`	Здесь мы вставляем имя страницы, которое будет отображаться в верхней части окна или вкладки браузера. Текст, написанный между этими открывающим `<title>` и закрывающим `</title>` тегами, будет отображаться на вкладке страницы или в строке заголовка браузера.
`<body>`	Здесь находится содержимое страницы. Тег открывает часть документа, отображаемую для пользователей, т.е. весь видимый или слышимый контент страницы. Никакой контент не должен добавляться после закрывающего тега `</body>`.

Дополнительная информация

- Подробнее с мета-тегом (<meta>) viewport и его атрибутами с их значениями можно ознакомиться по ссылке – URL: https://developer.mozilla.org/ru/docs/Glossary/Viewport (на русск. яз.)
- Простое руководство по элементам HTML <head> (гайд, содержащий список элементов, которые можно размещать в <head>) - URL: https://htmlhead.dev/ (на англ. яз.)

Комментарии в коде HTML

HTML-комментарии можно использовать, чтобы оставлять заметки для себя или других разработчиков об определенном месте кода. Они могут быть инициированы с помощью <!-- и завершены с помощью -->, например: <!-- Это HTML-комментарий -->.

Они могут занимать несколько строк, чтобы предоставить дополнительную информацию и могут быть встроены в другой контент **между тегами**, например:

```
<h1>Для комментария <!-- Это HTML-комментарий между тегами --></h1>.
```

Но комментарии **не могут появляться внутри самого HTML-тега**, например:

```
<h1 <!-- Это HTML-комментарий внутри тега -->>Для комментария</h1>
```

3 Вложенные элементы HTML

HTML организован как набор отношений генеалогического древа. Когда элемент содержится внутри другого элемента, он считается дочерним элементом этого элемента. Говорят, что дочерний элемент вложен внутрь родительского элемента. Пример представлен на листинге 1.2:

Листинг 1.2 – Пример дочернего элемента

```
<body>
    <p>Этот абзац - дочерний элемент body</p>
</body>
```

В приведенном выше примере элемент <p> вложен внутри элемента <body>. Элемент <p> считается дочерним по отношению к элементу <body>, а элемент <body> считается родителем.

Давайте рассмотрим более сложный пример, в котором используются некоторые новые теги (листинг 1.3):

Листинг 1.3 – Пример родственных элементов-потомков

```
<body>
    <div>
      <h1>Родственник p и потомок body</h1>
      <p>Родственник h1 и потомок body</p>
    </div>
</body>
```

В этом примере элемент <body> является родителем элемента . Оба элемента <h1> и <p> являются потомками элемента . Поскольку элементы <h1> и <p> находятся на одном уровне, они считаются родственными элементами и являются потомками элемента <body>.

Понимание иерархии HTML важно, потому что дочерние элементы могут наследовать поведение и стиль своего родительского элемента. Больше об иерархии веб-страниц можно будет узнать при изучении CSS.

4 Заголовки в HTML

Заголовки в HTML аналогичны заголовкам в других типах медиа. Например, в газетах большие заголовки обычно используются для привлечения внимания читателя. В других случаях заголовки используются для описания контента, например, названия фильма или образовательной статьи.

HTML следует аналогичному шаблону. В HTML существует шесть различных заголовков или элементов заголовков. Заголовки можно использовать для различных целей, например, для заголовков разделов, статей или других форм контента.

Ниже приведен листинг 1.4 с кодом для создания заголовков и подзаголовков, доступных в HTML, и результатом его выполнения (рисунок 1.3). Они упорядочены от большего размера к меньшему.

Листинг 1.4 – Создание заголовков и подзаголовков

```
<h1>Один</h1><!-- Используется для основных заголовков.
Все остальные заголовки используются для подзаголовков. -->
<h2>Два</h2>
<h3>Три</h3>
<h4>Четыре</h4>
<h5>Пять</h5>
<h6>Шесть</h6>
```

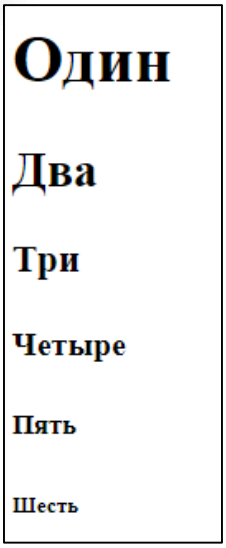

Рисунок 1.3 – Заголовки от `h1` до `h6`

Поисковые системы и другие пользовательские агенты обычно индексируют содержимое страницы на основе элементов заголовков, например, для создания оглавления, поэтому важно использовать правильную структуру заголовков.

Как правило, статья должна иметь один элемент `h1` для основного заголовка, за которым следуют подзаголовки `h2` — при необходимости спускаясь на уровень ниже. Если на более высоком уровне есть элементы `h1`, их нельзя использовать для описания контента более низкого уровня.

5 Форматирование текста в HTML

Добавить текст на HTML-страницу очень просто, используя элемент, открытый с помощью тега `<p>`, который создает новый абзац. Мы помещаем весь наш обычный текст внутрь элемента `<p>`. Также для добавления разрыва строки и вывода на страницу изначально заданного форматирования текста используются теги `<br>` и `<pre>`, соответственно (таблица 1.2).

Таблица 1.2 – Описание тегов для добавления текста на страницу

Тег (Tag)	Описание
`<p>`	Создает абзац, который можно представить как блок, внутри которого все, что находится, начинается с новой строки. Между блоками `<p>` появляется вертикальный отступ, величину которого в дальнейшем можно будет задать с помощью стилей (CSS).
` `	Тег указывает на место разрыва строки и принудительного переноса текста на новую строку *(по стандарту используется только для этой цели)*. Применяется, например, для деления стихотворения по строкам.

Продолжение таблицы 1.2

<pre>	Нужен для отображения предварительно отформатированного текста, если необходимо сохранить все пробелы и переносы в HTML.

Сравнение результатов применения тегов <p> и <pre> при добавлении в HTML-код предварительно отформатированного текста, а именно части стихотворения (листинги 1.5, 1.6 и рисунки 1.4, 1.5).

Листинг 1.5 – Использование тега абзаца <p> в коде

```
<body>
    <p>
        Серые глаза — рассвет,
        Пароходная сирена,
        Дождь, разлука, серый след
        За винтом бегущей пены.
        ...
    </p>
</body>
```

Серые глаза — рассвет, Пароходная сирена, Дождь, разлука, серый след За винтом бегущей пены. ...

Рисунок 1.4 – Результат выполнения кода с тегом абзаца <p>

Листинг 1.6 – Использование тега отформатированного текста <pre> в коде

```
<body>
    <pre>
        Серые глаза — рассвет,
        Пароходная сирена,
        Дождь, разлука, серый след
        За винтом бегущей пены.
        ...
    </pre>
</body>
```

```
Серые глаза — рассвет,
Пароходная сирена,
Дождь, разлука, серый след
За винтом бегущей пены.
...
```

Рисунок 1.5 – Результат выполнения кода с тегом отформатированного текста <pre>

В таблице 1.3 представлены основные теги для форматирования текста в HTML.

Таблица 1.3 – Основные теги для форматирования текста

Тег (Tag)	Описание	Назначение
`<b>`	**Bold** (жирный шрифт)	Используется для визуального выделения ключевых слов без добавления какой-либо важности или изменения тона.
`<strong>`	**Strong** («сильный/серьезный» шрифт)	Элемент сильной значимости указывает на то, что его содержание имеет большое значение и серьёзность.
`<i>`	*Italic* (курсив)	Представляет текст, который отличается от обычной прозы, например, иностранное слово, изречения вымышленного персонажа или, когда текст ссылается на определение слова вместо представления его семантического значения.
`<em>`	*Emphasised Text* (акцентированный текст)	Ударение на словесное содержание.
`<mark>`	Marked Text (выделенный текст)	Представляет текст, выделенный в справочных целях из-за своей актуальности в определённом контексте. Может быть использован на странице с результатом поиска, в которой выделяется каждый экземпляр искомого слова.
`<small>`	Small Text (мелкий шрифт)	Используется для установки маленького размера шрифта (например, для авторских прав, юридического текста) и для комментариев-сносок. Уменьшает размер шрифта на один размер (от среднего до мелкого, от x-large до крупного).
`<s>`	~~Strikethrough~~ (зачеркнутый)	Используется для указания того, что текстовое содержимое больше не является актуальным или точным.
`<del>`	~~Delete~~ (удаленный)	Используется для выделения замененного или удаленного текста.
`<u>`	Unarticulated Text (ранее Underlined – до HTML4) (неясный/невнятный текст)	Используется для текста, имеющего некоторую форму нетекстовой аннотации. Используется для подчеркивания слов с ошибками (элемент следует избегать, если его можно спутать с гиперссылкой).
`<ins>`	Inserted Text (вставленный текст)	Отображается с подчеркиванием, чтобы показать вставленный текст (например, в новую версию документа). Позволяет отследить сделанные в документе изменения.

Продолжение таблицы 1.3

<sub>	Subscript Text (подстрочный текст)	Подстрочное начертание текста. Используется, например, для обозначения некоторых математических переменных в виде текста нижнего индекса.
<sup>	Superscript Text (надстрочный текст)	Надстрочное начертание текста. Используется, например, для обозначения некоторых математических переменных в виде текста верхнего индекса.
<q>	«Quotation» (цитирование)	Используется для выделения в тексте кратких цитат с помощью кавычек.
<cite>	*Cite* (ссылка на источник цитаты)	Обозначается курсивным начертанием. Используется для названия произведения, например, названия книги или фильма. *Не путать:* для указания ссылки на первоисточник цитаты используется атрибут cite в сочетании с элементом <blockquote>.
<blockquote>	Blockquote (цитирование)	В HTML4 использовался для выделения в тексте длинных цитат, т.е. цитат, которые занимают несколько строк. В HTML5 указывает раздел, цитируемый из другого источника.
	Span (диапазон)	Представляет собой универсальный встроенный контейнер для встроенных элементов и содержимого. Он используется для группировки элементов с целью дальнейшего их изменения с помощью стилей (CSS) (с использованием атрибутов class или id). Лучший способ использовать его, когда нет другого семантического элемента.
<abbr>	Abbreviation (Аббревиатура)	Используется для определения аббревиатуры или краткой формы элемента. Теги <abbr> и <acronym> используются как сокращенные версии и используются для представления последовательности букв. Аббревиатура используется для предоставления полезной информации браузерам, системам перевода и поисковым системам. Применяется с атрибутом title. Он используется для указания дополнительной информации об элементе. Когда мышь перемещается по элементу, отображается информация.

Примечание: в дальнейшем оформление текста по некоторым вышеописанным стилям возможно через свойство CSS text-decoration.

6 Блочный элемент

Одним из самых популярных элементов HTML является элемент . — это сокращение от «division» или контейнера (*container*), который делит страницу на разделы. Эти разделы очень полезны для группировки элементов в HTML.

Элементы по своей сути не имеют визуального представления, но они очень полезны, когда необходимо применить пользовательские стили к нашим элементам HTML. Элементы позволяют группировать элементы HTML, чтобы применять одинаковые стили ко всем элементам HTML внутри. Можно стилизовать элемент в целом.

Элементы могут содержать любой текст или другие элементы HTML, такие как ссылки, изображения или видео.

7 Якоря и гиперссылки в HTML

Как вы могли заметить, Интернет состоит из множества ссылок. Почти все, на что вы нажимаете во время веб-серфинга, является ссылкой, которая ведет на другую страницу веб-сайта, который вы посещаете, или на внешний сайт. Ссылки включаются в атрибут, открываемый тегом <a>. В таблице 1.4 представлены атрибуты этого тега.

Таблица 1.4 – Атрибуты тега <a>

Атрибут (Attribute)	Описание
href	Указывает адрес назначения. Это может быть абсолютный или относительный URL-адрес или имя (name) якоря. Абсолютный URL — это полный URL-адрес веб-сайта, например, http://example.com/. Относительный URL-адрес указывает на другой каталог и/или документ на том же веб-сайте, например, /about-us/ указывает на каталог «о нас» внутри корневого каталога (/). При указании на другой каталог без явного указания документа веб-серверы обычно возвращают документ «index.html» внутри этого каталога.
hreflang	Идентифицирует язык текста по ссылке.
rel	Указывает связь между текущим документом и связанным документом.
target	Указывает, где открыть ссылку, например, в новой вкладке или окне. Возможные значения: _blank, _self, _parent, _top.
title	Указывает дополнительную информацию о ссылке. Информация чаще всего отображается в виде всплывающей подсказки при наведении курсора на ссылку. Этот атрибут не ограничивается ссылками, его можно использовать почти во всех тегах HTML.

Продолжение таблицы 1.4

download	Указывает, что цель будет загружена, когда пользователь щелкнет гиперссылку. Значением атрибута будет имя загружаемого файла. Ограничений на допустимые значения нет, а браузер автоматически определит правильное расширение файла и добавит его к файлу (.img, .pdf и т.д.). Если значение опущено, используется исходное имя файла.

7.1 Ссылка на другой сайт

Основное простое использование элемента <a>:

```
<a href="https://www.google.ru/">Пример ссылки</a>
```

Он создает гиперссылку на URL-адрес https://www.google.ru/, как указано в атрибуте href (hypertext reference - гипертекстовая ссылка), с текстом привязки «Пример ссылки».

Пример ссылки с атрибутами href и rel:

```
<a href="https://www.google.ru/" rel="external">Пример ссылки</a>
```

Атрибут rel со значением external показывает, что ссылка ведет на внешний интернет-ресурс.

7.2 Ссылка-якорь

Якоря можно использовать для перехода к определенным тегам на HTML-странице. Тег <a> может указывать на любой элемент, имеющий атрибут id. Якоря в основном используются для перехода к подразделу страницы и используются в сочетании с тегами заголовков.

Например, вы создали страницу (index.html) со множеством тем или разделов (листинг 1.7):

Листинг 1.7 – Создание тем на странице

```
<h2>Первая тема</h2>
<p>Описание первой темы</p>
<h2>Вторая тема</h2>
<p>Описание второй темы</p>
```

Если у вас есть несколько разделов, вы можете создать оглавление в верхней части страницы с быстрыми ссылками (или закладками) на определенные разделы. Для этого

необходимо присвоить темам или разделам атрибут `id`, чтобы можно было ссылаться на них (листинг 1.8).

Листинг 1.8 – Присвоение темам атрибута `id`

```
<h2 id="Topic1">Первая тема</h2>
<p>Описание первой темы</p>
<h2 id="Topic2">Вторая тема</h2>
<p>Описание второй темы</p>
```

И теперь вы можете использовать **якорь** – **символ #** и **`id`** темы или раздела – в своем оглавлении (содержании страницы) (листинг 1.9):

Листинг 1.9 – Создание оглавления через якоря

```
<h1>Содержание страницы</h1>
     <a href="#Topic1">Нажать для перехода на первую тему</a>
     <a href="#Topic2">Нажать для перехода на вторую тему</a>
```

Эти якоря также привязаны к веб-странице, на которой они находятся (`index.html`). Таким образом, вы можете ссылаться на тему/заголовок на другой странице с помощью названия файла самой страницы и имени якоря:

```
<a href="index.html#Topic1">Вернуться к первой теме на главной
                            странице</a>
```

7.3 Ссылка на номер телефона

Если значение атрибута `href` начинается с `tel:`, тогда ваше устройство будет набирать номер при нажатии на него. Это работает на мобильных устройствах или на компьютерах/планшетах с программным обеспечением, которое может совершать телефонные звонки:

```
<a href="tel:01234567890">Позвонить по номеру</a>
```

7.4 Ссылка на почтовый клиент

7.4.1 Основное использование

Если значение атрибута `href` начинается с `mailto:` браузер попытается открыть почтовый клиент по клику:

```
<a href="mailto:sendemail@gmail.com">Написать письмо</a>
```

Это поместит адрес электронной почты sendemail@gmail.com в качестве получателя для вновь созданного электронного письма.

7.4.2 Копия (CC) и скрытая копия (BCC)

Вы также можете добавить адреса для получателей копии (cc - carbon copy) или скрытой копии (bcc - blind carbon copy), используя следующий синтаксис:

```
<a href="mailto:sendemail@gmail.com?cc=copy@mail.ru&bcc=blindcopy@mail.ru">
                                Написать письмо</a>
```

7.4.3 Тема и основной текст

Вы также можете заполнить тему (subject) и тело (body) нового письма:

```
 <a href="mailto:sendemail@mail.ru?subject=Example+subject&body=Message+text">
                                Отправить письмо</a>
```

Эти значения должны быть закодированы в URL.

Щелчок по ссылке с mailto: попытается открыть почтовый клиент по умолчанию, указанный вашей операционной системой, или попросит вас выбрать, какой клиент вы хотите использовать. Не все параметры, указанные после адреса получателя, поддерживаются во всех почтовых клиентах.

8 Списки в HTML

HTML предлагает три способа задания списков: упорядоченные списки, неупорядоченные списки и списки описаний. Упорядоченные списки используют порядковые последовательности (числовые или алфавитные) для указания порядка элементов списка, неупорядоченные списки используют определенный символ, такой как маркер, для перечисления элементов в произвольном порядке, а списки описания используют отступы для перечисления элементов с их дочерними элементами.

8.1 Упорядоченный список (Ordered List)

Для создания упорядоченного списка используется парный тег <ol>. Внутри него добавляются элементы списка с помощью тега <li> (*list item*) (листинг 1.10).

Листинг 1.10 – Пример упорядоченного (нумерованного) списка

```
<ol>
  <li>Жарко</li>
  <li>Тепло</li>
  <li>Прохладно</li>
  <li>Холодно</li>
</ol>
```

После выполнения кода на веб-странице отобразится следующий список (рисунок 1.6).

1. Жарко
2. Тепло
3. Прохладно
4. Холодно

Рисунок 1.6 – Упорядоченный нумерованный список

8.1.1 Атрибуты `start`, `value`, `type` и `reversed`

Есть несколько способов установить, какие числа появляются в элементах списка в упорядоченном списке. *Первый способ* – установить начальный номер, используя атрибут `start` (листинг 1.11). Список начнется с этого определенного номера и продолжится увеличивается на единицу, как обычно (рисунок 1.7).

Листинг 1.11 – Применение атрибута `start`

```
<ol start="5">
  <li>Жарко</li>
  <li>Тепло</li>
  <li>Прохладно</li>
  <li>Холодно</li>
</ol>
```

5. Жарко
6. Тепло
7. Прохладно
8. Холодно

Рисунок 1.7 – Изменение начального номера списка

Вы также можете установить для определенного элемента списка свое значение, благодаря атрибуту `value`. Дальнейшие элементы списка после него будут продолжать увеличиваться на единицу от значения этого элемента списка, игнорируя, где находился родительский список (листинг 1.12, рисунок 1.8):

Листинг 1.12 – Применение атрибутов `start` и `value`

```
<ol start="5">
  <li>Жарко</li>
  <li value="9">Тепло</li>
  <li>Прохладно</li>
  <li>Холодно</li>
</ol>
```

5. Жарко
9. Тепло
10. Прохладно
11. Холодно

Рисунок 1.8 – Изменение номера определенного элемента списка

Примечание: в значения атрибутов `start` и `value` можно ставить только числа, даже если упорядоченный список настроен на отображение в виде римских цифр или букв (листинг 1.13, рисунок 1.9).

Листинг 1.13 – Применение атрибутов `start` и `value` со списком в алфавитном порядке

```
<ol type="A" start="5">
  <li>Жарко</li>
  <li value="9">Тепло</li>
  <li>Прохладно</li>
  <li>Холодно</li>
</ol>
```

E. Жарко
I. Тепло
J. Прохладно
K. Холодно

Рисунок 1.9 – Применение атрибутов `start` и `value` с буквенным списком

Исходя из вышеприведенного примера кода (листинг 1.13) можно понять, что для изменения типа (*type*) числа, отображаемого в маркере элемента списка по умолчанию, используется атрибут `type`.

Описание типов нумерации представлено в таблице 1.5.

Таблица 1.5 – Типы нумерации упорядоченных списков

Тип (type)	Описание	Примеры
`1`	Десятичные числа (по умолчанию)	`1, 2, 3, 4, 5`
`a`	Строчные буквы в алфавитном порядке	`a, b, c, d, e`
`A`	Заглавные буквы в алфавитном порядке	`A, B, C, D, E`
`i`	Строчные римские цифры	`i, ii, iii, iv, v`
`I`	Заглавные римские цифры	`I, II, III, IV, V`

Также вы можете изменить нумерацию, сделав ее в обратном порядке с помощью атрибута `reversed` в элементе `ol` (листинг 1.14, рисунок 1.10):

Листинг 1.14 – Применение атрибутов `reversed` и `value`

```
<ol reversed>
  <li>Жарко</li>
  <li value="9">Тепло</li>
  <li>Прохладно</li>
  <li>Холодно</li>
</ol>
```

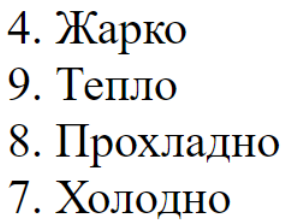

Рисунок 1.10 – Список с нумерацией в обратном порядке

Примечание: использование `<ol>` для отображения списка элементов необходимо в тех случаях, когда **важен порядок элементов и его следует подчеркнуть**. Если изменение порядка элементов не делает список неправильным, следует использовать `<ul>`.

8.2 Неупорядоченный список (Unordered List)

Неупорядоченный список можно создать с помощью тега `<ul>`, а каждый элемент списка также с помощью тега `<li>` (листинг 1.15, рисунок 1.11), и задать тип маркера (таблица 1.6).

Листинг 1.15 – Пример неупорядоченного (маркированного) списка

```
<ul>
  <li>Жарко</li>
  <li>Тепло</li>
  <li>Прохладно</li>
  <li>Холодно</li>
</ul>
```

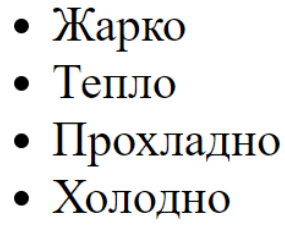

Рисунок 1.11 – Маркированный список

Таблица 1.6 – Типы маркеров неупорядоченного списка

Тип (type)		Описание
•	`disc`	Список с маркерами в виде круга
○	`circle`	Список с маркерами в виде окружности
▪	`square`	Список с квадратными маркерами

Примечание: использование `<ul>` для отображения списка элементов необходимо в тех случаях, когда **порядок элементов не важен**. Если изменение порядка элементов делает список неправильным, следует использовать `<ol>`.

8.3 Список описаний (Description List)

Список описаний (или список определений, как его называли до HTML5) можно создать с помощью элемента `dl`. Он состоит из групп «имя-значение» (*name-value*), где имя (термин) задается в элементе `dt` (*description list term*), а значение (описание) – в элементе `dd` (*description list description*) (листинг 1.16).

Листинг 1.16 – Список описаний

```
<dl>
     <dt>Термин 1</dt>
     <dd> Описание термина 1</dd>

     <dt>Термин 2</dt>
     <dd>Описание термина 2</dd>
     <dd>Описание термина 2</dd>

     <dt>Термин 3</dt>
     <dd>Описание термина 3</dd>
</dl>
```

Визуальное представление списка описаний на веб-странице показано на рисунке 1.12.

Термин 1
Описание термина 1
Термин 2
Описание термина 2
Описание термина 2
Термин 3
Описание термина 3

Рисунок 1.12 – Список описаний

Примечание: группа «имя-значение» может иметь более одного имени и/или более одного значения, которые представляют альтернативы.

9 Изображения в HTML

Тег HTML `<img>` (*image*) используется для добавления изображения в графическом формате GIF, JPEG, SVG или PNG на интернет-ресурс. В настоящее время веб-сайт не добавляет изображения на веб-страницу напрямую, поскольку изображения связаны с веб-страницами с помощью тега `<img>`, который содержит место для изображения. Также изображение можно сделать ссылкой на другую страницу, включив тег `<img>` в контейнер `<a>`. В таблице 1.7 представлены атрибуты тега `<img>`.

Пример синтаксиса с локальной ссылкой (значения `width` и `height` проставлены в px):

```
<img src="images/hello.png" alt="Hello World" width="300"
height="200">
```

Пример синтаксиса с URL-адресом:

```
<img src="https://www.pngall.com/wp-content/uploads/5/Hello-
PNG-Image.png" alt="Hello World">
```

Таблица 1.7 – Атрибуты тега `<img>`

Атрибут (Attribute)	**Описание**
`src`	Указывает локальный или URL-адрес изображения (*обязательный*).
`srcset`	Изображения для использования в различных ситуациях, например, на дисплеях с высоким разрешением, небольших мониторах и т.д. *Рекомендуется использовать с адаптивными изображениями.*
`alt`	Заменяющий текст для использования, когда изображения недоступны (*обязательный; при необходимости его значение можно оставлять пустым – для программ чтения с экрана, чтобы содержание страницы оставалось логичным в случае выполнения значения атрибута на странице*).
`width`	Ширина изображения. Используется в случае, когда нужно задать пространство, занимаемое изображением, чтобы обеспечить стабильность макета страницы перед его загрузкой
`height`	Высота изображения. Используется в случае, когда нужно задать пространство, занимаемое изображением, чтобы обеспечить стабильность макета страницы перед его загрузкой
`ismap`	Используется для указания изображения в качестве карты изображения на стороне сервера.
`usemap`	Используется для указания изображения в качестве карты изображения на стороне клиента.

Продолжение таблицы 1.7

`loading`	Используется, чтобы указать, должен ли браузер откладывать загрузку изображений до тех пор, пока не будут выполнены некоторые условия, или загружать изображение немедленно.
`decoding`	Представляет собой подсказку браузеру о том, как он должен декодировать изображение (синхронно, асинхронно, по умолчанию).
`referrerpolicy`	Он используется для указания того, какую информацию о реферере использовать при извлечении изображения, т.е. no-referrer, `no-referrer-when-downgrade`, `origin`, `origin-when-cross-origin`, `unsafe-url`.
`sizes`	Размеры изображений для различных макетов страниц. *Рекомендуется использовать с адаптивными изображениями.*
`crossorigin`	Обработка перекрестных запросов элементом.

Примечание: альтернативный текст (`alt`) используется программами чтения с экрана для слабовидящих пользователей и поисковыми системами. Поэтому важно написать хороший альтернативный текст для ваших изображений.

9.1 Иллюстрация с необязательной подписью <figure>

Элемент `<figure>` в HTML используется для добавления в документ самостоятельного содержимого, такого как иллюстрации, диаграммы, фотографии или коды. Это связано с основным потоком, но его можно использовать в любой позиции документа, и рисунок идет с потоком документа, и если его удалить, то это не должно влиять на поток документа. Иллюстрация часто добавляется с подписью (заголовком), которая указывается с помощью элемента `<figcaption>`. Иллюстрация и её подпись представляет собой единое целое. Этот тег является новым в HTML5.

Пример синтаксиса представлен в листинге 1.17:

Листинг 1.17 – Добавление элемента `<figure>`

```
<figure>
     <img src="/images/alpaca.jpg" alt="Альпака">
     <figcaption>Милая альпака</figcaption>
</figure>
```

Результат выполнения кода *(дополненного стилями)* на рисунке 1.4.

Рисунок 1.4 – Применение элемента <figure> и <figcaption> к изображению

9.2 Карта изображения

Карты изображений — это изображения с кликабельными областями, которые обычно действуют как гиперссылки. Изображение определяется тегом <img>, а карта определяется тегом <map> с тегами <area> для обозначения каждой кликабельной области. Атрибуты usemap и name используются для того, чтобы связать изображение и карту.

Есть три основных HTML-элемента, необходимых для создания сопоставленного изображения.

Карта <map>**:** используется для создания карты изображения с кликабельными областями.

Изображение <img>**:** используется в качестве источника изображения, для которого выполняется картографирование.

Область <area>**:** используется на карте для определения интерактивных областей.

Теги и атрибуты для создания карт изображений представлены в таблице 1.8.

Талица 1.8 – Связь тегов и атрибутов для карты изображения

Атрибут	Описание
usemap	Атрибут тега <img>. Имя карты с добавленным к нему символом # (решёткой). Например, для карты с name="map" изображение должно иметь usemap="#map".
name	Атрибут тега <map>. Название карты для ее идентификации. Взаимодействует с атрибутом usemap изображения.
alt	Атрибут тега <area>. Альтернативный текст в случае, если изображения не поддерживаются. Это необходимо только в том случае, если href также установлен в <area>.

Продолжение таблицы 1.8

coords	Атрибут тега <area>. Координаты, очерчивающие выбираемую область. Для shape="polygon" – координаты: "x, y", разделенных запятыми (shape="polygon" coords="x1, y1, x2, y2, x3, y3,... "). Для shape="rectangle" – координаты: left, top, right, bottom. Для shape="circle" – координаты: centerX, centerY, radius.
href	URL-адрес гиперссылки, если он указан. Если его опустить, <area> не будет представлять собой гиперссылку.
shape	Форма <area>. Позволяет установить значение по умолчанию для выбора всего изображения (атрибут coords не требуется), circle или circ для окружности, rectangle или rect для прямоугольника и polygon или poly для многоугольной области, заданной угловыми точками.

Пример синтаксиса для создания карты изображений (листинг 1.18):

Листинг 1.18 – Карты изображений

```
<img src="/images/mapshapes.jpg" usemap="#shapes">
<map name="shapes">
      <area shape="polygon" coords="79,6,5,134,153,134">
      <area shape="rectangle" coords="177,6,306,134">
      <area shape="circle" coords="397,71,65">
</map>
```

Примечание: браузер распознает области, когда курсор становится указателем.

10 Фавикон в HTML

Фавикон (*favicon*) – это крошечный значок, который отображается в адресной строке браузера, вкладках страниц и меню закладок.

Однако, большинство современных браузеров заменили фавикон в адресной строке изображением, указывающим, использует ли веб-сайт HTTPS.

Фавикон в стандартных условиях может быть размером 32 x 32 или 16 x 16 пикселей (если при сжатии фавиконка 32 x 32 пикселя будет размытой) и хранится в формате GIF, PNG или ICO. Также можно использовать формат SVG, если современный браузер умеет работать с векторными фавиконками.

Они используются для улучшения пользовательского опыта и обеспечения согласованности бренда. Например, когда в адресной строке браузера отображается знакомый значок, это помогает пользователям понять, что они находятся в нужном месте.

Фавикон подключается в коде между тегами <head> и </head> после элемента <title> (можно подключать сразу несколько форматов фавикон):

- PNG-фавиконка:

```
<link rel="icon" type="image/png" href="/favicon.png">
```

- icon-фавиконка (*.ico):

```
<link rel="icon" href="/favicon.ico" sizes="any">
```

- SVG-фавиконка:

```
<link rel="icon" type="image/svg+xml"
  href="images/favicons/icon.svg">
```

Примечание: фавиконку в формате ico необходимо класть в корень проекта, так как любой веб-сервер всегда ищет favicon.ico в корне проекта и пытается её подключить к сайту.

Практическое задание

1. **Создать файл «index.html»** и единую папку под практические работы (и под каждую из практических работ).

2. **Продумать тематику своей веб-страницы** для разработки ее элементов на практической работе №1 (и последующих) – если уже выбрали тему курсовой работы, можно взять ее. Темы интернет-ресурсов в рамках группы повторяться не могут.

3. При написании кода этой практической работы ко всем основным элементам страницы необходимо **добавлять комментарии**.

Список того, что обязательно должно быть на странице:

1. Заголовки от уровня h1 до уровня h3 (по желанию можно взять и меньшие).
2. Форматирование текста с тегами `<p>`, `<br>`, `<pre>` из таблицы 1.2.
3. Форматирование текста с тегами `<b>`, `<strong>`, `<i>`, `<em>`, `<u>`, `<sub>`, `<sup>`, `<cite>`, `<blockquote>`, `<abbr>` из таблицы 1.3.
4. Якоря на два раздела страницы: заголовок (уровня h1) и подзаголовок (h2) в конце страницы.
5. Ссылка на внешний ресурс по нажатию на слово/словосочетание в тексте.
6. Ссылка на номер телефона – контакт можно добавить внизу страницы.
7. Ссылка на почтовый клиент – можно добавить также внизу страницы.
8. Упорядоченный список с атрибутами `type` – не использовать значение атрибута по умолчанию – и `start`.
9. Вложенный список – маркированный список вложить в упорядоченный.
10. Список описаний – минимум 5 терминов (у двоих терминов должны быть альтернативные описания).
11. Два изображения (по желанию можно больше):
    - Изображение-ссылка на внешний ресурс (добавить атрибуты `width` и `height` к изображению);
    - Изображение-карта с тремя областями-ссылками (круг, прямоугольник, многоугольник).
12. Фавиконка.

Плагины и горячие клавиши Visual Studio Code (названия кликабельны)

1. **20 сочетаний клавиш для ускорения работы в VS Code (с gif – примерами)**
2. **10 горячих клавиш VS Code, которые ускорят вашу работу**
3. **Горячие клавиши Visual Studio Code**
4. **8 полезных плагинов VS Code для вёрстки**
5. **Сокращенные команды Emmet (для VSCode)**
6. **Тренажер по Emmet**

Сайты - справочники по HTML, CSS, JavaScript (названия кликабельны)

1. **htmlbook**
2. **html5book** (https://html5book.ru/)
3. **WebReference** (**Теги HTML и HTML5:** https://webref.ru/html)
4. **Веб-технологии для разработчиков**

Что учить и где учиться веб-разработчику? (названия кликабельны)

1. **Что учить веб-разработчику:** проект, на котором в удобной форме собраны большинство инструментов/тем с пояснениями и ссылками на ресурсы (*HTML, CSS, JS, PWA, фреймворки, тестирование, безопасность, архитектура, дизайн, алгоритмы, базы данных*)
2. **Интерактивные курсы htmlacademy (на русс. яз.)**
3. **Интерактивные курсы codecademy (на англ. яз.)**