**Вопросы для зачета:**

* HTML:

HyperText Markup Language — язык гипертекстовой разметки.

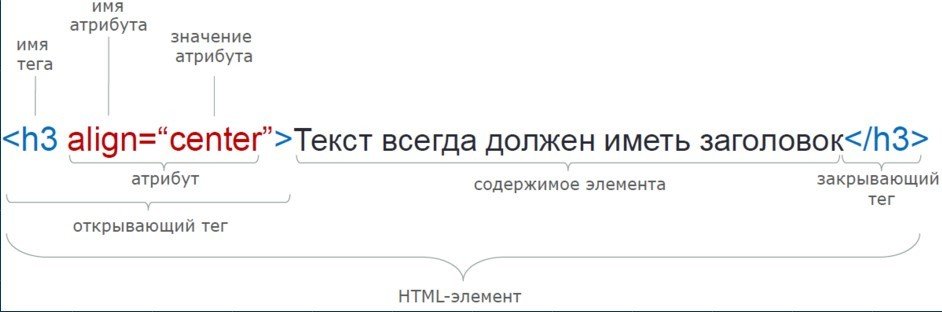
**(H)Hyper(T)Text** - это совокупность документов, содержащих текстовую, аудио и видеоинформацию, связанных между собой.

(**M)Mark-up** - это разметка. Вы размечаете текст так же, как вы делаете это в текстовых редакторах: выставляете заголовки, списки, выделяете текст жирным шрифтом и т. д.

**(L)Language** - это язык - HTML. В нём используется много английских слов.

## **Tag Structure**

* + что такое теги,
* <tag>
* Tag - это специальный управляющий код, записываемый в угловых скобках.
* Если HTML - это язык разметки, то tag - это элемент разметки. Теги на странице отображаются в преобразованном виде.



### **Тэги глобально деляться на парные и непарные:**

### **Тэги глобально деляться на парные и непарные:**

<div></div>

<span></span>

<header></header>

<a></a>

...и многие другие

<br />

<img />

<input />

<link />

<meta />

...и немногие другие

## **Атрибуты**

Для атрибутов тегов используются значения по умолчанию

Атрибуты без значений

<**form** action="self.php">

<**p**><**input** type="text"></**p**>

<**p**><**input** type="submit" disabled></**p**>

</**form**>

В данном примере используется атрибут disabled, у которого явно не задано значение. Подобная запись называется «сокращенный атрибут тега».

Порядок атрибутов в любом теге не имеет значения и на результат отображения элемента не влияет.

### **Формат атрибутов**

Текст

<**input** type="password">

Число

<**p**><**textarea** rows="10" cols="45" name="text"></**textarea**></**p**>

Ссылка

<**img** src="images/girl.png" width="189" height="255" alt="boy">

<**link** rel="stylesheet" href="iLoveIE.css">

Атрибуты могут располагаться как в одну строку, так и в несколько. Главное условие - наличие разделительного символа

<**img** src="images/girl.png" width="189" height="255" alt="boy">

<**img** src="images/girl.png"

width="189"

height="255"

alt="boy">

## **Атрибуты data-\***

data-  это атрибут, который позволяет хранить разную информацию, которая может помочь в работе скриптов, а также для оформления элементов через CSS

<**ul**>

<**li** data-damage-resistance="40" data-weight="45"

data-effect="+10 Сопротивляемость радиации">

Броня Купревича</**li**>

<**li** data-damage-resistance="40" data-weight="45"

data-effect="+15 Интелект">

Силовая броня Епама</**li**>

<**li** data-damage-resistance="50" data-weight="40"

data-effect="+25 Репутация">

Тяжелая броня Rolling scopes</**li**>

</**ul**>

* + какие основные теги существуют,

## **Document Structure**

**Структура документа**

1. <!DOCTYPE html> - актульная краткая запись

2. <html>

  2.1. раздел заголовка <head> - содержимое не

  отображается на странице, но влияет на отображение <body>

  2.2. раздел тела документа <body> - содержимое

  отображается на странице

<head>

    <title> - заголовок документа, отображается во вкладке, в поисковике, т.д.

    <link> - подключение сторонних файлов к документу - css, favicon и т.д.

    <meta> - информация для браузеров и поисковых систем (кодировка, ключевые слова)

    <sсript> - подключение внешних скриптов

 </head>

<body>

    Здесь записывается та информация, которая будет отображена на  экране (списки, заголовки, меню, параграфы, картинки и т.д.)

 </body>

**<h1>**Заголовок**</h1>**

HTML предлагает шесть текстовых заголовков разного уровня, которые показывают относительную важность секции, расположенной после заголовка. Так, тег **<h1>** представляет собой наиболее важный заголовок первого уровня, а тег **<h6>** служит для обозначения заголовка шестого уровня и является наименее значительным. По умолчанию, заголовок первого уровня отображается самым крупным шрифтом жирного начертания, заголовки последующего уровня по размеру меньше. Теги **<h1>...<h6>** относятся к блочным элементам, они всегда начинаются с новой строки, а после них другие элементы отображаются на следующей строке. Кроме того, перед заголовком и после него добавляется пустое пространство.

<!-- Комментарий -->

Некоторый текст можно спрятать от показа в браузере, сделав его комментарием. Хотя такой текст пользователь не увидит, он все равно будет передаваться в документе, так что, посмотрев исходный код, можно обнаружить скрытые заметки.

Комментарии нужны для внесения в код своих записей, не влияющих на вид страницы. Начинаются они тегом **<!--** и заканчиваются тегом **-->**. Все, что находится между этими тегами, отображаться на веб-странице не будет.

**<p>**Первый абзац.**</p>**

Тег **<p>** определяет абзац (параграф) текста. Если закрывающего тега нет, считается, что конец абзаца совпадает с началом следующего блочного элемента.

**<p>**Второй абзац.**</p>**

Тег **<p>** является блочным элементом, поэтому текст всегда начинается с новой строки, абзацы идущие друг за другом разделяются между собой отбивкой (так называется пустое пространство между ними). Это хорошо видно на рис. 4.1.

* + блочные vs строчные элементы,

В HTML вам, в основном, будут попадаться два типа элементов HTML.

**Блочные** элементы, вроде:

* абзацы <р>;
* списки: неупорядоченные (с маркером) <ul> или упорядоченные списки (с числами) <ol>;
* заголовки: от первого уровня <h1> до шестого уровня <h6>;
* статьи <article>;
* разделы <section>;
* длинные цитаты <blockquote>.

**Строчные** элементы, вроде:

* ссылки <a>;
* выделенные слова <em>;
* важные слова <strong>;
* короткие цитаты <q>;
* аббревиатуры <abbr>.

**Блочные** элементы предназначены для **структурирования** основных частей вашей страницы, путём разделения содержимого на логически связанные блоки.

**Строчные** элементы предназначены, чтобы разграничить часть текста и придать ему определённую функцию или смысл.

## **Открывающие и закрывающие теги**

**Все** блочные элементы содержат открывающие и закрывающие теги.

Как результат, самозакрывающие элементы являются **строчными**, просто потому, что их синтаксис не позволяет им содержать любой другой элемент HTML.

|  |  |  |
| --- | --- | --- |
|  | Содержит открывающие и закрывающие теги | Самозакрывающие |
| Блочные элементы | <p> </p> <ul> </ul> <ol> </ol> | **Невозможно** |
| Строчные элементы | <a> </a> <strong> </strong> <em> </em> | <input> <br> <img> |

## **Другие типы элементов HTML**

Есть несколько исключений у блочных/строчных элементов. Вот те, с которыми вы чаще всего столкнётесь:

* пункты списка <li>;
* таблица <table>;
* строки таблицы <tr>;
* ячейки таблицы <td>.
  + способы подключения скриптов/стилей,

## **Adding CSS to HTML**

### **1. Инлайновый (внутренний стиль для элемента)**

С помощью атрибута style задаются стилевые параметры конкретного элемента

<**body**>

<**div** style="background-color:red;">

</**div**>

</**body**>

### **2. Глобальные стили**

C помощью тега <style> таблица внедряется в любую часть HTML-документа

<**head**>

<**style**>

**div**{

background-color:red;

}

</**style**>

</**head**>

<**body**>

<**div**>Наш блок</**div**>

</**body**>

### **3. Через внешний файл (подключается в <head>)**

Таблица стилей хранится в отдельном файле с расширением  .css, который связывается с HTML-документом с помощью тега <link>

<**head**>

<**link** rel="stylesheet" href="css/styles.css" />

</**head**>

<**body**>

<**div**>Наш текст</**div**>

</**body**>

Внутри файла .css будет прописано:

**div**{

background-color:red;

}

### **4. Через правило @import внутри CSS.**

В текущую стилевую таблицу можно импортировать содержимое CSS-файла с помощью команды [@import](http://htmlbook.ru/css/import" \t "_blank). Этот метод допускается использовать совместно со связанными или глобальными стилями, но никак не с внутренними стилями

**@import** "style/all.css" all; /\* стили для всех устройств \*/

**@import** "style/scr\_hand.css" screen, handheld; /\* стили для мониторов и телефонов \*/

**@import** "style/print.css" print; /\* стили для печати \*/

**BODY** {

font-family: Arial, Verdana, Helvetica, sans-serif;

font-size: 90%;

background: white;

color: black;

}

/\*

Типы:

all, aural, braille, handheld, print, projection, screen, tv

\*/

## **Adding JavaScript to HTML**

### **1. В любое место HTML файла**

C помощью тега <script>

<**head**>

<**script**>

console.log("Сообщение 1");

</**script**>

</**head**>

<**body**>

<**script**>

console.log("Сообщение 2");

</**script**>

</**body**>

### **2. Через внешний файл**

Доступ к внешнему файлу обеспечивается с помощью атрибута src. При указании атрибута src содержимое тега игнорируется.

**Универсальная запись**

<**head**>

<**script** type="text/javascript" src="js/slider.js"></**script**>

<**head**>

<**body**>

<**h1**>This is JS!</**h1**>

<**script** type="text/javascript" src="js/menu.js"></**script**>

</**body**>

### **3. async, defer**

* If async is present: The script is executed asynchronously with the rest of the page (the script will be executed while the page continues the parsing)
* If async is not present and defer is present: The script is executed when the page has finished parsing

<**body**>

<**script** type="text/javascript" src="js/menu.js" async></**script**>

<**h1**>This is JS!</**h1**>

<**script** type="text/javascript" src="js/menu.js" defer></**script**>

</**body**>

* + что такое семантика,
  + что такое валидация HTML.
* CSS:

CSS (Cascading Style Sheets "каскадные таблицы стилей") - это набор правил, описывающих способы отображения и расположения элемента.

HTML описывает контент, а таблицы стилей определяют, как документ выглядит.

Получается, что CSS - это то, в каком виде наш HTML представлен.

### **Возможности CSS:**

 - управление отображением множества документов с помощью одного листа (таблицы) стилей;

 - более точный контроль над внешним видом страниц;

 - сложная и проработанная техника дизайна;

 - позволяют достаточно быстро и просто изменять внешний вид веб-страниц;

 - поскольку становится меньше объем документов, уменьшается время их загрузки

* + селекторы (специфичность, каскадность, наследование)
  + единицы измерения
  + блочная модель
  + виды позиционирования,
  + margin vs padding,
  + что такое flexbox,
  + что такое CSS Grid,
  + что такое псевдоэлемент/псевдокласс (с примером использования),
  + способы сделать страницу интерактивной с помощью CSS (по наведению мыши на определенный элемент).
* JS:
  + типы данных, type of
  + явное и неявное приведение типов
  + переменные в JS, var, const, let, область видимости переменных
  + циклы и ветвление
  + function declaration, function expression, arrow function
  + массивы
  + способ происка элементов DOM
  + обработка событий
  + создания/отмена таймеров
  + Promise, как обработать результат промиса
  + Event Loop
* React:
  + что такое JSX
  + что такое виртуальный DOM
  + в чем отличие между state и props
  + компоненты в реакт, жизненный цикл компонента
  + useState, useEffect
  + роутинг в реакт
  + стилизация компонентов
* TS:
  + что такое TypeScript
  + типы данных в TypeScript
  + union
  + enum
  + дженерики