***Hyper Text Marker Language*** – язык *HTML*, используется для создания каркаса сайта.

*HTML*-элемент состоит из открывающего и закрывающего тегов, а также размещенного между ними содержания.

Структура веб-страницы:

*<!DOCTYPE html>*

*<html>*

*<head lang="ru"> // указываем язык сайта*

*<meta charset="UTF-8"> // кодировка страницы*

*<title>Кнопка для Кати и Саши</title>*

*</head>*

*<body>*

*<button>Like</button>*

*</body>*

*</html>*

Для комментирования строк используются специальные символы:

*<!-- в HTML-коде комментарий окружён такими символами -->*

*/\* в CSS — такими \*/*

**Атрибуты** (или **свойства**) сообщают дополнительные сведения об элементе. Атрибут встраивается только в открывающий тег. Атрибуты пишутся подряд через пробел, порядок не имеет значение. Значение атрибута приводят в ‘одинарных’ или “двойных” кавычках (без разницы). Для *HTML* рекомендуется использовать двойные кавычки, а для *CSS* – одинарные.

Все возможные атрибуты элементов можно посмотреть на <https://doka.guide/>

***<head> </head>*** - теги метаданных страницы.

***<title>***Текст***</title>*** - теги названия страницы.

***<meta charset=”UTF-8”>*** - одиночный тег для подключения кодировки UTF-8.

***<link rel=”icon” href=”fav.ico” type=”image”>*** - одиночный тег для параллельного подключения внешних файлов. Например, иконки вкладки. Атрибут ***rel*** задает назначение, ***type*** – тип файла, ***href*** – адрес.

***<body> </body>*** - теги тела страницы (видимой части).

***<button>***Текст***</ button>*** - теги простой кнопки.

***<h1>***Текст***</h1>*** - теги заголовка. Всего их 6 размеров, *<h6>* самый маленький.

***<p>***Текст***</p>*** - теги абзаца. В них заключается основная часть текста.

***<a>***Текст***</a>*** - теги ссылки-якоря (anchor). Необязательный атрибут ***href*** хранит адрес ссылки. Необязательный атрибут ***target*** открывает ссылку в этой вкладке или в новой.

*<a href=”https://takiedela.ru/”>Перейти по ссылке</a>*

*<a target=”\_blank” href=”https://takiedela.ru/”>Перейти по ссылке</a>*  *// в новой вкладке*

***<header> </header>*** - теги элемента-блока шапки страницы, позволяющая объединять другие элементы. Расположен в *body* и занимает всю ширину страницы.

***<div> </div>*** - теги элемента-блока (абстрактная пустая коробка), позволяющая объединять другие элементы. Расположен в *body*.

***<section> </section>*** - теги элемента-блока страницы по аналогии с *div*, но лучше читается поисковиками. Расположен в *body*.

***<footer> </footer>*** - теги элемента-блока подвала страницы, позволяющая объединять другие элементы. Расположен в *body*.

Изображение создается только одним открывающимся тегом с ***img*** с обязательным атрибутом ***src***.

*<img src=”images/mobile.jpg”>*

Атрибут ***alt*** отображает текст, который пользователь видит, когда картинка не загружается:

*<img src=”images/mobile.jpg” alt=”здесь должна быть картинка, но что-то пошло не так…”>*

Необязательный атрибут ***width*** позволяет задать ширину картинки в пикселях. Это также позволит избежать рывков страницы при медленной загрузке изображений.

*<img src=”images/mobile.jpg” width=230>*

***Cascading Style Sheets*** – каскадные таблицы стилей отвечают за внешность веб-страницы. Чтобы придать элементу индивидуальный стиль, в тег встраивается атрибут ***style***. Ему присваивается название свойства и значение свойства. Например, атрибут ***color*** позволяет задать цвет.

*<h1 style=”color: springgreen; font-size: 72px;”>Текст</h1>*

Для изменения стиля сразу во многих элементах можно использовать **селектор**. Например, здесь тег h2 является селектором:

*h2 {*

*font-size: 32px;*

*color: skyblue;*

*}*

Селекторы могут быть **вложенными**:

*div p { // все p внутри div*

*margin-left: 20px*

*}*

Селекторы могут быть **комбинированными**:

*.app.best-choice { // элементы с классами app и best-choice одновременно*

*font-style: italic;*

*}*

Стили можно задавать как в самом *HTML*-документе, так и в отдельном файле. В первом случае внутри элемента *<head></head>* создается элемент *<style></style>* и заполняется селекторами. Во втором случае внутри элемента *<head></head>* создается ссылка на файл с *CSS*. Атрибут ***rel*** задает назначение, ***href*** – адрес.

*<head>*

*<link rel="stylesheet" href="reset.css"> // либо сброс всех стилей*

*<link rel="stylesheet" href="normalize.css"> // либо нормализация всех стилей*

*// либо через специальный селектор*

*// либо переписываем все вручную*

*<link rel=”stylesheet” href=”style.css”> // подключаем свои стили*

*</head>*

Рекомендуется добавлять специальный селектор для удаления значений по умолчанию:

*body, h1, h2, h3, h4, h5, h6, p, a {*

*margin: 0;*

*font-size: 100%;*

*font-weight: normal;*

*}*

Сразу после сброса стилей, можно задать общий для всех элементов селектор ***\****:

*\* {*

*box-sizing: border-box;*

*}*

Для подключения внешних файлов к CSS используется **директива импорта**:

*@import url(fonts.css)*

Для придания индивидуальности каждому элементу отдельно можно использовать атрибут ***class***. **Классы** – это уникальные селекторы, на которые ссылаются из *HTML*-кода.

*<!-- в html название класса указывается в кавычках атрибута class -->*

*<h1 class="white-text">Откуда к нам пришла зима</h1>*

*<p class="white-text">Не знаешь ты, никто не знает</p>*

*/\* в стилях селектор класса начинается с точки \*/*

*.white-text {*

*color: #fff;*

*}*

Одному элементу можно задать несколько классов, записав их через пробел.

*<p class="****text special-text****">Bonobo, Rhye — Break Apart</p>*

Задавать цвет можно не только английскими словами (147 штук), но и палитрами ***RGB*** или ***HEX-colors***. Не важно строчные или заглавные буквы используются. Если буквы повторяются, их можно сократить в два раза.

*rgb(255, 255, 0) // желтый*

*#FFFF00 // желтый*

*#FF0 // желтый*

Задать прозрачность можно с помощью **альфа-канала**. Величина прозрачности меняется от 1 до 0 (от непрозрачного к прозрачному). Дробь можно писать без нуля.

*rgba(255, 255, 0, 0.2) // желтый с 80% прозрачности*

За отображение фона отвечают свойства из группы ***background***:

* *background-color* позволяет задать фоновый цвет.Например:

*background-color: rgb(255, 255, 0);*

* *background-image* позволяет задать фоновую картинку. Например:

*background-image: url(https://yastatic.net/s3/auth2/\_/msoffice.d1ada09f.svg);*

* *background-position* позволяет задать положение фона. Например:

*background-position: center; // по центру и горизонтально и вертикально*

*background-position: right bottom; // прижать к нижнему правому углу*

*background-position: 150px -50px; // сдвиг на 150px вправо и на 50px вверх*

* *background-size* позволяет задать размер картинки фиксировано или в пропорциях. Например:

*background-size: cover; // сожмёт или растянет картинку, сохранив пропорции*

*background-size: contain; // картинка занимает всю доступную площадь, остальное - обрезается*

*background-size: 220px 400px; // задаст фикс размер ширина + высота*

* *background-repeat* позволяет отменить повторение изображения, если оно меньше поверхности блока. Например:

*background-repeat: no-repeat; // не повторяться*

*background-repeat: repeat; // с повтором*

*background-repeat: repeat-y; // с повтором только по вертикали*

За отображение шрифта отвечают свойства из группы ***font***:

* *font-size* позволяет задать размер шрифта.Например:

*font-size: 24px;*

* *font-weight* позволяет задать толщину текста (жирный). Можно задавать как словами, так и числами в диапазоне от 100 до 900. Например:

*font-weight: normal;*

* *font-family* позволяет задать семейство шрифта в приоритетном порядке. Например:

*font-family: 'Helvetica', 'Arial', sans-serif;*

* *font-style* позволяет задать стиль шрифта. Например:

*font-style: italic;*

***serif*** – системный шрифт с засечками

***sans-serif*** – системный шрифт без засечек

***monospace*** – шрифт с одинаковой шириной символов

***cursive*** – наклонный системный шрифт

***fantasy*** – декоративный системный шрифт

Для подключения нового шрифта, сначала нужно его декларировать (указать откуда его брать и что с ним делать). Это можно сделать с помощью директивы ***@font-face*** в начале *CSS*-файла:

*@font-face {*

*src: local(‘Best font ever’), // путь к локальному файлу*

*url(https://pictures.s3.yandex.net/fonts/Ibmplexserif.woff); // путь к файлу в сети*

*// подключение альтернативных форматов шрифтов для разных браузеров*

*src: url(https://pictures.s3.yandex.net/fonts/Ibmplexserif.woff) format(‘woff’),*

*url(https://pictures.s3.yandex.net/fonts/Ibmplexserif.woff2) format(‘woff2’);*

*font-family: 'Best Font Ever'; // сами задаем имя шрифту*

*font-display: swap;*

*}*

*// различные начертания шрифта подключаются отдельно*

*@font-face {*

*src: url(https://pictures.s3.yandex.net/fonts/Ibmplexserif.woff);*

*font-family: 'Best Font Ever';*

*font-style: italic;*

*font-weight: bold;*

*}*

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

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

*div {*

*// Если шрифт из файла не загрузился, использовать Arial или Helvetica,*

*// на крайний случай — любой шрифт без засечек в системе пользователя*

*font-family: 'Best Font Ever', 'Arial', 'Helvetica', sans-serif;*

*font-weight: bold;*

*font-style: italic;*

*}*

Пока скачивается файл *'Best Font Ever'* по ссылке из интернета, браузер отрисовывает весь текст доступным шрифтом *Arial* и лишь после скачивания начинает заменять шрифт текста. За логику замены шрифта отвечает свойство ***font-display***:

* *font-display: block;* — отрисует прозрачным альтернативным шрифтом текст, дождётся загрузки и перерисует видимым скачанным шрифтом весь текст.
* *font-display: swap;* — сначала покажет альтернативный шрифт, а когда загрузит нужный, заменит отображение текста. Может возникнуть мигание, но текст будет виден почти сразу.

Ссылка на самую большую библиотеку бесплатных шрифтов <https://fonts.google.com/> .

За поведение текстовых элементов отвечают свойства из группы ***text***:

* *text-align* позволяет выравнивать текст.Например:

*text-align: center; // выровнять по центру*

* *text-decoration* позволяет задать подчеркивание текста. Например:

*text-decoration: underline; // подчеркнуть текст*

*text-decoration: line-through; // перечеркнутый текст*

*text-decoration: none; // убирает подчеркивание по умолчанию для гиперссылок*

* *text-transform* позволяет задать регистр букв текста. Например:

*text-transform: uppercase; // перевести в верхний регистр*

*text-transform: lowercase; // перевести в нижний регистр*

*text-transform: capitalize; // перевести все первые буквы в словах верхний регистр*

* *line-height* позволяет задать высоту строки текста. Например:

*line-height: normal; // размер шрифта font-family умноженному на 1,2*

*line-height: 1,5; // размер шрифта font-family умноженному на 1,5*

*line-height: 3px; // длина в пикселях и других условных единицах*

*line-height: 120%; // за 100% берется размер шрифта font-family*

* *letter-spacing* позволяет задать расстояние между символами в тексте. Например:

*letter-spacing: normal; // размер задается аналогично line-height*

* *white-space* задает браузеру, как обрабатывать пробелы в тексте. Например:

*white-space: normal; // текст переносится по пробелам и достижении границы родителя*

*white-space: nowrap; // текст перестает переноситься по пробелам и выводится в одну строку*

*white-space: pre; // сохраняется форматирование, границы элемента игнорируются*

*white-space: pre-wrap; // сохраняется форматирование, границы элемента сохраняются*

*// сохраняется форматирование, но пробелы схлопываются, границы элемента сохраняются*

*white-space: pre-line;*

* *overflow* задает **родительскому блоку**, что делать с непомещающимся текстом. Например:

*overflow: scroll; // добавляет скролл для прокрутки*

*overflow: hidden; // скрывает лишний текст*

* *text-overflow* задает поведение для лишнего текста. Работает только вместе с *white-space: nowrap* и *overflow: hidden*. Например:

*text-overflow: clip; // обрежет лишний текст по краю*

*text-overflow: ellipsis; // появится многоточие*

* *overflow-wrap* задает поведение для переноса длинных слов. Например:

*overflow-wrap: normal; // перенесет слово целиком либо выйдет за границу родителя*

*overflow-wrap: break-word; // разобьет слово в любой части*

*overflow-wrap: anywhere; // разобьет слово в любой части с учетом правил мягкого переноса (слоги)*

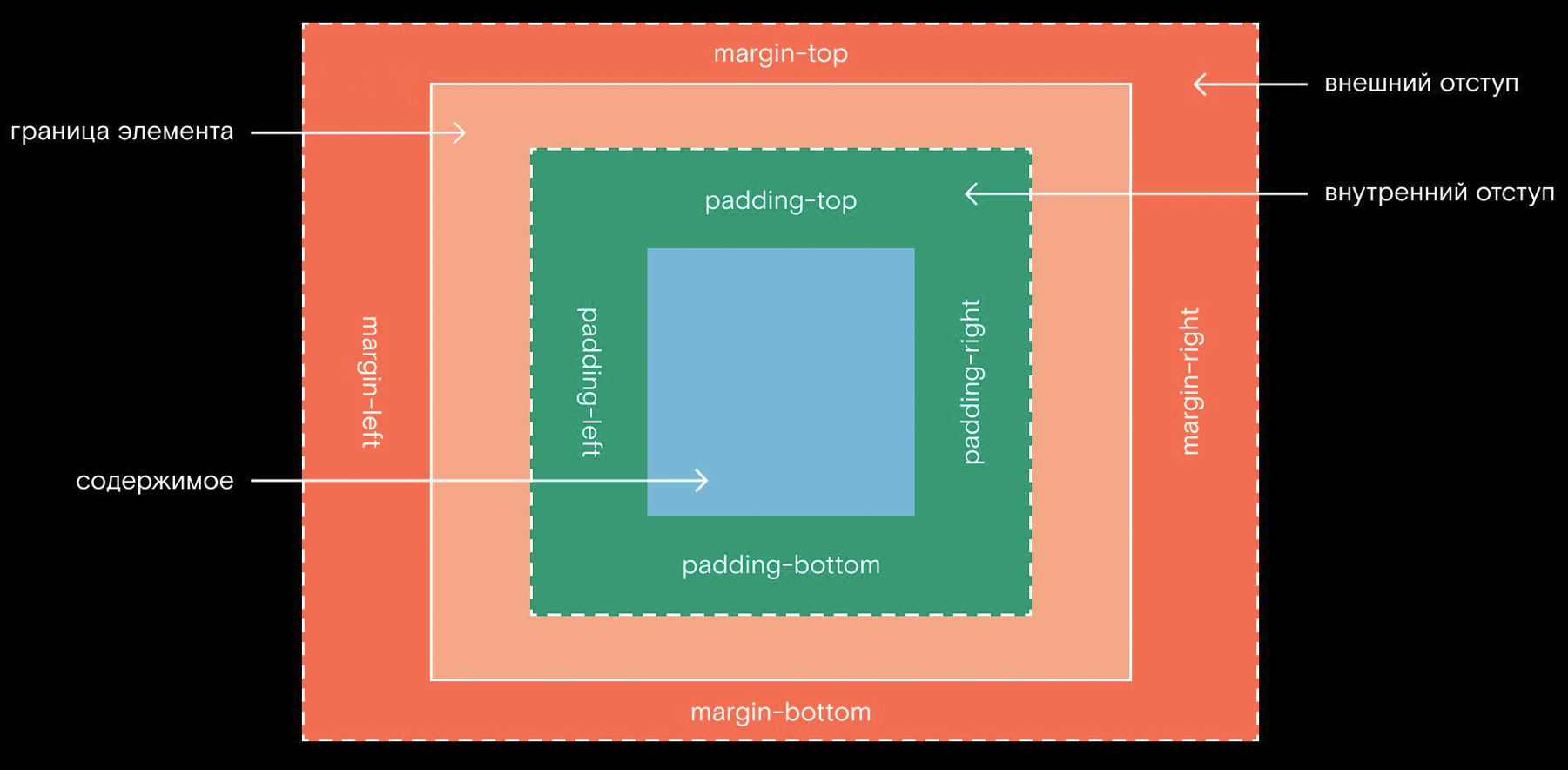
Указывать размер элементов можно в процентном соотношении. Это особенно важно при кроссплатформенной верстке – ведь *300px* на мониторе и на смартфоне занимают совершенно разное соотношение экрана. Элементы *h1-h6*, *p*, *div* и *header* занимают 100% ширины своего родителя. Значение элемента *body* равно видимой ширине окна браузера. Окно просмотра измеряется в единицах ***vw*** (*view width*) и ***vh*** (*view height*) относительно размеров окна браузера, или в %.

Стиль родительских элементов оказывает влияние на стиль дочерних (наследуется, если дочернему элементу не задано конкретное значение).

Порядок расположения элементов на странице называется **потоком**. Браузер читает код файла сверху вниз и также отрисовывает страницу, поэтому говорят, что элементы следуют друг за другом в потоке.

Элементы *div*, *section*, *header*, *h1-h6*, *p*, располагаясь в потоке, занимают всю ширину своего родителя, поэтому называются **блочными**. Модель любого блочного элемента содержит:

* содержимое с размерами *width* и *height*
* внешние отступы *margin*
* внутренние отступы *padding*
* границы



Свойство ***margin*** создает пустое пространство между элементом и его соседями. По умолчанию задается для всех сторон. Для задания отступа с конкретной стороны используются соответствующие свойства ***margin-top***, ***margin-right***, ***margin-bottom*** и ***margin-left***.

Если задать свойствам *margin-left* и *margin-right* значения ***auto***, то элемент получит максимальные отступы с обоих сторон и встанет по центру своего родителя.

Если задать *margin-top* для самого первого элемента в коробке *<div>*, то сдвинется не первый элемент, а сама коробка. Именно поэтому используется ***padding***.

Свойство ***padding*** позволяет отступить от границы текущего элемента внутрь. Для задания отступа с конкретной стороны используются соответствующие свойства ***padding-top***, ***padding-right***, ***padding-bottom*** и ***padding-left***.

Установить отступы *margin* или *padding* для элемента можно через короткую запись:

*/\* направление расстановки — по часовой стрелке, начиная сверху \*/*

*padding: 20px 15px 30px 15px;*

*/\* верх — 10px, по бокам — 20px, низ — 30px \*/*

*padding: 10px 20px 30px;*

*/\* верх и низ — 10px, по бокам — 20px \*/*

*padding: 10px 20px;*

*/\* 10px со всех сторон \*/*

*padding: 10px;*

За отображение границы элемента отвечают свойства из группы ***border***:

* *border-color* позволяет задать цвет границы.Например:

*border-color: #000;*

* *border-width* позволяет задать толщину границы элемента. Например:

*border-width: 1px;*

* *border-style* позволяет задать стиль границы элемента. Например:

*border-style: solid;*



Установить свойства *border* для элемента можно через короткую запись:

*border: 3px solid #000;*

Когда называют размер элемента, то имеют в виду размер содержимого до границ включительно, *margin* сюда не входит.

Атрибут ***box-sizing*** определяет поведение границ и внутренних отступов. По умолчанию для всех элементов действует правило *box-sizing: content-box* (границы и внутренний отступ расширяют элемент). Установив *box-sizing: border-box*, границы и внутренний отступ отрисуется внутрь.

Кроме блочных элементов (*div, section, body, header, footer*), в *HTML* есть **строчные**. Строчные элементы (*span*) занимают ровно столько места, сколько контента в них содержится. Если они идут подряд, то по умолчанию все расположены на одной строке. Они игнорируют указание размеров через стили.

Также существует комбинированный тип – **блочно-строчный**. С одной стороны, они не занимают собой всю горизонталь, с другой, восприимчивы к указанию размеров через *CSS*. Если они не помещаются на одной строке, то переносятся на следующую. Например, так ведут себя элементы *img*.

Атрибут ***display*** позволяет переопределить тип отображения элемента:

*display: block; /\* сделает элемент блочным \*/*

*display: inline; /\* сделает элемент строчным \*/*

*display: inline-block; /\* сделает элемент блочно-строчным \*/*

*display: flex; /\* сделает flex-элементом \*/*

Для того чтобы убрать зазор между блочно-строчными элементами, для коробки нужно задать:

*font-size: 0*;

Строчно-блочные элементы выравниваются по последней строке текста внутри них. Вертикальным выравниванием можно управлять с помощью атрибута ***vertical-align***.

Атрибут ***text-shadow*** отвечает за создание тени текста.

Атрибут ***box-shadow*** отвечает за создание тени элемента.

*div {*

*box-shadow: -2px 2px 5px #FD6969;*

*/\* сдвинута на 2px влево, на 2px вниз \*/*

*/\* радиус размытия — 5px \*/*

*/\* цвет красноватый, #FD6969 \*/*

*}*

Рассмотри пример, когда при наведении на элемент одного класса будет меняться прозрачность элементов другого класса. Этого можно достичь с помощью свойства ***hover***.

*.error {*

*font-size: 120px;*

*}*

*.message {*

*opacity: 0; // сделать прозрачным*

*}*

*.error:hover ~ .message {*

*opacity: 1; // сделать видимым*

*}*

Подробнее на <https://doka.guide/css/hover/>

Для сравнения макета и свёрстанного проекта используются несколько приемов:

1. Добавить макет в разметку, сделать полупрозрачным и поменять цвета.

*.pixel-perfect-image {*

*position: fixed;*

*top: 0;*

*left: 50%;*

*transform: translateX(-50%);*

*opacity: 0.5;*

*filter: sepia(0.6);*

*}*

1. Использовать плагин *PerfectPixel*.