**HTML –** HyperText Markup Language (Язык гипертекстовой разметки) стандартизированный язык разметки для описания структуры веб-страницы.

HTML представляет собой набор элементов, начало и конец которых помечаются тегами.

Тег – именованная метка. Определяет, каким образом будет отображаться или вести себя обернутый контент.

Элемент – основная структурная единица веб-страницы. Состоит из открывающегося тега, контента и закрывающегося тега.

Теги бывают **одиночные и парные**.

Одиночный тег используется самостоятельно, и у него отсутствует закрывающийся тег.

<br> - перенос на новую строку, <hr> - разделительная линия, <img> - вставка изображения, <meta>, <input>.

Парный тег образует контейнер, может включать себя другие теги или текст.

<h1>, <p>, <b>, <form>, <table>.

**Структура html-документа**

Элемент <!DOCTYPE> предназначен для указания типа текущего документа. Это необходимо, чтобы браузер понимал, как интерпретировать и отображать страницу. Ведь существуют разные версии html и xhtml.

< !DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"

"http://www.w3.org/TR/html4/strict.dtd" >

* **[Элемент верхнего уровня]** — корневой элемент на веб-странице, часто html/HTML;
* **[Публичность]** — этот параметр для файла может принимать два значения: PUBLIC (публичный) и SYSTEM (локальный);
* **[Регистрация]** — указывает на наличие регистрации у разработчика DTD внутри ISO (международная организация). Может быть + или -.
* **[Организация]** — название организации стандартизации (W3C);
* **[Тип]** — определение типа страницы, например DTD;
* **[Имя]** — имя документа;
* **[Язык]** — язык описания объектов (по умолчанию английский EN);
* **[URL]** — адрес страницы, на которой находится описание для указанного типа документа. Из этой спецификации браузер узнает, как отображать страницу с допустимыми тегами.

***Обязательные теги***

Тег <html> определяет начало HTML-файла, внутри него хранится заголовок(<head>) и тело документа(<body>).

<head> содержит заголовок <title>, favicon, тег <meta> в котором можно указать кодировку страницы, добавлять ключевые слова, описание документа.

<meta charset="utf-8">

Тег <body> содержит тело html документа, основную информацию.

<!-- Комментарий -->

**Атрибуты тегов**

Атрибуты – это дополнительные настройки тегов, для более детального описания свойств и поведения элементов.

Атрибут состоит из двух частей: имени и значения в кавычках, отделяемых знаком равенства.

Основные атрибуты, которые могут использоваться в большинстве тегов.

|  |  |  |
| --- | --- | --- |
| **Атрибут** | **Параметры** | **Описание** |
| **align** | right, left, center | Выравнивание тега по горизонтали |
| **valign** | top, middle, bottom | Вертикальное выравнивание тега. |
| **bgcolor** | числовые, шестнадцатеричные значения RGB | Цвет фона позади тега. |
| **background** | URL-адреса | Фоновое изображения за тегом. |
| **id** | Определяется пользователем | Уникальный идентификатор тега |
| **class** | Определяется пользователем | Имя тега для использования Каскадных таблиц Стилей CSS. |
| **width** | Числовое значение (%, px) | Задает ширину таблицы, изображения или ячейки таблицы. |
| **height** | Числовое значение (%, px) | Задает высоту таблицы, изображения или ячейки таблицы. |
| **title** | Определяется пользователем | "Pop-up" название элементов при наведении курсора или когда не загружаются. |
| **style** | Css стили | Css стили в пределах элементов. |

Можно добавлять тегам свои собственные атрибуты. Однако, чтобы html документ был валидным, необходимо чтобы он начинался с префикса data-.

**Форматирование текста**

Заголовки задаются с помощью тегов h, которые имеют 6 уровней. Браузер добавляет пустую строку после заголовка.

Тег <p> размещает текст в абзаце.

Тег <br> перенос текста на следующую строку.



<small> - уменьшение шрифта на единицу.

<ins> - подчеркивание текста.

<del> - перечеркнутый текст

<mark> - выделяет текст цветом, желтым.

<code>- вывод программного кода, моноширинный шрифт.

<font> в атрибутах можно задать несколько характеристик шрифта: color, size, face (семейство шрифтов)

**Форма**

Тег <form> создает html форму, которая используется для отправки введённых пользователем данных на сервер.

Атрибуты:

* **accept-charset**: указывает кодировку символов или список кодировок, разделенных пробелами, для введенных в форму данных, переданных на сервер для обработки. По умолчанию строка “UNKNOWN”, и браузер берет кодировку как у документа.
* **action**: Указывает адрес отправки данных из заполненной формы, где будет обрабатываться данная информация.
* **autocomplete**: включить или выключить автозаполнение формы.

**On**: браузер показывает значения, вводимые ранее пользователем (по умолчанию).

**Off:** ранее введенные значения показываться не будут.

* **enctype**: Указывает способ кодирования данных формы при отправке на сервер. Используется только с методом post.

Возможные значения:

* **application/x-www-form-urlencoded:** значение по умолчанию: все символы кодируются перед отправкой (пробелы преобразуются в символ "+", а специальные символы в значения ASCII HEX).
* **multipart/form-data:** символы не кодируются. Используется для элементов <input>, у которых в атрибуте type установлено значение "file".
* **text/plain:** пробелы преобразуются в символ "+", но не кодирует в шестнадцатеричные значения спецсимволы, такие как апострофы.
* **method**: метод передачи данных по http-протоколу.

**get**: данные передаются в адресной строке.

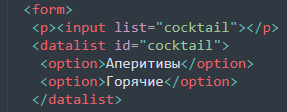
**post**: данные передаются в теле запроса.

* **novalidate:** указывает, что данные введенные в форму не будут проверятся перед отправкой.
* **target:** определяет в какой вкладке откроется ответ.

Тег <input> создает поле для ввода данных. Используется внутри тега <form>.

Атрибуты:

* **accept –** фильтр на типы загружаемых файлов, устанавливается для поля загрузки файлов.
* **accesskey** – устанавливает сочетание клавиш для быстрого доступа.
* **alt** - устанавливает альтернативный текст для поля с изображением (пока браузер не загрузит изображение).
* **autocomplete –** включить или выключить автозаполнение (по умолчанию on).
* **autofocus –** устанавливает фокус в поле формы (курсор сразу в поле).
* **border –** толщина рамки вокруг изображения.
* **checked –** предварительно активированный переключатель или флажок.
* **form –** связывает поле с формой по её идентификатору. Используется, когда поле располагается за пределами тега <form>.
* **formaction –** определяет адрес обработчика формы (для определенного инпута).
* **list –** указывает на список вариантов ввода, созданный с помощью тега <datalist>. Изначально этот список скрыт и становится доступным при получении полем фокуса.



* **max –** устанавливает верхнее значение для ввода числа или даты в поле формы.
* **min –** нижнее значение для ввода числа или даты.
* **maxlength –** максимальное число символов, которое может быть введено пользователем в текстовом поле.
* **multiple –** позволяет указать одновременно несколько файлов в поле для загрузки файлов, а также несколько email через запятую.
* **name –** определяет имя элемента формы. Используется для получения данных в обработчике формы, или для доступа к введенным данным через скрипты.
* **pattern –** регулярное выражение, которому должен соответствовать ввод.
* **placeholder –** подсказывающий текст внутри поля формы, исчезающий при получении фокуса (либо при начале ввода).
* **disabled –** блокирует доступ и изменение поля формы. Отображается серым. Состояние поля можно менять с помощью скриптов. Значение не передается на сервер при отправке формы.
* **readonly –** введенное значение не может изменяться пользователем. Применимо не ко всем видам полей.
* **required –** делает поле обязательным для заполнения.
* **size –** ширина текстового поля, определяется числом символов моноширинного шрифта (по умолчанию 20).
* **src –** адрес на изображение, которое отображается в поле type=”image”.
* **step –** шаг приращения для числовых полей (по умолчанию 1).
* **tabindex –** определяет порядок перехода между элементами при нажатии на клавишу tab.
* **type –** сообщает браузеру к какому типу относится элемент формы.
* **value –** определяет значение элемента формы.

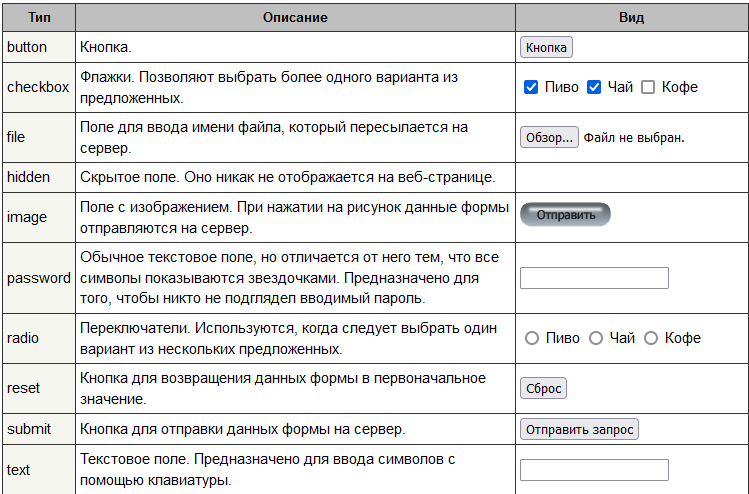
Для кнопок (input type="button | reset | submit") устанавливает текстовую надпись на них;

Для текстовых полей устанавливает предварительно введенную строку.

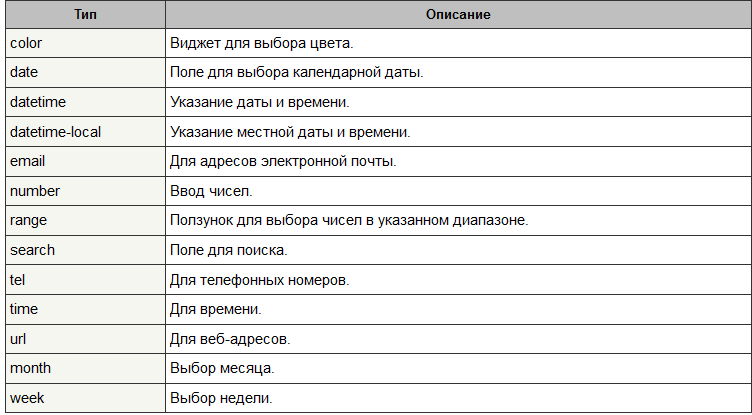
Для checkbox и radio уникально определяет каждый элемент, чтобы сервер мог однозначно установить, какой пункт выбрал пользователь.

Для файлового поля не оказывает влияния.

**Значения для атрибута type**



В HTML5 добавлены новые значения



Если в чекбоксе и радиокнопке не указать атрибут value, то при отправке формы значение будет установлено on.

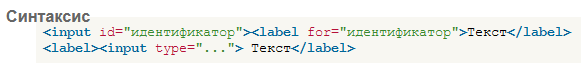
Браузер при отправке данных с формы не разрешает конфликт имен. Сервер получает строку, с параметрами в виде ключ=значение. Если в форме есть поля с одинаковыми именами, браузер отправит на сервер оба этих поля. Какой из них выбрать, или взять сразу оба решает сервер. Часто приоритетным является последнее полученное значение.

Тег **<select>** позволяет создать элемент интерфейса в виде раскрывающегося списка, а также список с одним или множественным выбором. Атрибут multiple позволяет одновременно выбрать несколько элементов. size – количество отображаемых строк списка. Внутри вложены теги <option>, представляющие пункты списка.

Тег **<optgroup>** представляет собой контейнер для тегов <option> и используется для группировки вариантов в выпадающем списке.

Поле **<textarea>** представляет область для ввода нескольких строк текста. Допустимо делать переносы строк. Атрибуты cols – ширина поля в символах, rows – высота поля в строках. Атрибут wrap указывает как переносится текст: soft – текст переносится на следующую строку, на сервер автоматический перенос не отправляется (только явный при нажатии Enter); hard – на сервер отправляется и автоматический перенос; off – текст не переносится автоматически, появляется полоса прокрутки.

Тег **<label>** - метка (подпись) элемента формы. Чаще всего используется с checkbox и radio. При нажатии на текст, будет изменяться значения чекбокса или радиокнопки.



Элемент **<fieldset>** предназначен для группировки элементов формы. Браузеры отображают его в виде рамки вокруг инпутов. Атрибут disabled блокирует поля в группе.

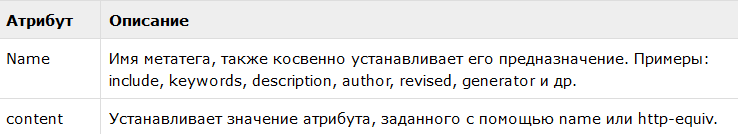
Тег <legend> определяет заголовок для группы <fieldset>.

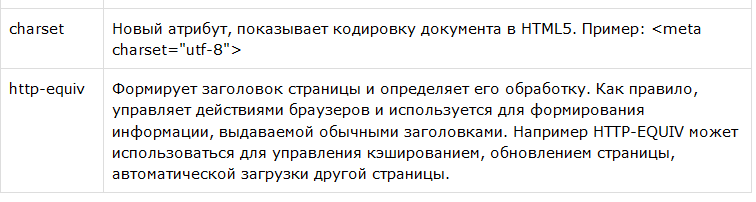
Тег **<output>** HTML используется для вывода результатов вычисления, полученных, как правило, при помощи скрипта.

**Тег meta**

<meta> определяет метатеги, которые используются для хранения информации предназначенной для браузеров и поисковых систем.

Как правило, атрибуты любого метатега сводятся к парам «имя=значение».





Значения атрибута name:

* **“keywords” -** список ключевых слов. Используется поисковыми системами.
* **“descriprion” –** краткое описание страницы. Поисковые системы учитывают его при индексации и при создании аннотации в выдаче по запросу. При его отсутствии выдается первая строка документа или отрывок, содержащий ключевые слова.
* **«Author» и «Copyright» -** Эти значения, как правило, не используются одновременно. Функция author и copyright — идентификация автора или принадлежности контента на странице. «Author» содержит имя автора веб-страницы, но в случае, если веб-сайт принадлежит какой-либо организации, целесообразнее использовать значение «Copyright».

**TABLE**

Тег <table> определяет HTML таблицу. Таблица состоит из строк и столбцов. Внутри <table> также можно использовать следующие элементы:

* <tr> - строка таблицы.
* <td> - ячейка таблицы, размещается внутри <tr>. Имеет следующие атрибуты:

**abbr –** краткое описание содержимого ячейки. Браузером не выводится. Доступен из скриптов;

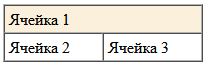
**align –** выравнивание содержимого ячейки по горизонтали;

**background –** задний фоновой рисунок в ячейке;

**bgcolor –** цвет фона ячейки;

**bordercolor –** цвет рамки;

**colspan –** число ячеек, которые должны быть объединены в одну по горизонтали. Значение 0 распространяет ячейку на всю родительскую группу колонок, объединенную тегом <colgroup>. Значения выше 1000 считаются неправильными, и устанавливаются в 1.



<tr>

<td colspan="2">Ячейка 1</td>

</tr>

**headers –** связывает ячейки таблицы с заголовками с помощью id. Используется в основном в речевых браузерах;

**height –** высота ячейки. Если содержимое ячейки больше, атрибут игнорируется, если меньше – добавляется пустое пространство;

**nowrap –** запрещает перенос текста в ячейке;

**rowspan –** количество ячеек, которые должны быть объединены по вертикали. Если значение установлено в 0, то ячейки объединяются до конца раздела таблицы (<thead>, <tbody> или <tfoot>) или самой таблицы. Максимально допустимое значение равно 65534;

**valign –** выравнивание содержимого ячейки по вертикали;

**width –** ширина ячейки.

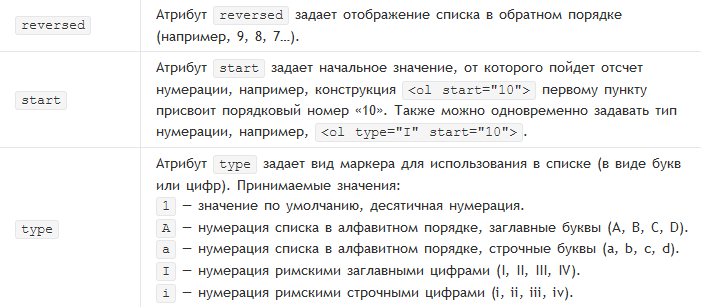
* <caption> - заголовок;
* <col> - задает ширину и другие характеристики одной или нескольких колонок таблицы;
* <colgroup> - позволяет объединять колонки в определенные группы, также при добавлении атрибута rules="groups" к тегу <table> браузер будет рисовать линию только между колонками, созданными с помощью <colgroup>;
* <tbody> - применяется для группировки строк таблицы. Позволяет задать единое оформление группе рядов;
* <tfoot> - «подвал» таблицы, отображается внизу таблицы. Предназначен для информации о колонках таблицы. В таблице разрешается только один <tfoot>;
* <th> - заголовочная ячейка. Текст обычно выделяется жирным и выравнивается по центру;
* <thead> - шапка таблицы, отображается вверху таблицы. В таблице разрешается только один <thead>.

**HTML-списки**

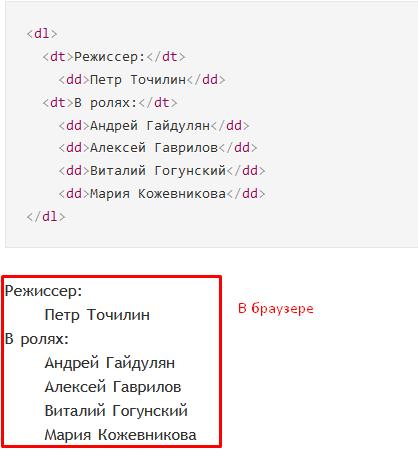
* **Маркированный список -** <ul> - каждый элемент списка отмечается маркером.
* **Нумерованный список -** <ol> - каждый элемент списка отмечается цифрой.
* **Список определений -** <dl> - состоит из пар термин <dt> - <dd> определение.

Списки можно вкладывать друг в друга.

В нумерованном списке для элемента <li> можно прописать атрибут value, чтобы изменить номер в списке по умолчанию. Следующее элементы также будут перенумерованы. Причем это значение число, даже если атрибут type=”a”. Атрибуты тега <ol>:



**Список определений**



**Типы верстки**

**Табличная –** устаревший вид верстки. Страницы конструируются из набора таблиц, внутри которых располагается контент.

**+** Простота по сравнению с блочной.

**+** Хорошая кроссбраузерность.

**-** Требуется больше HTML, сложно стилизовать через CSS, требуется много атрибутов.

**-** Медленная загрузка. Таблица не отображается, пока не будет полностью загружена.

**-** Плохая индексация поисковыми системами.

**Блочная верстка –** основным применяемым элементом является тег <div>. Все стили вынесены из html в css файлы.

**+** Меньший объем кода.

**+** Удобство изменения элементов. В основном путем правки стилей (например, перенести блок с правой стороны на левую). С таблицами это сделать сложнее.

**+** Большая гибкость, по сравнению с таблицами. Гораздо проще реализовать нестандартные расположения и оформления элементов.

**+** Возможность реализовывать более адаптивные страницы.

**+** Загрузка блоков происходит быстрее, чем таблиц. Браузер может кэшировать css.

**-** Большая сложность в освоении.

**-** Возможны проблемы с кроссбраузерностью.

**Семантическая вёрстка** — подход к разметке, который опирается не на содержание сайта, а на смысловое предназначение каждого блока и логическую структуру документа. Основные семантические теги:

* <header> - вводная часть смыслового раздела или всего сайта, обычно содержит подсказки и навигацию. Чаще всего повторяется на других страницах.
* <main> - основное, не повторяющееся на других страницах, содержание страницы.
* <footer> - заключительная часть смыслового раздела или всего сайта, обычно содержит информацию об авторах, список литературы, копирайт и т.д. Чаще всего повторяется на всех страницах сайта.
* <nav> - навигационный раздел со ссылками на другие страницы или другие части страниц.

**Span**

**<span> -** является строчным элементом и представляет из себя контейнер для текста. Используется когда нужно изменить стиль части текста или слова, не помещая их в блочный элемент.

**Ссылка**

Создается с помощью тега <a>. В атрибуте href задается адрес. Если установить атрибут name, то можно будет переходить к этому месту на странице (якорь). Атрибуты:

* target – определяет в какой вкладке открыть ссылку. \_blank в новой, self в текущей (по умолчанию). Если указывать blank (без нижнего подчеркивания), то ссылка будет открываться всегда в одной и той же вкладке.
* accesskey – устанавливает горячую клавишу для перехода по ссылке.
* title – всплывающая подсказка, при наведении курсора.

**Строчные (inline) и блочные(blok) элементы**

К строчным элементам относятся теги: <a>, <br>, <code>, <em>, <img>, <span>, <strong>, <sub>, <sup>.

**Строчные элементы** в общем потоке располагаются последовательно на одной строке один за другим.

На другую строку строчный элемент или часть его содержимого переходит только в конце строки или когда что-то принудительно заставляет их перейти на новую строку.

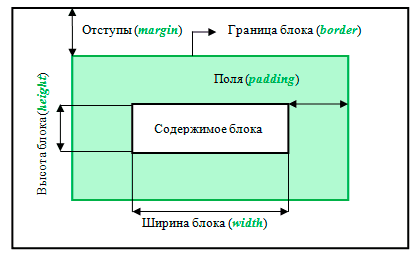
Строчные элементы могут располагаться внутри блочных элементов.

Применение значения block свойства display к строчному элементу делает его блочным.

К **блочным элементам** относятся теги: <div>, <dl>, <h1>...<h6>, <hr>, <ol>, <p>, <table>, <ul>.

*Блочные элементы* в общем потоке располагаются последовательно один под другим. По умолчанию два блочных элемента не могут располагаться на одной строке.

Один или несколько блоков могут располагаться внутри другого *(родительского)* блочного элемента.



Любой блок имеет форму прямоугольника.

Блочный элемент имеет границу и содержимое: это может быть текстовая информация, фотографии, логотип фирмы и т.п.

Любое содержимое блока или веб-страницы называется контентом *(англ. content — содержание)*.

Между содержимым и границей существуют внутренние расстояния - *поля блока* *(свойство padding)*. Если их явно не установить при помощи свойства padding, то граница будет вплотную примыкать к содержимому блока, либо отстоять от него на каком-то минимальном расстоянии.

*Отступы блока* (свойство margin) - это расстояния от его границ, до ближайших элементов, или, если их нет, то до краев окна браузера. Отступы так же, как и поля по умолчанию отсутствуют, либо имеют минимальную ширину, автоматически определяемую браузером.

Блочный элемент занимает всю доступную ширину, высота элемента определяется его содержимым.

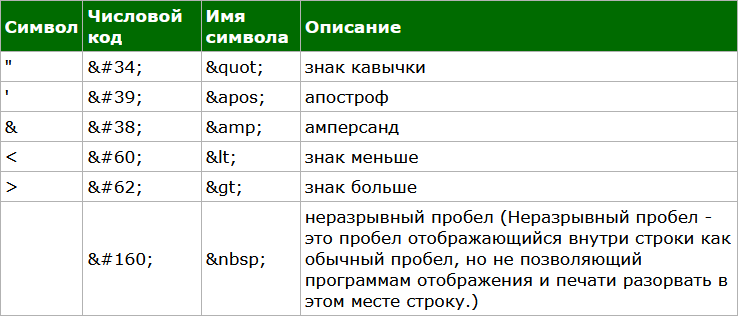
*Вставлять блочные элементы внутрь строчных запрещено.*

Если в одном блочном элементе содержатся строчные элементы наряду с блочными, то вокруг строчных элементов генерируется анонимный блок. Для таких блоков применяется стиль по умолчанию, а также наследуемый стиль, заданный для его родителя.

*Строчно-блочные элементы:* использовать css свойство inline-block;

* Внутрь строчно-блочных элементов допустимо помещать любые элементы.
* Высота рассчитывается исходя из содержимого блока.
* Ширина = содержимое + отступы + поля и границы.
* Несколько элементов располагаются на одной строке. Переносятся при необходимости.
* Эффект схлопывания отступов не действует

**Экранирование –** замена некоторых символов спецсимволами. Чтобы выводить как обычный текст символы, которых нет на клавиатуре, либо предназначенные для разметки страницы. Это помогает защитить приложение, ведь злоумышленники могут вставить в форме форума html-разметку или js-скрипты. Основные из них:



**CSS**

**Cascading Style Sheets** (Каскадные таблицы стилей) – формальный язык описания внешнего вида (стилизации) документа.

Основной целью создания CSS являлось разделение описания логической структуры веб-страницы (HTML) от описания внешнего вида этой страницы.

Преимущества css:

* Разделение кода и оформления. Упрощается разработка, меньшее количество кода (уменьшается количество повторений описания стилей).
* Один и тот же документ может иметь различные стили (экранное представление, печать, голосовое).
* Ускорение загрузки сайта. Файлы css могут кэшироваться браузерами.
* Централизованное управление стилями. Меняем в одном месте, меняется везде.

Таблицы стилей могут быть подключены различными способами:

* Внешний файл со стилями. Подключается с помощью тега <link> в блоке head. В атрибуте href указывается адрес к таблице стилей.
* Директива @import. Располагается в документе в элементе style.

@import url("header.css");

* Встроенная таблица стилей. Стили описанs в самом документе между тегами style. Такие стили можно применить только в данном документе.
* Инлайн-стили. Стили прописываются непосредственно в теге в атрибуте style. Действует только на данный тег. Стили смешаны со структурой, эти стили не рекомендуется использовать. Еще в них нельзя применять псевдоэлементы.

CSS правило состоит из селектора – определяет к каким элементам будет применен стиль, и блока объявлений. Блок объявлений заключен в фигурные скобки. Объявление - это сочетание имени свойства и значения, разделенных знаком «:». Селекторы могут **группироваться** в одной строке через запятую. Тогда стили применяются к каждому из них.

**Виды селекторов:**

* *\* {}* универсальный селектор. Правило применяется к любому элементу. Может использоваться в комбинации с псевдоклассами и псевдоэлементами.
* *p {}* селектор элемента. Применяется ко всем элементам данного типа.
* *.class {}* селектор класса. Применяются к элементам с совпадающим атрибутом class. Если элемент имеет несколько классов, то они разделяются пробелом.
* *#id {}* селектор идентификатора. Применяется к элементу с указанным идентификатором.
* *ul li {}* селектор потомков. Применяется к элементам, расположенным внутри элемента контейнера.
* *p > b* дочерний селектор. Применяется ко всем элементам, являющимися прямыми потомками контейнера.
* *h1 + p* сестринский селектор. Сестринские элементы – те которые имеют общего родителя. Выбирается при этом только первый элемент, следующий после h1.

*h1 ~ p* выберет все сестринские элементы, следующие за h1.

* *p[attribute=”value”]* селектор атрибута. Выбирает элементы на основе имени атрибута или его значения
* *a:active* селектор псевдокласса. позволяют применить CSS-правила к элементам при совершении события или подчиняющимся определенному правилу.
* *p:first-letter* селектор псевдоэлемента.

Псевдоклассы определяют динамическое состояние элементов, которое изменяется с помощью действий пользователя, а также положение в дереве документа.

Условно все псевдоклассы делятся на три группы:

* **определяющие состояние элементов;**

1. active – активация элемента (щелчок мыши);
2. link – непосещенная ссылка;
3. focus – элемент получивший фокус (например курсор в поле для ввода);
4. hover – при наведении мышки на элемент;
5. visited – посещенная ссылка;

* имеющие отношение к дереву элементов;

1. first-child – первый дочерний элемент селектора.

* указывающие язык текста.
  1. lang() – определяет язык, который используется в документе или его фрагменте.

**Псевдоэлемент** в CSS — это ключевое слово, добавляемое к селектору, которое позволяет стилизовать определённую часть выбранного элемента.

* 1. after – применяется для вставки контента после содержимого элемента. Вставляемый контент определяется свойством content в css правиле.
  2. before – для вставки контента до содержимого элемента.
  3. first-letter – определение стиля первого символа в тексте.
  4. first-line – стиль для первой строки блочного текста.

**Наследование –** механизм, с помощью которого значения свойств элемента-родителя передаются его элементам-потомкам. Потомки могут переопределять свой стиль.

Не все свойства наследуются. Чтобы унаследовать ненаследуемые свойства используется слово inherit. Чтобы запретить наследование используется *initial*. В этом случае элемент применит свойства браузера.

*Наследуемые свойства:* отображающее параметры текста (fon-size, font-family, color, text-align, direction), cursor, visibility, list-style

**Каскадирование стилей –** заключается в том, что если к элементу применяется несколько css-правил, то результирующий стиль будет сформирован из всех присутствующих в этих правилах свойств.

В случае конфликтов, будет применено свойство с наибольшим приоритетом.

* + 1. Стиль браузера;
    2. Стиль пользователя (устанавливается в настройках браузера).
    3. Стиль автора страницы

Стили унаследованные от родителей.

Стили заданные во внешних таблицах.

Стили внутренних таблиц в атрибуте <style>.

Инлайн-стили.

Стили помеченные !important.

В случае конфликта на одном уровне, конфликт решается с помощью сравнения специфичности селекторов. Приоритетнее те, что имеют больше идентификаторов; затем классов, атрибутов и псевдоклассов; элементов и псевдоэлементов. В случае одинаковой специфичности, применяется стиль, расположенный ниже в таблице.

**ПРОЦЕСС ЗАГРУЗКИ СТРАНИЦЫ**

1. Браузер преобразует HTML документ в объектную модель документа (DOM).
2. Когда парсер встречает внешний ресурс (файл css или js) он загружает его.
3. CSS файлы преобразуются в объектную модель CSS (CSSOM).
4. Выполнение JavaScript.
5. Объединение DOM и CSSOM, для построения дерева рендеринга.
6. Расчет макета страницы (положение и размер каждого элемента) – **Layout.**
7. **Paint –** отрисовка элементов.