CSS

CSS (Cascading Style Sheets)

CSS (Cascading Style Sheets) — каскадные таблицы стилей
Позволяет задать оформление практически для любого элемента написанного языком разметки. Обычно **CSS-стили** используются для создания и изменения стиля элементов веб-страниц и пользовательских интерфейсов, написанных на языке HTML.

Каскадные таблицы стилей описывают правила форматирования элементов с помощью свойств и допустимых значений этих свойств. Для каждого элемента можно использовать ограниченный набор свойств, остальные свойства не будут оказывать на него никакого влияния.

Структура CSS

В **CSS** не существует ни элементов ни тегов. За основу берется правило для определенного элемента.

Способы добавления стилей

Существует несколько способов добавление стилей:

- Внутри элемента
- Подключить внешнюю таблицу стилей
- Прописать стили непосредственно в документе

Внешние стили

Внешняя таблица стилей

Внешняя таблица стилей представляет собой текстовый файл с расширением .css, в котором находится набор **CSS-стилей** элементов.

Внешняя таблица стилей подключается к веб-странице с помощью тега <link>, расположенного внутри раздела <head></head>. Такие стили работают для всех страниц сайта.

```
<head>
    <link rel="stylesheet" href="css/style.css">
    <link rel="stylesheet" href="css/assets.css">
</head>
```

Внутренние стили

Внутренние стили

Внутренние стили встраиваются в раздел <head></head> **HTML-документа** и определяются внутри тега <style></style>. **Внутренние стили** имеют приоритет над **внешними**, но уступают **встроенным стилям**.

```
<head>
	<style>
		div {display: inline-block; background: green;}
	</style>
</head>
```

Встроенные стили

Свойства прописанные в данном теге, строго ограничиваются внутри него.

Чтобы прописать непосредственно в теге определенной стиль используют атрибут style:

```
<a style="font-size:14px; font-style: italic; ">Наш любимый Lorem ipsum</a>
```

Правило @import

Правило @import позволяет загружать внешние таблицы стилей.Чтобы директива @import работала, она должна располагаться в таблице стилей перед всеми остальными правилами:

```
<style>
        @import url(mobile.css);
        p {font-size: 0.9em; color: grey;}
</style>
```

Правило @import также используется для подключения веб-шрифтов:

```
@import  url(https://fonts.googleapis.com/css?family=Open+Sans&subset=latin);
```

Виды селекторов

Селекторы представляют структуру веб-страницы. С их помощью создаются правила для форматирования элементов веб-страницы. Селекторами могут быть элементы, их классы и идентификаторы, а также псевдоклассы и псевдоэлементы.

Универсальный селектор

Применяется ко всему документу, или ко всем элементам.

```
* {font-size:15px; color:green;}
```

Селектор элемента

```
a {font-size:12px; color:#000}
```

Применяется непосредственно ко всем <a>

Виды селекторов

Селектор класса

Для написания стиля для селектора по классу используют .(точку) перед именем

```
.green{
        font-size: 14px;
        font-family: arial,
        color: green;
}
```

Селекторы класса позволяют задавать стили для одного и более элементов с одинаковым именем класса, размещенных в разных местах страницы или на разных страницах сайта.
Пример для того чтобы заработал .green, нужно к тегу <a> добавить атрибут class=”green” (без точки).
Стили, созданные с помощью класса, можно применять к другим элементам, не обязательно данного типа.

Специфика селекторов

- Классы могут использоваться в коде неоднократно
- Имена классов чувствительно к регистру
- К любому тегу можно добавить несколько классов, перечисляя их в атрибуте class через пробел

Виды селекторов

Селектор идентификатора

Селектор идентификатора позволяет форматировать **один** конкретный элемент. Идентификатор id должен быть уникальным и на одной странице может встречаться только один раз.

```
<div id="sidebar"></div>
```

```
#sidebar {
	width: 300px;
	height: 300px;
	border: 1px solid black;
}
```

Специфика идентификаторов

- В коде документа каждый идентификатор уникален и используется лишь один раз
- Имя идентификатора чувствительно к регистру
- Стиль идентификатора имеет приоритет выше, чем у классов

Виды селекторов:

Селектор потомка

Селекторы потомков применяют стили к элементам, расположенным внутри элемента-контейнера. Например, div p {text-transform: uppercase;} — выберет все элементы p, являющиеся потомками всех элементов div.

Если нужно отформатировать потомки определенного элемента, этому элементу нужно задать стилевой класс:

div.first p {color: green;} — данный стиль применится ко всем параграфам, потомкам тега с классом first;

div .first p {color: green;} — если добавить пробел, то будут стилизованы параграфы, расположенные внутри любого тега класса .first, который является потомком элемента ;

Виды селекторов:

Дочерний селектор

Дочерний элемент является прямым потомком содержащего его элемента. У одного элемента может быть несколько дочерних элементов, а родительский элемент у каждого элемента может быть только один. Дочерний селектор позволяет применить стили только если дочерний элемент идёт сразу за родительским элементом и между ними нет других элементов, то есть дочерний элемент больше ни во что не вложен.

Например, ul > li — выберет все элементы li, являющиеся дочерними по отношению к элементу ul.

Виды селекторов:

Сестринский селектор

Сестринские отношения возникают между элементами, имеющими общего родителя. Селекторы сестринских элементов позволяют выбрать элементы из группы элементов одного уровня.

h2 + p — выберет все первые абзацы, идущие непосредственно за любым тегом <h2>, не затрагивая остальные абзацы;

h2 ~ p — выберет все абзацы, являются сестринскими по отношению к любому заголовку h2 и идущие сразу **после него**.

Виды селекторов:

Селектор атрибута

Селекторы атрибутов выбирают элементы на основе имени или значения атрибута:

[атрибут] — все элементы, содержащие указанный атрибут. [alt] — все элементы, для которых задан атрибут alt;

селектор[атрибут] — элементы данного типа, содержащие указанный атрибут. input[value] — только поля ввода, для которых задан атрибут value;

селектор[атрибут="значение"] — элементы данного типа, содержащие указанный атрибут с конкретным значением, input[value="name"] — все поля ввода, значение которых содержит слово name;

Виды селекторов:

^ - циркумфлекс
селектор[атрибут^="значение"] — элементы, значение атрибута которых начинается с указанного значения, a[href^="#anchor"] — все ссылки, начинающиеся на #anchor;

$ - доллар
селектор[атрибут$="значение"] — элементы, значение атрибута которых заканчивается указанным значением, img[src$=".png"] — все картинки в формате png;

* - звездочка или астериск
селектор[атрибут*="значение"] — элементы, значение атрибута которых содержит в любом месте указанное слово, a[href*="auto"] — все ссылки, название которых содержит auto.

Виды селекторов:

Селектор псевдокласса

Псевдоклассы — это классы, фактически не прикрепленные к **HTML-тегам**. Они позволяют применить **CSS-правила** к элементам при совершении события или подчиняющимся определенному правилу.

:link — не посещенная ссылка;

:visited — посещенная ссылка;

:hover — любой элемент, по которому проводят курсором мыши;

:focus — интерактивный элемент, к которому перешли с помощью клавиатуры или активировали посредством мыши;

:active — элемент, который был активизирован пользователем;

:checked — выделенные (выбранные пользователем) элементы формы.

Виды селекторов:

Селектор структурных псевдоклассов

Структурные псевдоклассы отбирают дочерние элементы в соответствии с параметром, указанным в круглых скобках:

:nth-child(odd) — нечетные дочерние элементы;

:nth-child(even) — четные дочерние элементы;

:nth-child(3n) — каждый третий элемент среди дочерних;

:nth-child(3n+2) — выбирает каждый третий элемент, начиная со второго дочернего элемента (+2);

Виды селекторов:

:nth-child(n+2) — выбирает все элементы, начиная со второго;

:nth-child(3) — выбирает третий дочерний элемент;

:nth-last-child() — в списке дочерних элементов выбирает элемент с указанным местоположением, аналогично с :nth-child(), но начиная с последнего, в обратную сторону;

:first-child — позволяет оформить только самый первый дочерний элемент тега;

:last-child — позволяет форматировать последний дочерний элемент тега;

Виды селекторов:

Селектор псевдоэлемента

Псевдоэлементы используются для добавления содержимого, которое генерируется с помощью свойства content:

:before — вставляет генерируемое содержимое перед элементом;

:after — добавляет генерируемое содержимое после элемента.

Группировка селекторов

Один и тот же стиль можно одновременно применить к нескольким элементам. Для этого необходимо в левой части объявления перечислить через запятую нужные селекторы:

```
h1, h2, p, a:focus {
        color: blue;
}
```

Наследование и каскадирование:

Наследование и каскад — два фундаментальных понятия в CSS, которые тесно связаны между собой. Наследование заключается в том, что элементы наследуют свойства от своего родителя (элемента, их содержащего). Каскад проявляется в том, как разные виды таблиц стилей применяются к документу, и как конфликтующие правила переопределяют друг друга.

Наследование

Наследование является механизмом, с помощью которого определенные свойства передаются от предка к его потомкам. Спецификацией CSS предусмотрено наследование свойств, относящихся к текстовому содержимому страницы, таких как color, font, letter-spacing, line-height, list-style, text-align, text-indent, text-transform, visibility, white-space и word-spacing. Во многих случаях это удобно, так как не нужно задавать размер шрифта и семейство шрифтов для каждого элемента веб-страницы.

Свойства, относящиеся к форматированию блоков, не наследуются. Это background, border, display, height и width,margin,min-max-height и -width, outline, overflow, padding, position, text-decoration, vertical-align и z-index.

Каскад

Каскадирование — это механизм, который управляет конечным результатом в ситуации, когда к одному элементу применяются разные **CSS-правила**. Существует три критерия, которые определяют порядок применения свойств — правило !important, специфичность и порядок, в котором подключены таблицы стилей.

Правило !important

Вес правила можно задать с помощью ключевого слова !important, которое добавляется сразу после значения свойства, например,

```
span {font-weight: bold!important;}
```

Правило необходимо размещать в конец объявления перед закрывающей скобкой, без пробела. Такое объявление будет иметь приоритет над всеми остальными правилами. Это правило позволяет отменить значение свойства и установить новое для элемента из группы элементов в случае, когда нет прямого доступа к файлу со стилями.

Специфичность

Для каждого правила браузер вычисляет **специфичность селектора**, и если у элемента имеются конфликтующие объявления свойств, во внимание принимается правило, имеющее наибольшую специфичность.

встроенный стиль	`<p style="color: red;"></p>`
идентификатор	`#identificator {...}`
класс	`.class {...}`
элемент	`a {...}`

Порядок подключённых таблиц

Вы можете создать несколько внешних таблиц стилей и подключить их к одной веб-странице. Если в разных таблицах будут встречаться разные значения свойств одного элемента, то в результате к элементу применяется правило, находящееся в таблице стилей, идущей в списке ниже.