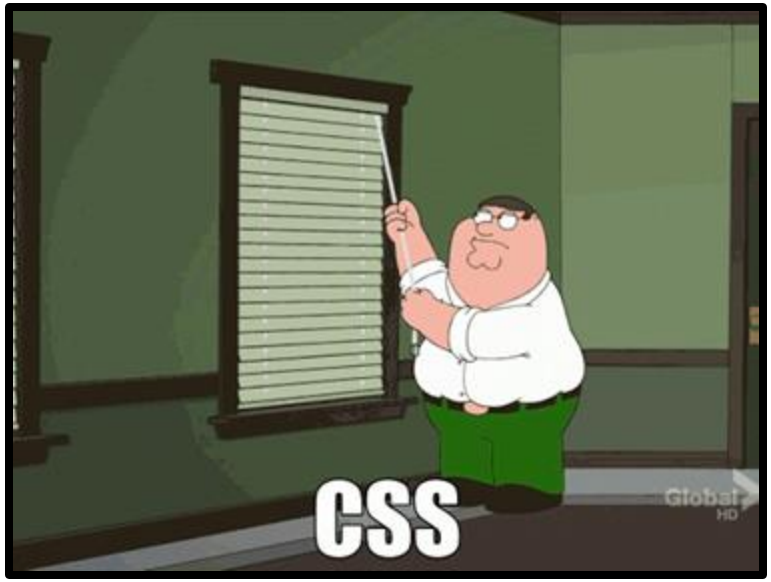
DATA GROUP

Full-stack

GEN AI

Технологии

- HTML
- CSS
- Javascript



Поля CSS

Ниже приведен список наиболее распространенных полей CSS и примеры их использования:

- `color: #FF0000;  red;  rgba(255,00,100,1.0); //различные способы указания цветов`
- `background-color: red;`
- `background-image: url('file.png');`
- `font: 18px 'Tahoma';`
- `border: 2px solid black;`
- `border-top: 2px solid red;`
- `border-radius: 2px; //для сглаживания углов и придания им более округлой формы`
- `margin: 10px; //расстояние от границы до внешних элементов`
- `padding: 2px; //расстояние от границы до внутренних элементов`
- `width: 100%;  300px;  1.3em;  //множество различных способов указания расстояний`
- `height: 200px;`
- `text-align: center;`
- `box-shadow: 3px 3px 5px black;`
- `cursor: pointer;`
- `display: inline-block;`
- `overflow: hidden;`

CSS-селекторы

Вы также можете указать теги по их контексту, например: теги, которые находятся внутри тегов, соответствующих селектору. Просто разделите селекторы пробелом:

```
div#main p.intro { ... }
```

Это повлияет на теги p класса intro, которые находятся внутри тега div с id main

```
<div id="main">
    <p class="intro"&gt;....</p&gt;  ← Влияет на этот
</div>

<p class="intro">....</p>  ← но не на этот
```

CSS-селекторы

Вы можете комбинировать селекторы, чтобы еще больше сузить поиск.

```
div#main.intro:hover { ... }
```

применит CSS к любому тегу div с id main и классом intro, если курсор мыши находится над ним.

И вам не нужно указывать тег, вы можете использовать селекторы класса или id без тега, это означает, что он будет влиять на любой узел с id main

```
#main { ... }
```

CSS-селекторы

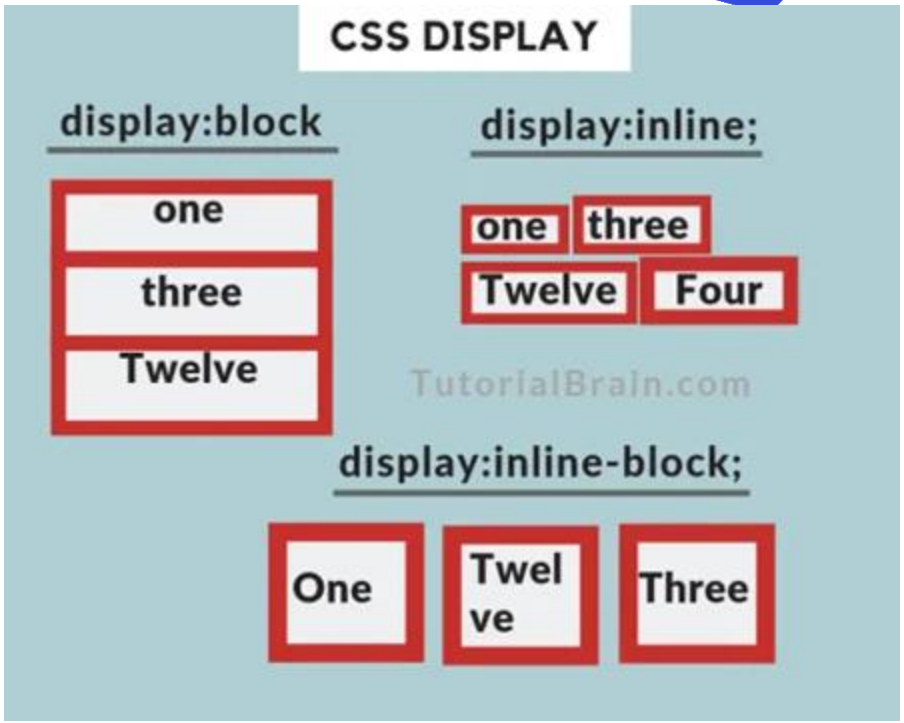
Если вы хотите выбрать только элементы, которые являются прямыми потомками одного элемента (а не имеют предка с этим правилом), используйте символ >:

```
ul.menu > li { ... }
```

Наконец, если вы хотите применить одни и те же CSS-действия к нескольким селекторам, вы можете использовать символ запятой ,:

```
div, p { … }  ← это будет применяться ко всем тегам div и p
```

HTML-размещение



Важно понимать, как браузер располагает элементы на экране.

Ознакомьтесь с этим учебником, в котором объясняются различные способы расположения элементов на экране.

Вы можете изменить способ расположения элементов с помощью свойства display:

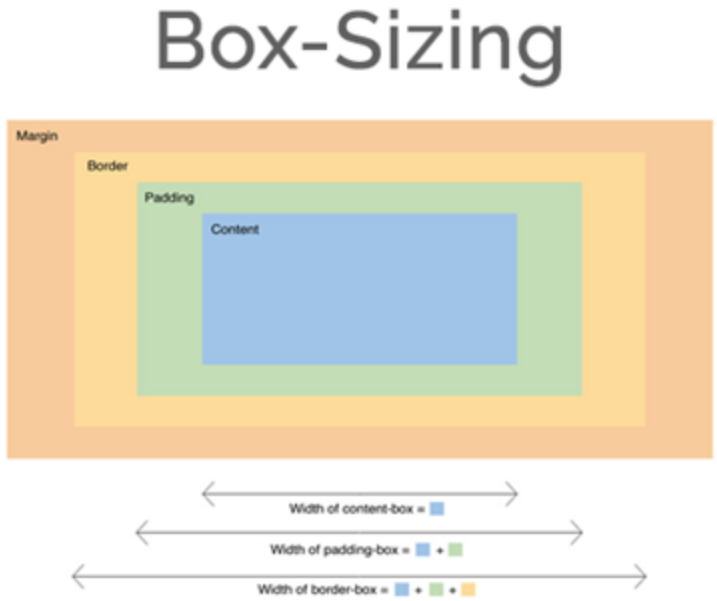
```
div { display: inline-block; }
```

Модель блока

Важно отметить, что по умолчанию любая ширина и высота, указанные для элемента, не учитывают его отступы, поэтому div с шириной 100 пикселей и отступом 10 пикселей будет отображаться на экране размером 120 пикселей, а не 100 пикселей.

Это может стать проблемой, нарушающей ваш макет.

Вы можете изменить это поведение, изменив модель блока элемента, чтобы ширина использовала самую

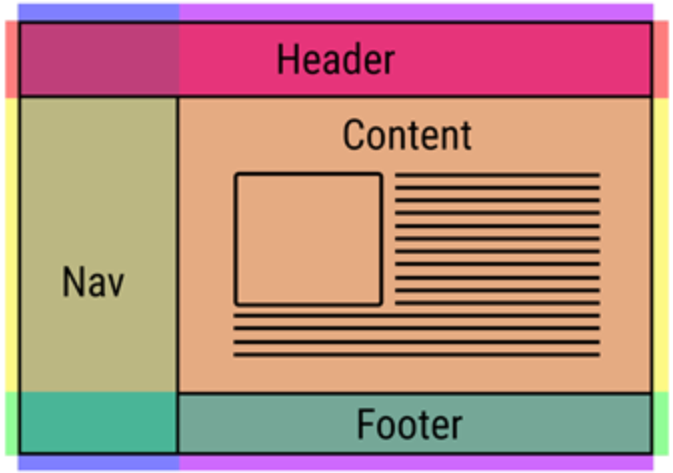


Макет

Одной из самых сложных частей CSS является интерпретация макета вашего веб-сайта (структуры внутри окна).

По умолчанию HTML имеет тенденцию размещать все в одной колонке, что не является идеальным решением.

В CSS было много предложений по решению этой проблемы (таблицы, фиксированные div, flex, grid и т. д.).



Flexbox

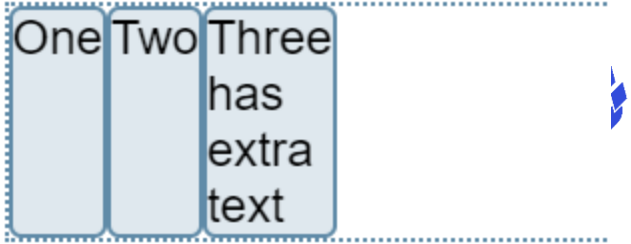
Первым крупным предложением по решению проблемы макета стала модель flexbox.

Эта модель позволяет очень легко располагать элементы в одном направлении (по вертикали или по горизонтали).

Вы даже можете выбрать расположение справа налево (в обратном порядке).

Ее также можно использовать для расположения ряда элементов в разных строках.

Более подробную информацию см. в учебном пособии.



HTML

```
<div class="box">
  <div>Один</div>
  <div>Два</div>
  <div>Три
    <br>первая строка
    <br>вторая строка
  </div>
</div>
```

CSS

```
.box {
  display: flex;
}
```