



CSS

Cascading Style
Sheets
(Каскадные
таблицы стилей.)



ПЛАН

1

Основы позиционирования.

2

Базовые стили.

3

Display, фоновые картинки,
цвета, ссылки.

4

Opacity, z-index, комбинаторы.

ОСНОВЫ ПОЗИЦИОНИРОВАНИЯ

Основы позиционирования в CSS позволяют контролировать расположение элементов на веб-странице. Существуют несколько методов позиционирования, которые определяют, как элементы будут взаимодействовать с другими элементами и родительским контейнером.

Свойство **position** определяет тип метода позиционирования и используется для элемента, имеет значения (static, relative, fixed или absolute).



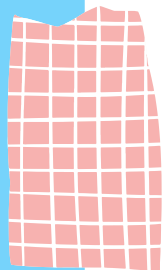
ОСНОВЫ ПОЗИЦИОНИРОВАНИЯ

1

2

3

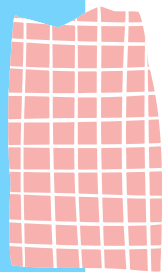
4



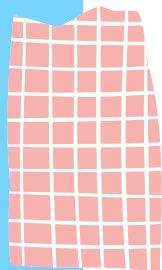
1. **static**: Это значение по умолчанию для всех элементов. Элементы с позиционированием `static` следуют нормальному потоку документа и игнорируют свойства позиционирования, такие как `top`, `left`, `right` и `bottom`.
2. **relative**: Элементы с позиционированием `relative` позиционируются относительно своего нормального местоположения в потоке документа. Они могут использовать свойства `top`, `left`, `right` и `bottom` для смещения относительно своего исходного положения.
3. **fixed**: Элементы с позиционированием `fixed` позиционируются относительно окна просмотра (`viewport`) и остаются на месте даже при прокрутке страницы. Они также используют свойства `top`, `left`, `right` и `bottom` для указания местоположения.
4. **absolute**: Элементы с позиционированием `absolute` позиционируются относительно ближайшего родительского элемента с позиционированием `relative`, `absolute` или `fixed`. Они используют свойства `top`, `left`, `right` и `bottom` для указания точного местоположения.
5. **sticky**: Элементы с позиционированием `sticky` позиционируются относительно своего родительского контейнера или окна просмотра (`viewport`). Они ведут себя как `relative` до тех пор, пока не достигают пороговой позиции, а затем становятся `fixed`.

DISPLAY

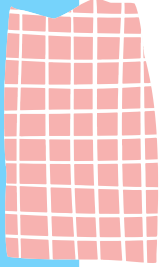
Свойство display наиболее важное свойство CSS для управления макетом.



Свойство display в CSS используется для определения типа отображения элемента. Оно позволяет контролировать, как элемент будет отображаться на веб-странице и как он будет взаимодействовать с другими элементами.



1. **block**: Элемент отображается как блочный элемент, занимающий всю доступную ширину на странице. Блочные элементы начинаются с новой строки и могут содержать другие блочные и строчные элементы.
2. **inline**: Элемент отображается как строчный элемент, который не нарушает текущий поток и занимает только необходимое пространство для своего содержимого. Строчные элементы не создают новую строку и не могут содержать другие блочные элементы внутри себя.
3. **inline-block**: Элемент комбинирует свойства блочного и строчного элементов. Он отображается внутри строки, но при этом может иметь ширину и высоту, а также применять внутренние и внешние отступы.
4. **none**: Элемент не будет отображаться на странице. Он будет полностью скрыт и не займет места на макете. Это полезно, когда требуется временно скрыть элемент или когда элемент должен быть создан динамически с помощью JavaScript.
5. **flex**: Элемент отображается как блочный элемент с гибким макетом. Он позволяет создавать гибкие макеты, распределять пространство между элементами и управлять их поведением при изменении размеров окна.
6. **grid**: Элемент отображается как блочный элемент с использованием гибкой сетки. Он позволяет создавать сложные макеты с помощью сетки, размещая элементы в ячейки сетки.

- 
1. По сравнению со встроенным блоком `display:inline`, основная разница в том, что линейный блок `display:inline-block` позволяет задать **ширину** и **высоту** элемента.
 2. Также, с линейным блоком `display:inline-block`, соблюдаются верхние и нижние поля и отступы, а со встроенным `display:inline` это не так.
 3. По сравнению с блоком `display:block`, основная разница в том, что линейный блок `display:inline-block` не добавляет **разрыв строки** после элемента, чтобы элемент могли находиться рядом.

CSS ФОН

С помощью CSS свойства **background** определяют фон, эффекты для элементов.

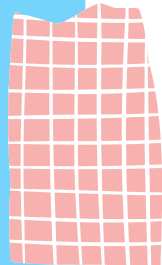
В CSS цвет чаще всего задается с помощью:

- Название цвета - например "red"
- HEX значение - например "#ff0000"
- RGB значение - например "rgb(255,0,0)"

Вы можете установить цвет фона для любых HTML элементов:

OPACITY

Свойство opacity определяет непрозрачность/прозрачность элемента. Оно может принимать значение от 0.0 до 1.0.
Чем меньше значение, тем прозрачнее:



Примечание: При использовании свойства opacity для добавления прозрачности к фону элемента, все его дочерние элементы наследуют одинаковую прозрачность. Может сделать текст внутри, полностью прозрачным элементом, трудным для чтения.

CSS BACKGROUND-REPEAT

По умолчанию свойство background-image повторяет изображение как по горизонтали, так и по вертикали.

Чтобы повторить изображение горизонтали установите свойство (background-repeat: repeat-x;), фон будет выглядеть лучше.

Чтобы повторить изображение по вертикали, установите свойство background-repeat: repeat-y;

Отображение фонового изображения только один раз также задается свойством background-repeat: no-repeat;

СВОЙСТВО **BACKGROUND-POSITION**

Определяет положение фонового изображения внутри элемента.

`background-position: right top;`

CSS BACKGROUND-ATTACHMENT

Свойство `background-attachment` указывает, должно ли фоновое изображение прокручиваться или быть фиксированным (не будет прокручиваться вместе с остальной частью страницы):

CSS BACKGROUND - СОКРАЩЕННОЕ СВОЙСТВО

Чтобы сократить код, можно также указать все свойства фона в одном единственном свойстве. Это называется свойство background

При использовании свойства сокращен порядок значений свойств:

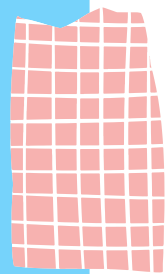
- background-color
- background-image
- background-repeat
- background-attachment
- background-position

Не имеет значения, если одно из значений свойства отсутствует, пока остальные находятся в этом порядке. Обратите внимание, что мы не используем свойство background-attachment в приведенных выше примерах, поскольку оно не имеет значения.

COLORS (ЦВЕТА)

Цвета в CSS могут быть заданы различными способами, позволяя выбирать из широкой палитры цветов. Вот основные способы задания цветов в CSS

1. Имена цветов: CSS предоставляет predetermined имена для некоторых цветов, например, red (красный), blue (синий), green (зеленый) и так далее. Можно использовать эти имена напрямую.
2. Коды цветов в шестнадцатеричной системе: Цвета также могут быть представлены с помощью шестнадцатеричной системы. Код цвета начинается с символа #, за которым следуют шестнадцатеричные значения для красного (RR), зеленого (GG) и синего (BB) компонентов цвета:
3. Коды цветов в RGB: Цвета также можно задавать с помощью значений красного (R), зеленого (G) и синего (B) компонентов в десятичной системе счисления, используя функцию `rgb()` или `rgba()`. Значения компонентов варьируются от 0 до 255:
4. Прозрачность: Значение альфа-канала (прозрачность) может быть добавлено в коды цветов с использованием функции `rgba()`. Значение альфа-канала варьируется от 0 (полностью прозрачный) до 1 (полностью непрозрачный).



1. В CSS цвет также может быть задан с помощью значений RGB, шестнадцатеричных значений HEX, значений HSL, значений RGBA и значений HSLA:
2. То же самое, что и название цвета "Tomato":

`rgb(255, 99, 71)`

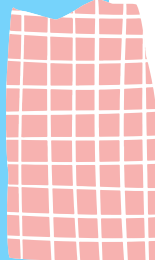
`#ff6347`

`hsl(9, 100%, 64%)`

То же самое, что и название цвета "Tomato", но на 50% прозрачный:

`rgba(255, 99, 71, 0.5)`

`hsla(9, 100%, 64%, 0.5)`



Имена цветов: CSS предоставляет predefined имена для некоторых цветов, например, red (красный), blue (синий), green (зеленый) и так далее. Можно использовать эти имена напрямую.

```
div {  
  color: red;  
}
```


Цвет HEX

В CSS цвет может быть задан с помощью HEX шестнадцатеричного значения в виде:

#rrggbb

Где rr (red), gg (green) и bb (blue) - шестнадцатеричные значения между 00 и ff (такие же, как десятичные 0-255).

Например, #ff0000 отображается красным цветом, поскольку красный цвет имеет самое высокое значение (ff), а остальные самое низкое значение (00).

Коды цветов в шестнадцатеричной системе: Цвета также могут быть представлены с помощью шестнадцатеричной системы. Код цвета начинается с символа #, за которым следуют шестнадцатеричные значения для красного (RR), зеленого (GG) и синего (BB) компонентов цвета:

```
div {  
    background-color: #FF0000; /* Красный цвет */  
}
```

Цвет RGB

В CSS цвет может быть задан как значение RGB, используя эту формулу:

`rgb(red, green, blue)`

Каждый параметр (red, green, и blue) определяет интенсивность цвета в диапазоне от 0 до 255.

Например, `rgb(255, 0, 0)` отображается красный, потому что red имеет самое высокое значение (255), а остальные 0.

Чтобы отобразить черный цвет, установите все цветовые параметры равными 0, например: `rgb(0, 0, 0)`.

Чтобы отобразить белый цвет, установите все цветовые параметры равными 255, например: `rgb(255, 255, 255)`.

```
div {  
  background-color: #F00; /* Красный цвет */  
}
```

Цвет RGBA

Значения цвета RGBA - это расширение значений цвета RGB с альфа каналом, который определяет **непрозрачность** для цвета.

Значение цвета RGBA задается с помощью:

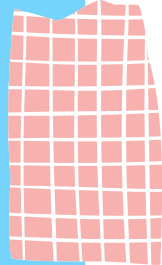
`rgba(red, green, blue, alpha)`

Альфа параметр - это число между 0.0 (полностью прозрачное) и 1.0 (совсем не прозрачное):

```
div {  
  background-color: rgba(0, 0, 255, 0.5); /* Полупрозрачный синий цвет */  
}
```

CSS КОМБИНАТОР

Комбинатор - это то, что объясняет взаимосвязь между селекторами

- 
1. Есть четыре различных комбинатора в CSS3:
 2. селектор потомок пространство
 3. селектор ребенок >
 4. селектор смежный брат +
 5. селектор общий брат ~

Селектор потомок

1. Селектор потомок сопоставляется всем элементам, которые являются потомками указанного элемента.

```
div p {  
    background-color: yellow;  
}
```

Селектор потомок

1. Селектор ребенок выбирает все элементы, которые являются непосредственными детьми указанного элемента.

```
/*child*/  
div > p {  
    background-color: yellow;  
}
```

Селектор соседний брат

1. Селектор соседний брат выбирает все элементы, которые являются смежными элементами указанного элемента.
2. Дочерние элементы должны иметь тот же родительский элемент, "соседний" значит "сразу же после".

```
div + p {  
  background-color: yellow;  
}
```

Селектор общий брат

1. Селектор `div ~ p` в CSS является комбинатором соседних элементов (general sibling combinator). Он выбирает все элементы `<p>`, которые являются соседними (следуют сразу за) элементом `<div>`.

```
/*same bro ~*/  
div ~ p {  
  background-color: yellow;  
}
```

ССЫЛКИ



С помощью CSS, стили ссылок могут быть разные.

Четыре состояния ссылок:

- a:link - по умолчанию, непосещенная ссылка
- a:visited - пользователь посетил, посещенная ссылка
- a:hover - курсор мыши, при наведении на ссылку
- a:active - на данный момент, нажатая ссылка

При определении стиля для нескольких состояний ссылок, есть некоторые правила порядка:

- a:hover Должно происходить после a:link и a:visited
- a:active Должно происходить после a:hover

```
/*link*/  
  
/* непросмотренная ссылка */  
a:link {  
    color: red;  
}  
  
/* посещенная ссылка */  
a:visited {  
    color: green;  
}  
  
/* наведите указатель мыши на ссылку */  
a:hover {  
    color: hotpink;  
}  
  
/* выбранная ссылка */  
a:active {  
    color: blue;  
}
```


Оформление текста ссылки

Свойство text-decoration в основном используется, чтобы удалить подчеркивание ссылок:

```
/* text-decoration */  
a:active {  
    color: blue;  
}  
  
a:link {  
    text-decoration: none;  
}  
  
a:visited {  
    text-decoration: none;  
}  
  
a:hover {  
    text-decoration: underline;  
}  
  
a:active {  
    text-decoration: underline;  
}
```

Цвет фона ссылки

Свойство background-color может использоваться, чтобы указать цвет фона для ссылки:

```
/*link-background*/  
  
a:link {  
    background-color: yellow;  
}  
  
a:visited {  
    background-color: cyan;  
}  
  
a:hover {  
    background-color: lightgreen;  
}  
  
a:active {  
    background-color: hotpink;  
}
```

Z-INDEX

Свойство z-index определяет порядок расположения элемента в стеке (какой элемент должен быть размещен перед другими или позади них).

Элемент может иметь положительный или отрицательный порядок стека

Без z-index

Если два позиционированных элемента перекрывают друг друга без указания z-index, элемент определенный последним в HTML-коде, будет показан сверху.



**THANK
YOU!**

Have a
great day
ahead.