CSS BOX-MODEL

AIT
Hi-tech team

НАШИ ПРАВИЛА

Включенная камера

Вопросы по поднятой руке

Не перебиваем друг друга

Все вопросы, не связанные с тематикой курса (орг-вопросы и т. д.), должны быть направлены куратору

Подготовьте свое рабочее окружение для возможной демонстрации экрана (закройте лишние соцсети и прочие приложения)

Повторение ;)

Сколькими способами можно подключить css?

Какая разница между div и span?

Какие основные типы селекторов?

ЦЕЛЬ

Изучить концепцию блочной модели в css. Научиться отличать и использовать margin, padding, border

ПЛАН ЗАНЯТИЯ

- комбинированные стили
- padding
- margin
- border
- единицы измерения

Комбинированные стили

Группировка селекторов

Если нескольким селекторам нужно задать одно и то же правило, то можно написать длинно:

```
h1 {
    color: red;
    }
h2 {
    color: red;
    }
h3 {
    color: red;
    }
```

А можно перечислить все селекторы через запятую и написать всего одно CSS-правило:

```
h1, h2, h3 {
    color: red;
    }
```

Группировка селекторов

При группировке любые виды селекторов записываются в произвольном порядке через запятую и в таком случае правила будут применяться к каждой из групп:

```
h1, .block, #id, h3 {
    color: red;
}
```

Сложение стилей

Если разные правила для одного элемента содержат свойства, которые не конфликтуют, то они объединяются в один стиль, т.е. каждое новое правило добавляет новую информацию о стиле к тому правилу, которое находится выше:

```
h1 {
  color: gray;
  font-family: sans-serif;
}
h1 {
   border-bottom: 1px solid black;
}
```

```
h1 {
  color: gray;
  font-family: sans-serif;
  border-bottom: 1px solid black;
}
```

Обычно дополнительные правила для элемента указываются в тех случаях, когда был задан один стиль сразу для нескольких элементов, но помимо этого необходимо добавить что-то ещё для определённого элемента:

```
h1, h2, h3 {
  color: gray;
  font-family: sans-serif;
}
h2 {
   border-bottom: 1px solid black;
}
```

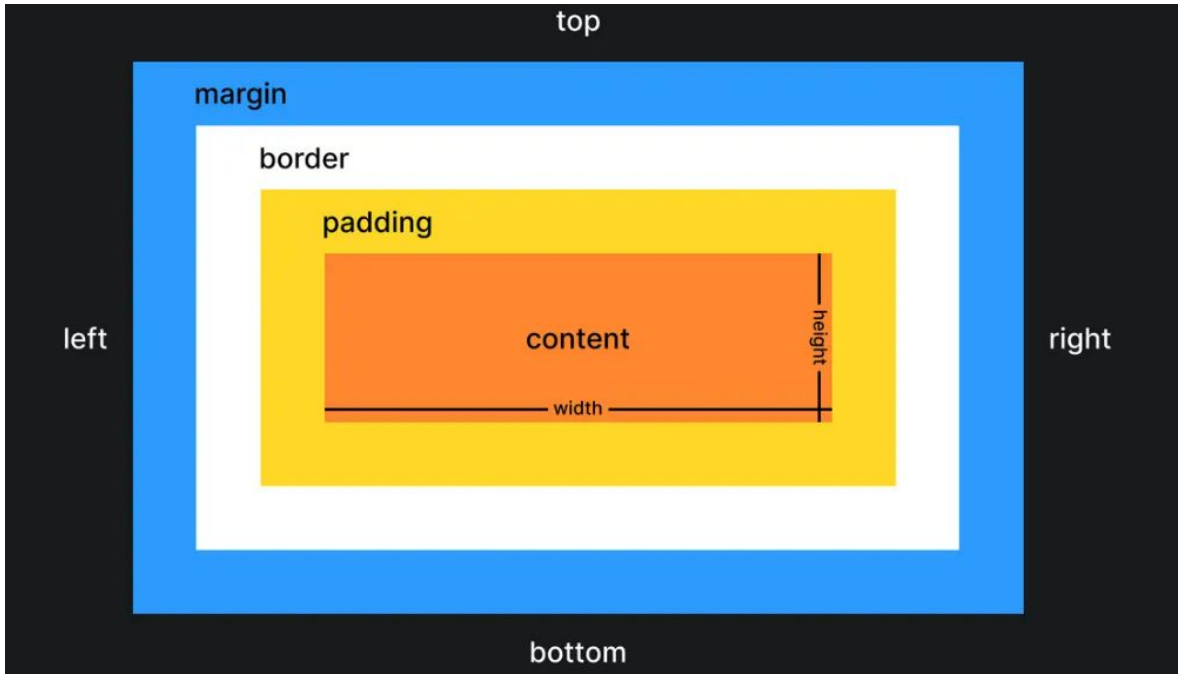
Список CSS свойств

https://developer.mozilla.org/en-US/docs/Web/CSS/color

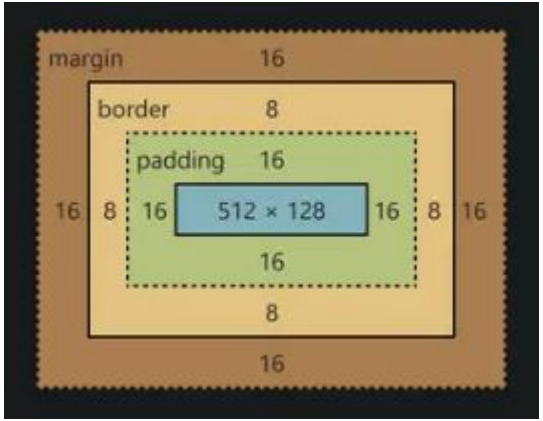
Box model

Блочная модель

Блочная модель, она же box model — это алгоритм расчёта размеров каждого отдельного элемента на странице, которым браузеры пользуются при отрисовке.

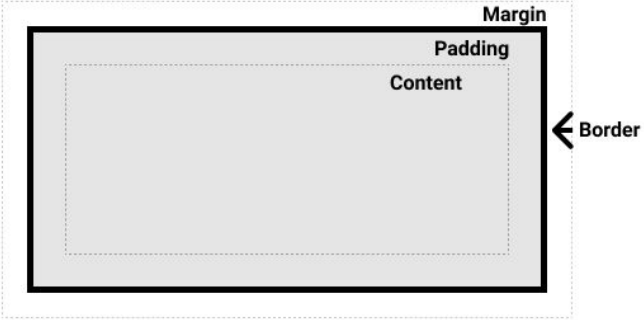


Ровно такую же схему, но в других цветах можно увидеть в инструментах разработчика любого из браузеров. Например, так выглядит блочная модель элемента в Chrome:

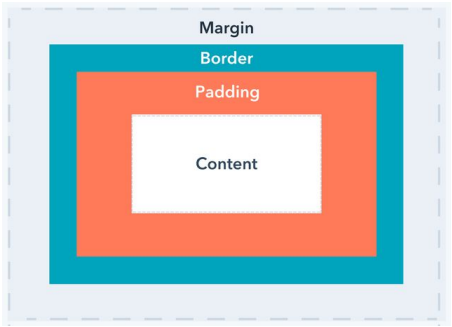


Блочная модель состоит из нескольких CSS-свойств, влияющих на размеры элемента:

- **width — ширина элемента;**
- **height — высота элемента;**
- **padding — внутренние отступы от контента до краёв элемента;**
- **border — рамка, идущая по краю элемента;**
- **margin — внешние отступы вокруг элемента**



Внешние и внутренние отступы



Внутренний отступ

Внутренний отступ - пустое пространство между содержимым элемента и его рамкой (если она установлена). Для добавления и управления шириной внутренних отступов со всех четырех сторон элемента используется свойство `padding`.

```
padding: 10px 15px 22px 18px;
```

top right bottom left

Внешний отступ

Внешний отступ - пустое пространство, отделяющее элемент от других элементов или краев окна браузера. Для добавления и управления шириной внешних отступов со всех четырех сторон элемента используется свойство `margin`.

```
margin: 10px 15px 22px 18px;
        top  right bottom left
```

Внутренние и внешние отступы

Внутренние отступы могут регулироваться отдельными свойствами:

- **padding-top**
- **padding-right**
- **padding-bottom**
- **padding-left**

Внешние отступы могут регулироваться отдельными свойствами:

- **margin-top**
- **margin-right**
- **margin-bottom**
- **margin-left.**

Общие свойства:

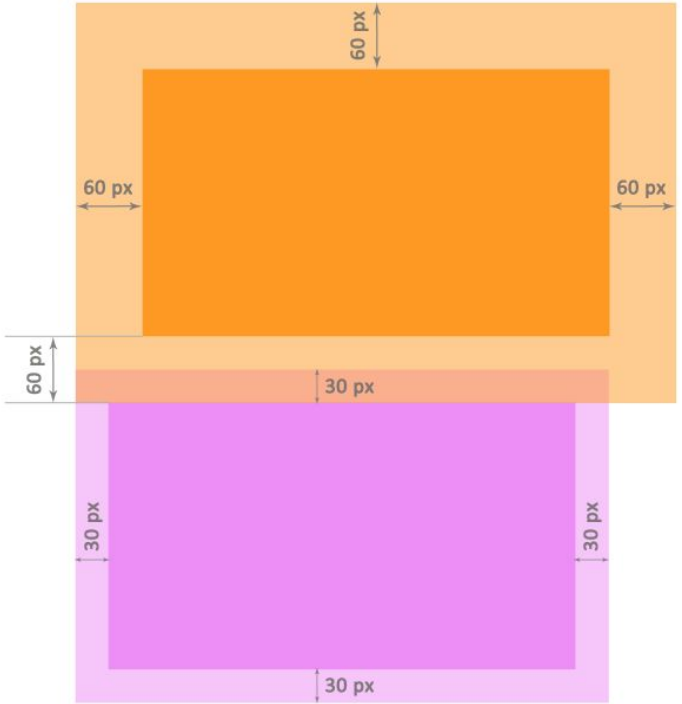
- padding
- margin

Схлопывание margin (Margin Collapse)

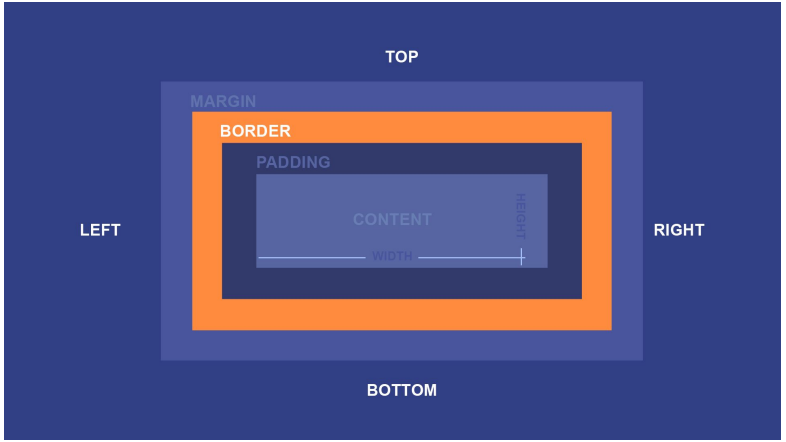
Представьте ситуацию: два блочных элемента находятся друг над другом и им заданы поля margin. Для верхнего блока установлено значение `margin`: 60px, а для нижнего – `margin`: 30px. Логично было бы предположить, что два граничащих поля двух элементов просто соприкоснутся и в итоге промежуток между блоками будет равен 90 пикселям.

Однако дела обстоят по-другому. На самом деле в такой ситуации проявляется эффект, который называют схлопыванием, когда из двух примыкающих полей элементов выбирается наибольший по размеру. В нашем примере итоговый промежуток между элементами будет равен 60 пикселям.

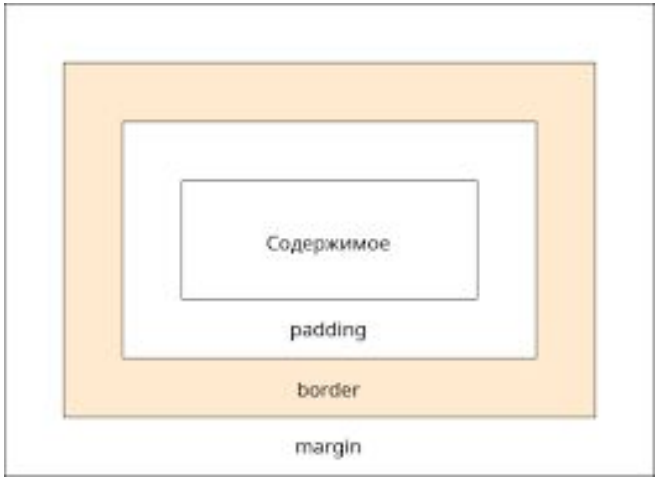
Схлопывание `margin` работает только для верхних и нижних полей



Параметры рамки. Цвет и стиль рамки



CSS-рамка элемента представляет собой одну или несколько линий, окружающих содержимое элемента и его поля.



Стиль рамки - border-style

По умолчанию рамки всегда отрисовываются поверх фона элемента, фон распространяется до внешнего края элемента. Стиль рамки определяет ее отображение, без этого свойства рамки не будут видны вообще. Для элемента можно задавать рамку для всех сторон одновременно с помощью свойства `border-style` или для каждой стороны отдельно с помощью уточняющих свойств `border-top-style` и т.д. Данное свойство не наследуется.

Стиль рамки - border-style

border-style

(border-top-style, border-right-style, border-bottom-style, border-left-style)

Значения:

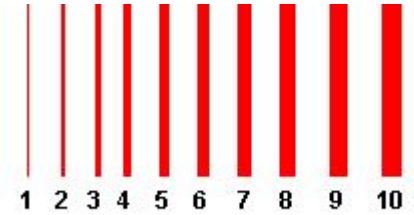
`none`	Значение по умолчанию, означает отсутствие рамки. Также убирает рамку элемента из группы элементов с установленным значением данного свойства.
`hidden`	Эквивалентно `none`.
`dotted`	dotted
`dashed`	dashed
`solid`	solid
`double`	double

Стиль рамки - border-style

`groove`	groove
`ridge`	ridge
`inset`	inset
`outset`	outset
`{1,4}`	Одновременное перечисление четырех разных стилей для рамок элемента, только для свойства `border-style` : `{border-style: solid dotted none dotted;}`
`initial`	Устанавливает значение свойства в значение по умолчанию.
`inherit`	Наследует значение свойства от родительского элемента.

Ширина рамки - border-width

Ширина рамки задается с помощью единиц измерения длины или ключевых слов. Если для свойства `border-style` задано значение none, и для рамки элемента установлена какая-то ширина, то в данном случае ширина рамки приравнивается к нулю. Свойство не наследуется.



Ширина рамки - border-width

border-width

(border-top-width, border-right-width, border-bottom-width, border-left-width)

Значения:

`thin / medium / thick`	Ключевые слова, устанавливают ширину рамки относительно друг друга. Первое значение уже, чем второе, второе — тоньше третьего. Значение по умолчанию — `medium`
`width (px, em)`	`{border-width: 5px;}`
`{1,4}`	Возможность одновременного задания четырех разных ширин для рамок элемента, только для свойства `border-width`: `{border-width: 5px 10px 15px 3px;}`
`initial`	Устанавливает значение свойства в значение по умолчанию.
`inherit`	Наследует значение свойства от родительского элемента.

Цвет рамки - border-color

Свойство задаёт цвет рамок всех сторон одновременно. С помощью уточняющих свойств можно установить свой цвет для рамки каждой стороны элемента. Если для рамки цвет не задан, то он будет таким же, как и цвет текста элемента. Если в элементе нет текста, то цвет рамки будет таким же, как и цвет текста родительского элемента. Свойство не наследуется.

Цвет рамки - border-color

border-color

(border-top-color, border-right-color, border-bottom-color, border-left-color)

Значения:

`transparent`	Устанавливает прозрачный цвет для рамки. При этом ширина рамки остается. Можно использовать для смены цвета рамки при наведении курсора мыши на элемент, чтобы избежать смещение элемента.
цвет	Цвет рамок задается при помощи значений свойства color.

```
{border-color: #cacd58;}
```

`{1,4}`	Одновременное перечисление четырех разных цветов для рамок элемента, только для свойства `border-color` : `{border-color: #cacd58 #5faf8a #b9cea5 #aab238;}`
`initial`	Устанавливает значение свойства в значение по умолчанию.
`inherit`	Наследует значение свойства от родительского элемента.

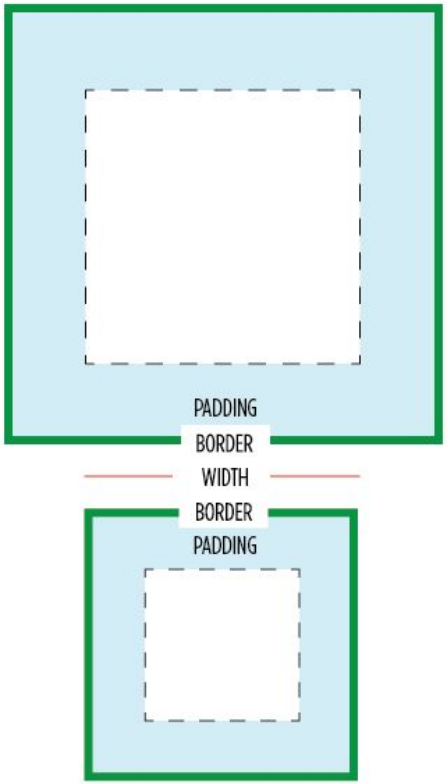
Задание рамки одним свойством

Свойство border позволяет объединить в себе следующие свойства: `border-width`, `border-style`, `border-color`, например:

```
div {
width: 200px;
height: 300px;
border: 1px solid grey;
}
```

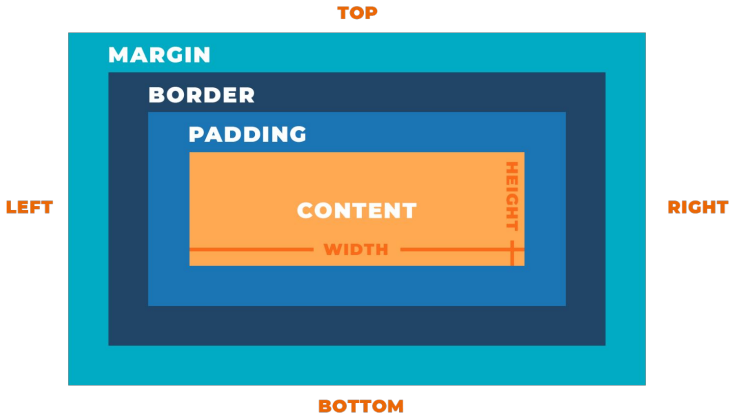
Примечание: Если какое-то из значений не указано, его место займет значение по умолчанию.

box-sizing

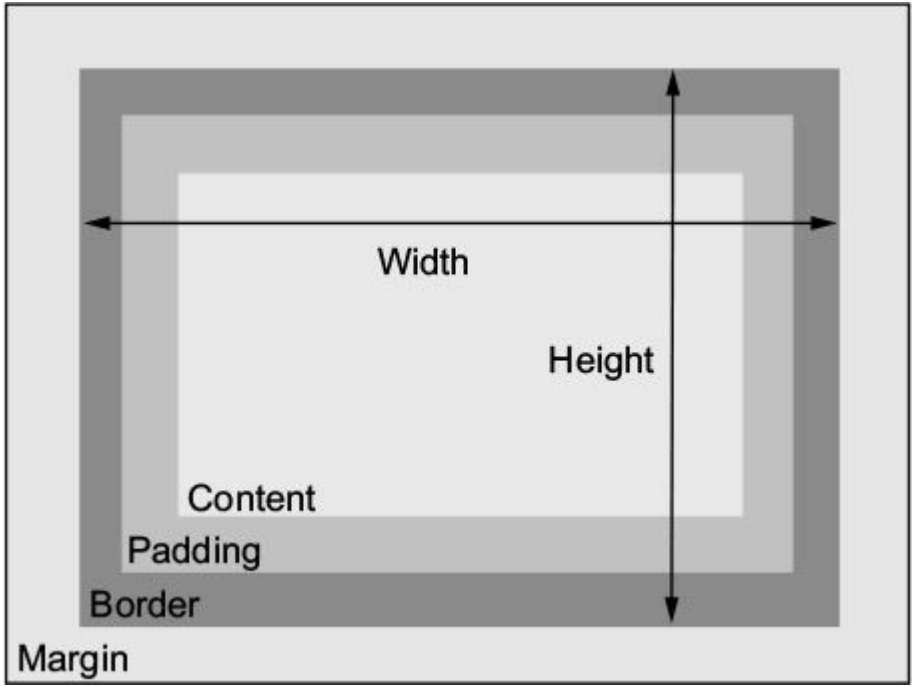


Свойство box-sizing в CSS определяет, как браузер рассчитывает общую ширину и высоту элемента, учитывая его содержимое, отступы (padding), границы (border) и внешние отступы (margin).

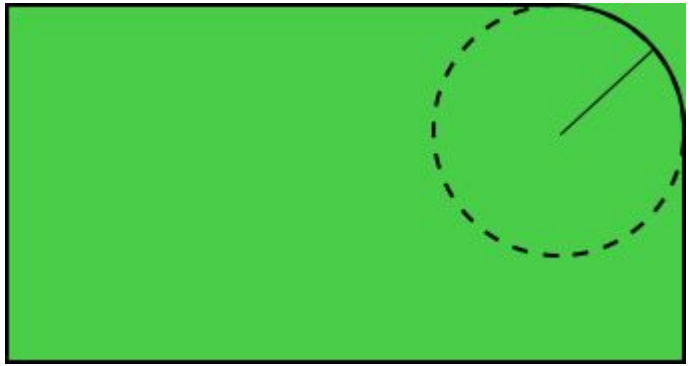
По умолчание значение равно: box-content - ширина и высота, которые будут заданы для элемент, будут включать только контент



Когда свойство box-sizing установлено в значение border-box, размеры элемента включают в себя содержимое, отступы и границы, но не включают внешние отступы. Это позволяет более прозрачно управлять размерами элементов



Скругление границ



border-radius

в CSS это свойство, которое используется для задания скругления углов для границы элемента. Оно позволяет создавать элементы с закругленными углами, делая интерфейс более эстетичным и приятным для восприятия.

```
border-radius: <размер>;
```

Значение <размер> указывает радиус скругления угла в пикселях (px), процентах (%) или других единицах измерения.

border-radius

- Указание одного значения для всех углов

```
.element {
    border-radius: 10px;
}
```

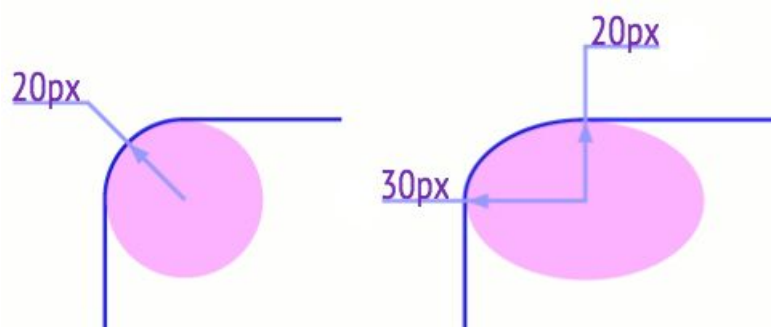
- Указание значений для каждого угла по отдельности (по часовой стрелке, начиная с верхнего левого угла):

```
.element {
    border-radius: 10px 20px 30px 40px;
}
```

border-radius

- **Указание значений для горизонтальных и вертикальных углов:**

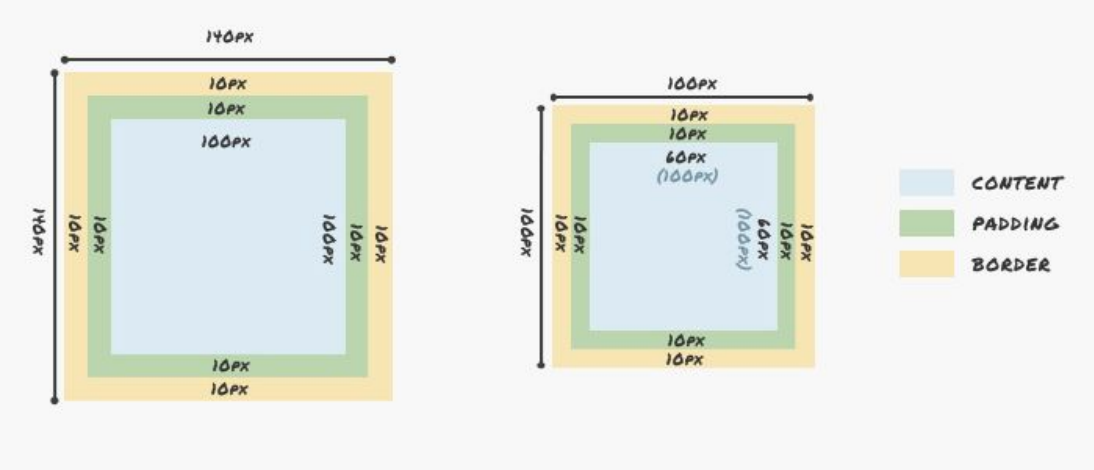
```
.element {
    border-radius: 10px 20px 10px 20px / 20px 10px 20px 10px;
}
```



Размеры

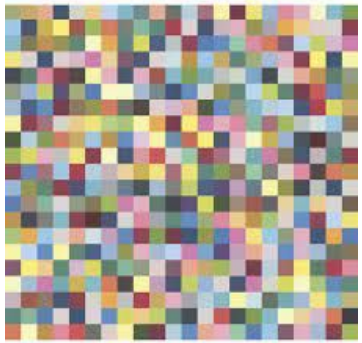
Основные способы задания размеров в CSS:

1. Пиксели (px):
2. Проценты (%)
3. EM и REM
4. Viewport Units (vw, vh)



Пиксели - это наиболее распространенная единица измерения в CSS и представляют собой точки на экране. Задавая размеры в пикселях, вы указываете конкретное количество пикселей в ширине, высоте или других размерах элемента.

```
div {
  width: 200px;
  height: 100px;
}
```



Проценты могут быть использованы для установки размеров элементов относительно размеров их родительских контейнеров.

```
.container {
  width: 80%;
}
```

Эм (em) и относительные эм (rem) - это единицы измерения, которые зависят от текущего размера шрифта.

em измеряет размер относительно размера текущего шрифта элемента, в то время как **rem** измеряет размер относительно размера шрифта корневого элемента (обычно <html>).

```
p {
  font-size: 1.2em;
}
```

```
.container {
  padding: 1rem;
}
```

Viewport Units (vw, vh):

Viewport units - это единицы измерения, которые зависят от размеров видимой области браузера (viewport).

Например, **1vw** представляет 1% ширины видимой области, а **1vh** представляет 1% высоты видимой области.

Это полезно для создания адаптивных макетов, которые реагируют на размер экрана пользователя.

```
.full-width {
  width: 100vw; /* Элемент будет занимать всю ширину видимой области браузера
}
```

```
.half-height {
  height: 50vh; /* Элемент будет занимать половину высоты видимой области
}
```

Абсолютные

px — соответствуют пикселям на экране. 1px интерпретируется как 1/96 дюйма.
cm — сантиметры. 1cm = 96px / 2.54.
in — дюймы. 1in = 96px = 2.54cm.
mm — миллиметры. 1mm = 1/10cm.
Q — четверть миллиметра. 1Q = 1/40cm.
pc — пики. 1pc = 1/16in.
pt — пункты. 1pt = 1/72in.

Ваша новая IT-профессия – Ваш новый уровень жизни

Программирование с нуля в
немецкой школе AIT TR GmbH

AIT
Hi-tech team