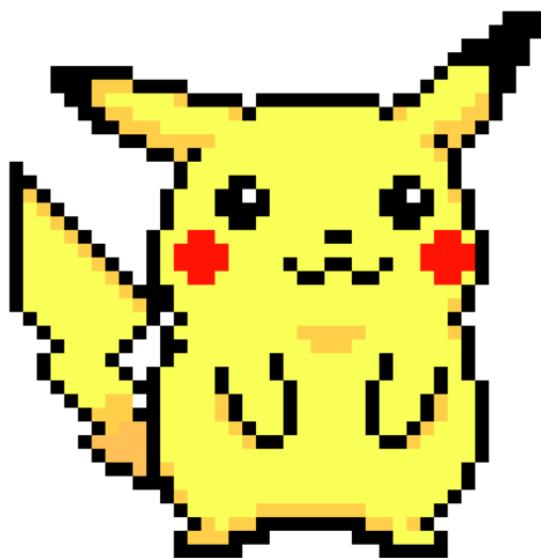


Модель отображения

Часть 1

Кристина Ильенко

Всё есть прямоугольники



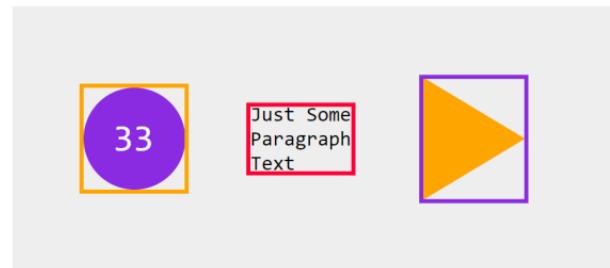
Are You Serious?



Как видит человек:



Как видит браузер:



← → 🔒 https://yandex.ru

Екатеринбург Конфиденциальность

Все плюсы здесь Настройка

Почта Завести почту

Войти Получить Плюс

Ленинский Сотовый оператор моя-20 дней по 30 раз в день посыпает мне сообщение-10 раз к ним обрат... koshiy_52@mail.ru

Завтра снегопад

Фильм Легенда 2015 - триллер Вам может понравиться

FORBES RUSSIA Серый кардинал YouTube: кто помогал Дудю, Саше Слиберг и Ивана зарабатывать миллионы

Яндекс

Найдётся всё. Например, сколько башен в московском кремле

HONOR 9X
Фитнес-трекер
HONOR Band 5 в ПОДАРОК

25.10.07.11

Погода
 -3° Ночью -5,
утром -5

Посещаемое
Недвижимость — нужна
ли регистрация
Беру — новинки электроники
Авто.ру — зимние шины для машин

Пробки
 8 Многокилометровые пробки

Телепрограмма Эфир
18:15 Что делать? Культура
18:15 Трофейный лес... ТВ Центр
18:20 Гамбургский счёт ОТР

Карта Екатеринбурга
Такси Расписания Сообщество соседей

Эфир Фильмы Сералии
Брат 1997, боевик
События ОТВ 24
Книга Илай 2010, боевик

КиноПоиск онлайн

Чаты

The screenshot shows the Yandex search engine interface. At the top, there's a navigation bar with links for 'Екатеринбург' (Yekaterinburg), 'Конфиденциальность' (Confidentiality), 'Все плюсы здесь' (All benefits here), and 'Настройка' (Settings). The main search area displays a news feed with several headlines, including 'Дания разрешила строительство «Северного потока-2»' (Denmark has approved the construction of the 'North Stream-2' pipeline) and 'Посольство Израиля в России приостановило работу' (The Israeli embassy in Russia has suspended its work). Below the news is a box for 'USD MOEX' at 63.91 and 'EUR MOEX' at 71.00, followed by a section for 'НЕФТЬ' (Oil) at 61.30. To the right, there's a sidebar with a 'Почта' (Email) section, a 'Ленинский' (Leninsky) news feed, and a 'Завтра снегопад' (Snowfall tomorrow) forecast. The bottom half of the page features a large advertisement for the HONOR 9X smartphone, showing the device and its accessories. There are also weather forecasts for Yekaterinburg (-3°C), traffic updates ('Пробки' - 8), and local services like 'Карта Екатеринбурга' (Map of Yekaterinburg) and 'Пассажирские перевозки' (Passenger transport).

https://github.com

Search or jump to...

Pull requests Issues Marketplace Explore

chrighter

Repositories

Find a repository...

[chrighter/chrighter.github.io](#)

[chrighter/CryptoProtocols](#)

[chrighter/cs](#)

[urfu-2017/teams](#)

[chrighter/webdev-task-4](#)

[chrighter/webdev-task-8](#)

[chrighter/webdev-task-7](#)

Show more

Your teams

Find a team...

[urfu-2018/mentors](#)

Learn Git and GitHub without any code!

Using the Hello World guide, you'll create a repository, start a branch, write comments, and open a pull request.

[Read the guide](#) [Start a project](#)

You've been added to the urfu-2018 organization!

Here are some quick tips for a first-time organization member.

- Use the switch context button in the upper left corner of this page to switch between your personal context ([chrighter](#)) and organizations you are a member of.  defunkt -
- After you switch contexts you'll see an organization-focused dashboard that lists out organization repositories and activities.

Securing software together
Introducing new ways to identify and prevent security vulnerabilities across your code base.

Welcome to the new dashboard. Get closer to the stuff you care about most.

Explore repositories

[microsoft/dts-gen](#)
dts-gen creates starter TypeScript definition files for any module or library.

[TypeScript](#)  4.1.4

[microsoft/TypeScript-Handbook](#)
The TypeScript Handbook is a comprehensive guide to the TypeScript language.

[JavaScript](#)  4.1.4

[microsoft/TypeScript-Node-Starter](#)
A starter template for TypeScript and Node with a detailed README describing how to use the two together.

[CSS](#)  6.3.4

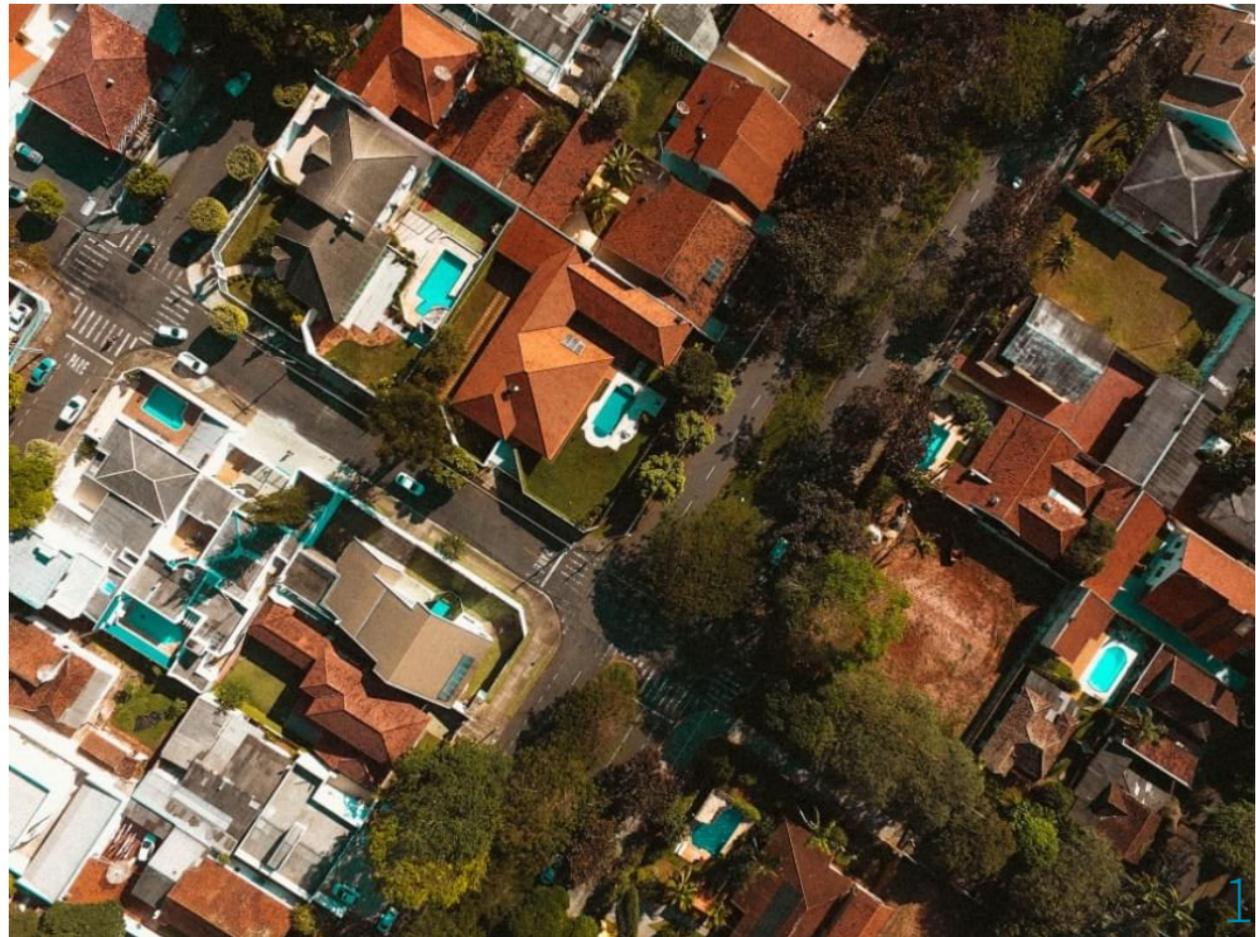
Explore more →

Каждый прямоугольник содержит
внутри себя еще несколько
прямоугольников

5 важных свойств элемента

- `width` – ширина
- `height` – высота
- `padding` – внутренний отступ
- `border` – граница
- `margin` – внешний отступ





Давайте поймем, что все 5 свойств прямоугольника

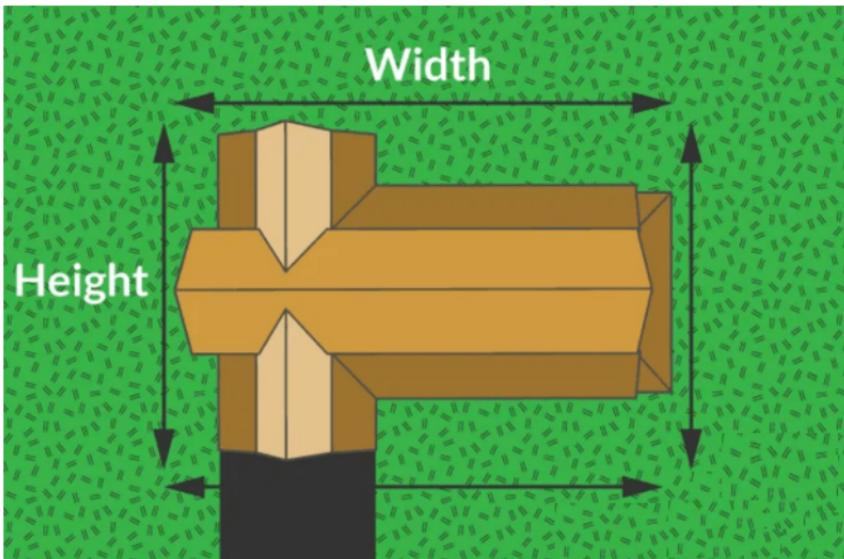
- `width` – ширина
- `height` – высота
- `padding` – внутренний отступ
- `border` – граница
- `margin` – внешний отступ

можно разделить на 3 зоны. И найдем аналогии

3 разные зоны:

Зона 1: высота (height) и ширина (width)
фактического элемента

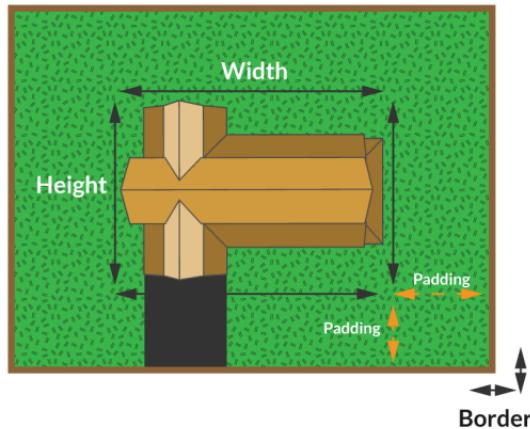
Зона 1 – дом



3 разные зоны:

Зона 2: внутренняя область вокруг элемента
(padding) и граница (border)

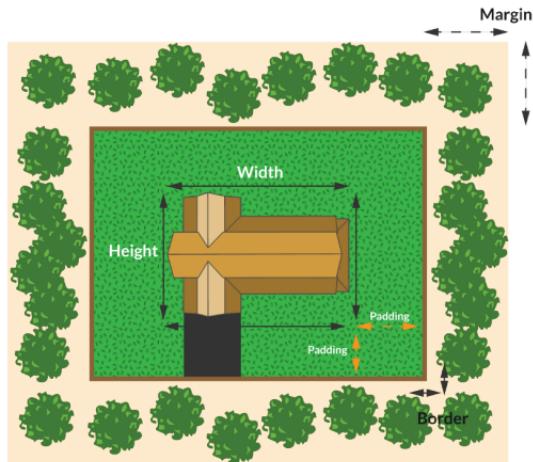
Зона 2 – двор и забор

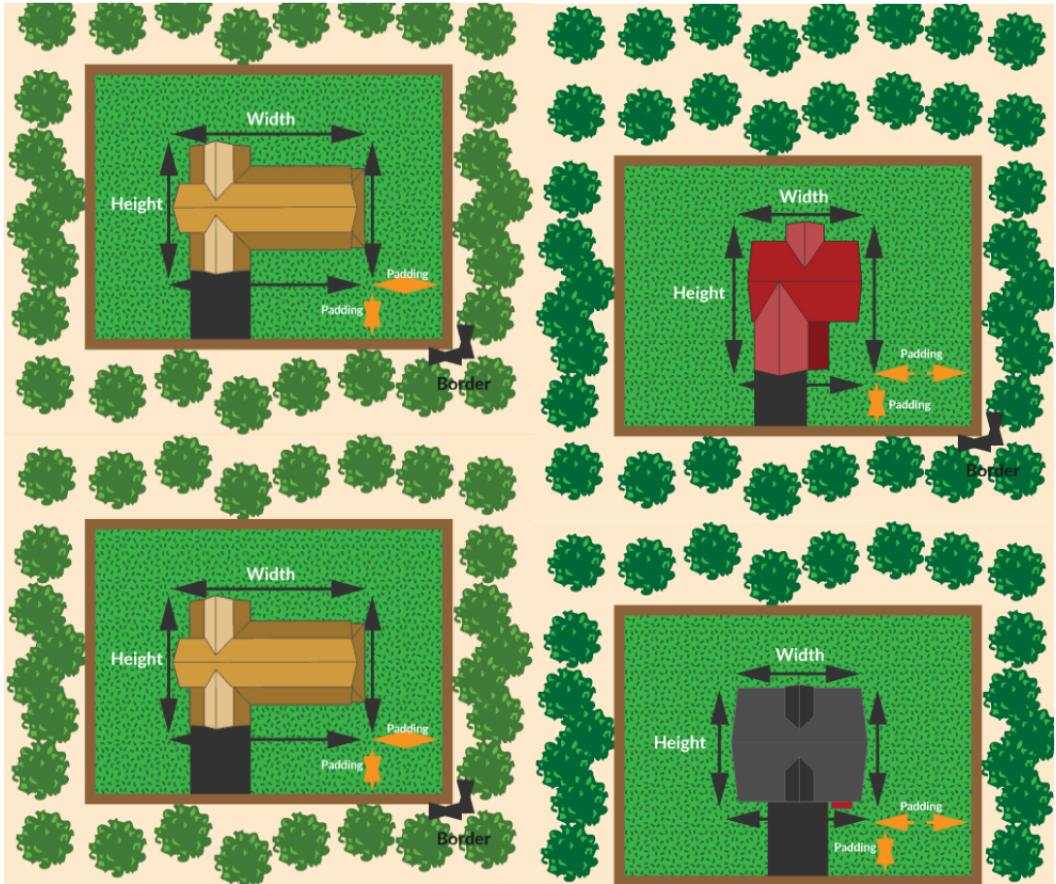


3 разные зоны:

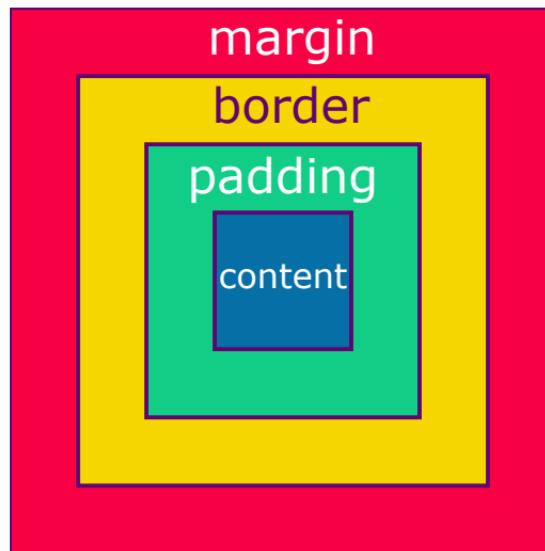
Зона 3: пустое пространство, отделяющее элемент от соседних (margin)

Зона 3 – деревья между соседними дворами





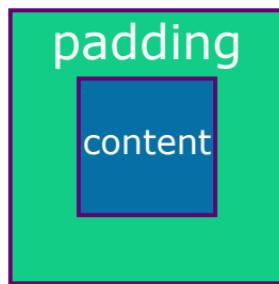
Боксовая модель CSS



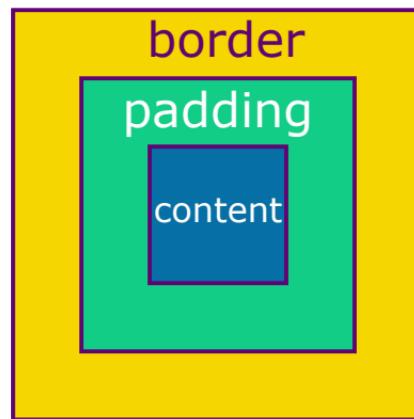
Content-box

content

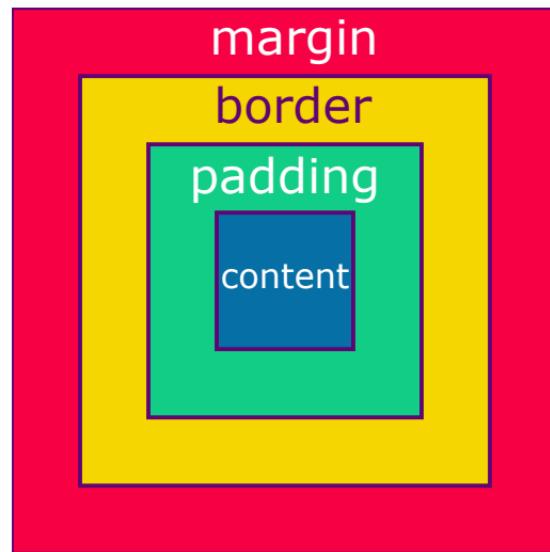
Padding-box

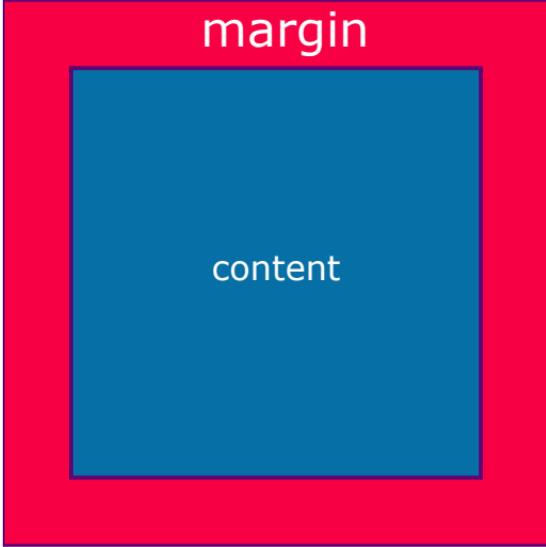


Border-box



Margin-box





margin

content



A diagram illustrating the structure of a block-level element. It consists of two nested rectangular layers. The outer layer is red and labeled "margin". The inner layer is blue and labeled "content".

margin

content

Модель визуального форматирования

Это алгоритм, который обрабатывает документ и отображает его на экране устройства.

Модель визуального форматирования задаёт трансформацию каждого элемента в документе и создаёт ноль, один или несколько боксов, согласно боксовой модели CSS

[MDN Web Docs](#)

На положение боксов на странице
влияет
размер бокса:

- точно задан
- вычисляется

На положение боксов на странице
влияет
тип бокса:

- block (блочный)
- inline (строчный)

На положение боксов на странице
влияет
схема позиционирования

Это алгоритм, используемый движком CSS
для расположения блоков на макете

На положение боксов на странице
влияет
схема позиционирования

Виды:

- Нормальный поток
- Схема float
- Схема абсолютного позиционирования

На положение боксов на странице
влияют
отношения между элементами DOM:
дочерними и соседними

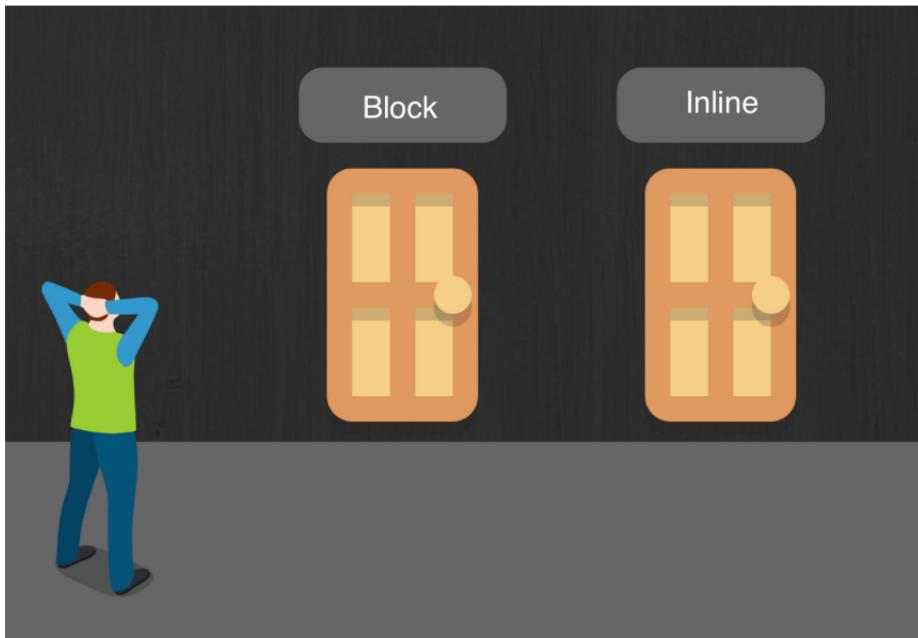
На положение боксов на странице
влияют

размеры и расположение окна
просмотра (viewport)

Viewport - это видимая пользователю
область веб-страницы, то, что может
увидеть пользователь, не прибегая к
прокрутке.

На положение боксов на странице
влияют
внутренние размеры содержащихся
изображений

Типы боксов



Боксы блочного уровня

Особенности

Каждый элемент блочного уровня генерирует один главный бокс блочного уровня, который

- содержит все дочерние блоки
- участвует в позиционировании элемента

Боксы блочного уровня

Особенности

Но некоторые элементы кроме главного бокса генерируют дополнительные, например

```
<div class="fake-list">  
    I will display as a list item  
</div>
```

```
.fake-list {  
    display: list-item;  
    list-style-position: inside;  
}
```

- I will display as a list item

Боксы блочного уровня

Блоchные элементы по умолчанию:

p div pre ol ul dl

h1 h2 h3 h4 h5 h6

address blockquote fieldset form hr table

Боксы блочного уровня

Необходимое CSS-свойство

Говорят, что элемент является боксом блочного уровня, когда:
`display: block; /* а так же list-item, table, flex, ... */`

Почему значение `flex`,
а элемент `блочный`?

Свойство display

Согласно спецификации **CSS Display 3**,
свойство **display** отвечает за:

- Внешнее поведение элемента — т.е. в каком контексте форматирования он сам участвует;
- Внутреннее поведение элемента — т.е. какой контекст форматирования действует в нем самом

Почему значение flex, а элемент блочный?

Элемент с `display: flex` участвует в **блочном контексте форматирования**, а для своих дочерних элементов он сам создает **гибкий (flex) контекст форматирования**, о котором расскажут в следующих лекциях

Боксы блочного уровня

Занимают всю доступную ширину и
выстраиваются сверху вниз

Боксы блочного уровня

Участвуют в блочном контексте
форматирования

Блочный контекст форматирования

Это часть визуального CSS-рендеринга веб-страницы

В нем действуют правила, изложенные в блочной модели CSS, которая определяет, как поля, границы и отступы элемента взаимодействуют с другими блоками в том же контексте.

[MDN Web Docs](#)

Боксы блочного уровня

Могут создавать вокруг себя анонимные
блочные боксы

Анонимные блочные боксы

Что это?

В случае смешанного контента алгоритм визуального форматирования добавляет дополнительную обёртку для текстового содержимого — анонимный бокс.

Анонимные блочные боксы

Пример

```
<div>Какой-то текст  
<p>с параграфом внутри</p>  
и текстом в конце.</div>
```

Какой-то текст
с параграфом внутри
и текстом в конце.

Анонимные блочные боксы

Особенности

- Анонимный блок не связан ни с одним элементом
- Анонимный блок не имеет названия, поэтому к нему нельзя применить CSS-стили для оформления
- Анонимные боксы наследуют свойства окружающего блока, а не наследуемые свойства принимают первоначальное значение.

Содержащий блок

Почему это важно?

На размеры и положение элемента часто влияет его содержащий блок

Определение содержащего блока элемента

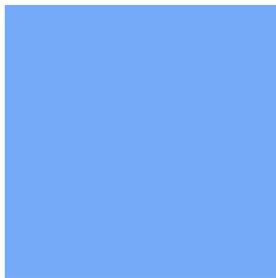
полностью зависит от значения свойства
position

В этой лекции мы рассматриваем случай,
когда position не задан (используется
его дефолтное значение),
и под содержащим блоком понимаем
родительский блок элемента

Математика блочных элементов

Ширина и высота

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



```
div {  
    width: 20%;  
    height: 10%;  
}
```



```
div {  
    width: 5em;  
    height: 5em;  
}
```



Ширина и высота

```
<div class="wrapper">
  <div class="inner">
    Yeah, you can be the greatest,<br>
    you can be the best
  </div>
</div>
```

```
.wrapper {
  width: 400px;
  height: 100px;
}
```

Yeah, you can be the greatest,
you can be the best

Ширина и высота

```
<div class="wrapper">
  <div class="inner">
    You can be the King Kong bangin'<br>
    on your chest
  </div>
</div>
```

```
.wrapper {
  width: 400px;
  height: 100px;
}
```

```
.inner {
  margin: 0 20px;
}
```

You can be the King Kong bangin'
on your chest

Ширина и высота

```
<div class="wrapper">
  <div class="inner">
    You can beat the world, you can win the war<br>
    You can talk to God, go bangin' on his door<br>
    You can throw your hands up, you can beat the clock
  </div>
</div>

.wrapper {
  width: 400px;
  height: 100px;
}

.inner {
  margin: 0 20px;
}
```

You can beat the world,
you can win the war

You can talk to God, go
bangin' on his door
You can throw your hands
up, you can beat the clock

Ширина и высота

```
<div class="wrapper">
  <div class="inner">
    You can move a mountain, you can break rocks<br>
    Some will call that practice, some will call that luck<br>
    But either way your going to the history books
  </div>
</div>

.wrapper {
  width: 400px;
  height: 100px;
}

.inner {
  height: 100px;
  margin: 0 20px;
}
```

You can move a
mountain, you can

break rocks

Some will call that
practice, some will
call that luck

But either way your
going to the history
books

Ширина и высота

```
<div class="wrapper">  
  <div class="inner">  
    Standing in the Hall of Fame<br>  
    And the world's gonna know your name<br>  
    'Cause you burn with the brightest flame  
  </div>  
</div>
```

```
.wrapper {  
  width: 500px;  
  height: 100px;  
}  
  
.inner {  
  width: 600px;  
  height: 100px;  
  margin: 0 20px;  
}
```

Standing in the Hall of Fame
And the world's gonna know your name
'Cause you burn with the brightest flame

При заданных фиксированных размерах
элемент не будет менять свои размеры,
но при этом вложенные элементы могут из
него выпадать

margin и padding

margin: 20px 10px 5px 34px;

margin-top: 20px;

margin-right: 10px;

margin-bottom: 5px;

margin-left: 34px;



Аналогично для padding

margin и padding

	top	right	bottom	left
margin:	2px	30px	400px	5000px;
margin:	2px	30px	400px /* 2px	30px */;
margin:	2px /* 2px	2px	2px	2px */;

Аналогично для padding

Процентные значения

Ширина, заданная в процентах, считается относительно ширины содержащего блока

```
.wrapper {  
    width: 816px;  
    background: #3af4d5;  
}  
.inner {  
    width: 50%;  
}
```

You can go the distance, you can
run a mile
You can walk straight through hell
with a smile

Высота, заданная в процентах, считается относительно высоты содержащего блока,

НО ТОЛЬКО ЕСЛИ ВЫСОТА СОДЕРЖАЩЕГО БЛОКА
ЗАДАНА ЯВНО

```
.wrapper {  
    width: 470px;  
    height: 400px;  
}  
  
.inner {  
    height: 50%;  
    width: 60%;
```

You can be a hero, you
can get the gold
Breaking all the
records they thought
would never be broke

Высота, заданная в процентах, считается относительно высоты содержащего блока,

но только если высота содержащего блока задана явно

```
.wrapper {  
    width: 457px;  
}  
.inner {  
    height: 50%;  
    width: 70%;
```

Yeah, do it for your
people,
do it for your pride
How are you ever gonna
know if you never even
try?

- margin-left/right и padding-left/right, заданные в процентах, считаются относительно **ширины** содержащего блока
- margin-top/bottom и padding-top/bottom, заданные в процентах, считаются относительно **ширины** содержащего блока

border

border: <размер> <тип> <цвет>;

border-left: <размер> <тип> <цвет>;

border-width: <размер> <размер> <размер> <размер>;

border-type: <тип> <тип> <тип> <тип>;

border-color: <цвет> <цвет> <цвет> <цвет>;

border-left-width: <размер>;

border-left-type: <тип>;

border-left-color: <цвет>;

border

```
div {  
    border: 20px dotted blue;  
}
```

• Do it for your country, do it for your name •

border

```
div {  
  border: 20px dashed blue;  
}
```

'Cause there's gonna be a day, when you're

border

```
div {  
  border: 20px solid blue;  
}
```

Standing in the Hall of Fame

border

```
div {  
  border-width: 20px;  
  border-style: solid;  
  border-color: blue red;  
}
```

And the world's gonna know your name

border

```
div {  
    border-width: 30px 20px;  
    border-style: solid dotted;  
    border-color: blue red;  
}
```

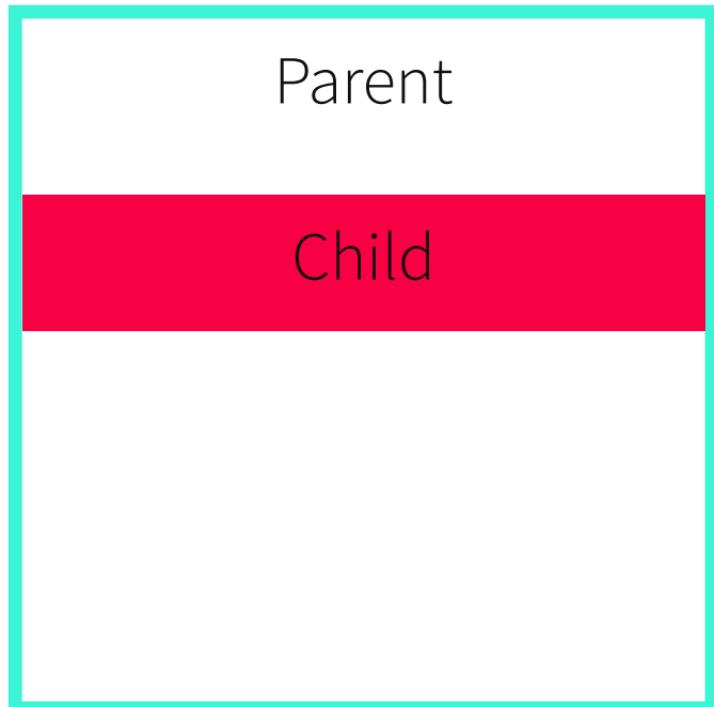
'Cause you burn with the brightest flame

Перерыв



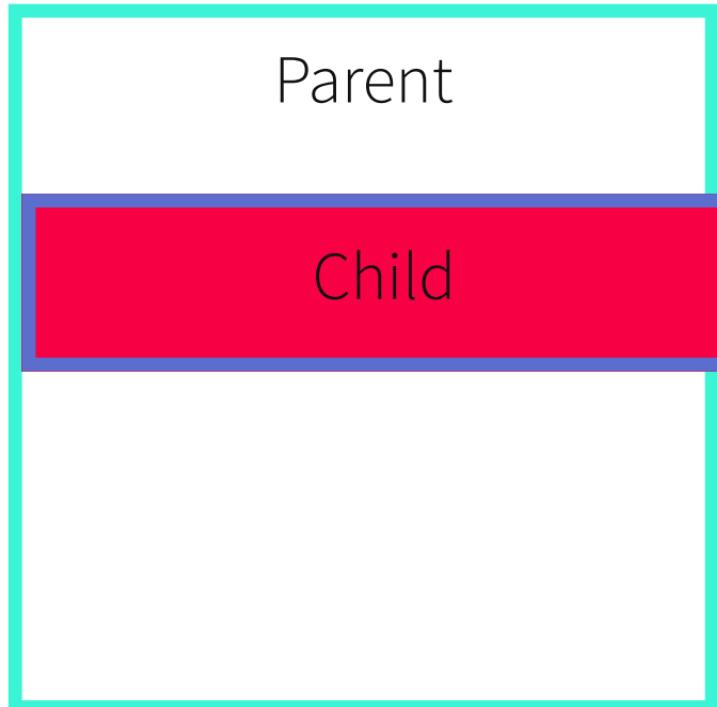
Пример #1

```
.parent {  
    width: 500px;  
    height: 500px;  
  
    border: 10px solid #3af4d5;  
}  
  
.child {  
    width: 100%;  
    height: 100px;  
    margin: 2em auto;  
    background-color: #f70044;  
}
```



Пример #2

```
.parent {  
    width: 500px;  
    height: 500px;  
  
    border: 10px solid #3af4d5;  
}  
  
.child {  
    width: 100%;  
    height: 100px;  
    margin: 2em auto;  
    background-color: #f70044;  
    /* добавим границу */  
    border: 10px solid #5b6dc9;  
    /* и внутренний отступ: */  
    padding: 5px;  
}
```



Что происходит?

По умолчанию в блочной модели CSS выставленная **ширина** и **высота** применяются только к **content-box** элемента

Если элементу дополнительно заданы **border** или **padding**, то их значения **добавляются к ширине и высоте**

Если размер содержащего элемента явно заданы, а сумма размеров **border**, **padding** и размеров элемента превышают их, элемент не поместится в содержащем

[MDN Web Docs](#)

Как быть?

Правильно использовать свойство
box-sizing

Какие варианты использования?

- content-box (по умолчанию)
- border-box

box-sizing: content-box

Ширина и высота, которую устанавливаем элементу, применяется к его content-box

Если задаем элементу дополнительно border или padding с ненулевыми размерами, то эти значения будут добавлены к заданным выше

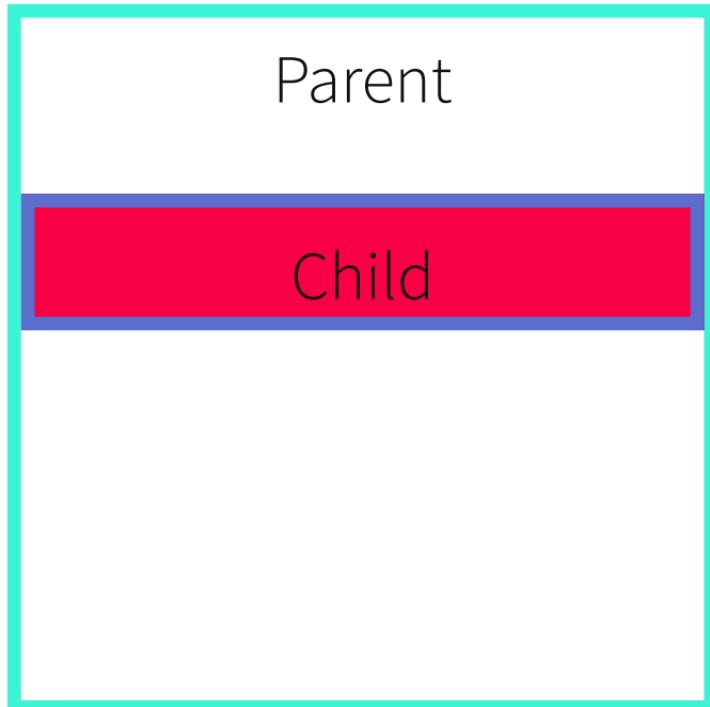
Поэтому размеры элемента при рендеринге станут больше, чем указанные первоначально

box-sizing: border-box

При рендеринге браузер будет учитывать заданные значения `border` или `padding` так, чтобы не выходить за указанные значения высоты и ширины

Пример #3

```
.parent {  
    width: 500px;  
    height: 500px;  
  
    border: 10px solid #3af4d5;  
}  
  
.child {  
    width: 100%;  
    height: 100px;  
    margin: 2em auto;  
    background-color: #f70044;  
    /* content-box -> border-box */  
    box-sizing: border-box;  
    border: 10px solid #5b6dcd;  
    padding: 5px;  
}
```



Нетрииальная математика блочных элементов

Пример

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    width: 400px;  
    margin: 0 auto;  
    background: #00bbf0;  
}
```

And you'll be on the walls of the
Hall of Fame

Что сделал браузер?

Браузер самостоятельно посчитал отступы

Нетрииальная математика блочных элементов

- Пусть `box-sizing: content-box`
- $\text{margin} + \text{border} + \text{padding} + \text{width} = \text{width}$ содержащего блока
- $\text{margin} + \text{width} = \text{width}$ содержащего блока - $\text{border} - \text{padding}$

При этом `border/padding`: либо заданы, либо `auto`, т.е.

Значит значения в правой части формулы известны

Браузеру остается вычислить `margin-left`, `margin-right` и ширину блока

Пример #1

Не задано 1 значение – margin-left

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    width: 300px;  
    margin-right: 300px;  
}
```



margin-left = 0, margin-right вычисляется

margin-left + margin-right + width = width c.6. - border - padding

$$0 + [366\text{px}] + 300\text{px} = 666\text{px}$$

Пример #2

Не задано 1 значение – margin-right:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-left: 100px;  
    width: 300px;  
}
```



margin-right вычисляется

margin-left + margin-right + width = width c.6. - border - padding

100px + [266px] + 300px = 666px

Пример #3

Не задано 1 значение – width:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-left: 100px;  
    margin-right: 200px;  
}
```



width вычисляется

margin-left + margin-right + width = width c.6. - border - padding

100px + 200px + [366px] = 666px

Пример #4

Не задано 2 значения – margin-left и margin-right:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    width: 300px;  
}
```



margin-left равен 0, margin-right вычисляется

margin-left + margin-right + width = width c.6. - border - padding

$$0 + [366px] + 300px = 666px$$

Пример #5

Не задано 2 значения – margin-left и width:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-right: 200px;  
}
```



margin-left равен 0, width вычисляется

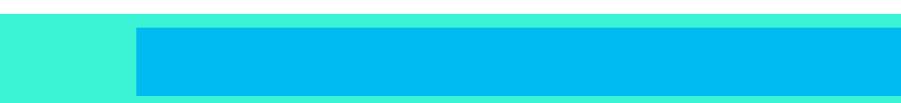
margin-left + margin-right + width = width c.6. - border - padding

$$0 + 200\text{px} + [466\text{px}] = 666\text{px}$$

Пример #6

Не задано 2 значения – margin-right и width:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-left: 100px;  
}
```



margin-right равен 0, width вычисляется

margin-left + margin-right + width = width c.6. - border - padding

100px + 0 + [566px] = 666px

Пример #7

Заданы margin-left: auto и width:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-left: auto;  
    width: 300px;  
}
```



margin-right равен 0, margin-left вычисляется

margin-left + margin-right + width = width c.6. - border - padding

$$[366\text{px}] + 0 + 300\text{px} = 666\text{px}$$

Пример #8

Заданы margin-right: auto и width:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-right: auto;  
    width: 300px;  
}
```



margin-left равен 0, margin-right вычисляется

margin-left + margin-right + width = width c.6. - border - padding

$$0 + [366px] + 300px = 666px$$

Пример #9

Заданы margin-right: auto, margin-left: auto и width:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin: 0 auto;  
    width: 300px;  
}
```



расчетное пространство распределяется равномерно

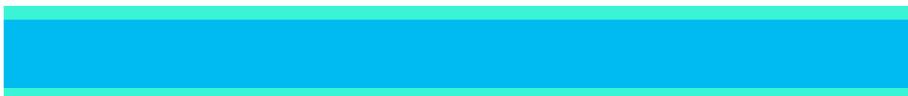
margin-left + margin-right + width = width c.6. - border - padding

$$[183\text{px}] + [183\text{px}] + 300\text{px} = 666\text{px}$$

Пример #10

Задан любой из margin в auto, width не задана:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-left: auto;  
}
```



оба margin = 0, width вычисляется

margin-left + margin-right + width = width c.6. - border - padding

0 + 0 + [666px] = 666px

Пример #11

Заданы все 3 значения:

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
    margin-left: 100px;  
    margin-right: 300px;  
    width: 300px;
```

пересчитываем margin-right

$\text{margin-left} + \text{margin-right} + \text{width} = \text{width c.6. - border - padding}$

$100\text{px} + [266\text{px}] + 300\text{px} = 666\text{px}$

Пример #12

Не задано ни одно из всех 3 значений

```
.wrapper {  
    width: 666px;  
    background: #3af4d5;  
    padding: 10px 0;  
}  
  
.inner {  
    height: 50px;  
    background: #00bbf0;  
}
```



margin-left = margin-right = 0, **width** вычисляется

margin-left + margin-right + **width** = width c.6. - border - padding

$$0 + 0 + [666\text{px}] = 666\text{px}$$

Пример #13 (Самостоятельный)

Заданы margin-left: auto и width:

```
.wrapper {  
    width: 666px;  
    padding: 10px 0;  
}  
  
.inner {  
    width: 500px;  
    margin-left: auto;  
}
```

margin-left + margin-right + width = width c.6. - border - padding



margin-right равен 0, margin-left вычисляется

$$[166\text{px}] + 0 + 500\text{px} = 666\text{px}$$

Как браузер это делает?

$\text{margin-left} + \text{margin-right} + \text{width} = \text{width}$ содержащего блока - border - padding

- Если не задано 1 значение:
 - 1) Если не задан margin-left, он равен 0, margin-right вычисляется
 - 2) Если не задан margin-right, он вычисляется
 - 3) Если не задана width, она вычисляется
- Если не задано 2 значения:
 - 1) Если не заданы margin-left и margin-right, то margin-left равен 0, margin-right вычисляется
 - 2) Если не заданы margin-left и width, то margin-left равен 0, width вычисляется
 - 3) Если не заданы margin-right и width, то margin-right равен 0, width вычисляется

Как браузер это делает?

$\text{margin} + \text{width} = \text{width} \text{ содержащего блока} - \text{border} - \text{padding}$

- Если задано что-то в значении auto и width:
 - 1) Если margin-left: auto, то margin-right = 0, margin-left вычисляется
 - 2) Если margin-right: auto, то margin-left = 0, margin-right вычисляется
 - 3) Если оба margin заданы в auto, то расчетное пространство распределяется равномерно
- Если задано что-то в значении auto и не задана width:
Если любой из margin в auto, то оба margin 0, width вычисляется
- Если заданы все 3 значения:
Пересчитываем margin-right
- Если не заданы все 3 значения:
 $\text{margin-left} = \text{margin-right} = 0$, width вычисляется

Схлопывание отступов

MDN Web Docs

Пример

```
.first { margin-bottom: 30px; }
```

```
.second { margin-top: 60px; }
```

БЛОКИ

Ожидание

Реальность

Be students, be
teachers

Be politicians, be
preachers

Be students, be
teachers

Be politicians, be
preachers

Be students, be
teachers

Be politicians, be
preachers

Произошло схлопывание
внешних "пересекающихся" отступов
элементов-братьев

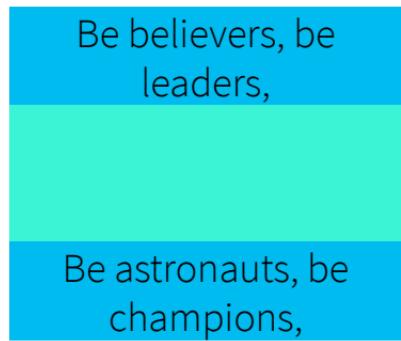
Высчитывание результирующего margin'a

Если у двух блочных элементов пересекаются margin'ы, то

1. Берутся максимальный и минимальный среди всех margin'ов
2. Если максимальный < 0, то максимум — 0
3. Если минимальный > 0, то минимум — 0.
4. Результирующий margin = max + min

Если у двух блочных элементов пересекаются margin'ы, то если максимальный < 0 , то $\max = 0$, а в нашем случае $\max = 100\text{px}$ если минимальный > 0 , то $\min = 0$, тогда в нашем случае $\min = 0$
total margin = max + min, в нашем случае $100\text{px} + 0 = 100\text{px}$

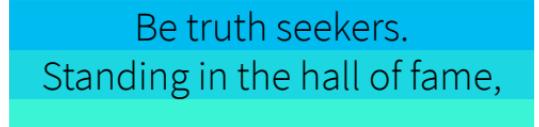
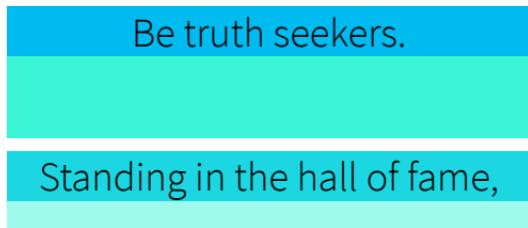
```
.first { margin-bottom: 60px; }  
.second { margin-top: 100px; }
```



Если у двух блочных элементов пересекаются margin'ы, то если максимальный < 0 , то $\max = 0$, а в нашем случае $\max = 60\text{px}$ если минимальный > 0 , то $\min = 0$, а в нашем случае $\min = -60\text{px}$

total margin = $\max + \min$, в нашем случае $60\text{px} + (-60\text{px}) = 0$

```
.first { margin-bottom: 60px; }  
.second { margin-top: -60px; }
```



Если у двух блочных элементов пересекаются margin'ы, то если максимальный < 0 , то $\max = 0$, а в нашем случае $\max = 60\text{px}$ если минимальный > 0 , то $\min = 0$, а в нашем случае $\min = -80\text{px}$
total margin = max + min, в нашем случае $60\text{px} + (-80\text{px}) = -20\text{px}$

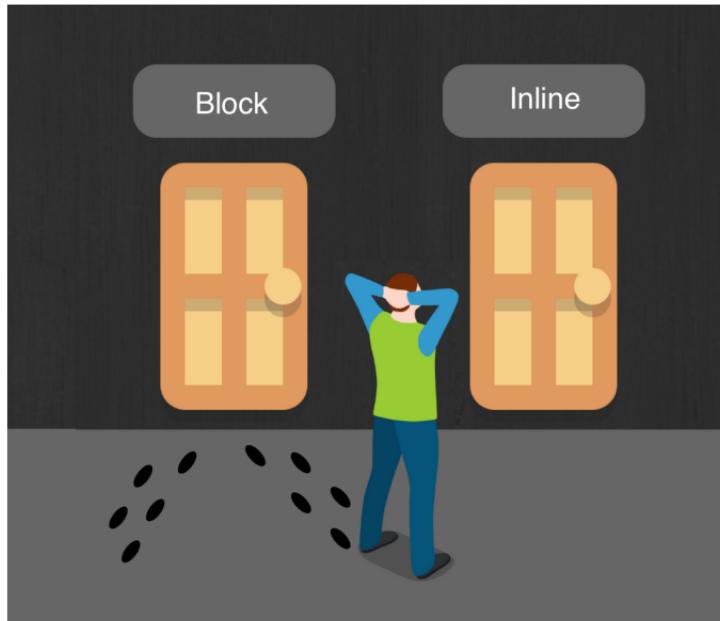
```
.first { margin-bottom: 60px; }  
.second { margin-top: -80px; }
```

And the world's gonna
know your name,

Cause you burn with
the brightest flame

And the world's gonna
Cause you burn with
the brightest flame

Боксы строчного уровня (inline)



Боксы строчного уровня

Строчные элементы генерируют
бокс(ы) строчного уровня

Боксы строчного уровня

Строчные элементы по умолчанию:

b	big	i	small	textarea	abbr
acronym	cite	code	strong	img	q
span	button	sub	sup	input	label

Боксы строчного уровня

Необходимое CSS-свойство

Говорят, что элемент является боксом строчного уровня, когда:

```
display: inline; /* а так же inline-block, inline-table,  
    inline-flex, ... */
```

Боксы строчного уровня

Особенности

Выглядят как строки

Боксы строчного уровня

Особенности

Выстраиваются слева направо, сверху вниз

Боксы строчного уровня

Особенности

Участвуют в строчном контексте
форматирования

Боксы строчного уровня

Особенности

Могут создать вокруг себя анонимные
строчные боксы

```
<div>Some inline  
text <span>followed  
by a span</span>  
  
followed by  
more inline text.</div>
```

Some inline text followed by a span followed by
more inline text.

Математика inline элементов

Инлайновые блоки не реагируют на
задание ширины и высоты

То есть сколько текста положим внутрь инлайн блока,
такая ширина этого блока и будет

Математика inline элементов

Не реагируют на вертикальные margin'ы

Реагируют на горизонтальные margin'ы

Математика inline элементов

Инлайновые padding'и не затрагивают
высоту строки

Но при этом сами padding'и можно сделать
видимыми

Поверстаем!

Хотим получить:

Помощь ?

Попытка #1

```
<div>Помощь
    <span class="button">
        ?
    </span>
</div>

.button {
    width: 63px;
    height: 63px;
    background: #ff0;
    text-align: center;
    cursor: pointer;
}
```

Помощь ?

Попытка #2

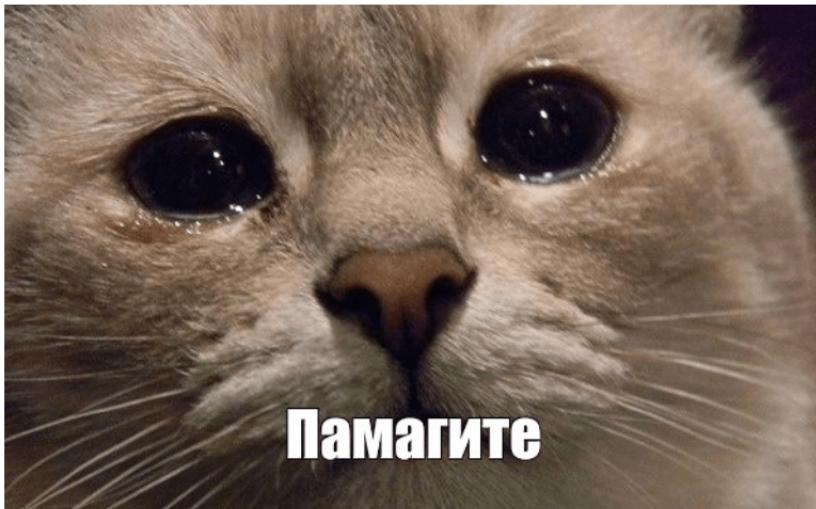
```
<div>Помощь
    <div class="button">
        ?
    </div>
</div>

.button {
    width: 63px;
    height: 63px;
    background: #ff0;
    text-align: center;
    cursor: pointer;
}
```

Помощь

?

Что делать?



Использовать
display: inline-block!

FINISH HIM!

```
<div>Помощь
    <div class="button">
        ?
    </div>
</div>

.button {
    display: inline-block;
    width: 63px;
    height: 63px;
    background: #ff0;
    text-align: center;
    cursor: pointer;
}
```

Помощь ?

inline-block элементы

Особенности

Реагируют на ширину и высоту

inline-block элементы

Особенности

Реагируют на вертикальные margin'ы

inline-block элементы

Особенности

Padding'и влияют на высоту строки

inline-block элементы

Особенности

Внешнее поведение как у inline элементов, а
внутри создают блочный контекст
форматирования

ИТОГИ

- Элементы на странице отображаются в виде боксов, каждый из которых имеет контентную часть (content), отступы (padding и margin) и границу (border)
- Боксы бывают двух видов – блочного уровня и строчного уровня, задать можно с помощью CSS-свойства `display`
- Для блочных боксов размеры и отступы можно задать либо явно, либо оставить вычисляемыми
- Строчные боксы не будут реагировать на заданные ширину, высоту и вертикальные отступы
- `inline-block` элементы – элементы, участвующие в строчном контексте форматирования, внутри которых действует блочный контекст форматирования

Спасибо!