CSS, Сетки

Азы CSS

CSS — это язык для управления внешним видом HTML-документа. С помощью CSS можно задавать параметры отображения любого тега: ширину и высоту, отступы, цвет и размер шрифта, фон и так далее.

```
<p style="color: red;">Текст</p>
```

Синтаксис свойство: значение;

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

НИИТ

CSS: общий синтаксис

CSS-правило:

```
селектор {
   свойство: значение;
   свойство: значение;
   свойство: значение;
}
```

НИИТ

Классы в CSS

```
<p class="text-class1">Текст</p>
<p class="text-class2">Текст</p>

.text-class1 {
  color: red;
}
.text-class2 {
  color: green;
}
```

НИИТ

Группы свойств

В CSS существует огромное количество свойств. Их можно разбить на следующие группы:

- оформление текста;
- работа с размерами и отступами;
- позиционирование элементов;
- создание сеток;
- декоративные: цвета, фон, тени;
- другие.

Оформление текста

```
.underline {
   text-decoration: underline;
}
.bold {
   font-weight: bold;
}
.italic {
   font-style: italic;
}
```

Размеры и отступы

```
.p-first {
  margin-left: 50px;
}
.p-second {
  font-weight: padding: 20px;
}
.p-third {
  font-style: width: 50%
}
```

Позиционирование элементов

```
.p-absolute {
  position: absolute;
  left: 100px;
  bottom: 200px;
}
```

Сетки

```
.left-column {
   float: left;
}
.right-column {
   float: right;
}
.footer {
   clear: both;
}
```

Декоративные свойства

```
.alert {
  background-color: #dff0d8;
  color: #468847;
  border-radius: 5px;
}
```

Каскадность

```
.p {
  padding: 10px;
}
.truth {
  background-color: #dff0d8;
}

//.p{background-color: #dff0d8;}
//.truth{background-color: #aaddff;}
```

Приоритеты каскадности

Когда для одного и того же элемента есть несколько CSS-правил с одинаковыми свойствами, браузер использует понятия приоритетов и специфичности, чтобы выбрать значение свойства из нескольких возможных.
Упрощённо, можно сказать что:

1. CSS-правила в значении атрибута <style> самые приоритетные,
2. за ними идёт селектор с <id>,
3. затем селектор с классом,
4. затем селектор с именем тега.

Наследование

```
body {
   color: red;
}
ul {
   font-style: italic;
}
```

Наследование работает не для всех свойств. Некоторые свойства применяются только к самому элементу и не переходят к его потомкам. К таким ненаследуемым свойствам относятся: ширина, высота, отступы, режим позиционирования и другие.

Селекторы

Контекстные(вложенные) селекторы

```
p strong { … }
ul .hit{ … }
footer .menu a{ … }
```

Соседние селекторы

```
p strong { … }
ul .hit{ … }
footer .menu a{ … }
```

Дочерние селекторы

```
<ul>
   <li><span>...</span> </li>
   <li><span>...</span> </li>
</ul>

ul > li { … }
ul > li > span{ … }
```

Псевдоклассы

Псевдоклассы — это дополнения к обычным селекторам, которые делают их ещё точнее и мощнее.

```
селектор:псевдокласс { … }
a:visited { … }
li:last-child { … }
.alert:hover { … }
```

Псевдокласс first-child позволяет выбрать первый дочерний элемент родителя, а last-child — последний дочерний элемент.

НИИТ

Псевдокласс :nth-child

С помощью псевдокласса nht-child можно выбирать теги по порядковому номеру, не используя классы.
Синтаксис псевдокласса: селектор:nth-child(выражение).
Выражением может быть число или формула.
Например:

```
li:nth-child(2) { … }
li:nth-child(4) { … }
li:nth-child(2n) { … }
```

НИИТ

Псевдокласс :hover

Этот псевдокласс позволяет выбрать элемент, когда на него наведён курсор мыши и кнопка мыши не нажата.
Примеры:

```
a:hover { … }
tr:hover { … }
.menu-item:hover { … }
```

Динамические эффекты :hover

Этот псевдокласс позволяет выбрать элемент, когда на него наведён курсор мыши и кнопка мыши не нажата.
Примеры:

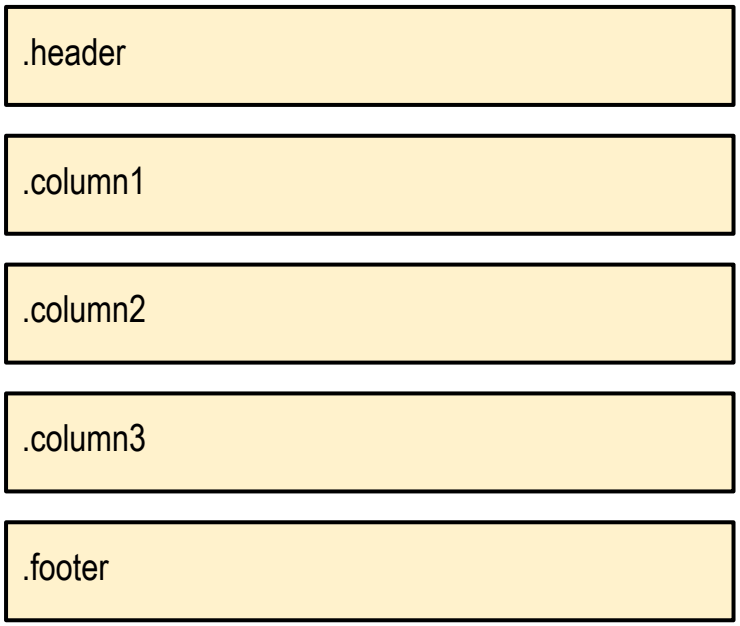
```
li.top ul.submenu {
   display: none;
}

li.top:hover ul.submenu {
   display: block;
}
```

Что такое сетка?

Что такое поток документа?

Что такое поток документа?

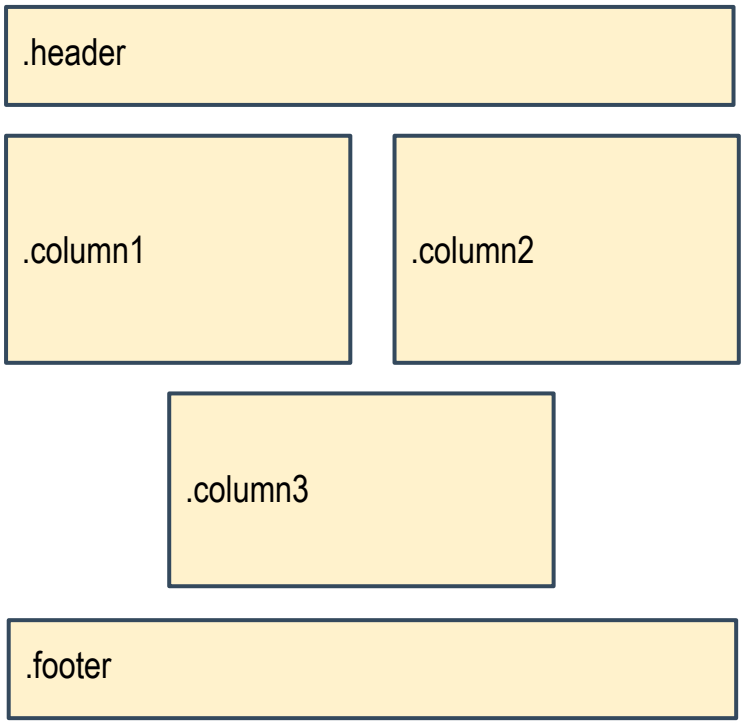
```
<body>
    <div class="header"></div>
    <div class="column1"></div>
    <div class="column2"></div>
    <div class="column3"></div>
    <div class="footer"></div>
</body>
```



НИИТ

Что такое поток документа?

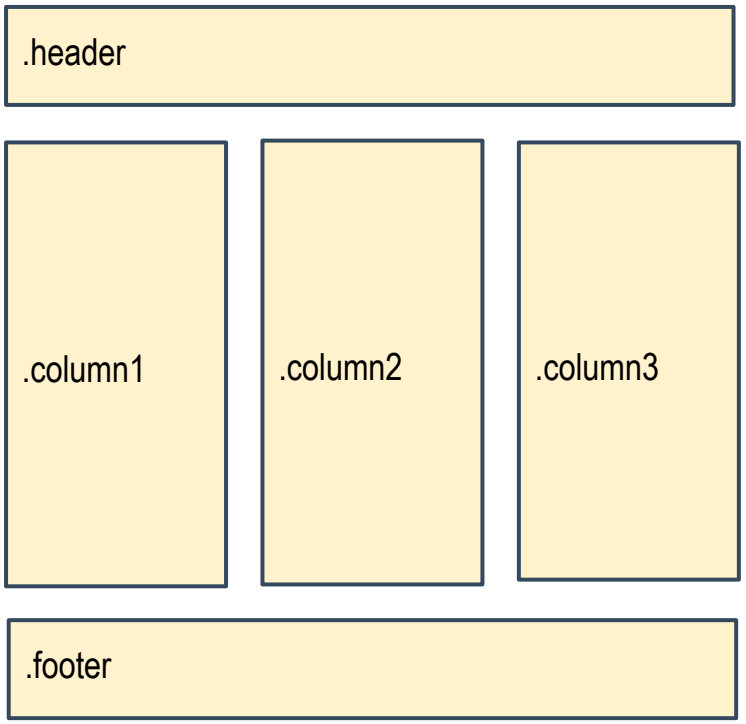
```
<body>
    <div class="header"></div>
    <div class="column1"></div>
    <div class="column2"></div>
    <div class="column3"></div>
    <div class="footer"></div>
</body>
```



НИИТ

Что такое поток документа?

```
<body>
    <div class="header"></div>
    <div class="column1"></div>
    <div class="column2"></div>
    <div class="column3"></div>
    <div class="footer"></div>
</body>
```



Как управлять потоком?

Как управлять потоком?

- Управляя потоком, можно строить необходимые сетки.
- Для управления потоком нужно знать, как работает блочная модель документа.

Блочная модель документа

CSS–свойства, относящиеся к блочной модели:

- Влияют на поведение элементов в потоке (изменяют его тип).
- Изменяют размер элементов и занимаемую ими площадь.

Типы элементов

Базовые:

- блочные,
- строчные.

Дополнительные:

- блочно-строчные,
- табличные,
- другие.

Блочные элементы

Блочные элементы — прямоугольные области на странице.

Блочными по умолчанию являются:

<section>

<h1>…<h6><p>

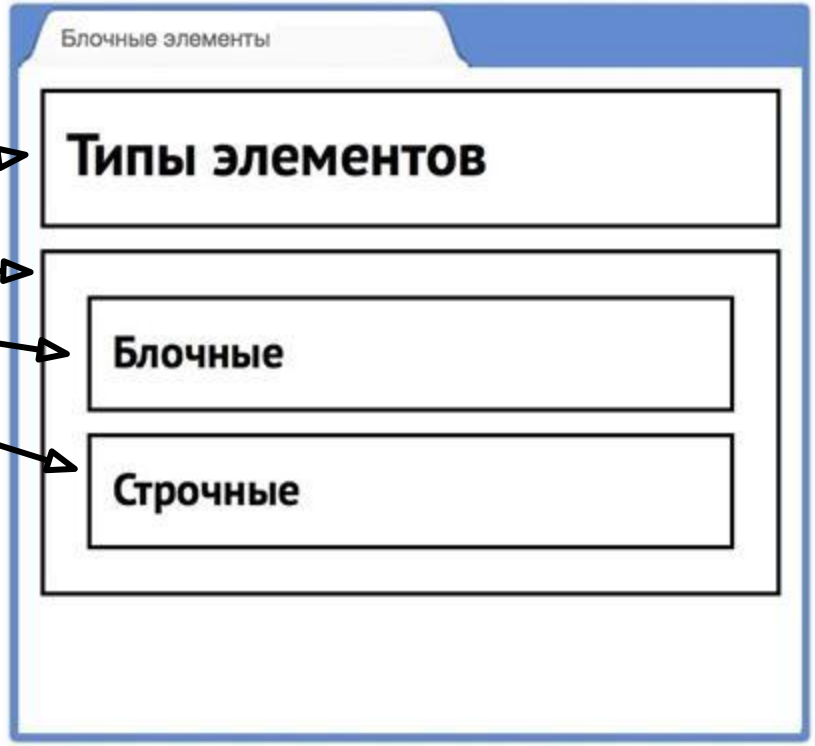
/* и другие теги для выделения крупных блоков текста */

НИИТ

Особенности блочных

- Принудительный перенос строки до и после.
- Воспринимают ширину, высоту, внутренние и внешние отступы.
- Занимают всё доступное по ширине пространство.
- По высоте подстраиваются под содержимое.

```
1  <!DOCTYPE html>                                    HTML
2  <html>
3      <head>…</head>
8      <body>
9          <h1>Типы элементов</h1>
10         <div>
11             <h2>Блочные</h2>
12             <h2>Строчные</h2>
13         </div>
14     </body>
15 </html>
```



Строчные элементы

Строчные элементы — фрагменты текста.

Строчными по умолчанию являются:

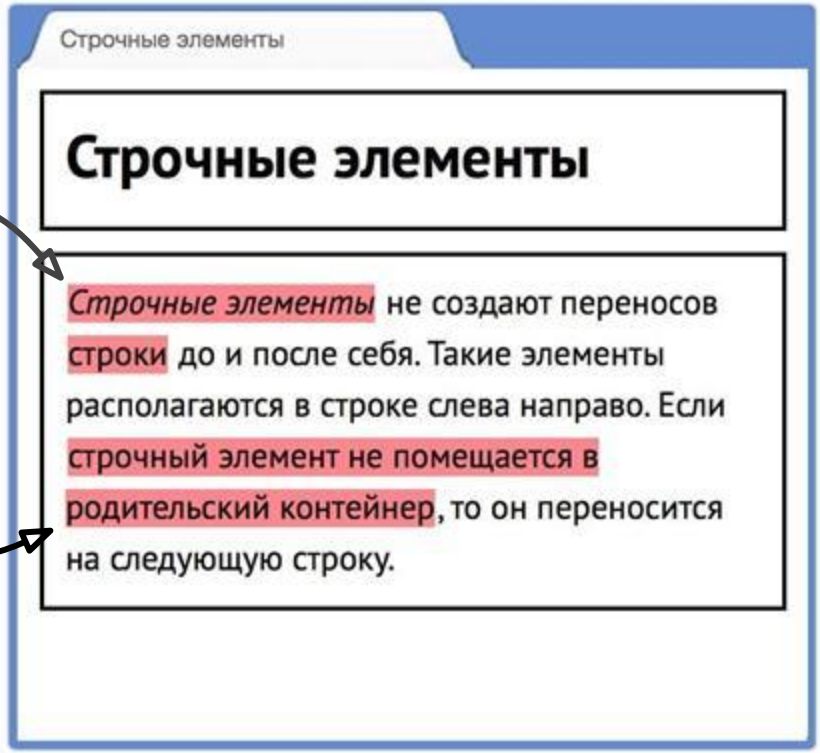
<a><b><i><time>

/* и другие теги для выделения небольших текстовых фраз */

Особенности строчных

1. Нет переносов строки до и после — можно располагать в одной строке.
2. Ширина и высота зависят только от содержания, задать размеры с помощью CSS нельзя.
3. Воспринимают только горизонтальные отступы.
4. Ведут себя как текст.

```html
<!DOCTYPE html>
<html>
    <head></head>
    <body>
        <h1>Строчные элементы</h1>
        <p>
            <em>Строчные элементы</em> не создают переносов <span>строки</span> до и после себя. Такие элементы располагаются в строке слева направо.
            Если <span>строчный элемент не помещается в родительский контейнер</span>, то он переносится на следующую строку.</p>
    </body>
</html>
```



Свойства блочной модели, 1 часть

Влияют на размер элемента:

- width
- height
- margin
- padding
- border

width /* ширина содержимого */

```
width: 500px;
width: 50%;
width: auto;    /* по умолчанию */
```

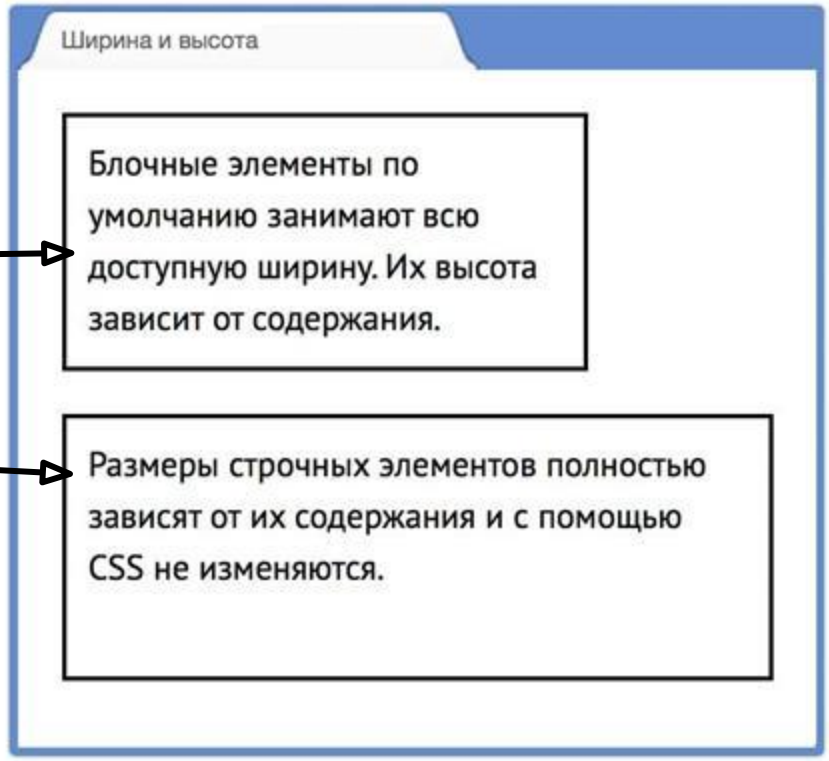
НИИТ

height /* высота содержимого */

```
height: 500px;
height: auto;  /* по умолчанию */
```

Пример width и height

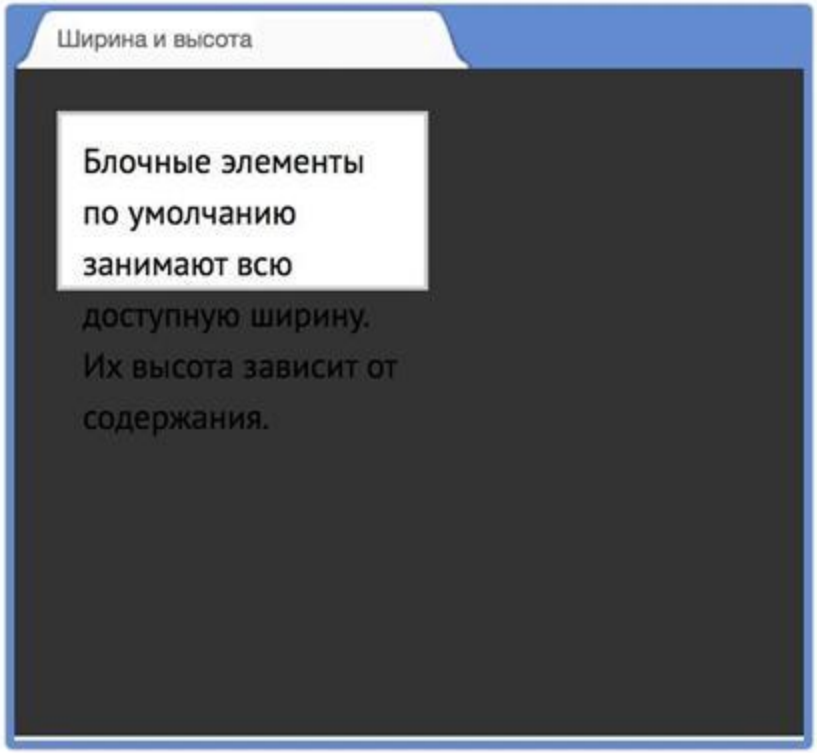
```css
div{
    margin:20px;
    padding:10px;
}

.example1{
    width: 220px;
}

.example2{
    height: 100px;
}
```



НИИТ

Пример width одновременно с height

```css
body {
    background: #333;
}

.example1{
    width: 150px;
    height: 60px;
    margin:20px;
    padding:10px;
    background: white;
    border-color: #ccc;
}
```



Советы

1. Старайтесь не использовать одновременно **width** и **height**, если это не декоративный элемент с фиксированными размерами.
2. Старайтесь не задавать фиксированную высоту.
3. Если всё-таки нужна высота, то лучше использовать **min-height**.

Дополнительные свойства

Позволяют управлять размерами гибче:

- `min-width`
- `min-height`
- `max-width`
- `max-height`

padding /* внутренние отступы */

```
padding: 10px;
padding: 5%;
padding: 2em;
```

padding с разных сторон

Либо значения через пробелы

```
padding: 10px; /* одинаково со всех сторон */
padding: 10px 20px; /* сверху+снизу, справа+слева */
padding: 10px 20px 30px; /* сверху, справа+слева, снизу */
padding: 10px 20px 30px 40px; /* сверху, справа, снизу, слева */
```

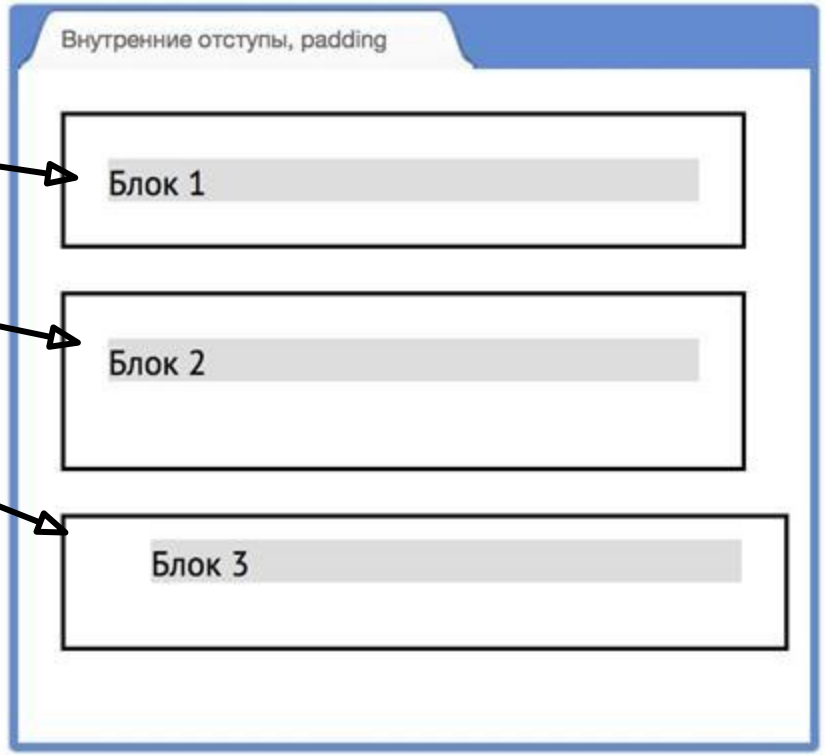
Либо отдельные свойства и их комбинации

```
padding-left: 10px;
padding-right: 20px;
padding-top: 30px;
padding-bottom: 40px;
```

НИИТ

Примеры padding

```css
.b1{
    padding: 20px;
}
.b2{
    padding: 20px;
    padding-bottom: 40px;
}
.b3{
    padding: 10px 20px 30px 40px;
}
```



margin /* внешние отступы */

```
margin: 10px;
margin: 5%;
margin: auto;  /* особый эффект, об этом позже */
margin: 2em;
```

НИИТ

margin с разных сторон

Либо значения через пробелы

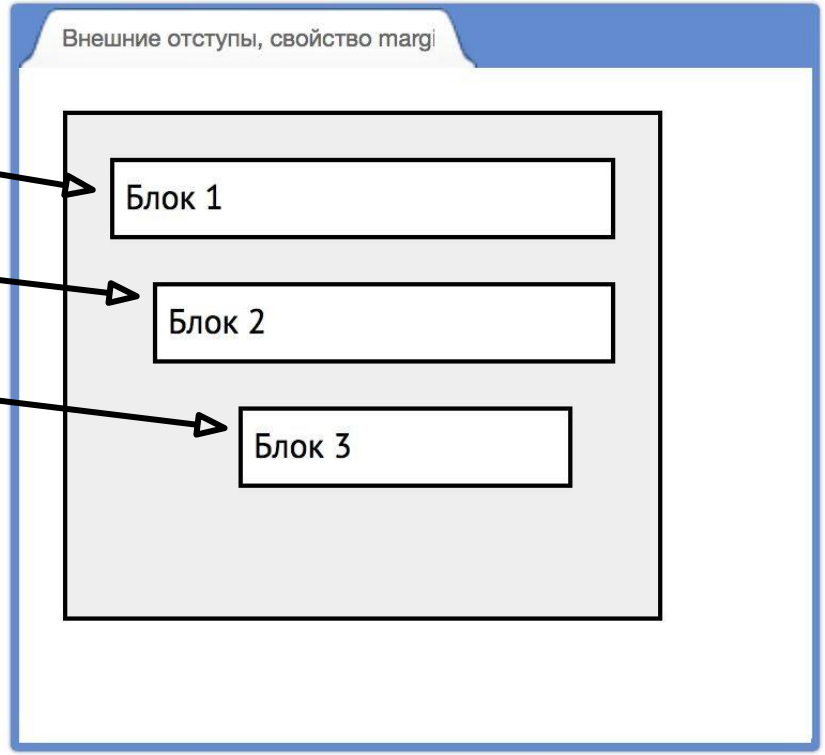
```
margin: 10px; /* одинаково со всех сторон */
margin: 10px 20px; /* сверху+снизу, справа+слева */
margin: 10px 20px 30px; /* сверху, справа+слева, снизу */
margin: 10px 20px 30px 40px; /* сверху, справа, снизу, слева */
```

Либо отдельные свойства и их комбинации

```
margin-left: 10px;
margin-right: 20px;
margin-top: 30px;
margin-bottom: 40px;
```

Примеры margin

```css
.b1{
    margin: 20px;
}
.b2{
    margin: 20px;
    margin-left: 40px;
}
.b3{
    margin: 20px 40px 60px 80px;
}
```



border /* рамки */

Состоит из трёх компонентов:

1. Ширина рамки.
2. Стиль рамки.
3. Цвет рамки.

Пример:

```
border: 1px solid #ff0000;
```

НИИТ

Компоненты и стороны border

```
/* Компоненты: */
border-width
border-style
border-color

/* Стороны: */
border-right
border-left
border-top
border-bottom
```

```
/* Комбинируем: */
border-right-width
border-right-style
border-right-color
border-left-width
border-left-style
border-left-color

/* и так далее */
```

НИИТ

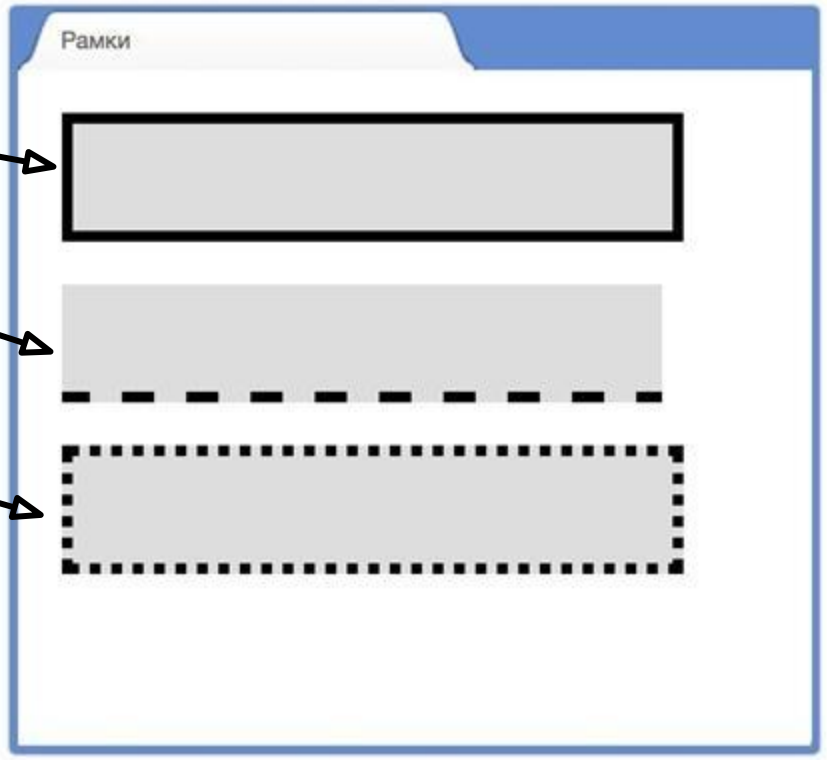
Примеры border

```
div { /* белая точечная рамка шириной 5px */
  border-width: 5px;
  border-style: dotted;
  border-color: #ffffff;
}

div { /* жёлтая пунктирная рамка снизу шириной 5px
         ниже переопределяем её цвет на оранжевый */
  border-bottom: 5px dashed yellow;
  border-bottom-color: orange;
}
```

НИИТ

Примеры border

```css
.b1 {
    border: 5px solid black;
}
.b2 {
    border-bottom: 5px dashed black;
}
.b3 {
    border-width: 5px;
    border-style: dotted;
    border-color: black;
}
```



Блочная модель и строчные

- Не реагируют на **width** и **height**.
- Воспринимают только горизонтальные **margin** и **padding**.
- Рамки **border** отображаются со всех сторон, но увеличивают размер элемента только в горизонтальном направлении.
- То есть рамки могут «залезть» на соседние строки.

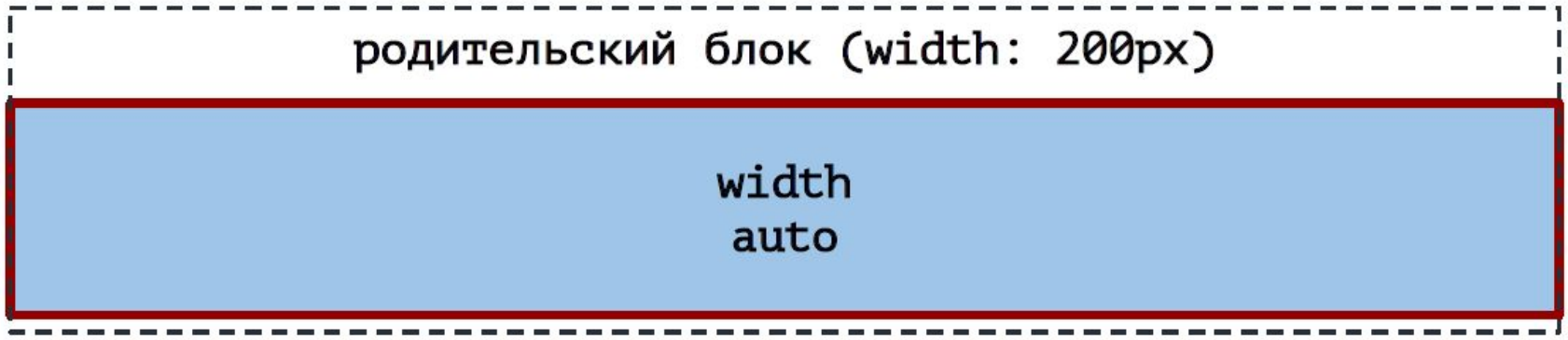
Блочная модель и строчные

Если вдруг для элемента не работают отступы или размеры, то проверьте его тип.

«А не строчный ли это элемент?»

Расчёт полного размера элемента

Если ширина **не** задана, общая ширина равна доступному месту в родителе.

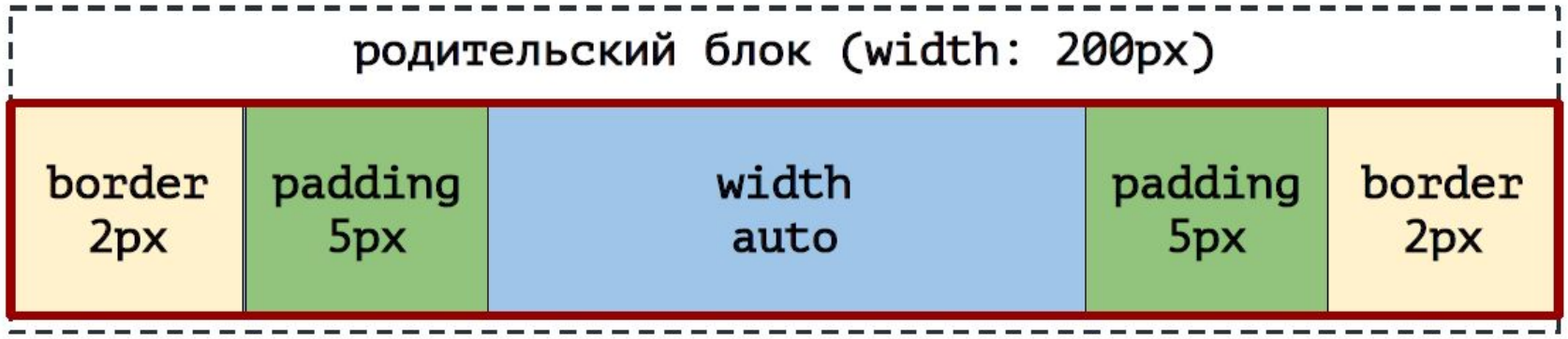


Расчёт полного размера элемента

При добавлении внутренних отступов ужимается содержимое.

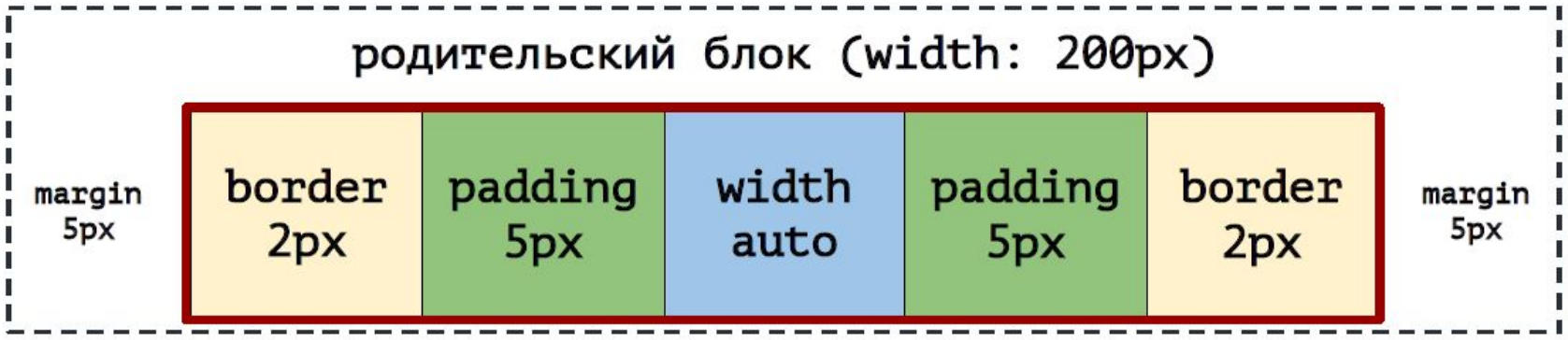
Расчёт полного размера элемента

Рамки также приводят к уменьшению области содержимого.



Расчёт полного размера элемента

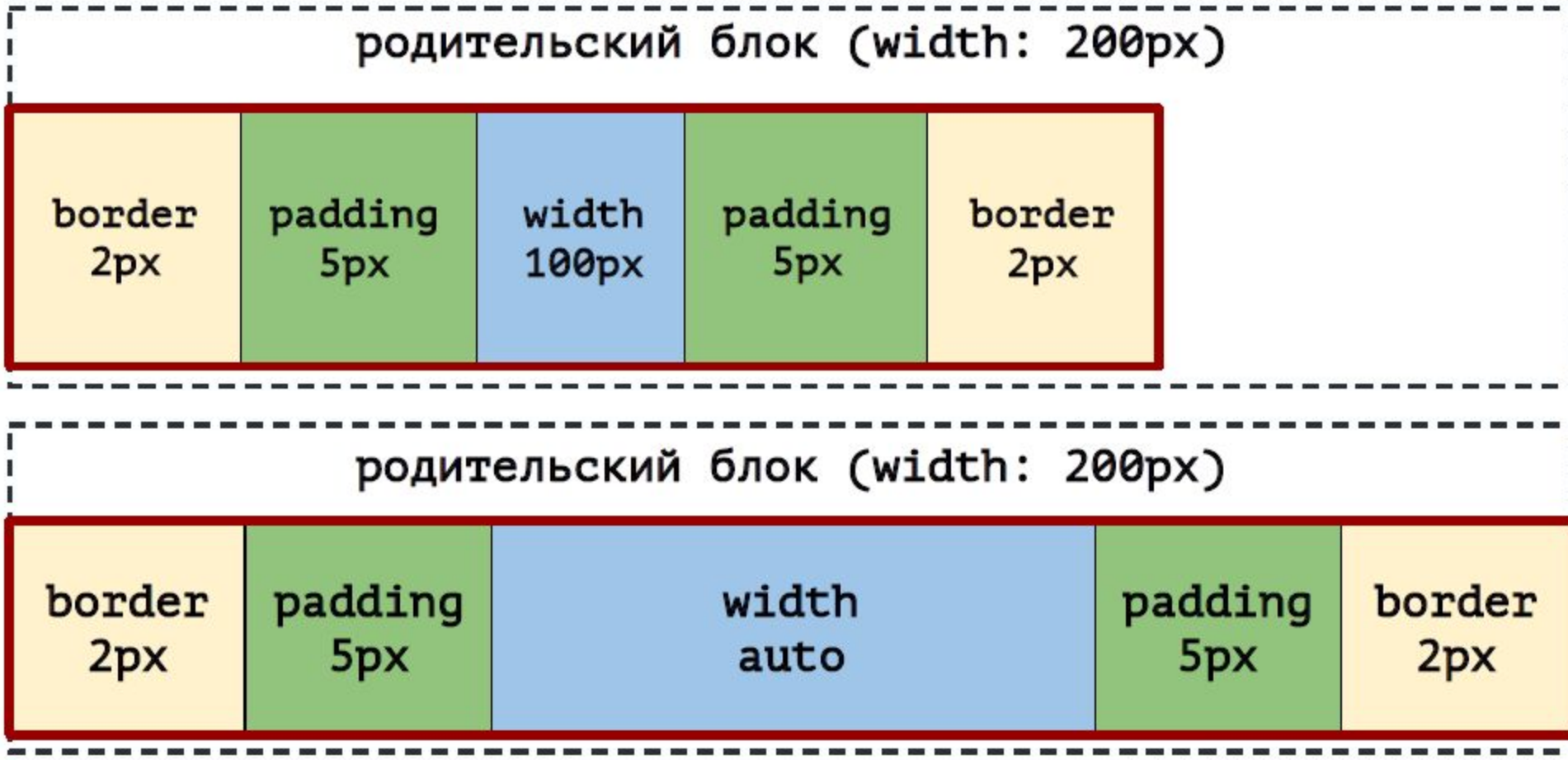
Внешние отступы изменяют общую ширину блока.



Расчёт полного размера элемента

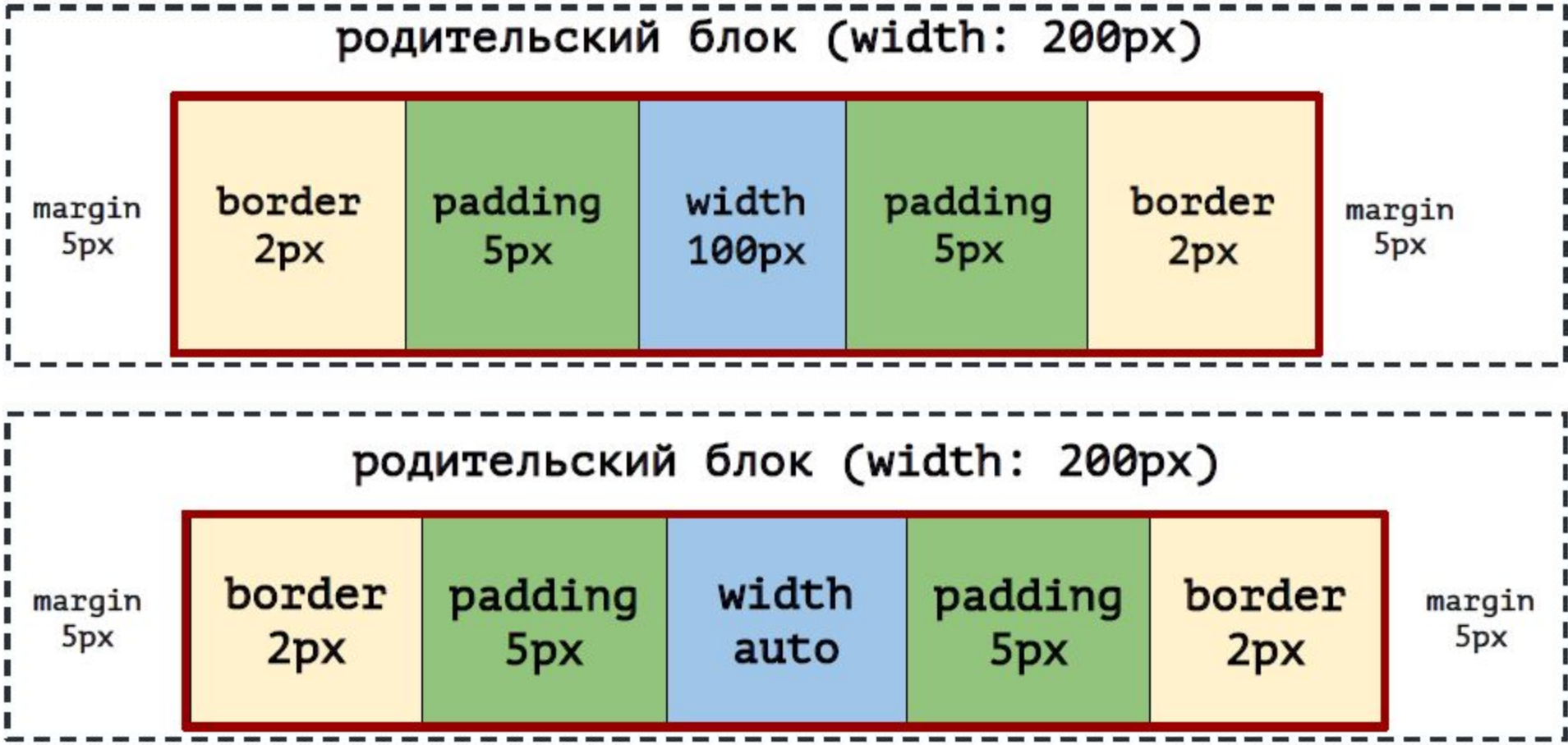
Если ширина задана, то общая ширина равна:

ширина контента + внутренние отступы + ширина рамок



Расчёт полного размера элемента

Внешние отступы не влияют на общую ширину блока с заданной шириной.



Расчёт полного размера элемента

1. При **height: auto** общая высота равна:

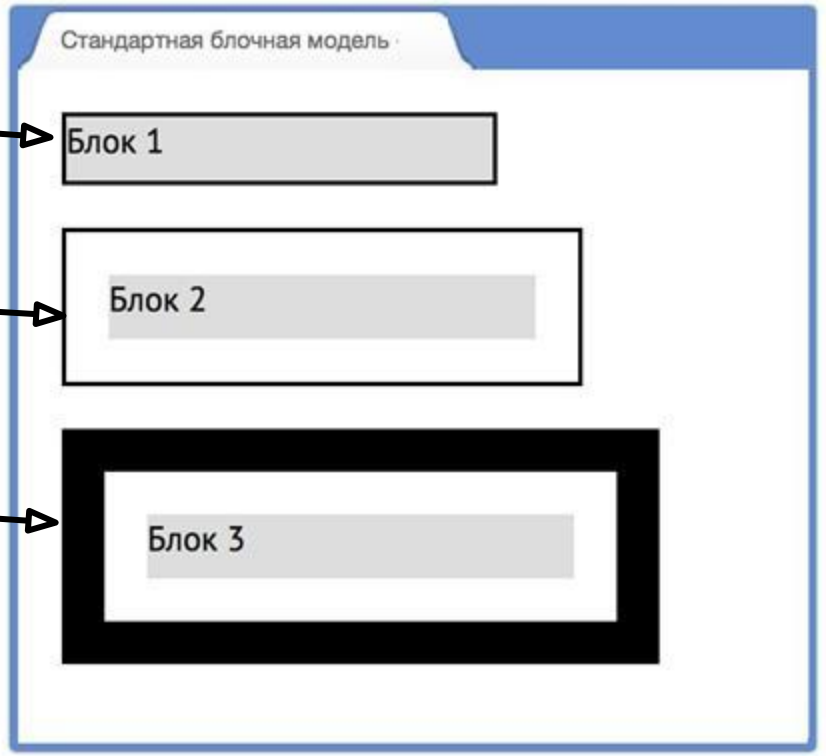
высота содержимого + внутренний отступ + рамки

2. Если не **height: auto**, общая высота равна:

высота элемента + внутренний отступ + рамки

Расчёт полного размера элемента

```css
.block{
    width: 200px;
    height: 30px;
    margin: 20px;
}
.block2{
    width: 200px;
    height: 30px;
    padding: 20px;
}
.block3{
    width: 200px;
    height: 30px;
    padding: 20px;
    border-width: 20px;
}
```



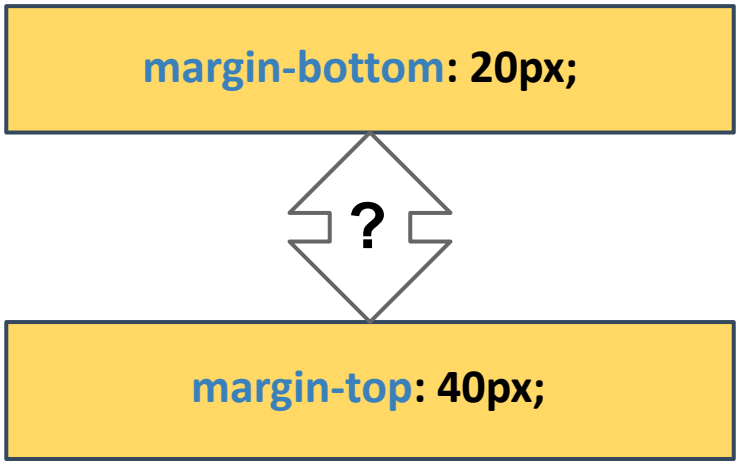
Расчёт полного размера элемента

- Общая ширина и высота элемента, это не то же самое, что свойства **width** и **height**, а обычно больше.
- **Но!** Это поведение можно изменить.

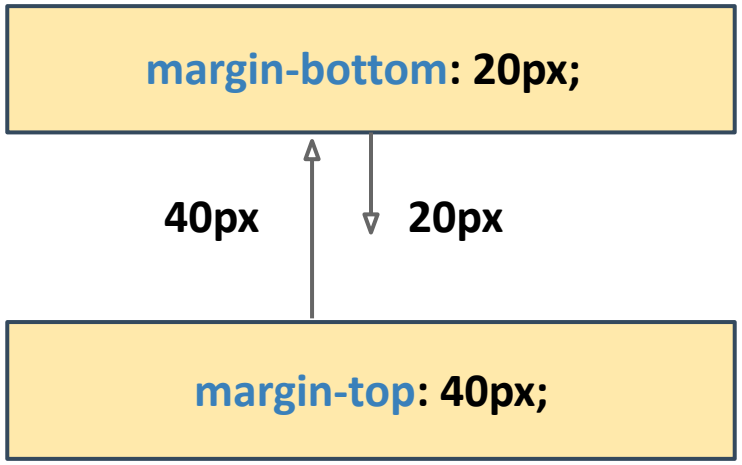
Тонкости блочной модели

- Схлопывание внешних отступов (**margin**).
- Выпадание внешних отступов (**margin**).
- Как расположить элемент по центру.
- Ширина по умолчанию и 100%.
- **box-sizing**.

Схлопывание внешних отступов

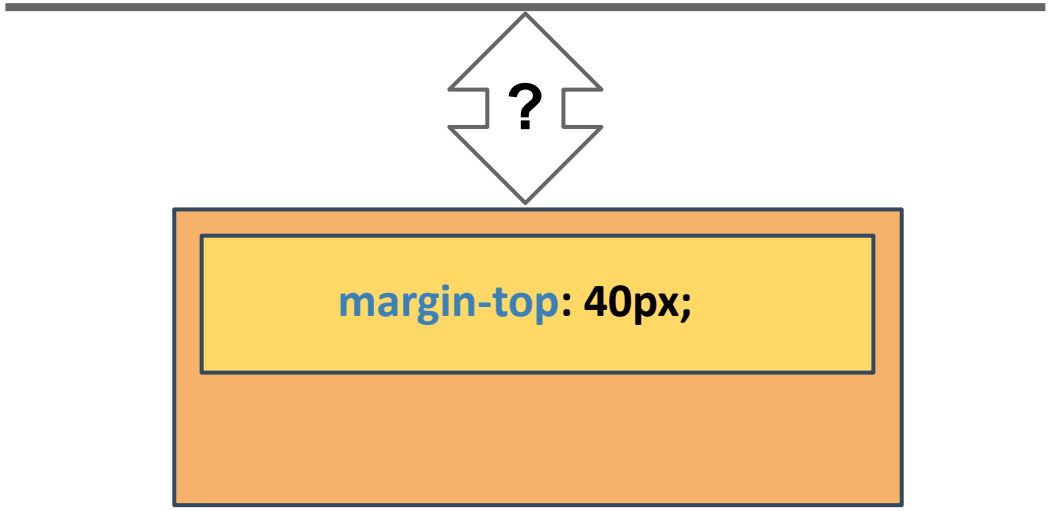


Схлопывание внешних отступов

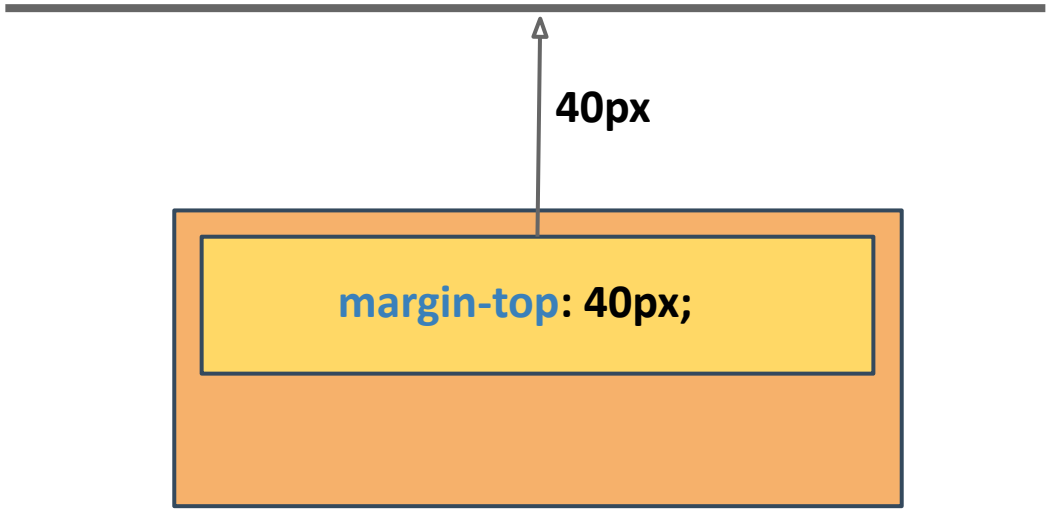


В вертикальном направлении внешние отступы (**margin**) не складываются, а выбирается максимальный из доступных.

Выпадание внешних отступов



Выпадание внешних отступов



В вертикальном направлении внешний отступ (**margin**) вложенного блока может выпадать из родительского и отталкивать оба блока.

Как бороться с выпаданием?

Родительскому блоку можно задать одно из следующих свойств:

- `overflow: hidden; /* использовать осторожно */`
- `padding-top: 1px; /* или */`
  `padding-bottom: 1px;`
- `border-top: 1px solid transparent; /* или */`
  `border-bottom: 1px solid transparent;`

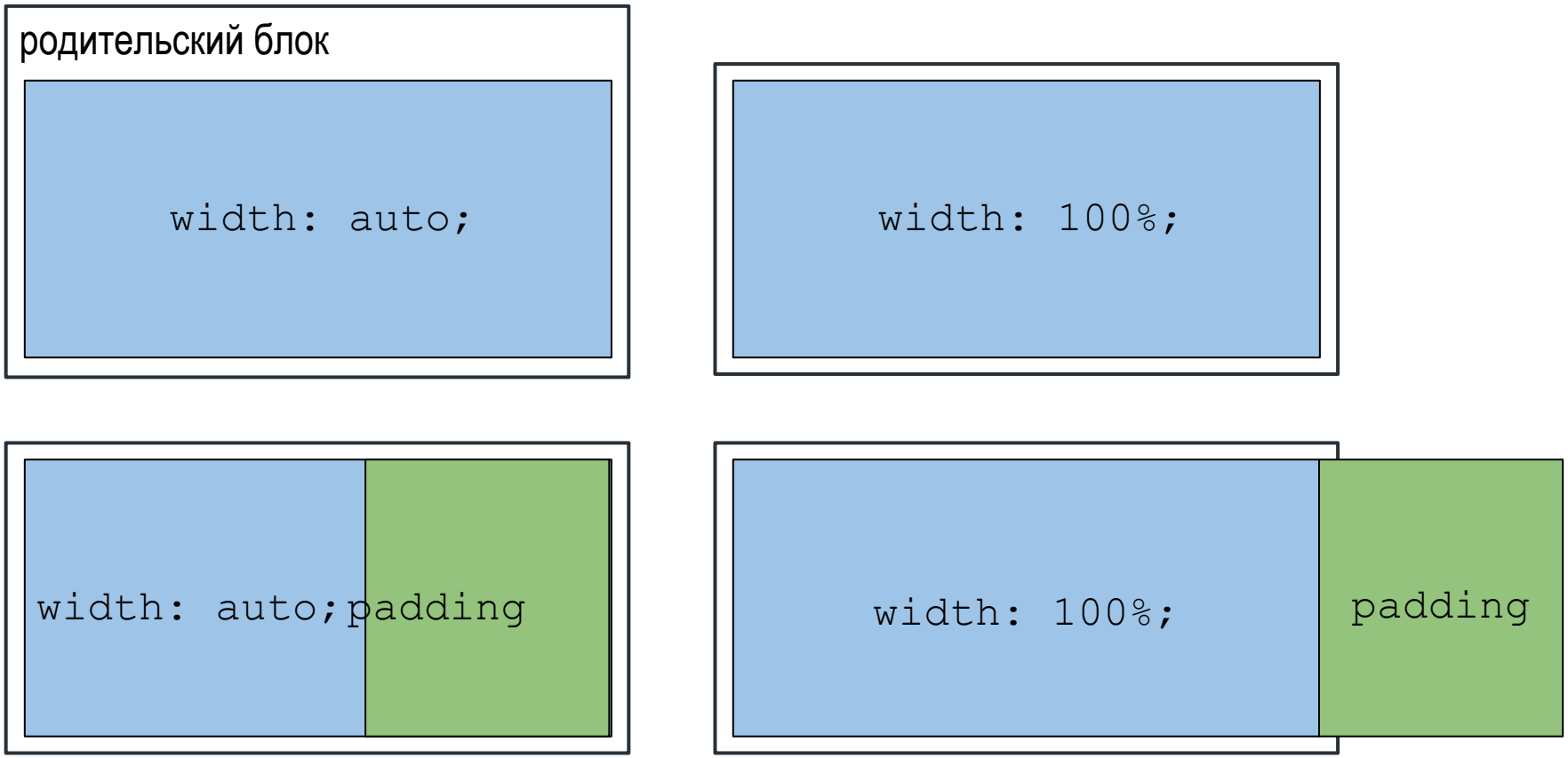
НИИТ

Как расположить элемент по центру

```
div {
   width: [меньше чем ширина родителя]px;
   margin-left: auto;
   margin-right: auto;

   /* Часто пишут так */
   margin: 0 auto;
}
```

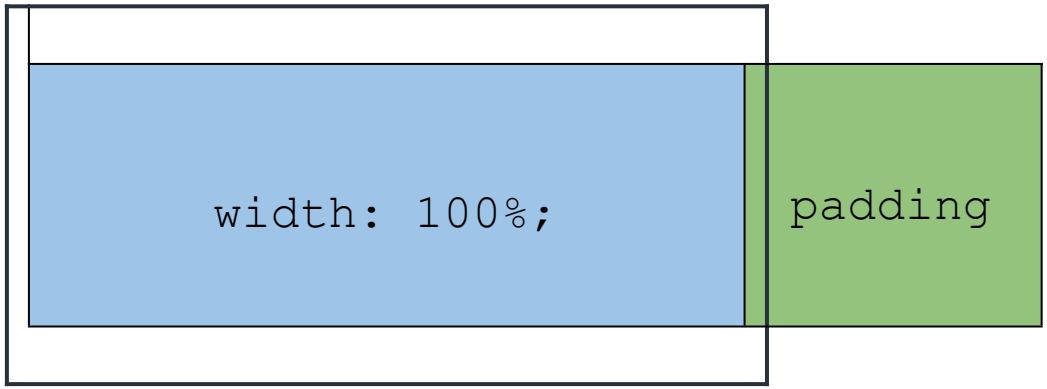
НИИТ

width: auto; и width: 100%;

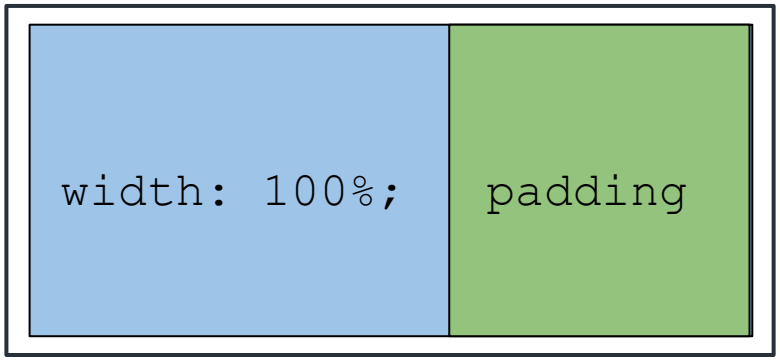


Свойство box-sizing

box-sizing: content-box;



box-sizing: border-box;



Content

The content of the box, where text and images

Padding

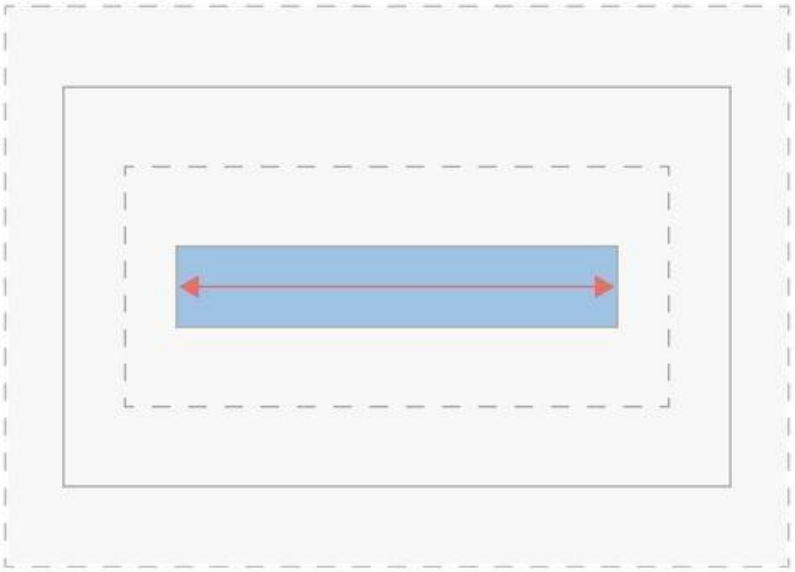
Clears an area around the content.

Border

A border that goes around the padding.

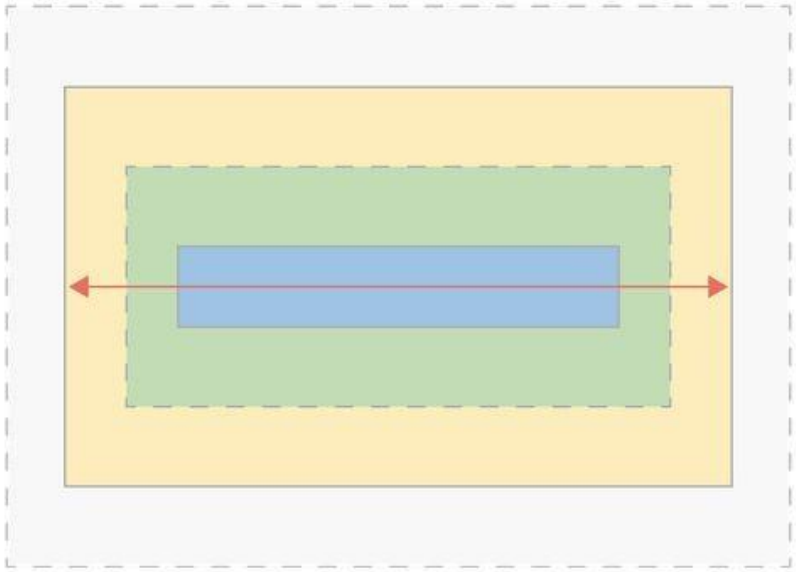
Margin

Clears an area around the border.



Without box-sizing: border-box

Margin, borders and padding are drawn outside the set width of your content.



With box-sizing: border-box

Borders and padding are drawn inside the set width of your content. The margin is drawn outside.

h

Как использовать box-sizing

Аккуратно и точечно.

Старайтесь не использовать его для разметки сеток, а применять точечно в узких местах, где его отказ не вызовет серьёзных проблем.

Например, при оформлении полей форм.

Свойства блочной модели, 2 часть

Управляют поведением в потоке:

- `display`
- `float` `/* а также clear */`

Свойство display

Самые часто используемые значения:

- `display: block`
- `display: inline`
- `display: inline-block`
- `display: none`

Да, вы можете сделать блочный элемент строчным и наоборот!

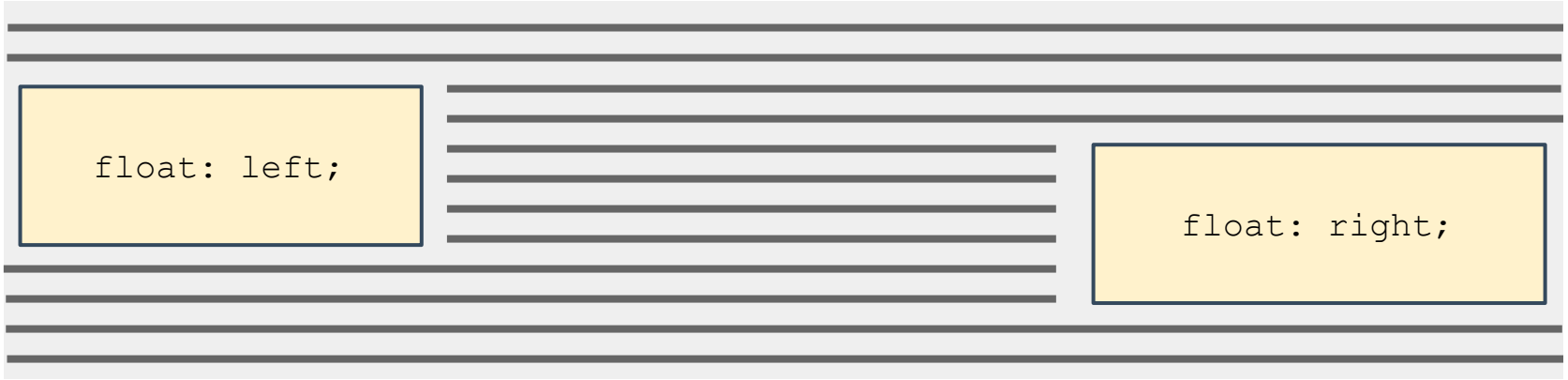
НИИТ

Приёмы построения сеток

- `float`
- `display: inline-block`
- `display: table`
- `display: flex`

Свойство float

- Задумывалось для обтекания блоков текстом.
- Возможные значения: **left, right, none**.
- Прижимает элемент к левому или правому краю родителя.

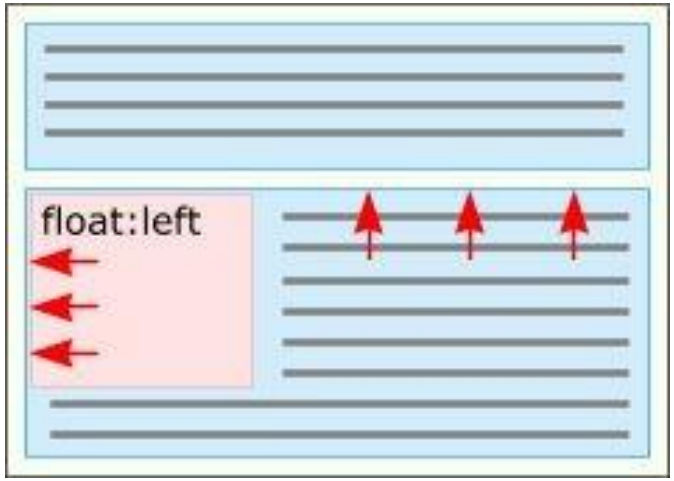


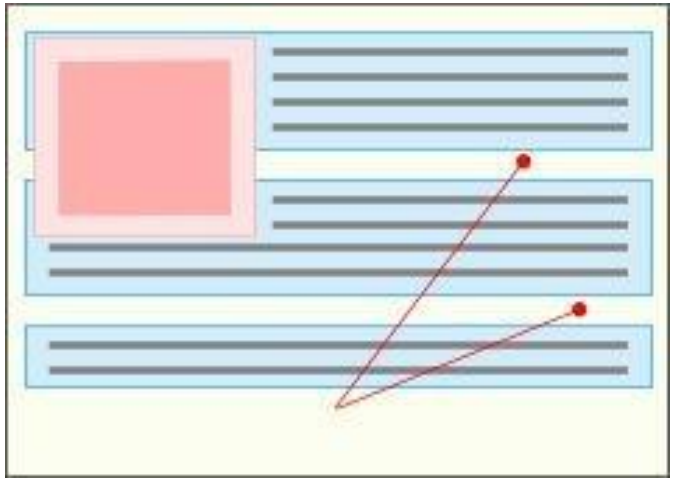
Свойство float

- Элементы со свойством float называют «плавающими».
- Плавающий элемент ужимается под контент.
- Плавающий элемент частично выпадает из потока.

Частичное выпадение из потока

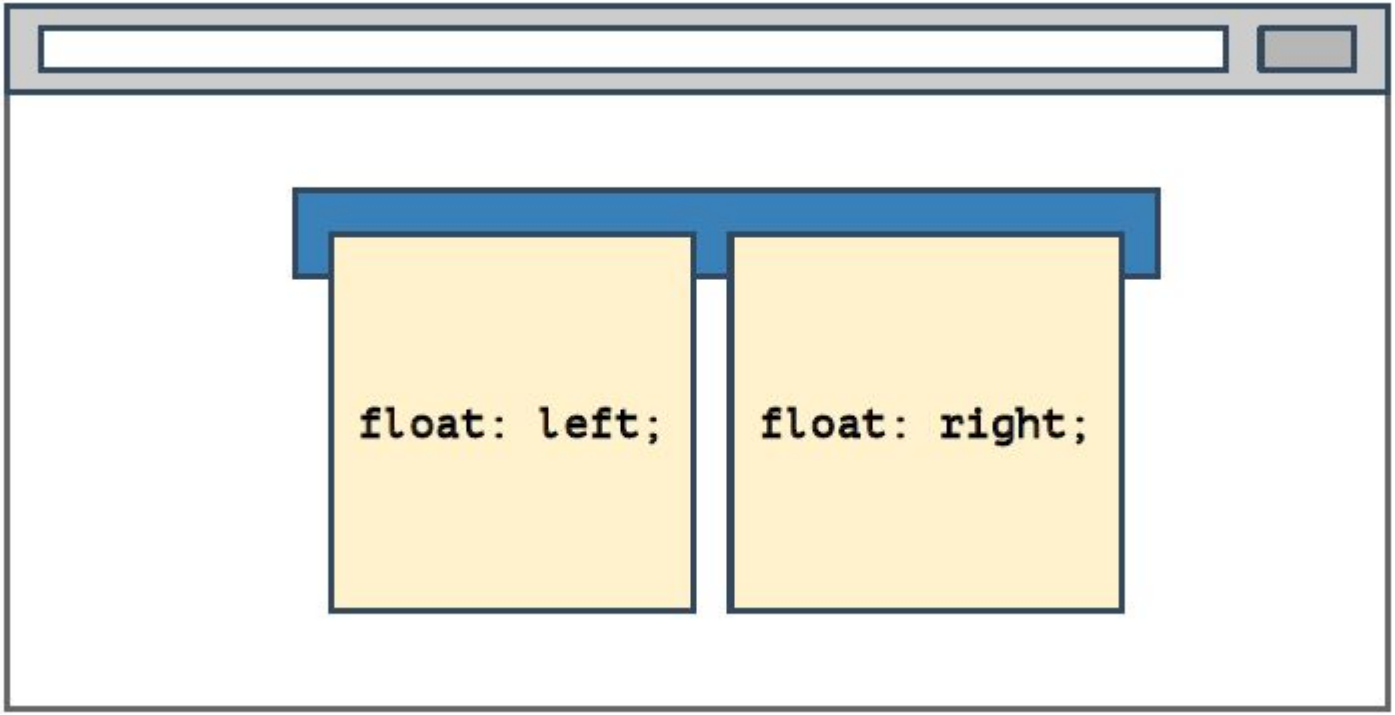
- Плавающий элемент «не виден» последующим блочным элементам.
- Последующие строчные элементы его обтекают.





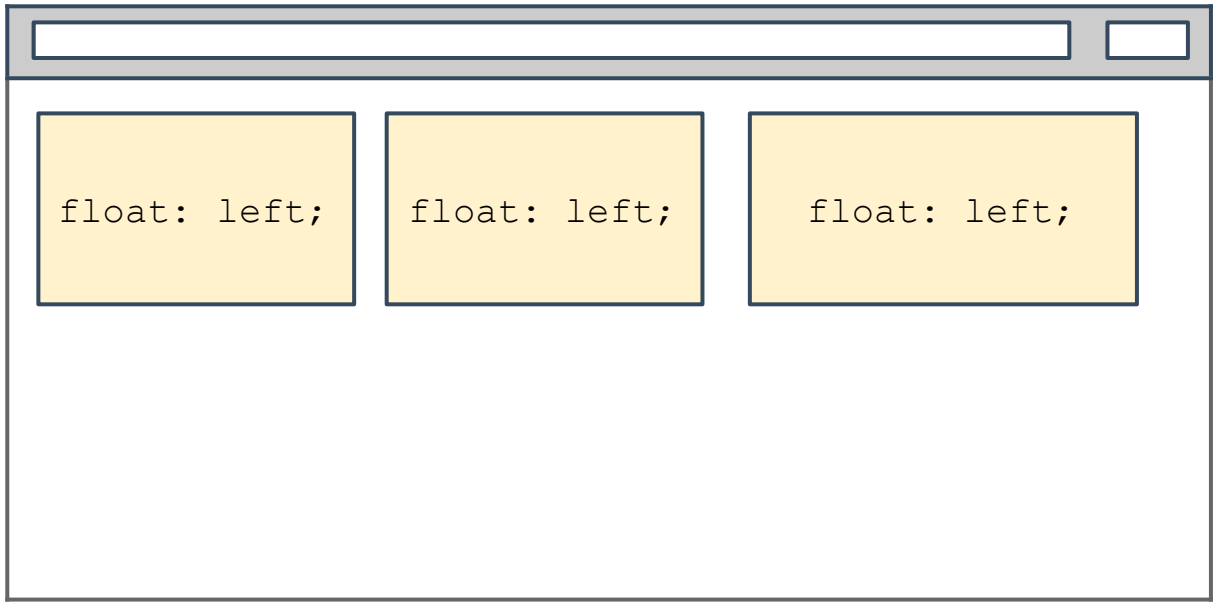
Частичное выпадение из потока

- Выпадают и из родительских блоков



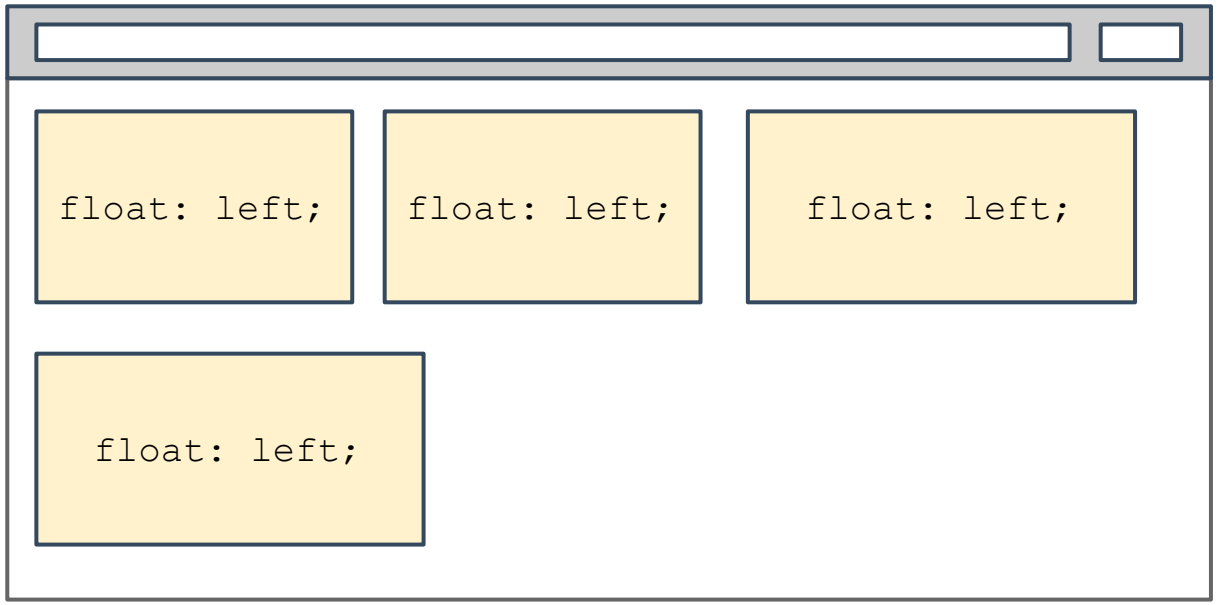
Частичное выпадение из потока

- Плавающие элементы пытаются встать друг за другом в ряд, если есть место.
- Если места не хватает — встают ниже.



Частичное выпадение из потока

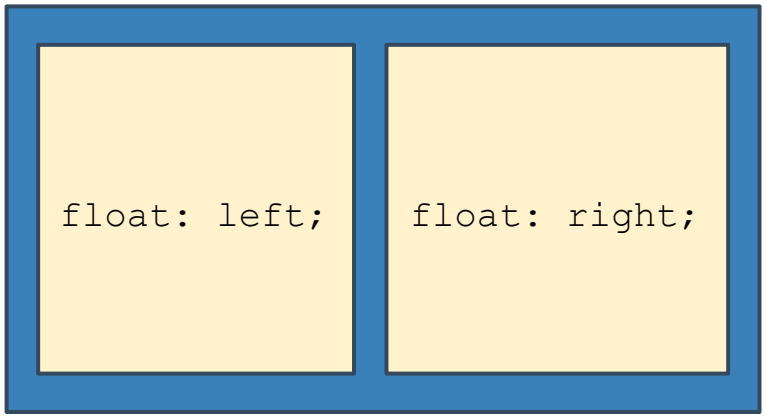
- Плавающие элементы пытаются встать друг за другом в ряд, если есть место.
- Если места не хватает — встают ниже.



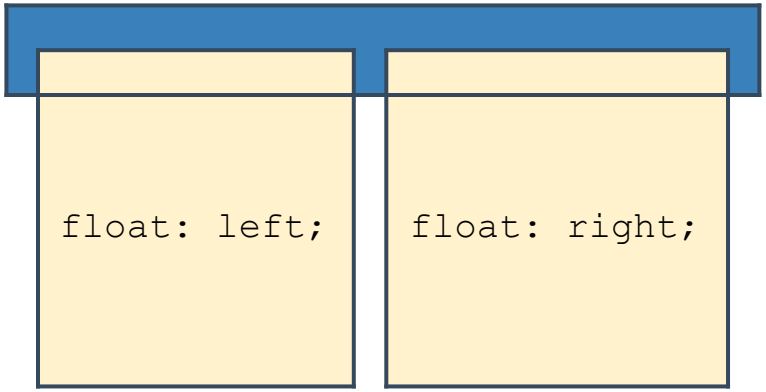
Частичное выпадение из потока

- Вывод: плавающие элементы позволяют создавать колонки.
- Проблема: выпадают из родителя.

Ожидание

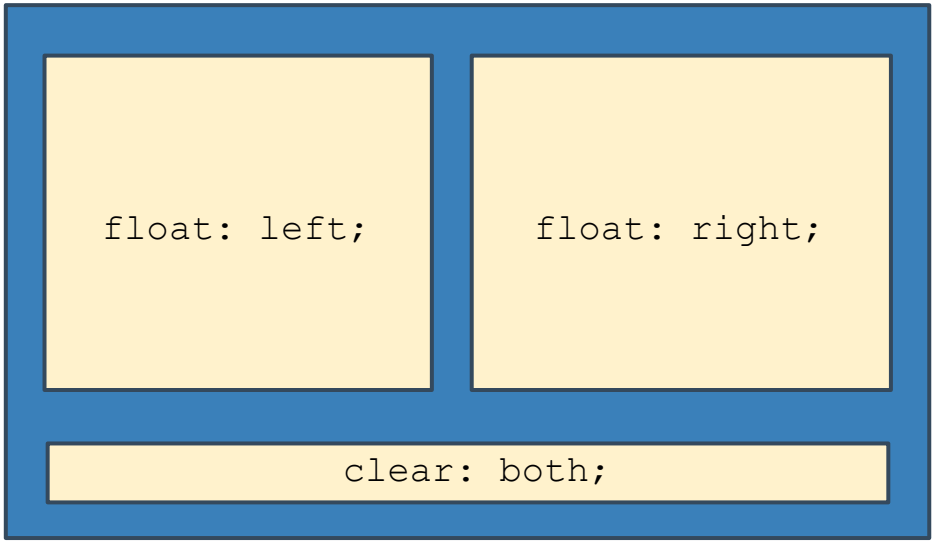


Реальность



Как предотвратить выпадение?

Приём «распорка»: использовать свойство clear: both у элемента расположенного после плавающих.



НИИТ

Как предотвратить выпадение?

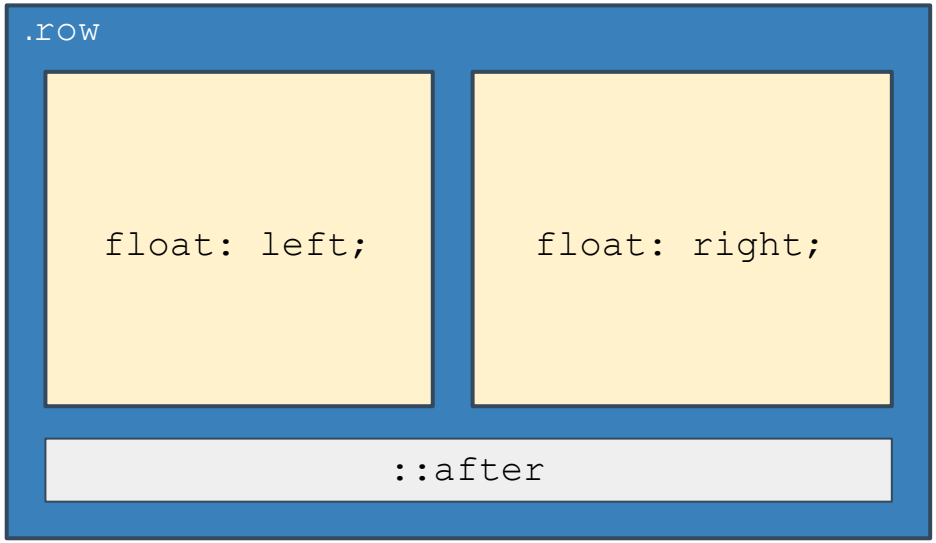
Приём «псевдораспорка»: контейнеру, содержащему флоаты, добавляются псевдоэлемент с `clear:both`.

```
.row::after {
  content: "";
  display: table;
  clear: both;
}
```

НИИТ

Построение сеток на флоатах

Рецепт: добавляем контейнеру с плавающими элементами «псевдораспорку». Следим, чтобы колонки помещались в ряд.




```
.row::after {
  content: "";
  display: table;
  clear: both;
}
```

НИИТ

display: inline-block;

Особенности:

- Снаружи как строчный, внутри как блочный.
- Воспринимает ширину, высоту, внутренние и внешние отступы.
- Ширина по умолчанию ужимается под содержание.

display: inline-block;

Особенности:

- Можно располагать на одной строке (отсутствуют принудительные переносы).
- Воспринимают «текстовые свойства», например vertical-align или text- align.
- Логичное поведение при переносе строк.

Построение сеток на inline-block

Блочно-строчные используют для вёрстки:

- декоративных элементов,
- кнопок,
- многоколоночных списков (товары в каталоге).

Особенно, если блоки в дизайне не «впритирку».