CSS

Дополнительный материал

УНИВЕРСАЛЬНЫЙ СЕЛЕКТОР

```
*{
    background-color: red;
}
```

СТИЛИЗАЦИЯ ГРУППЫ СЕЛЕКТОРОВ

Например, мы хотим применить ко всем заголовкам подчеркивание. В этом случае мы можем перечислить селекторы всех элементов через запятую:

```
h1, #header, .redBlock{

    color: red;
}
```

СЕЛЕКТОРЫ ПОТОМКОВ

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Селекторы CSS</title>
        <style>
            li .redLink{
                color: red;
            }
        </style>
    </head>
    <body>
        <ul>
            <li>Самсунг: <a class="redLink">Galaxy S7 Edge</a></li>
            <li>Apple: <a>iPhome SE</a></li>
            <li>LG: <a class="redLink">LG G5</a></li>
            <li>Microsoft: <a>Lumia 650</a></li>
        </ul>
    </body>
</html>
```

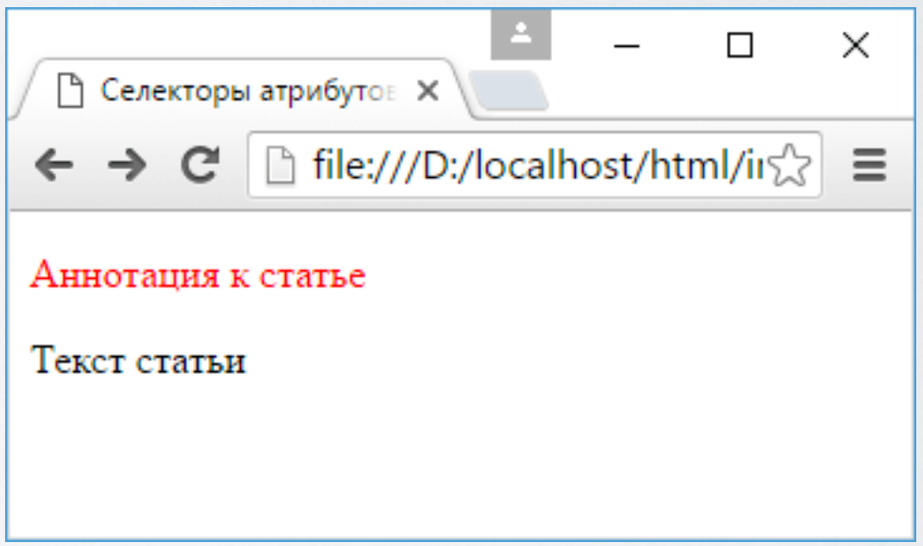
Это другой стиль:

```
li.redLink{
    color: red;
}
```

СЕЛЕКТОРЫ ДОЧЕРНИХ ЭЛЕМЕНТОВ

В блоке с классом article есть два параграфа. Селектор .article > p выбирает только те параграфы, который находятся непосредственно в блоке article:

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Селекторы атрибутов в CSS3</title>
        <style>
            .article > p{

                color: red;
            }
        </style>
    </head>
    <body>
        <div class="article">
            <p>Аннотация к статье</p>
            <div class="content">
                <p>Текст статьи</p>
            </div>
        </div>
    </body>
</html>
```



ПСЕВДОКЛАССЫ

:root: позволяет выбрать корневой элемент веб-страницы, наверное наименее полезный селектор, так как на правильной веб-странице корневым элементом практически всегда является элемент <html>

:link: применяется к ссылкам и представляет ссылку в обычном состоянии, по которой еще не совершен переход

:visited: применяется к ссылкам и представляет ссылку, по которой пользователь уже переходил

:active: применяется к ссылкам и представляет ссылку в тот момент, когда пользователь осуществляет по ней переход

:hover: представляет элемент, на который пользователь навел указатель мыши. Применяется преимущественно к ссылкам, однако может также применяться и к другим элементам, например, к параграфам

:focus: представляет элемент, который получает фокус, то есть когда пользователь нажимает клавишу табуляции или нажимает кнопкой мыши на поле ввода (например, текстовое поле)

:not: позволяет исключить элементы из списка элементов, к которым применяется стиль

:lang: стилизует элементы на основании значения атрибута lang

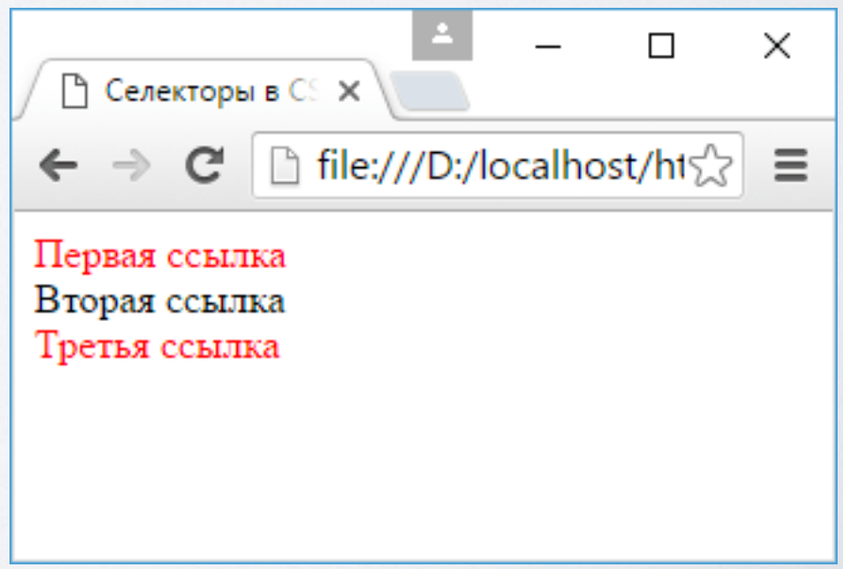
:empty: выбирает элементы, которые не имеют вложенных элементов, то есть являются пустыми

Селектор **:not()** позволяет выбрать все элементы кроме определенных, то есть исключить некоторые элементы из выбора.

ПСЕВДОКЛАССЫ

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Псевдоклассы в CSS3</title>
        <style>
            a:link    {color:blue; text-decoration:none}
            a:visited {color:pink; text-decoration:none}
            a:hover   {color:red; text-decoration:underline}
            a:active  {color:yellow; text-decoration:underline}
            input:hover {border:2px solid red;}
            :lang(ru) {color: red;}
        </style>
    </head>
    <body>
        <a href="index.html">Учебник по CSS3</a>
        <input type="text" />
        <p lang="ru-RU">Я изучаю CSS3</p>
    </body>
</html>
```

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Селекторы в CSS3</title>
        <style>
            a:not(.blueLink) { color: red;
}
        </style>
    </head>
    <body>
        <a>Первая ссылка</a><br/>
        <a class="blueLink">Вторая
ссылка</a><br/>
        <a>Третья ссылка</a>
    </body>
</html>
```



ПСЕВДОКЛАССЫ ДОЧЕРНИХ ЭЛЕМЕНТОВ

:first-child: представляет элемент, который является первым дочерним элементом

:last-child: представляет элемент, который является последним дочерним элементом

:only-child: представляет элемент, который является единственным дочерним элементом в каком-нибудь контейнере

:only-of-type: выбирает элемент, который является единственным элементом определенного типа (тега) в каком-нибудь контейнере

:nth-child(n): представляет дочерний элемент, который имеет определенный номер n, например, второй дочерний элемент

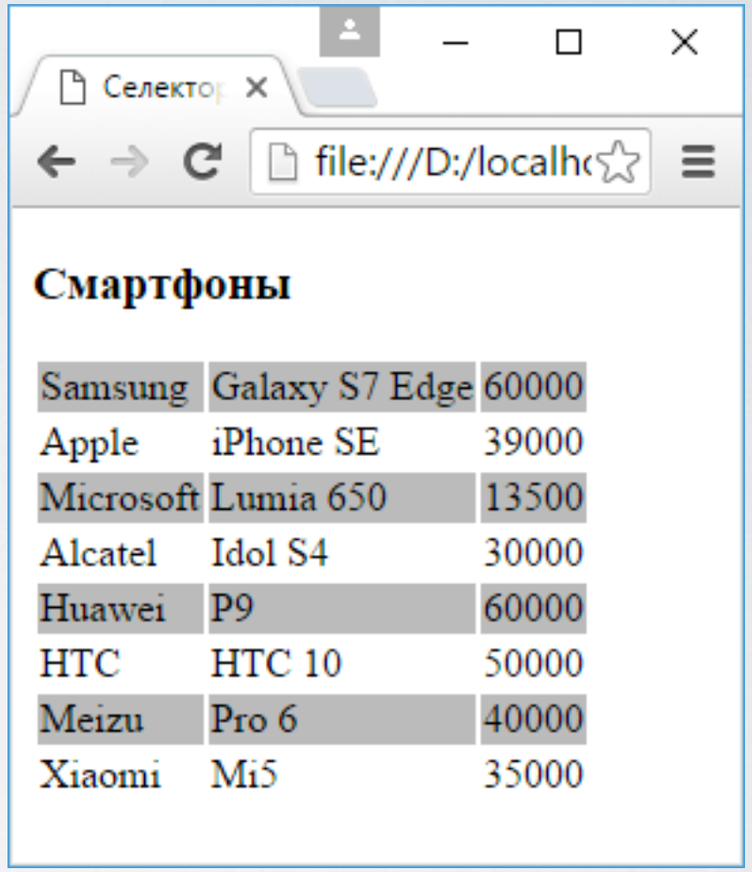
:nth-last-child(n): представляет дочерний элемент, который имеет определенный номер n, начиная с конца

:nth-of-type(n): выбирает дочерний элемент определенного типа, который имеет определенный номер

:nth-last-of-type(n): выбирает дочерний элемент определенного типа, который имеет определенный номер, начиная с конца

ПСЕВДОКЛАССЫ ДОЧЕРНИХ ЭЛЕМЕНТОВ

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Селекторы в CSS3</title>
        <style>
            tr:nth-child(odd) { background-color: #bbb; }
            tr:nth-child(even) { background-color: #fff; }
            tr:nth-child(3) { background-color: #bbb; }
            tr:nth-child(2n+1) { background-color: #bbb; }
            tr:nth-child(3n+2) { background-color: #bbb; }
        </style>
    </head>
    <body>
        <h3>Смартфоны</h3>
        <table>
            <tr><td>Samsung</td><td>Galaxy S7 Edge</td><td>60000</td></tr>
            <tr><td>Apple</td><td>iPhone SE</td><td>39000</td></tr>
            <tr><td>Microsoft</td><td>Lumia 650</td><td>13500</td></tr>
            <tr><td>Alcatel</td><td>Idol S4</td><td>30000</td></tr>
            <tr><td>Huawei</td><td>P9</td><td>60000</td></tr>
            <tr><td>HTC</td><td>HTC 10</td><td>50000</td></tr>
            <tr><td>Meizu</td><td>Pro 6</td><td>40000</td></tr>
            <tr><td>Xiaomi</td><td>Mi5</td><td>35000</td></tr>
        </table>
    </body>
</html>
```



ПСЕВДОКЛАССЫ ФОРМ

:enabled: выбирает элемент, если он доступен для выбора (то есть у него не установлен атрибут disabled)

:disabled: выбирает элемент, если он не доступен для выбора (то есть у него установлен атрибут disabled)

:checked: выбирает элемент, если у него установлен атрибут checked (для флажков и радиокнопок)

:default: выбирает элементы по умолчанию

:valid: выбирает элемент, если его значение проходит валидацию HTML5

:invalid: выбирает элемент, если его значение не проходит валидацию

:in-range: выбирает элемент, если его значение находится в определенном диапазоне (для элементов типа ползунка)

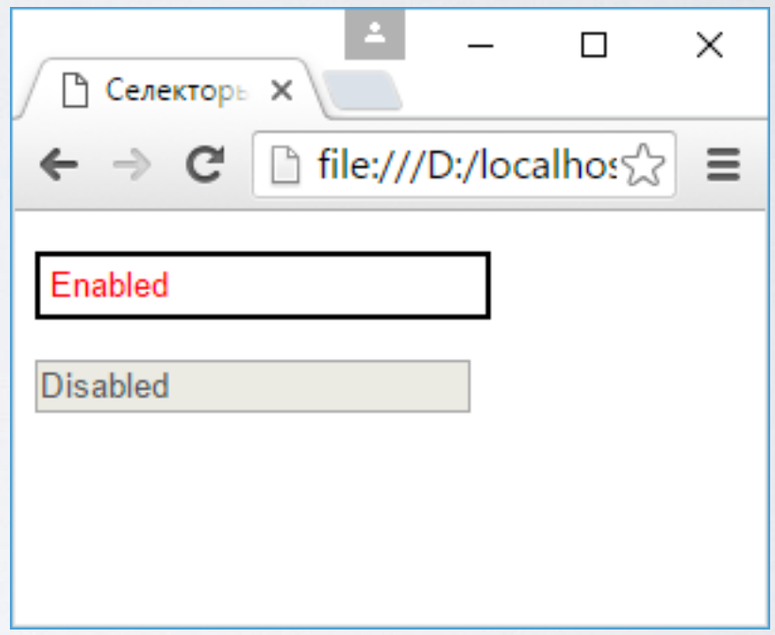
:out-of-range: выбирает элемент, если его значение не находится в определенном диапазоне

:required: выбирает элемент, если у него установлен атрибут required

:optional: выбирает элемент, если у него не установлен атрибут required

ПСЕВДОКЛАССЫ ФОРМ

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Селекторы в CSS3</title>
        <style>
            :enabled {
                border: 2px black solid;
                color: red;
            }
        </style>
    </head>
    <body>
        <p><input type="text" value="Enabled" /></
p>
        <p><input type="text" disabled
value="Disabled" /></p>
    </body>
</html>
```



ПСЕВДОЭЛЕМЕНТЫ

::first-letter: позволяет выбрать первую букву из текста

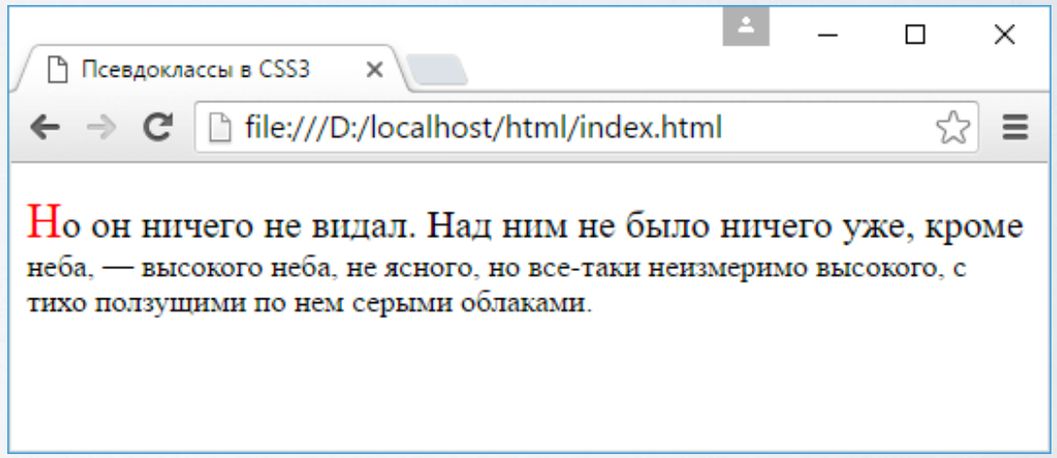
::first-line: стилизует первую строку текста

::before: добавляет сообщение до определенного элемента

::after: добавляет сообщение после определенного элемента

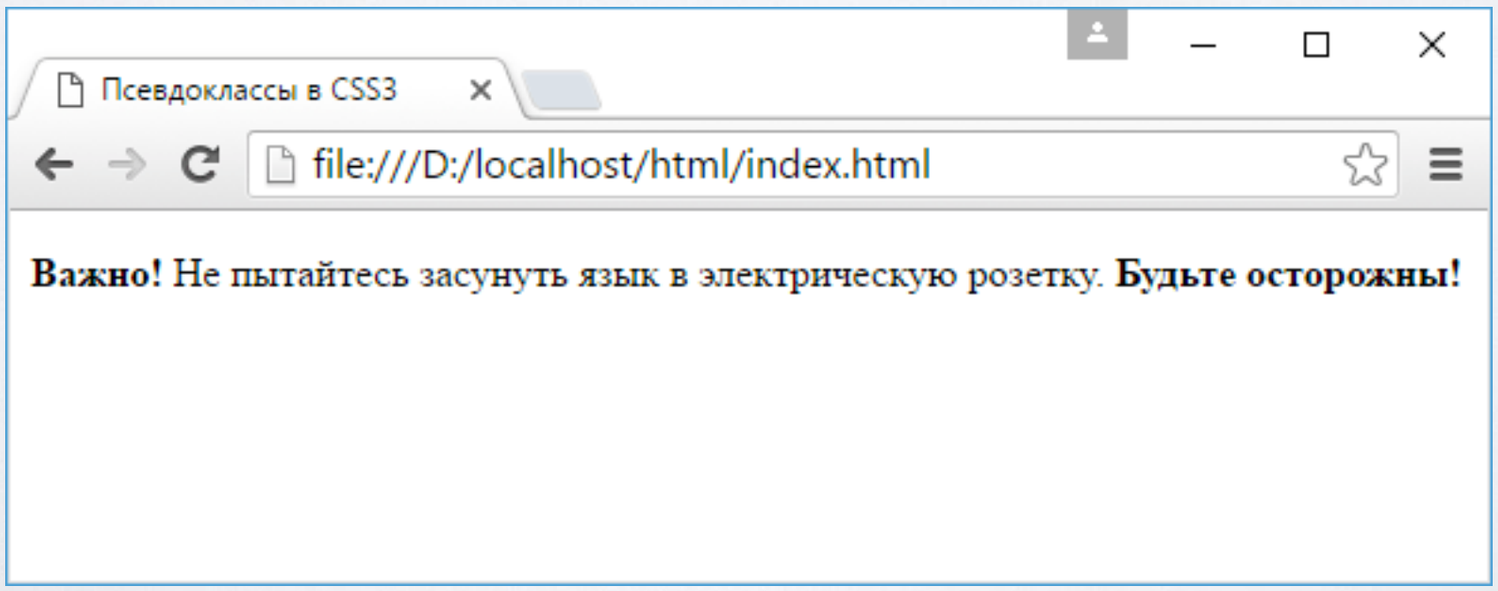
::selection: выбирает выбранные пользователем элементы

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Псевдоклассы в CSS3</title>
        <style>
            ::first-letter { color:red; font-size: 25px; }
            ::first-line { font-size: 20px; }
        </style>
    </head>
    <body>
        <p>Но он ничего не видал. Над ним не было ничего уже, кроме неба.</p>
    </body>
</html>
```



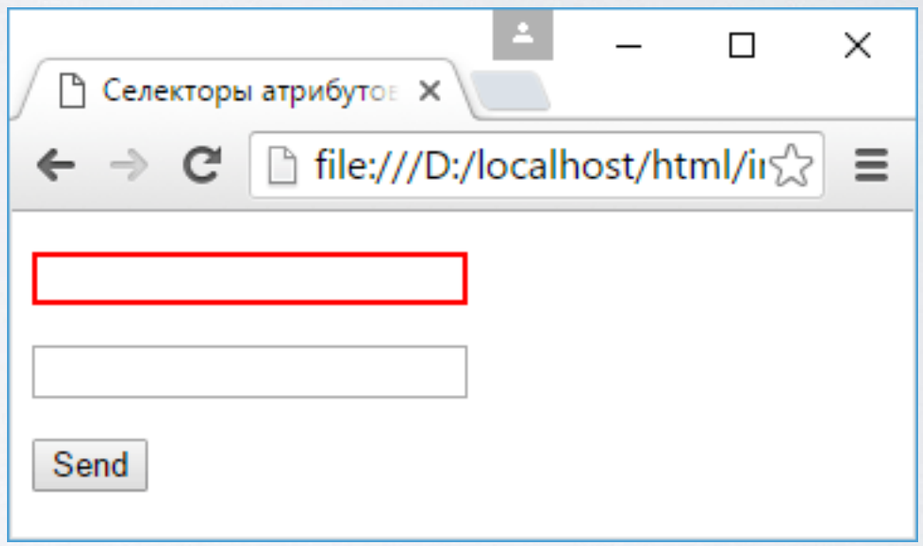
ПСЕВДОЭЛЕМЕНТЫ

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Псевдоклассы в CSS3</title>
        <style>
            .warning::before{ content: "Важно! "; font-weight: bold; }
            .warning::after { content: " Будьте осторожны!"; font-weight: bold;}

            ::selection {
                color: white;
                background-color: black;
            }

        </style>
    </head>
    <body>
        <p><span class="warning">Не пытайтесь засунуть язык в электрическую розетку.</span></p>
    </body>
</html>
```



СЕЛЕКТОРЫ АТРИБУТОВ

```
<html>
    <head>
        <meta charset="utf-8">
        <title>Селекторы атрибутов в CSS3</title>
        <style>
            input[type="text"]{
                border: 2px solid red;
            }
           .link[href="http://apple.com"]{

                color: red;
            }
            a[href^="https://"]{

                color: red;
            }
            img[src$=".jpg"]{

                width: 100px;
            }
            a[href*="microsoft"]{

                color: red;
            }


    </style>
</head>
<body>
    <p><input type="text" id="login" /></p>
    <p><input type="password" id="password" /></p>
    <input type="submit" value="Send" />
</body>
```



ЦВЕТ И ПРОЗРАЧНОСТЬ

```
div{
    background-color: red;
    background-color: #006600;
    background-color: rgb(28, 68, 99);
    background-color: rgb(100%,100%,100%);
    opacity: 0.4;
}
```

РАЗМЕРЫ

```
div{
    width: 150px;
    width: 75%;
    height: 15em;
    min-width: 200px;
    max-width: 300px;

}
```

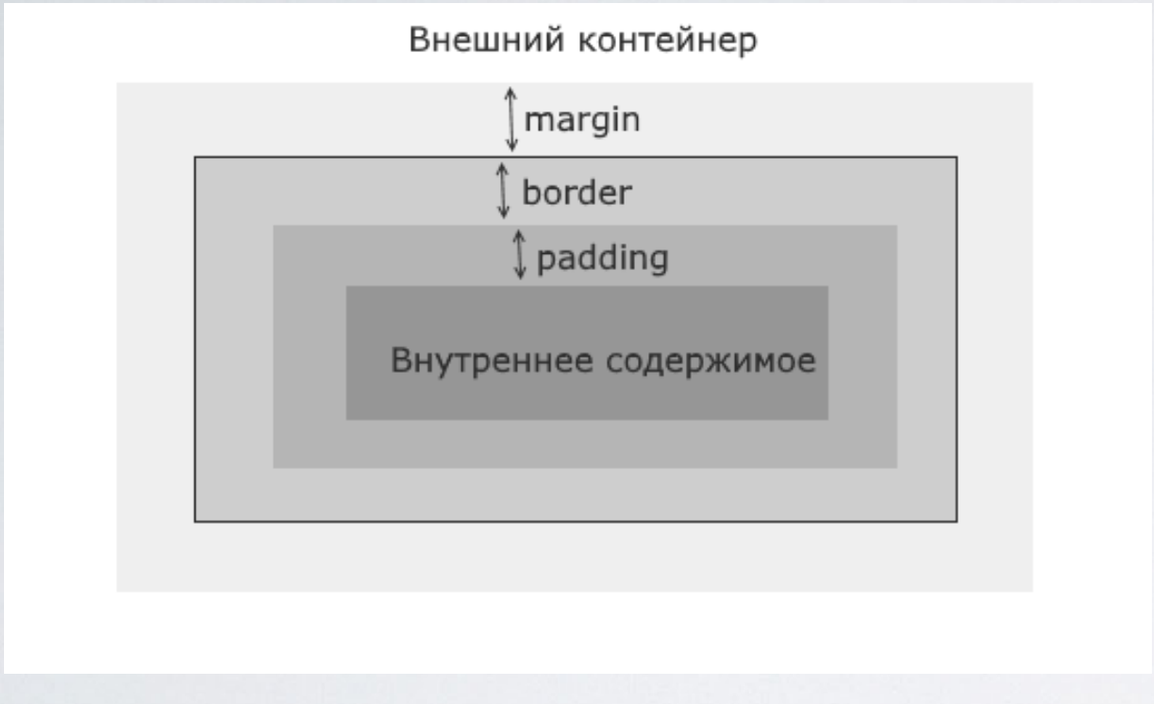
ВНЕШНИЕ ШРИФТЫ

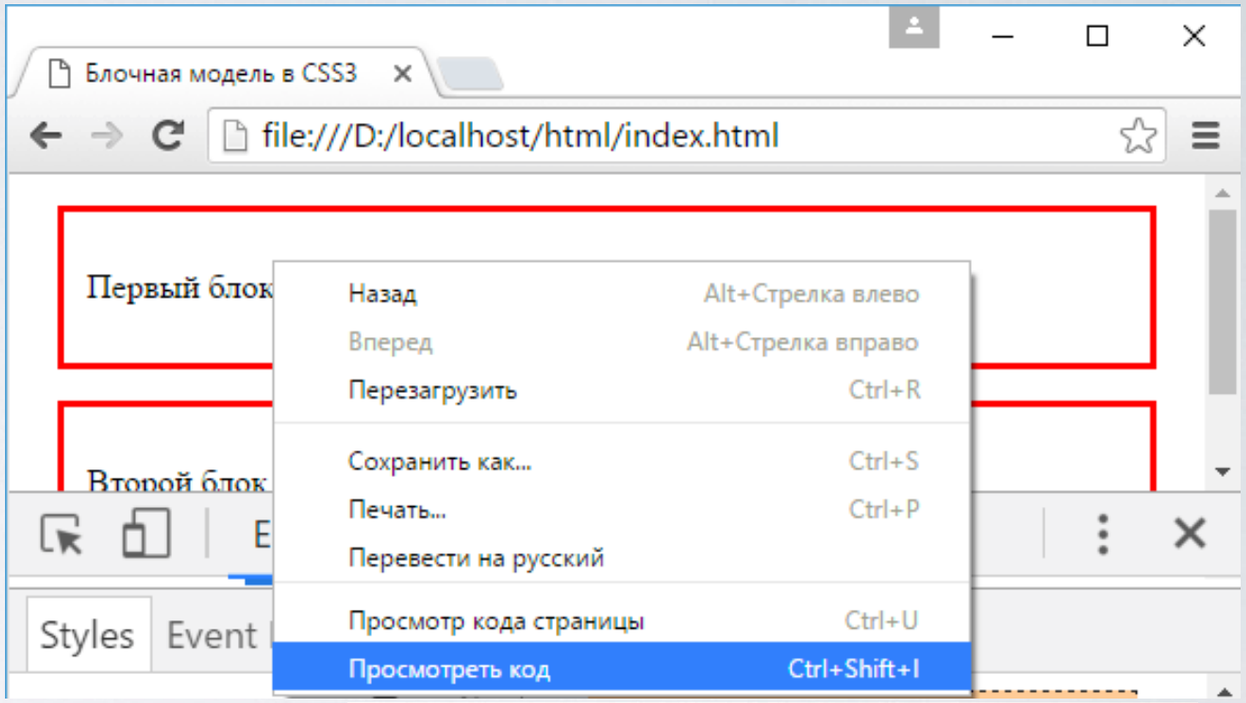
```
@font-face {

    font-family: 'Roboto';
    src: url(http://fonts.gstatic.com/s/roboto/v15/
mErvLBYg_cXG3rLvUsKT_fesZW2xOQ-xsNqO47m55DA.woff2);
}

p{

    font-family: Roboto;
}
```

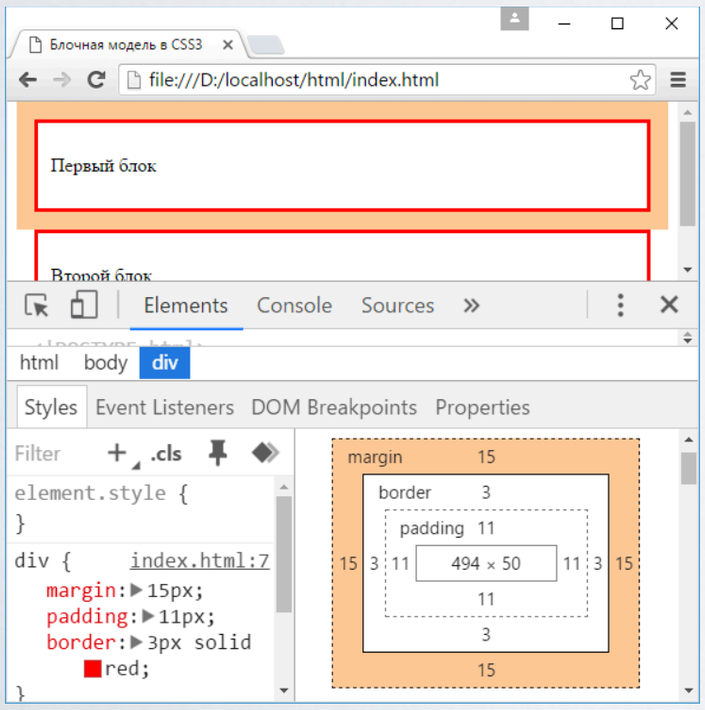
ФОН ЭЛЕМЕНТА

```
div{
    background-color: #ff0507;
    background-image: url(dubi.png);
    background-image: url(../images/
someimage.png); /* путь относительно html-
документа */
    background: url(dubi.png) no-repeat;
    background-size: 200px auto;    /* ширина
200 пикселей, автоматическая высота */
    background-position: top right;

}
```

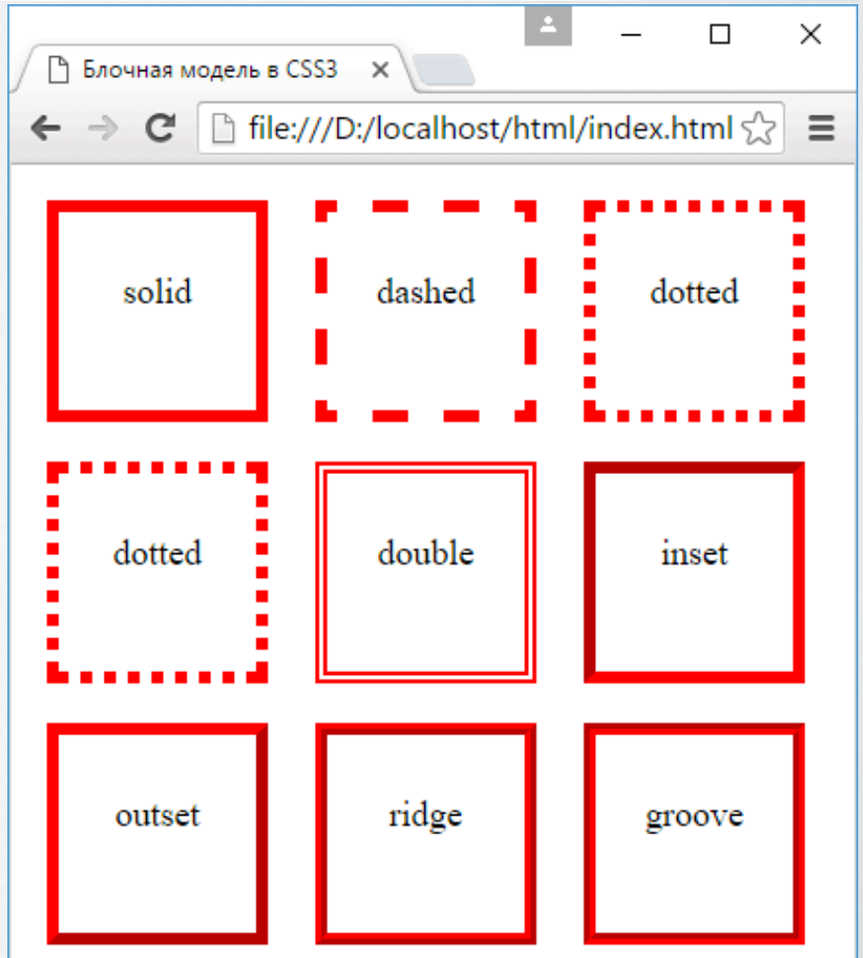
ЭЛЕМЕНТЫ КАК КОНТЕЙНЕРЫ





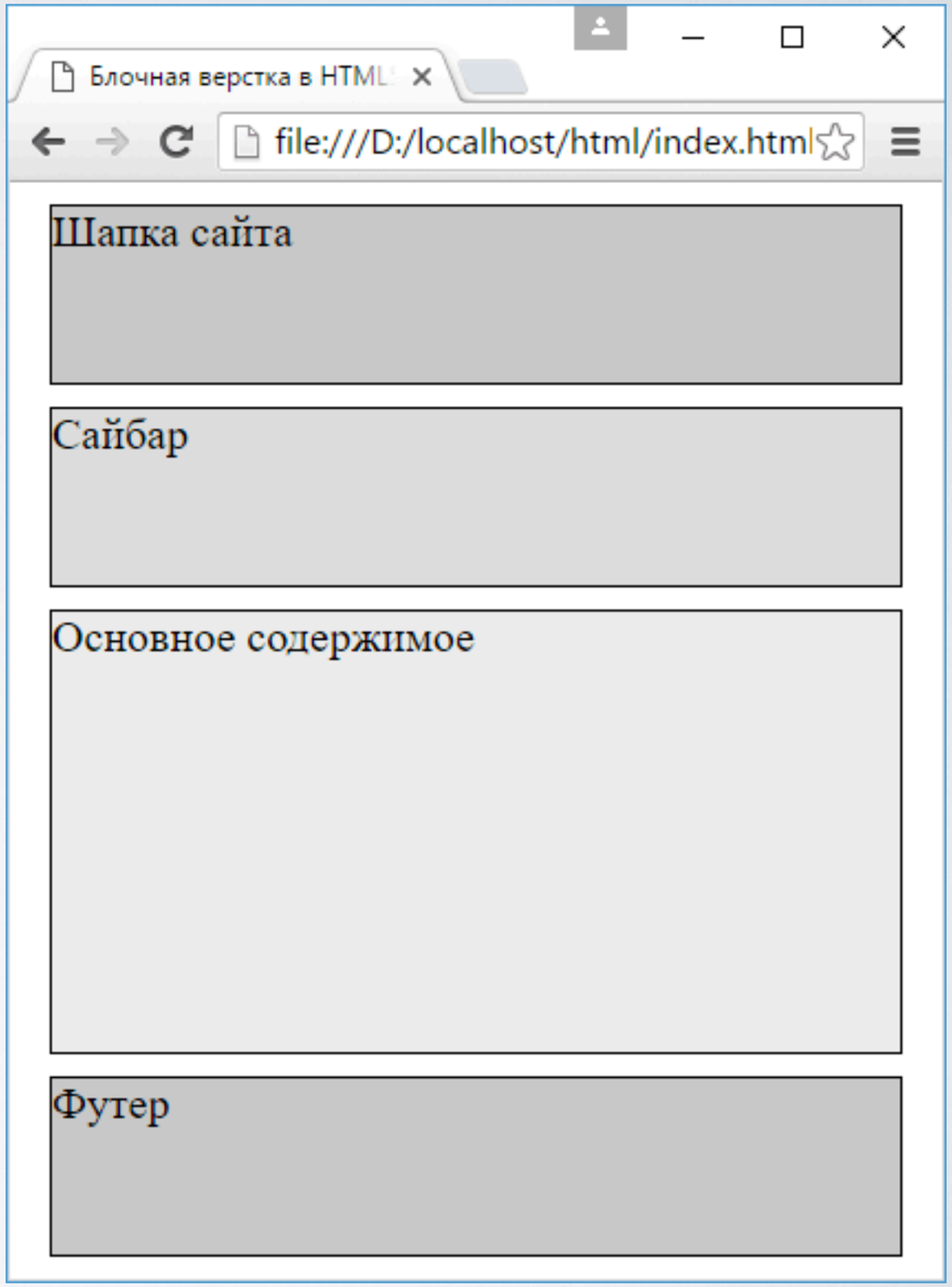



```
div{
    margin: 15px; /* внешний отступ */
    padding: 11px; /* внутренний отступ */
    border: 3px solid red;
}
```

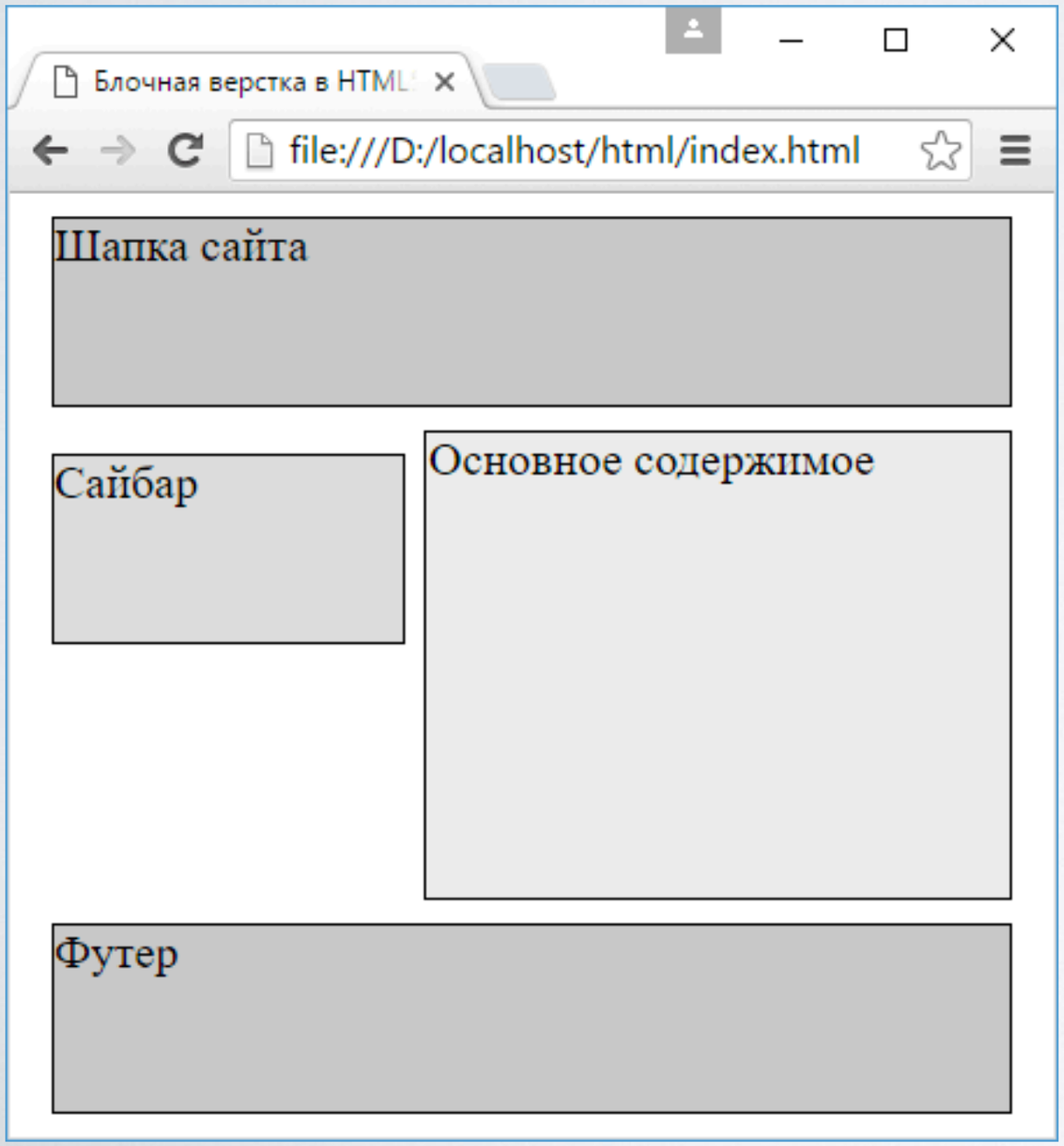


БЛОЧНАЯ ВЕРСТКА

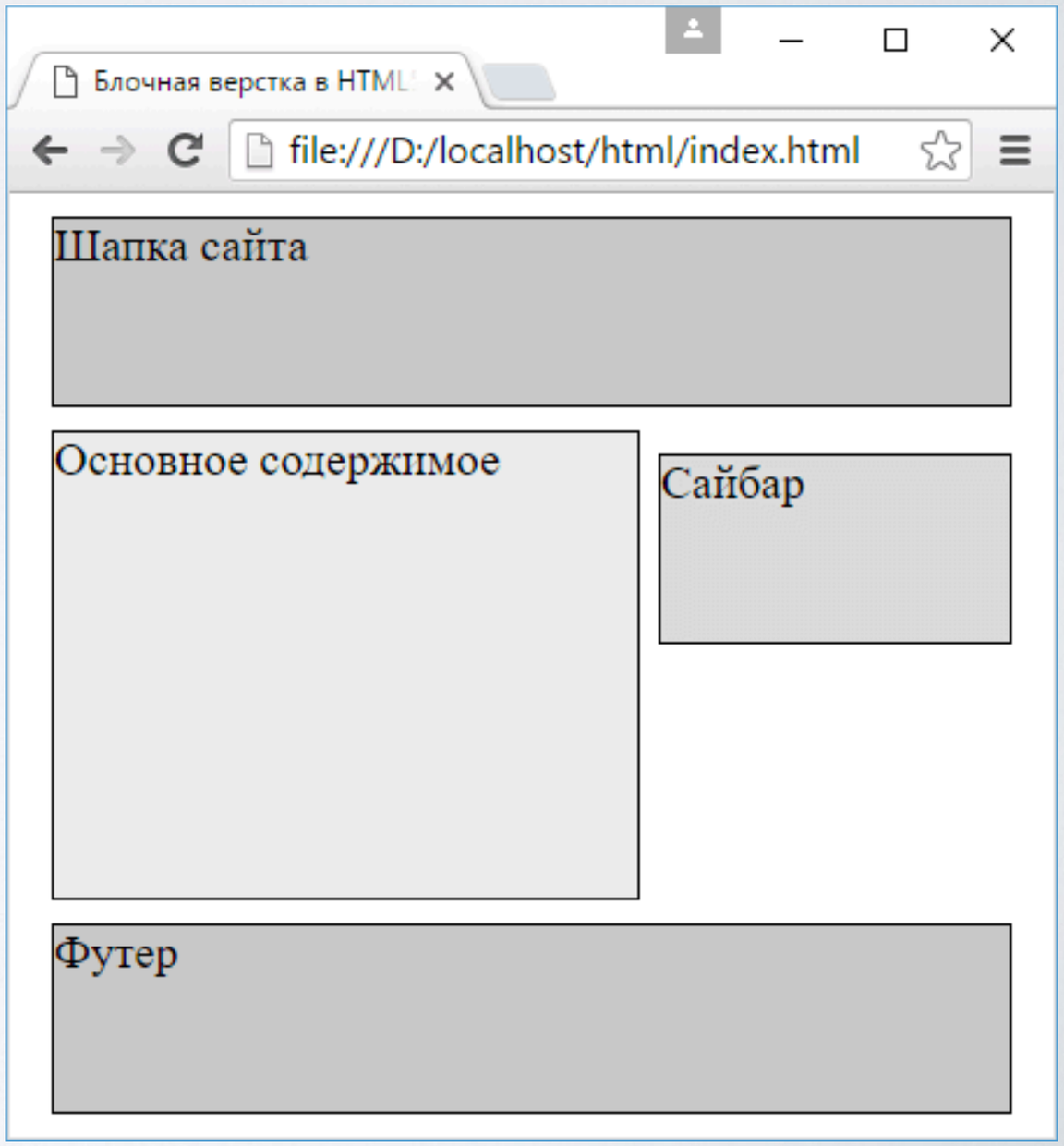
```
<html>
    <head>
        <meta charset="utf-8">
        <title>Блочная верстка в HTML5</title>
        <style>
            div{
                margin: 10px;
                border: 1px solid black;
                font-size: 20px;
                height: 80px;
            }
            #header{
                background-color: #ccc;
            }
            #sidebar{
                background-color: #ddd;
            }
            #main{
                background-color: #eee;
                height: 200px;
            }
            #footer{
                background-color: #ccc;
            }
        </style>
    </head>
    <body>
        <div id="header">Шапка сайта</div>
        <div id="sidebar">Сайбар</div>
        <div id="main">Основное содержимое</div>
        <div id="footer">Футер</div>
    </body>
</html>
```



```
#sidebar{
    background-color: #ddd;
    float: left;
    width: 150px;
}
#main{
    background-color: #eee;
    height: 200px;
    margin-left: 170px; /* 150px (ширина сайдбара) + 10px +
10px (2 отступа) */
}
```

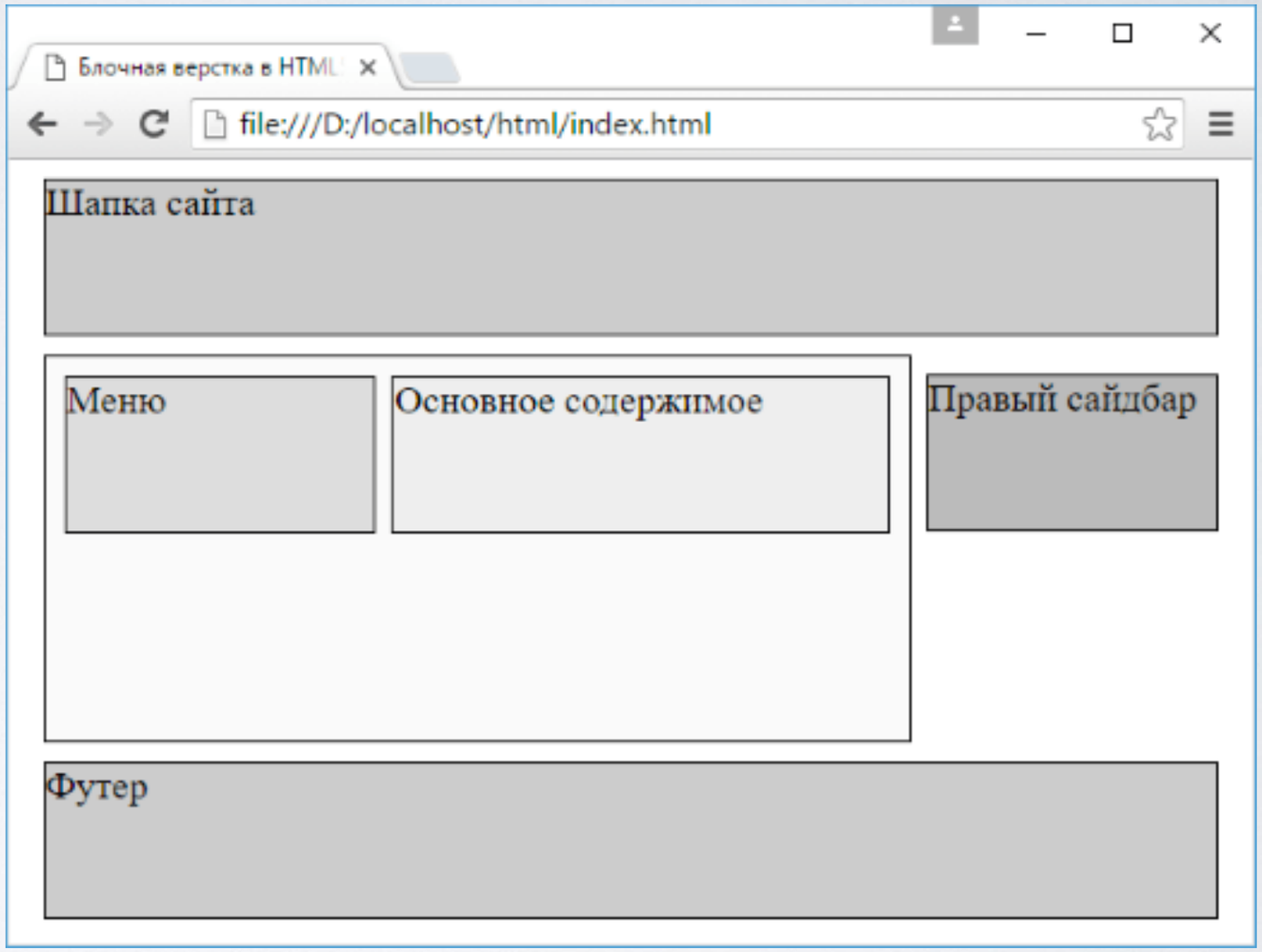



```
#sidebar{
    background-color: #ddd;
    float: right;
    width: 150px;
}
#main{
    background-color: #eee;
    height: 200px;
    margin-right: 170px;
}
```



ВЛОЖЕННЫЕ ПЛАВАЮЩИЕ БЛОКИ

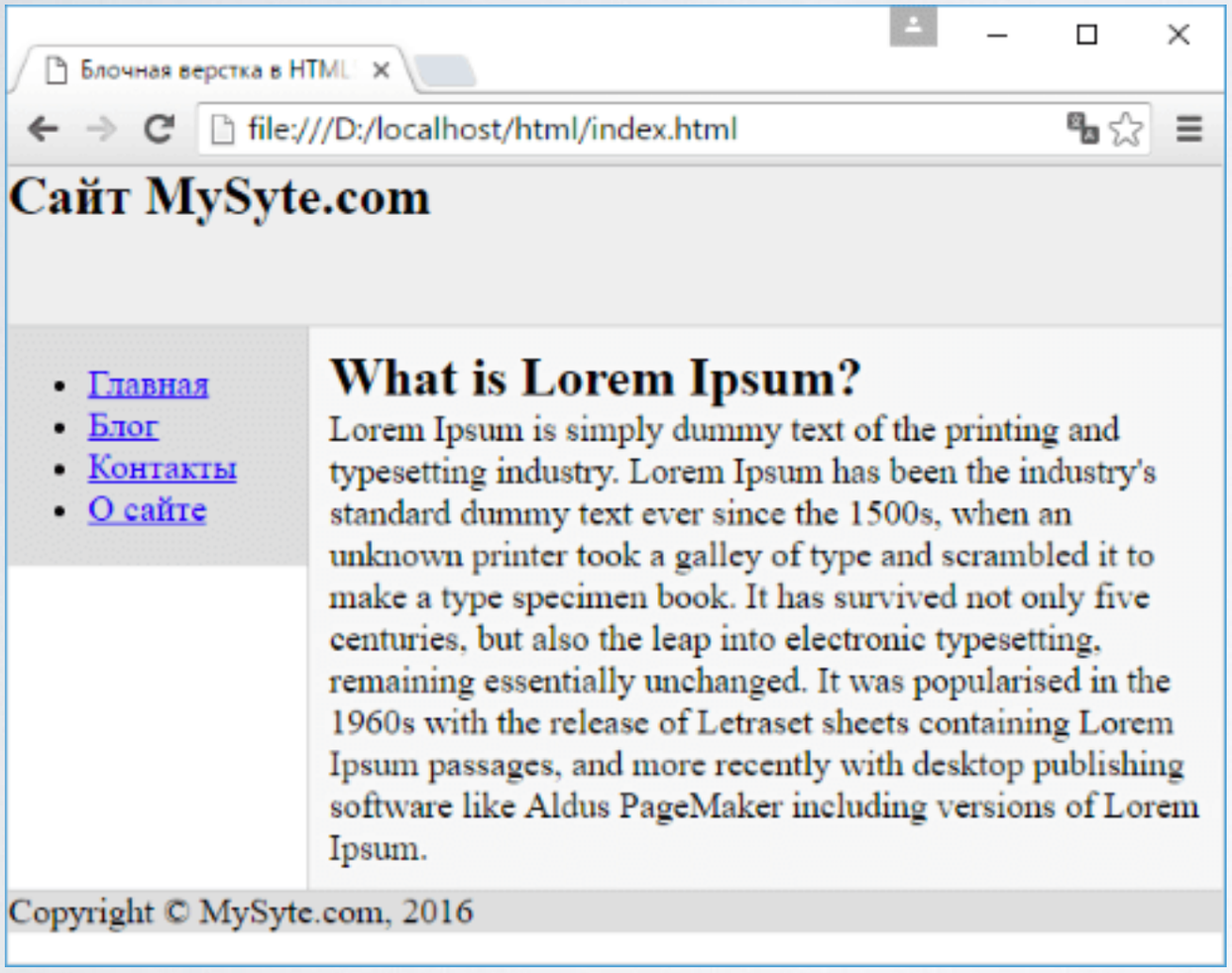
```
<style>
    div{
        margin: 10px;
        border: 1px solid black;
        font-size: 20px;
        height: 80px;
    }
    #header{ background-color: #ccc;}
    #sidebar{
        background-color: #bbb;
        float: right;
        width: 150px;
    }
        }
        #main{
            background-color: #fafafa;
            height: 200px;
            margin-right: 170px;
        }
        #menu{ float: left; background-color: #ddd; }
        #content{ background-color: #eee;}
        #footer{ background-color: #ccc; }
    </style>

<body>
    <div id="header">Шапка сайта</div>
    <div id="sidebar">Правый сайдбар</div>
    <div id="main">
        <div id="menu">Меню</div>
        <div id="content">Основное содержимое</div>
    </div>
    <div id="footer">Футер</div>
</body>
```

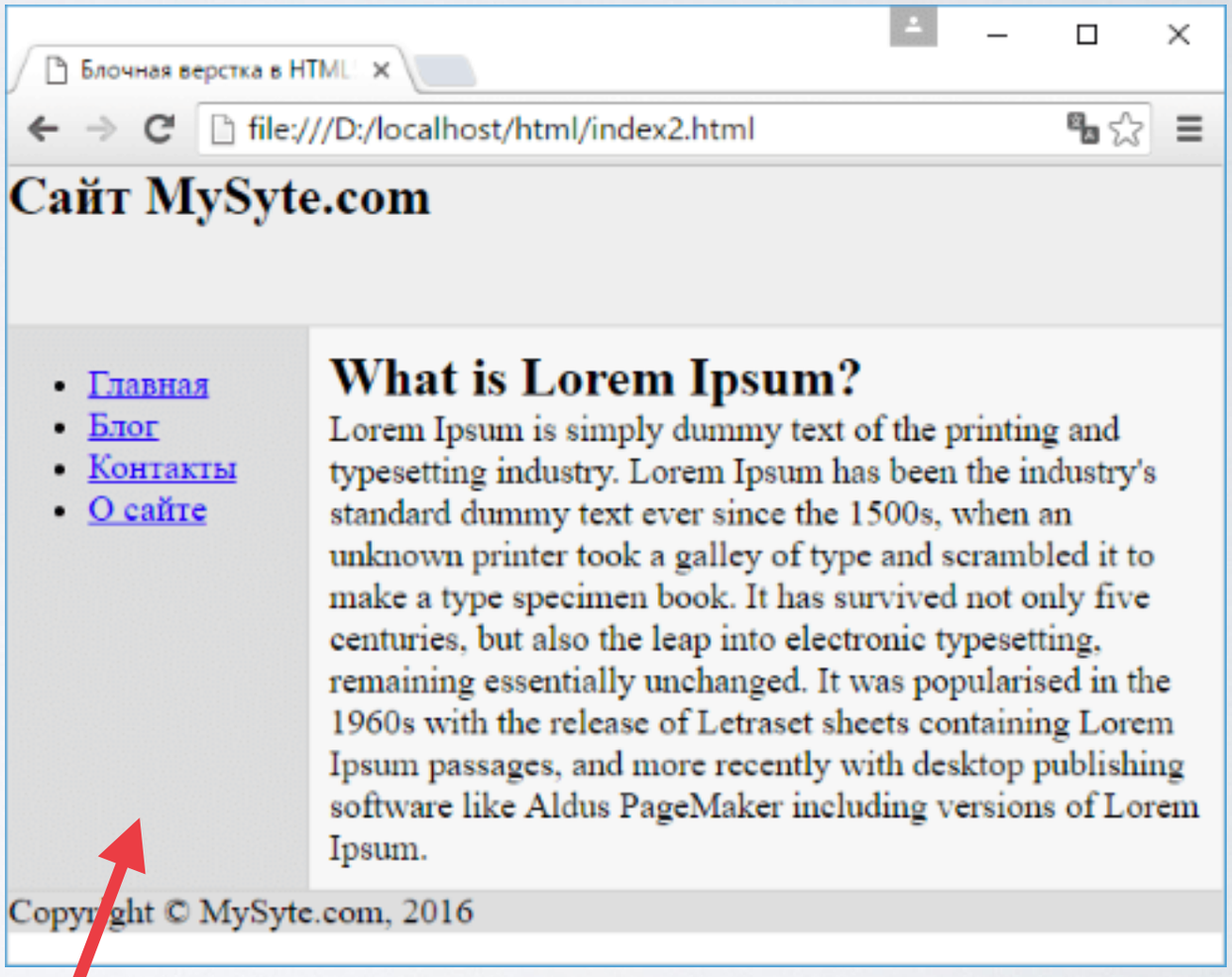


ВЫРАВНИВАНИЕ СТОЛБЦОВ ПО ВЫСОТЕ

Обычно так

Нужно так





Код 1

Код 2

Нужно обернуть плавающий элемент и блок, его обтекающий, в отдельный элемент, у которого устанавливается фон. Затем этот фон используется наименьшим по длине блоком. В итоге получается иллюзия, что блоки имеют равную длину, а фон у блоков установлен корректно.

Код 1

```
<body>
        <div id="header"><h2>Сайт MySyte.com</h2></div>
        <div id="menu">
            <ul>
                <li><a href="#">Главная</a></li>
                <li><a href="#">Блог</a></li>
                <li><a href="#">Контакты</a></li>
                <li><a href="#">О сайте</a></li>
            </ul>
        </div>
        <div id="main">
            <h2>What is Lorem Ipsum?</h2>
            <p>Lorem Ipsum is simply dummy text of the
printing and typesetting industry. Lorem Ipsum
                has been the industry...</p>
        </div>
        <div id="footer">
            <p>Copyright © MySyte.com, 2016</p>
        </div>
    </body>
```

Код 2

```
<body>
        <div id="header"><h2>Сайт MySyte.com</h2></div>
        <div id="wrapper">
            <div id="menu">
                <ul>
                    <li><a href="#">Главная</a></li>
                    <li><a href="#">Блог</a></li>
                    <li><a href="#">Контакты</a></li>
                    <li><a href="#">О сайте</a></li>
                </ul>
            </div>
            <div id="main">
                <h2>What is Lorem Ipsum?</h2>
                <p>Lorem Ipsum is simply dummy text of the printing and
typesetting industry. Lorem Ipsum
                    has been the industry...</p>
            </div>
        </div>
        <div id="footer">
            <p>Copyright © MySyte.com, 2016</p>
        </div>
    </body>
```

Одинаково у обоих вариантов

```
#menu{ float: left; width: 150px; }
```

```
#wrapper{ background-color: #ddd; }
```

Отличие

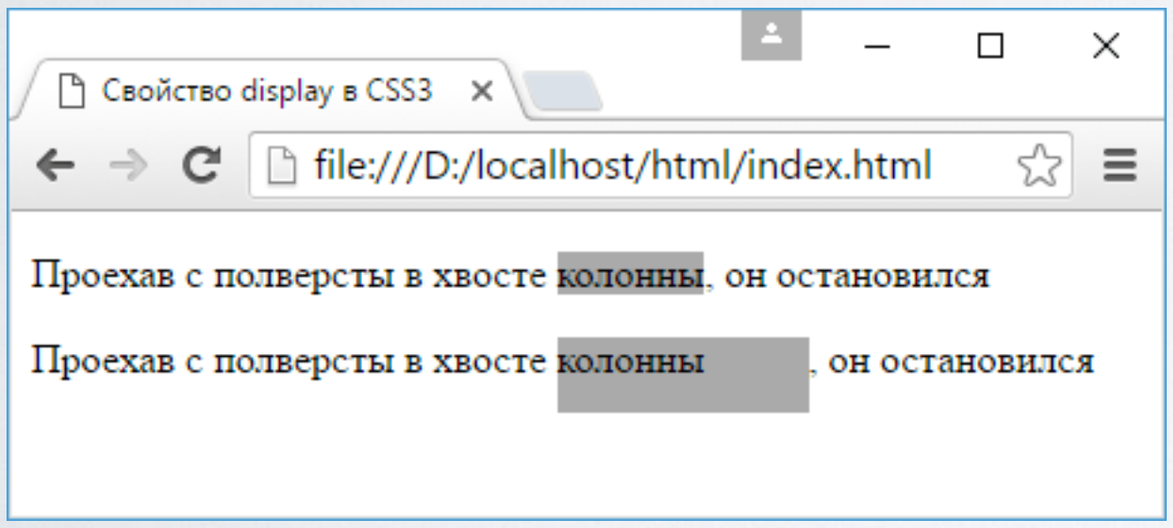
СВОЙСТВО DISPLAY

Кроме свойства float, которое позволяет изменять позицию элемента, в CSS есть еще одно важное свойство - display. Оно позволяет управлять блоком элемента и также влиять на его позиционирование относительно соседних элементов.

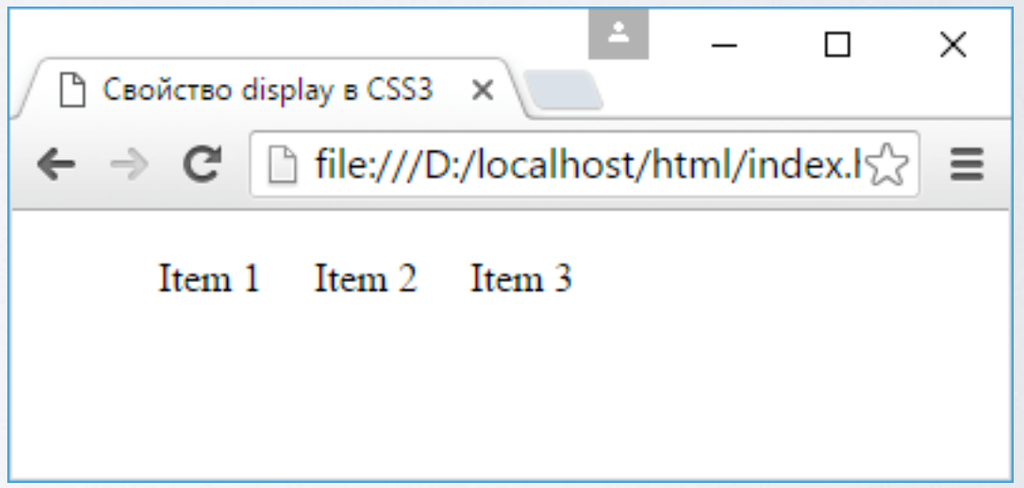
Это свойство может принимать следующие значения:

- **inline**: элемент становится строчным, подобно словам в строке текста
- **block**: элемент становится блочным, как параграф
- **inline-block**: элемент располагается как строка текста
- **list-item**: элемент позиционируется как элемент списка обычно с добавление маркера виде точки или порядкового номера
- **run-in**: тип блока элемента зависит от окружающих элементов
- **flex**: позволяет осуществлять гибкое позиционирование элементов
- **table, inline-table**: позволяет расположить элементы в виде таблицы
- **none**: элемент не виден и удален из разметки html

```
        <style>
            span{
                width: 100px;
                height: 30px;
                background-color: #aaa;
            }
            .inineBlockSpan{
                display: inline-block;
            }
        </style>

    <body>
        <p>Проехав с полверсты в хвосте <span>колонны</span>, он
остановился</p>
        <p>Проехав с полверсты в хвосте <span
class="inineBlockSpan">колонны</span>, он остановился</p>
    </body>
```




```
        <style>
            ul{
                display: table;
                margin: 0;
            }
            li{
                list-style-type: none;
                display: table-cell;
                padding: 10px;
            }
        </style>
    </head>
    <body>
        <ul>
            <li>Item 1</li>
            <li>Item 2</li>
            <li>Item 3</li>
        </ul>
    </body>
```



ПОЗИЦИОНИРОВАНИЕ

Основным свойством, которые управляют позиционированием в CSS, является свойство **position**. Это свойство может принимать одно из следующих значений:

- **static**: стандартное позиционирование элемента, значение по умолчанию
- **absolute**: элемент позиционируется относительно границ элемента-контейнера, если у того свойство position не равно static
- **relative**: элемент позиционируется относительно его позиции по умолчанию. Как правило, основная цель относительного позиционирования заключается не в том, чтобы переместить элемент, а в том, чтобы установить новую точку привязки для абсолютного позиционированния вложенных в него элементов
- **fixed**: элемент позиционируется относительно окна бразуера, это позволяет создать фиксированные элементы, которые не меняют положения при прокрутке

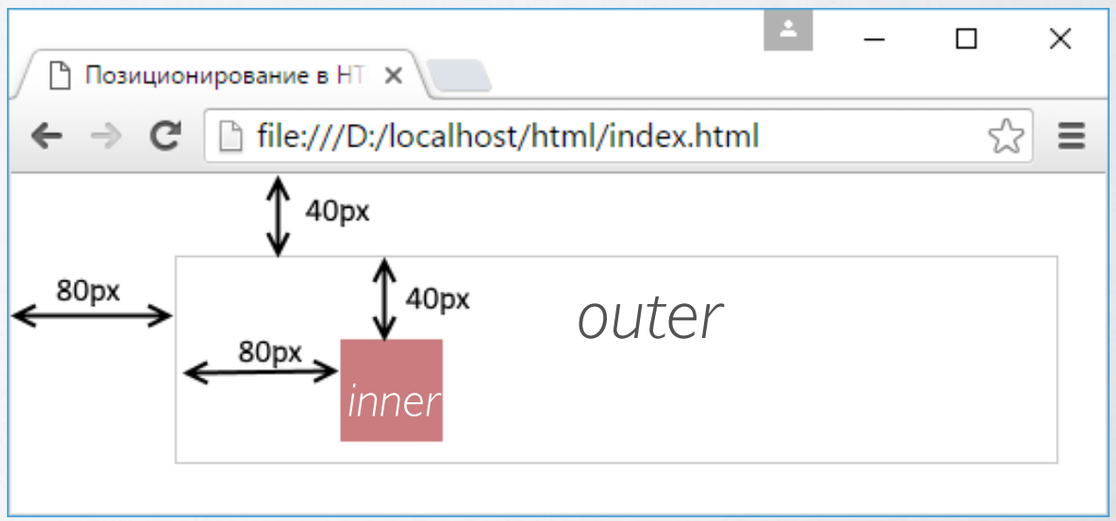
Не следует одновременно применять к элементу свойство float и любой тип позиционирования, кроме static (то есть тип по умолчанию).

Абсолютное позиционирование

```
    <style>
        .header {
            position: absolute;
            left: 100px;
            top: 50px;
            width: 430px;
            height: 100px;
            background-color: rgba(128, 0, 0, 0.5);
        }
    </style>
<body>
    <div class="header"></div>
        <p>HELLO WORLD</p>
</body>
```

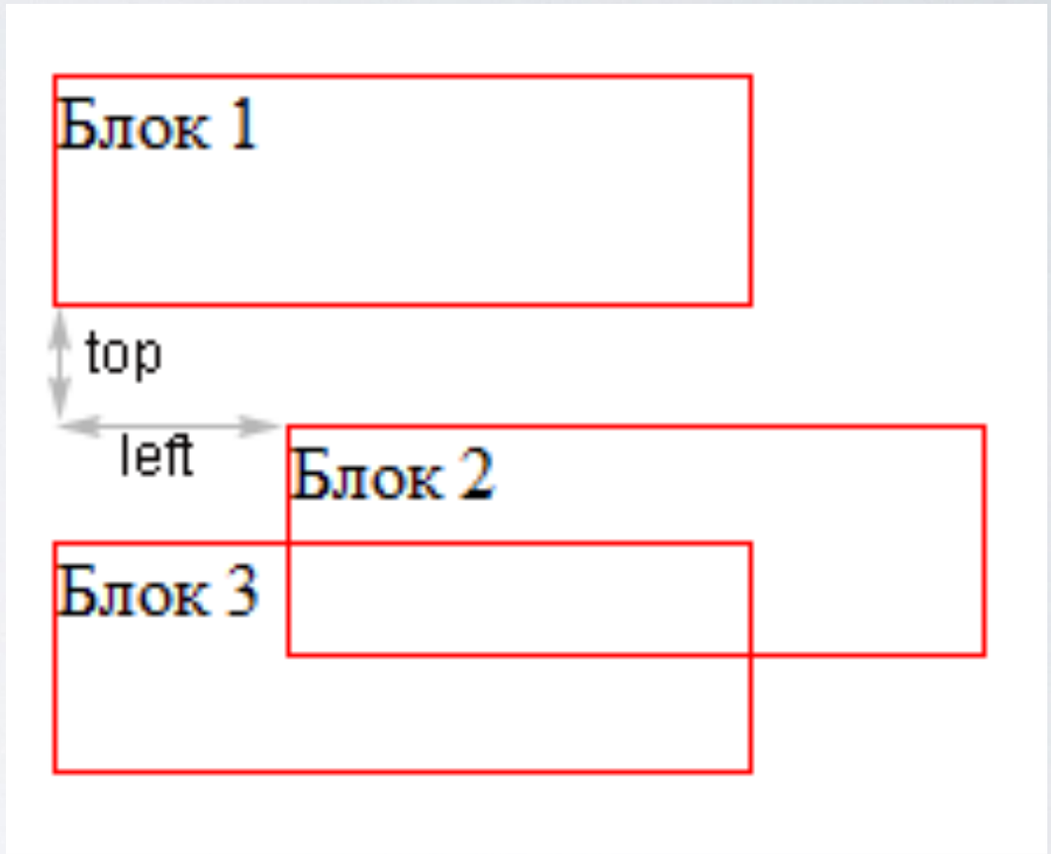
Абсолютное позиционирование внутри контейнера

```
.outer {
    position: absolute;
    left: 80px;
    top: 40px;
    width: 430px;
    height: 100px;
    border: 1px solid #ccc;
}
.inner{
    position: absolute;
    left: 80px;
    top: 40px;
    width: 50px;
    height: 50px;
    background-color: rgba(128, 0, 0, 0.5);
}
```



Относительное позиционирование

```
#block_2{
    position: relative;
    top: 25px;
    left: 50px;
}
```



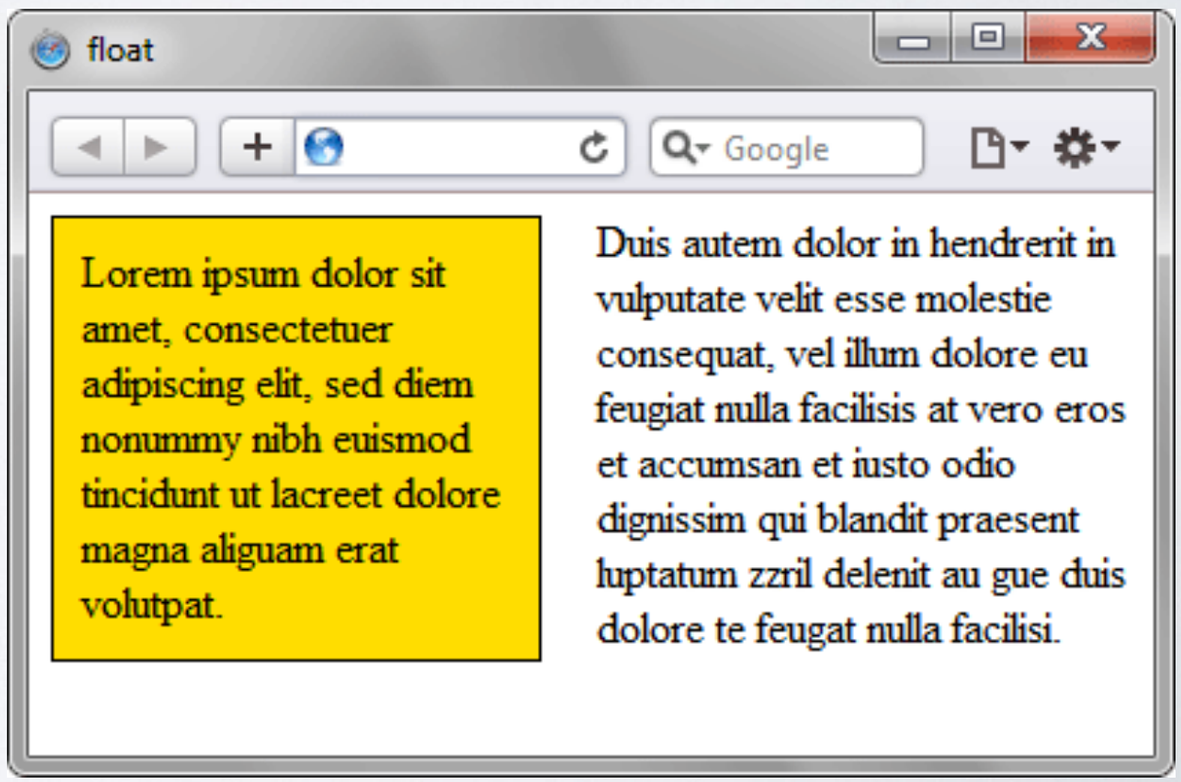
Относительно спозиционированный элемент на самом деле никуда не смещается.
Он остаётся в потоке ровно там же, где и был, а смещается *иллюзорная копия* блока.

Плавающие блоки

Блоки могут свободно перемещаться по странице, подобным образом ведут себя картинки в HTML, выровненные с помощью параметра align.

Float определяет, по какой стороне будет выравниваться элемент, при этом остальные элементы будут обтекать его с других сторон.

float: left | right | none | inherit



Flex

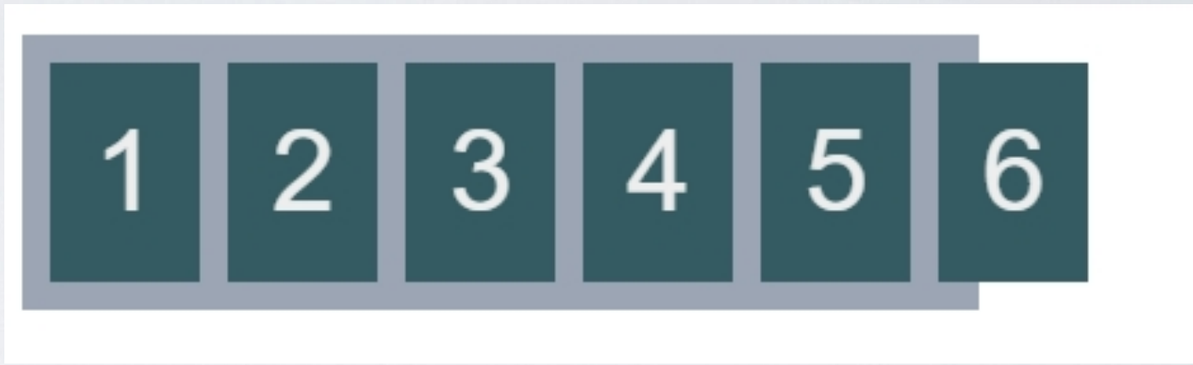
Свойство flex - это сокращенное свойство, определяющее способность гибкого элемента растягиваться или сжиматься для заполнения собой доступного свободного пространства.

Flex-direction изменяет направление главной оси контейнера.

```
.block{
    display: flex;
    flex-direction: row;
}
```

flex-wrap управляет переносом непомещающихся в контейнер элементов.

flex-wrap: nowrap



flex-wrap: wrap

flex-grow задает коэффициент увеличения элемента при наличии свободного места в контейнере. По умолчанию flex-grow: 0 т.е. никакой из элементов не должен увеличиваться и заполнять свободное место в контейнере.

```
Item_1{
    flex-grow: 1;
}

Item_2{
    flex-grow: 2;
}
```