Лекция 3

Позиционирование элементов с помощью CSS. Горизонтальное, вертикальное выравнивание. Flex. Position.

Блочные элементы

Блочным называется элемент, который отображается на веб-странице в виде прямоугольника. Такой элемент занимает всю доступную ширину, высота элемента определяется его содержимым, и он всегда начинается с новой строки.

К блочным элементам относятся теги **<address>**, **<blockquote>**, ** **, **<fieldset>**, **<form>**, **<h1>**,...,**<h6>**, **<hr>**, **<ol>**, **<p>**, **<pre>**, **<table>**, **<ul>** и некоторые устаревшие. Также блочным становится элемент, если в стиле для него свойство **display** задано как **block**, **list-item**, **table** и в некоторых случаях **run-in**.

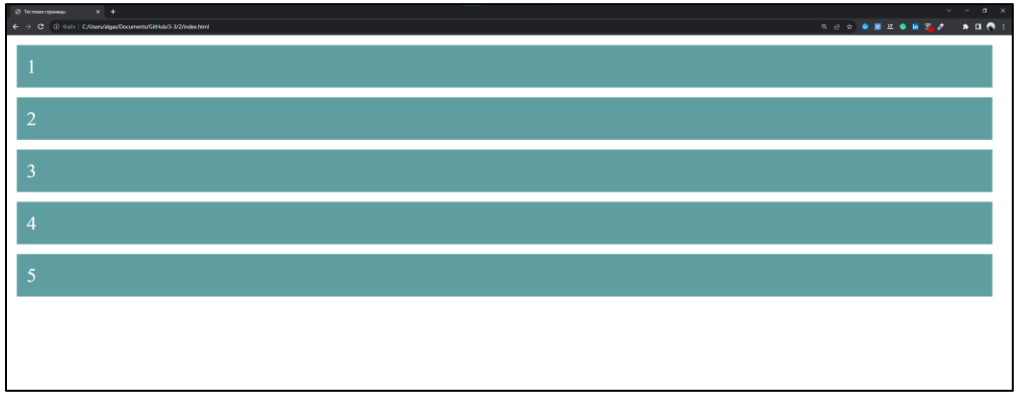
- Блоки располагаются по вертикали друг под другом.
- На прилегающих сторонах элементов действует эффект схлопывания отступов.
- Запрещено вставлять блочный элемент внутрь строчного. Например, **<a><h1>Заголовок</h1>** не пройдёт валидацию, правильно вложить теги наоборот — **<h1><a>Заголовок</h1>**.
- По ширине блочные элементы занимают всё допустимое пространство.
- Если задана ширина контента (свойство **width**), то ширина блока складывается из значений **width**, полей, границ, отступов слева и справа.
- Высота блочного элемента вычисляется браузером автоматически, исходя из содержимого блока.
- Если задана высота контента (свойство **height**), то высота блока складывается из значения **height**, полей, границ, отступов сверху и снизу. При превышении указанной высоты контент отображается поверх блока.
- На блочные элементы не действуют свойства, предназначенные для строчных элементов, вроде **vertical-align**.
- Текст по умолчанию выравнивается по левому краю.

Блочные элементы

Не все блочные теги допустимо вкладывать один в другой, поэтому при создании структуры кода активную роль выполняет ** ** как универсальный кирпичик вёрстки. Тег ** ** допустимо вкладывать один в другой, другие блочные элементы также можно помещать внутрь ** **.

```html
<body>
    <div class="container">
        <div class="item-div">1</div>

        <div class="item-div">2</div>

        <div class="item-div">3</div>

        <div class="item-div">4</div>

        <div class="item-div">5</div>
    </div>

</body>
```

```css
body {
    margin: 0;
}

.container {
    margin: 10px;
}

.item-div {
    padding: 10px;
    margin-right: 10px;
    margin-bottom: 10px;
    background-color: cadetblue;
    color: white;
    font-size: 20px;
}
```



Строчные элементы

Строчными называются такие **элементы** документа, которые являются непосредственной частью строки. К строчным элементам относятся теги **<img>**, ****, **<a>**, **<q>**, **<code>** и др., а также элементы, у которых **свойство** **display** установлено как **inline**. В основном они используются для изменения вида текста или его логического выделения

- Внутрь строчных **элементов** допустимо помещать текст или другие строчные элементы. Вставлять блочные элементы внутрь строчных запрещено.
- Эффект схлопывания отступов не действует.
- Свойства, связанные с размерами (**width**, **height**) не применимы.
- Ширина равна содержимому плюс значения отступов, полей и границ.
- Несколько строчных элементов идущих подряд располагаются на одной строке и переносятся на другую строку при необходимости.
- Можно выравнивать по вертикали с помощью свойства **vertical-align**.

Строчные элементы

Строчные элементы удобно использовать для изменения вида и стиля текста, в частности, отдельных символов и слов. Для этой цели обычно применяется универсальный тег ****, который самостоятельно никак не модифицирует содержимое, но легко объединяется со стилями через классы или идентификаторы. За счёт чего с помощью этого тега можно легко управлять видом и положением отдельных фрагментов текста или рисунков.

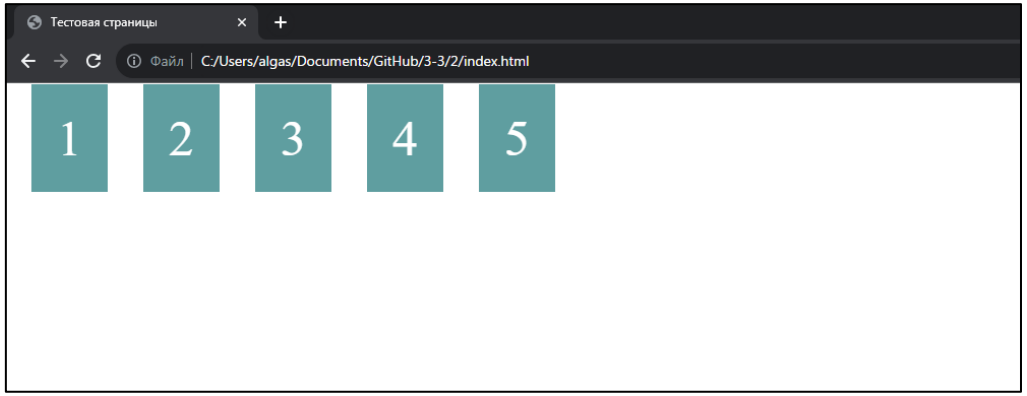
Для вёрстки строчные элементы применяются реже, чем блочные. Это связано в основном с тем, что внутрь строчных элементов не допускается вкладывать контейнеры ** **, **<p>** и подобные широко распространённые теги. Тем не менее, блочные и строчные элементы удачно дополняют друг друга, поскольку позволяют на всех уровнях менять вид составляющих веб-страниц.

Строчные элементы можно превращать в блочные с помощью свойства **display** и его значения **block**. Также возможно и обратное действие через значение **inline**

Строчные элементы

```html
<body>
    <div class="container">
        <span class="item-span">1</span>

        <span class="item-span">2</span>

        <span class="item-span">3</span>

        <span class="item-span">4</span>

        <span class="item-span">5</span>
    </div>

</body>
```

```css
body {
    margin: 0;
}

.container {
    margin: 10px;
}

.item-span {
    padding: 10px;
    margin-right: 10px;
    margin-bottom: 10px;
    background-color: cadetblue;
    color: white;
    font-size: 20px;
}
```



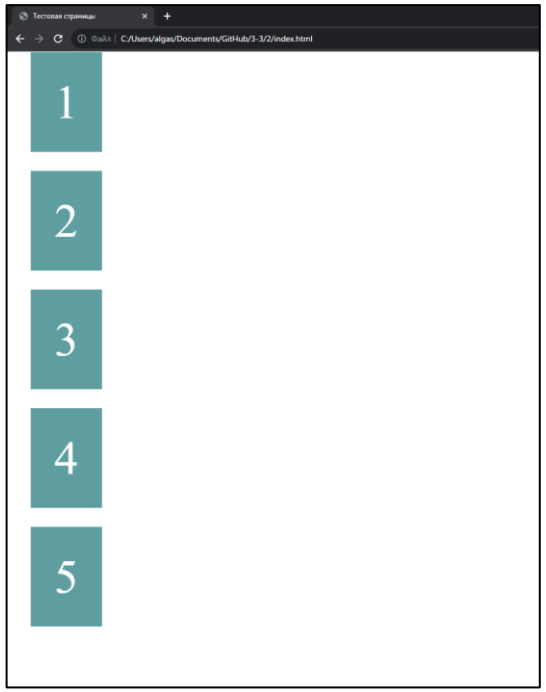
Комбинирование блочных и строчных элементов

Блочные и строчные элементы отлично дополняют друг друга при вёрстке, занимая каждый свою определённую нишу.

Если для формирования секций использовать тег ** **, как блочный элемент он будет каждый раз начинаться с новой строки. Для строчных элементов нельзя задать цвет фона всей секции и установить её размеры.

```html
<body>
    <div class="container">
        <div class="item-wrapper">
            <span class="item-span">1</span>
        </div>

        <div class="item-wrapper">
            <span class="item-span">2</span>
        </div>

        <div class="item-wrapper">
            <span class="item-span">3</span>
        </div>

        <div class="item-wrapper">
            <span class="item-span">4</span>
        </div>

        <div class="item-wrapper">
            <span class="item-span">5</span>
        </div>
    </div>
</body>
```

```css
body {
    margin: 0;
}

.container {
    margin: 10px;
}

.item-wrapper {
    margin-bottom: 10px;
    height: 40px;
}

.item-span {
    padding: 10px;
    margin-right: 10px;
    margin-bottom: 10px;
    background-color: cadetblue;
    color: white;
    font-size: 20px;
}
```



Выравнивание текста

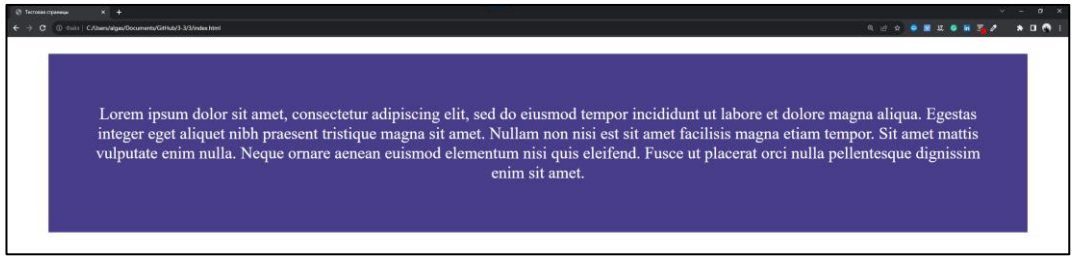
CSS позволяет выровнять текст, используя свойство **text-align** с 4 основными значениями:

- **left** — по левому краю. Используется по умолчанию
- **center** — по центру
- **right** — по правому краю
- **justify** — по ширине

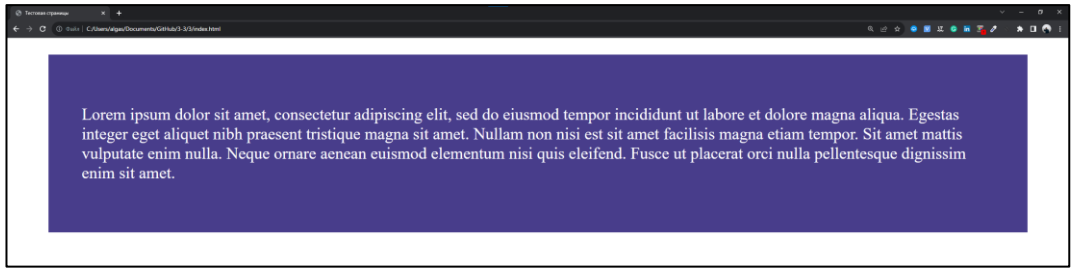
```html
<body>
    <div class="container">
        <div class="text-wrapper">
            <p class="text">
                Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
                dolore magna aliqua. Egestas integer eget aliquet nibh praesent tristique magna sit amet. Nullam non
                nisi est sit amet facilisis magna etiam tempor. Sit amet mattis vulputate enim nulla. Neque ornare
                aenean euismod elementum nisi quis eleifend. Fusce ut placerat orci nulla pellentesque dignissim enim
                sit amet.
            </p>
        </div>
    </div>
</body>
```

```css
body {
    margin: 0;
}

.container {
    margin: 10px;
}

.text-wrapper {
    background-color: darkslateblue;
    color: white;
    font-size: 20px;
    padding: 40px;
    margin: 20px 40px;

}

.text {
    text-align: center;
    /* text-align: left; */
    /* text-align: right; */
    /* text-align: justify; */
}
```

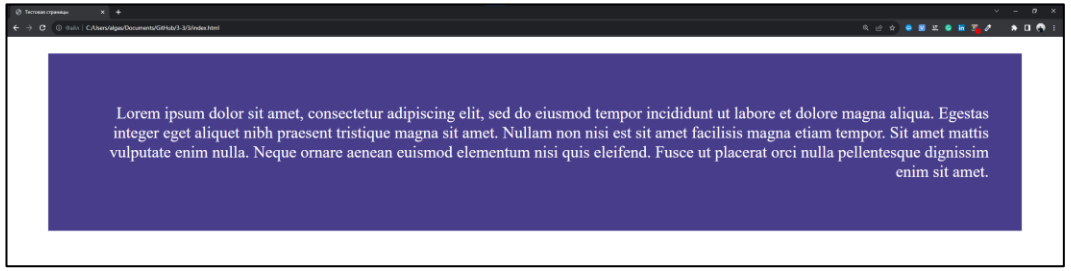
Выравнивание текста



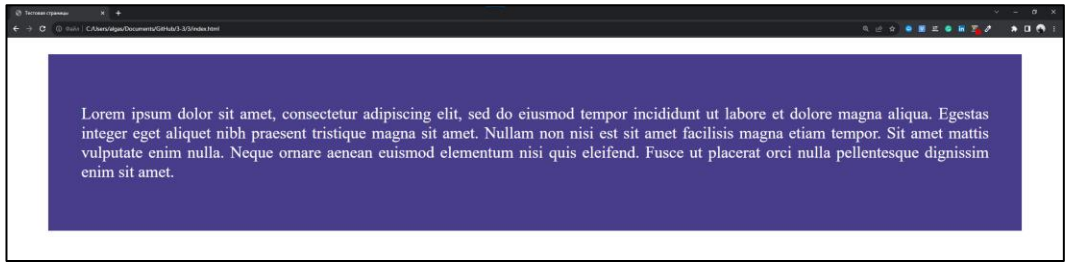

```
text-align: center;
```




```
text-align: left;
```




```
text-align: right;
```




```
text-align: justify;
```

Выравнивание текста

CSS свойство **text-align** позволяет выравнивать только текст. С блочными и строчными элементами он не работает

```html
<body>
    <div class="container">
        <div class="text-wrapper">
            <div class="block"></div>
        </div>
    </div>
</body>
```

```css
.text-wrapper {
    background-color: darkslateblue;
    color: white;
    font-size: 20px;
    padding: 40px;
    margin: 20px 40px;

}

.block {
    /* не работает, потому что это блок, а не текст */
    text-align: center;
    width: 100px;
    height: 100px;
    background-color: red;
}
```

Горизонтальное выравнивание элементов

Чтобы горизонтально выровнять по центру блоковый элемент (например, ** **), можно использовать свойство **margin: auto;**. Определение ширины элемента предотвратит его вытягивание до границ контейнера. При таких установках элемент займет заданную ширину, а оставшееся пространство будет поровну поделено между двумя отступами.

Выравнивание по центру не будет работать, если свойство **width** не установлено (или установлено в 100%).

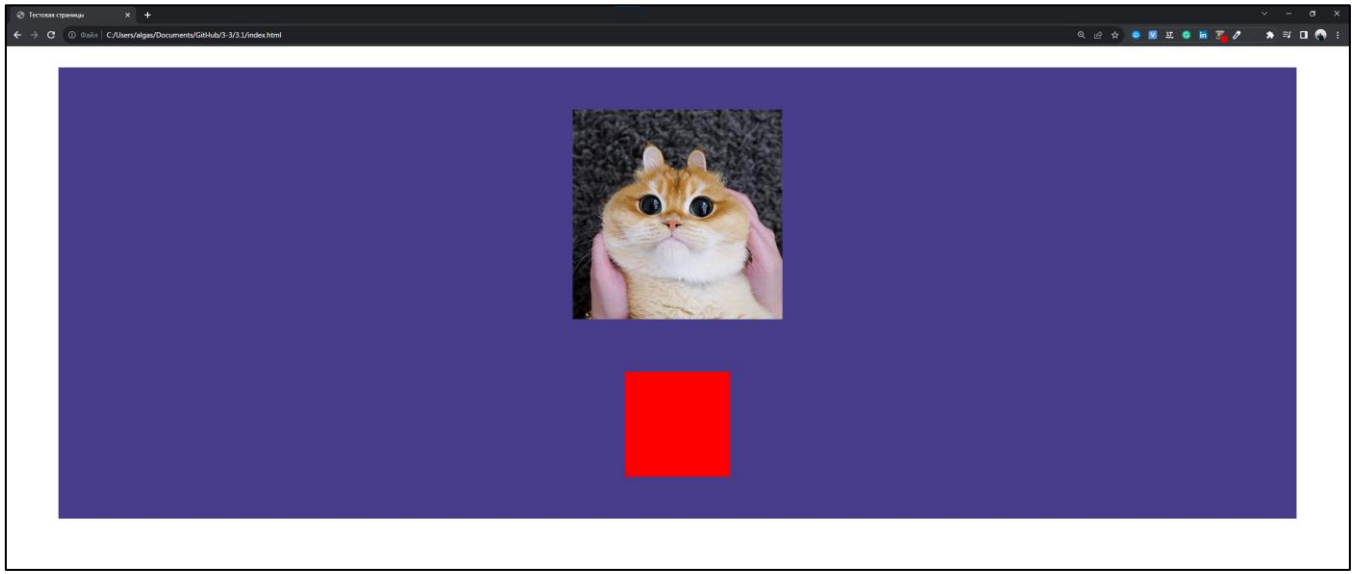
```html
<body>
    <div class="container">
        <div class="text-wrapper">
            <p class="text">
                Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Egestas integer eget aliquet nibh praesent tristique magna sit amet. Nullam non nisi est sit amet facilisis magna etiam tempor. Sit amet mattis vulputate enim nulla. Neque ornare aenean euismod elementum nisi quis eleifend. Fusce ut placerat orci nulla pellentesque dignissim enim sit amet.
            </p>
        </div>
    </div>
</body>
```


```css
.text-wrapper {
    background-color: darkslateblue;
    color: white;
    font-size: 20px;
    padding: 40px;
    margin: 20px 40px;

}

.text {
    width: 200px;
    margin-left: auto;
    margin-right: auto;
}
```

Горизонтальное выравнивание элементов

```html
<body>
    <div class="container">
        <div class="text-wrapper">
            <img class="image" src="../images/cat1.jpg" />

            <div class="block"></div>
        </div>
    </div>
</body>
```

```css
.image {
    /*
       мы превратили строчный элемент в
       блочный с помощью свойства display
    */
    display: block;
    height: 200px;
    margin-left: auto;
    margin-right: auto;
    margin-bottom: 50px;
}

.block {
    width: 100px;
    height: 100px;
    background-color: red;
    margin-left: auto;
    margin-right: auto;
}
```

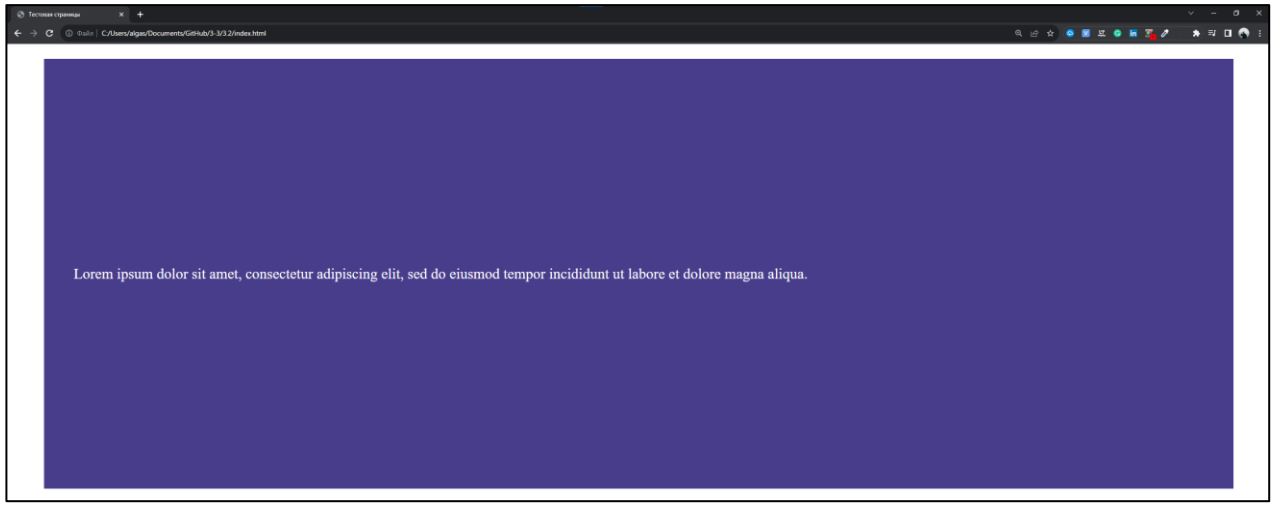


Вертикальное выравнивание элементов

В CSS существует множество способов центрирования элемента по вертикали. Одним из самых простых способов является добавление родительскому элементу свойства **display: flex;** и свойства **align-items: center;**.

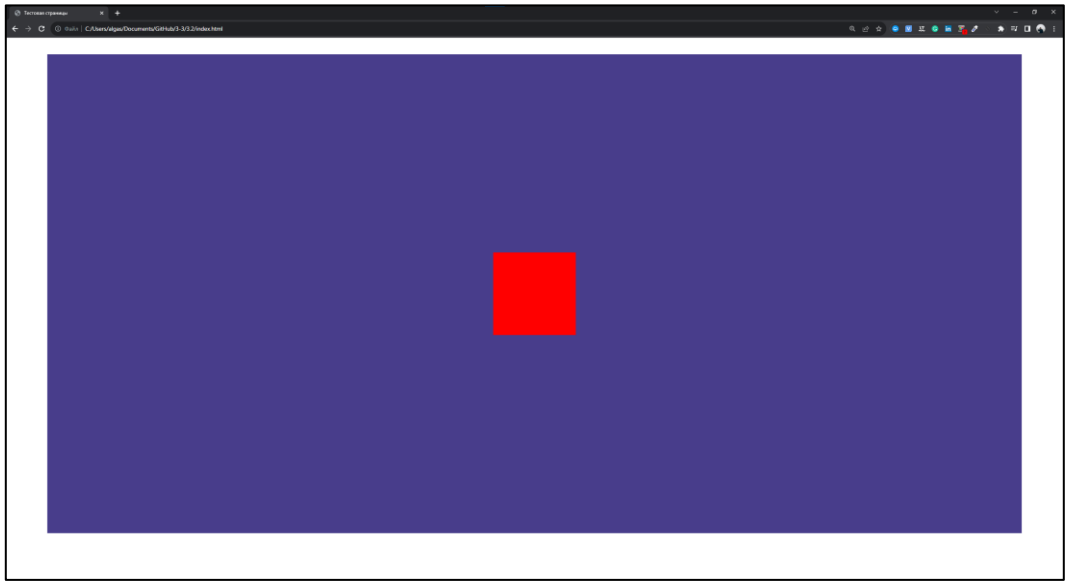
```html
<body>
    <div class="container">
        <div class="text-wrapper">
            <p class="text">
                Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et
                dolore magna aliqua.
            </p>
        </div>
    </div>
</body>
```

```css
.text-wrapper {
    height: 500px;
    display: flex;
    align-items: center;
    background-color: darkslateblue;
    color: white;
    font-size: 20px;
    padding: 40px;
    margin: 20px 40px;
}
```



Комбинирование вертикального и горизонтального выравнивания

```html
<body>
    <div class="container">
        <div class="text-wrapper">
            <div class="block"></div>
        </div>
    </div>
</body>
```



```css
.text-wrapper {
    height: 500px;
    display: flex;
    align-items: center;
    background-color: darkslateblue;
    color: white;
    font-size: 20px;
    padding: 40px;
    margin: 20px 40px;
}

.block {
    width: 100px;
    height: 100px;
    background-color: red;
    margin-left: auto;
    margin-right: auto;
}
```

Flexbox

Flexbox (или просто flex) — это способ позиционирования элементов в CSS. С помощью этой функции можно быстро и легко описывать, как будет располагаться тот или иной блок на веб-странице. Элементы выстраиваются по заданной оси и автоматически распределяются согласно настройкам.

Принцип работы флексбокса — выстраивание элементов внутри какого-то крупного блока по прямой оси.

Создание контейнера. Крупный блок, внутри которого расположены элементы, называется флекс-контейнером. Чтобы превратить блок во флекс-контейнер, нужно задать для него CSS-свойство: **display: flex;**.

Эту команду можно прочитать как «способ отображения: флексбокс». После этого все, что находится внутри контейнера, будет подчиняться правилам флекса — превратится во флекс-элементы. Но это касается только прямых потомков — блоков непосредственно в контейнере. Если внутри них тоже вложены какие-то элементы, на них правила не распространятся.

Распределение по оси. Когда блок превращается во флекс-контейнер, его содержимое автоматически выстроится вдоль прямой оси. По умолчанию она горизонтальная и «смотрит» слева направо, но это поведение можно изменить с помощью CSS.

Flexbox

С помощью свойства **flex-direction** можно изменить направление оси. У свойства есть четыре значения:

- **row** — ряд, горизонтальная линия слева направо;
- **reverse-row** — ряд справа налево;
- **column** — колонка, вертикальная линия сверху вниз;
- **reverse-column** — колонка снизу вверх.

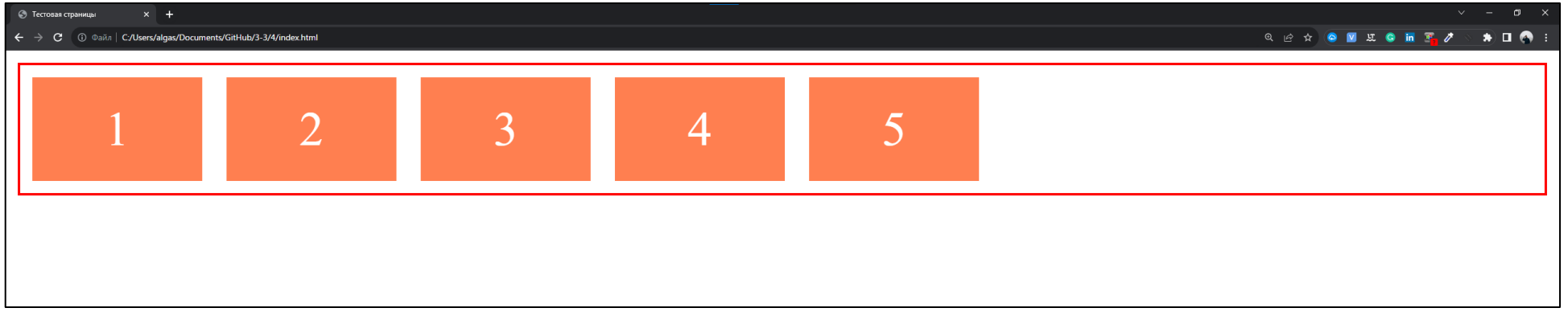
Чаще всего пользуются значениями **row** и **column**. **Row** хорошо подходит для горизонтальных меню, списков и так далее — элементов, которые расположены «в строку». А **column** — для наборов элементов, расположенных вертикально, «в столбик».

Flexbox

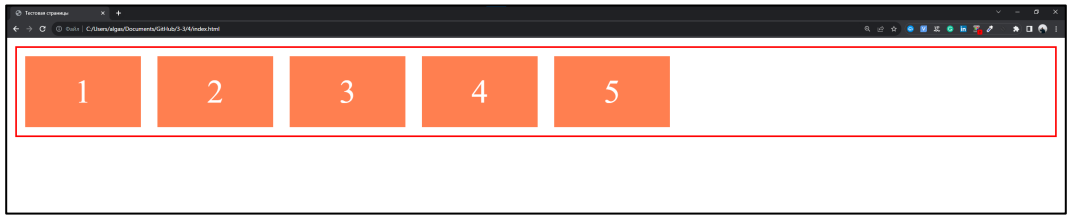
```html
<body>
    <div class="container">
        <div class="flex-container">
            <div class="item">1</div>
            <div class="item">2</div>
            <div class="item">3</div>
            <div class="item">4</div>
            <div class="item">5</div>
        </div>
    </div>
</body>
```

```css
.flex-container {
    border: 2px solid red;
    display: flex;
    flex-direction: row;
    /* flex-direction: row-reverse; */
    /* flex-direction: column; */
    /* flex-direction: column-reverse; */
}
```

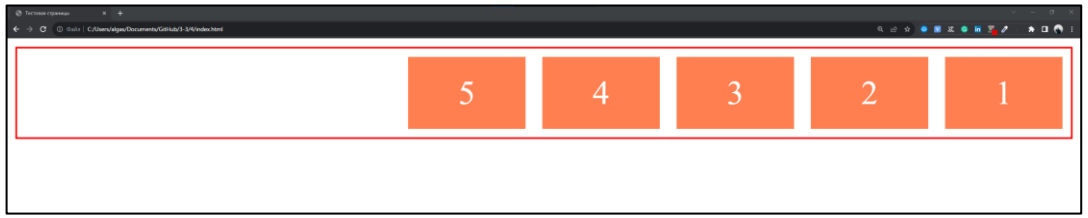
```css
.item {
    width: 100px;
    margin: 10px;
    padding: 20px;
    background-color: coral;
    font-size: 40px;
    color: white;
    text-align: center;
}
```



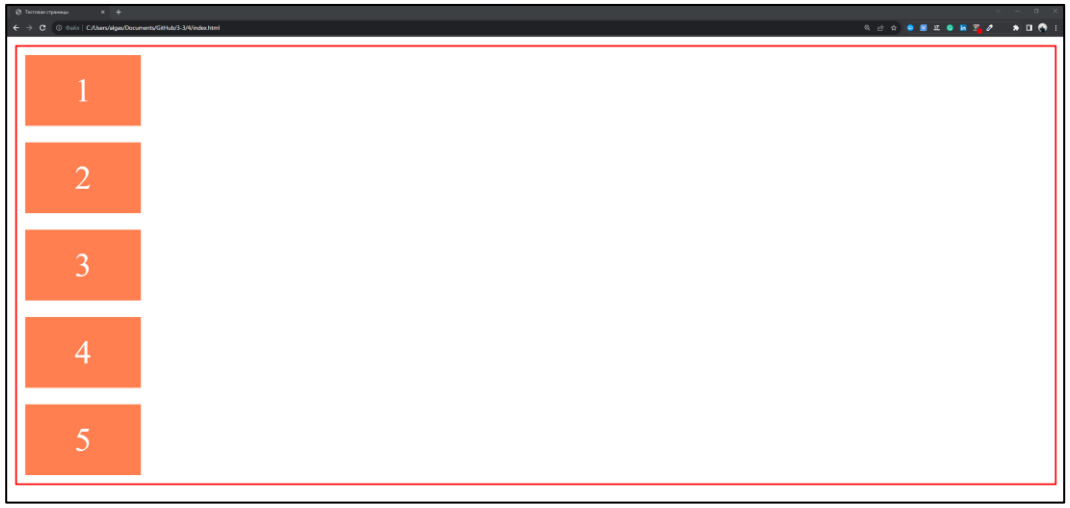
Flexbox



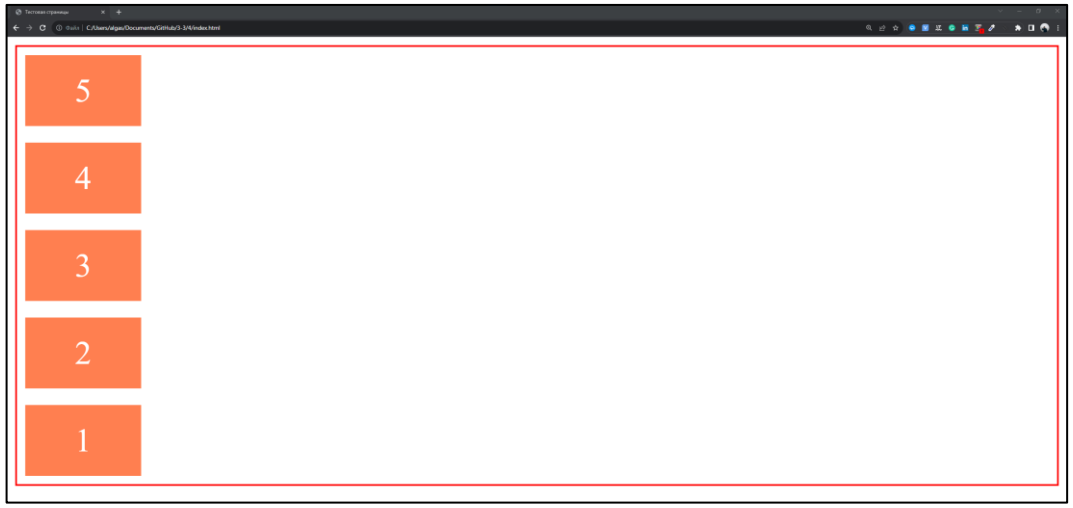
`flex-direction: row;`



`flex-direction: row-reverse;`



`flex-direction: column;`



`flex-direction: column-reverse;`

Flexbox

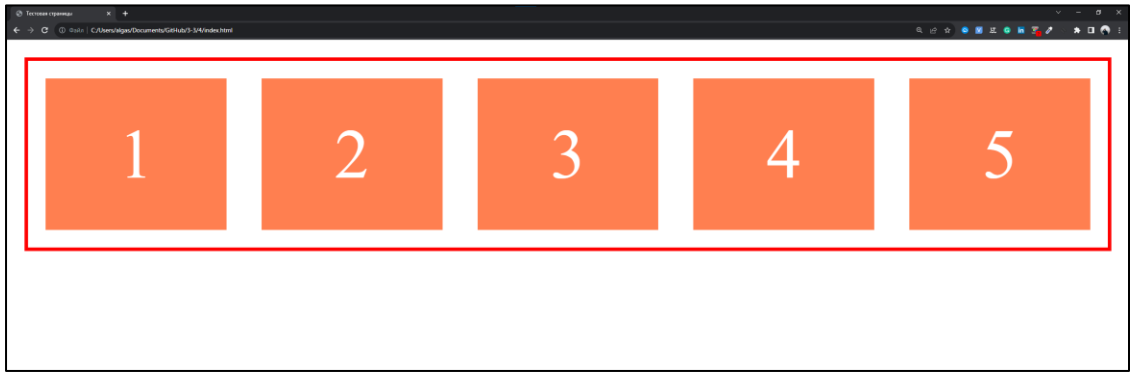
Что, если элементов станет слишком много и они не поместятся на оси? В таком случае есть два варианта поведения:

- элементы автоматически сожмутся, чтобы все уместились в одном ряду или колонке;
- элементы не будут сжиматься, а излишки перенесутся на другую строку / в другую колонку.

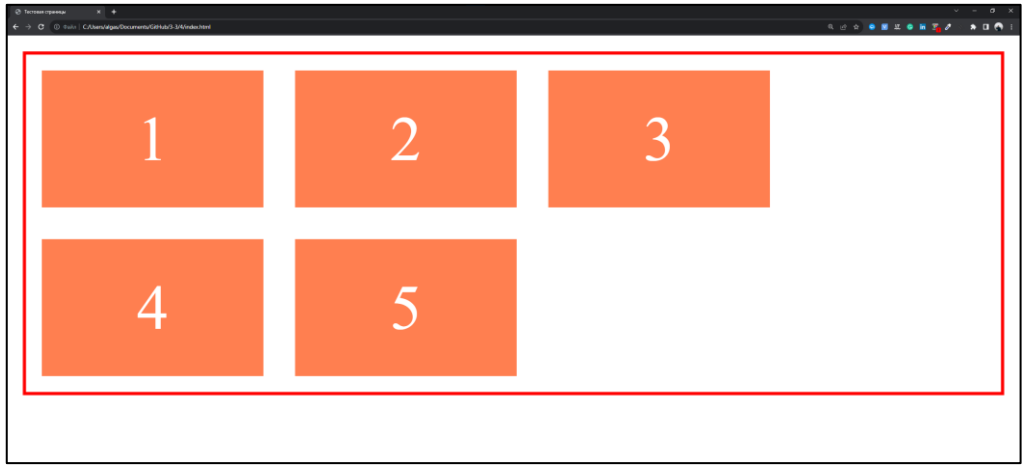
Этим поведением управляет свойство **flex-wrap**. По умолчанию его значение — **nowrap**, то есть не переносить, а сжимать. Если указать значение **wrap**, элементы будут переноситься на другую строку, а **wrap-reverse** — будут переноситься, но в противоположном направлении. То есть новая строка возникнет не снизу, а сверху.

```
.flex-container {
    border: 2px solid red;
    display: flex;
    flex-direction: row;

    /* flex-wrap: nowrap; */
    flex-wrap: wrap;
}
```

Flexbox



flex-wrap: nowrap;



flex-wrap: wrap;

Flexbox

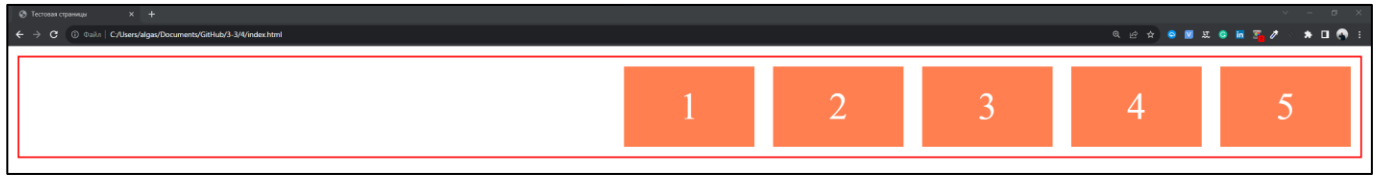
Когда нужное направление оси указано, элементы автоматически выстроятся вдоль него. По умолчанию они прижаты к левому или верхнему краю оси в зависимости от ее направления — горизонтального или вертикального. Но способ построения можно изменить с помощью свойства **justify-content**, выровнять контент.

- **flex-start** или просто **left** — выравнивать по «начальной точке» оси, то есть по левому краю для row или по верхнему для column;
- **flex-end** или **right** — выстраивать по «конечной точке», правой или нижней;
- **center** — концентрировать элементы в центре;
- **space-between** — прижимать крайние элементы к левому и правому краям, а элементы между ними распределять равномерно;
- **space-around** — то же самое, но с отступами от левого и правого края для крайних элементов. Каждый отступ от края в два раза меньше, чем отступы между элементами.

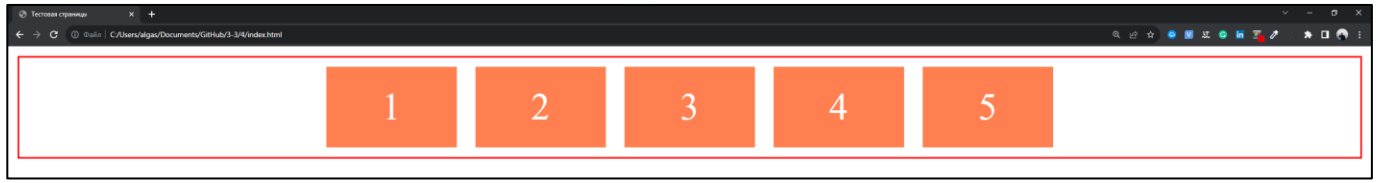
```
.flex-container {
    border: 2px solid red;
    display: flex;
    flex-direction: row;

    justify-content: left;
    /* justify-content: right; */
    /* justify-content: center; */
    /* justify-content: space-between; */
    /* justify-content: space-around; */
}
```

Flexbox


```
justify-content: left;
```

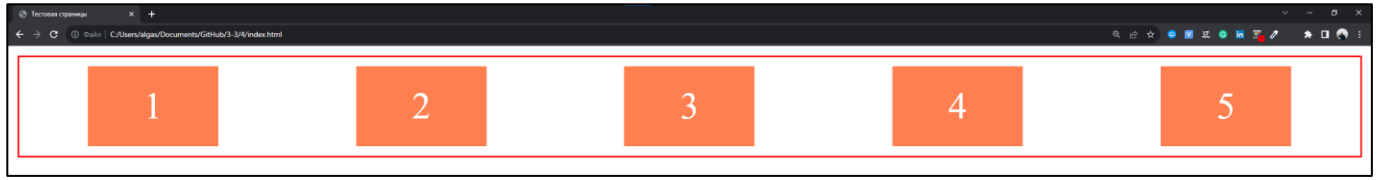



```
justify-content: right;
```




```
justify-content: center;
```


```
justify-content: space-between;
```




```
justify-content: space-around;
```

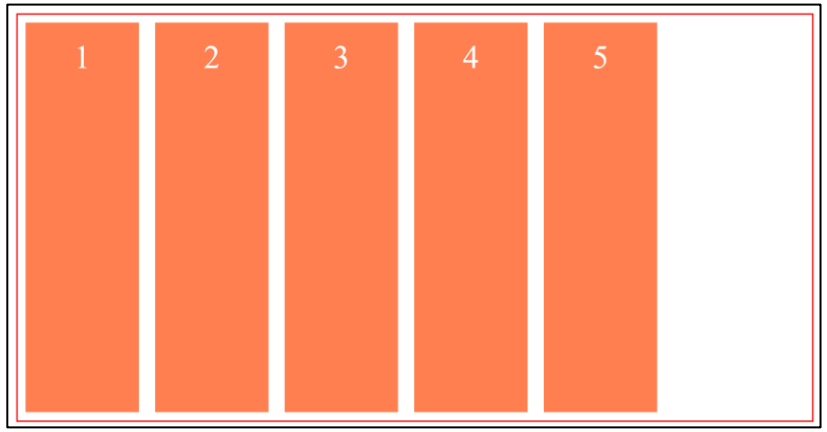
Flexbox

Можно выстроить элементы не только вдоль оси, но и «поперек» нее: для **row** — по вертикали, для **column** — по горизонтали. Поведение описывается свойством **align-items**.

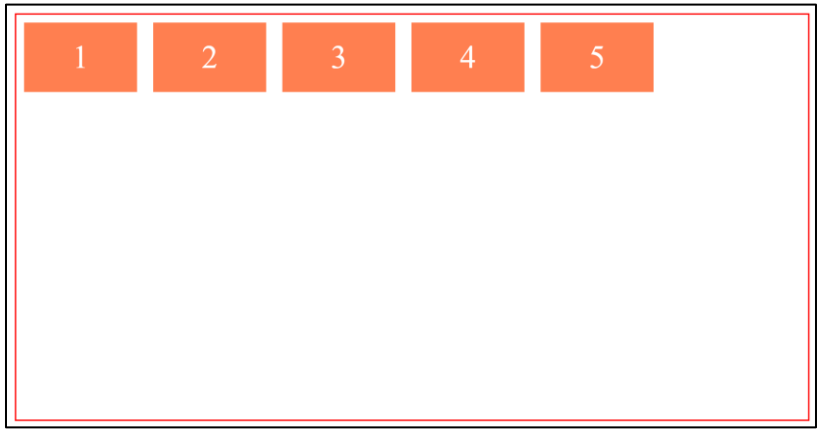
- **stretch** — элементы растягиваются поперек оси и заполняют все свободное пространство по обе стороны от нее;
- **start** — элементы сохраняют свой размер и выравниваются по верхнему или левому краю;
- **center** — элементы группируются по центру. Можно сказать, что их центральная часть «закреплена» на оси;
- **end** — элементы выравниваются по нижнему или правому краю;
- **baseline** — выравнивание происходит по «базовой линии» контента.

```
.flex-container {
    border: 2px solid red;
    display: flex;
    flex-direction: row;

    height: 500px;
    align-items: stretch;
    /* align-items: flex-start; */
    /* align-items: flex-end; */
    /* align-items: center; */
}
```

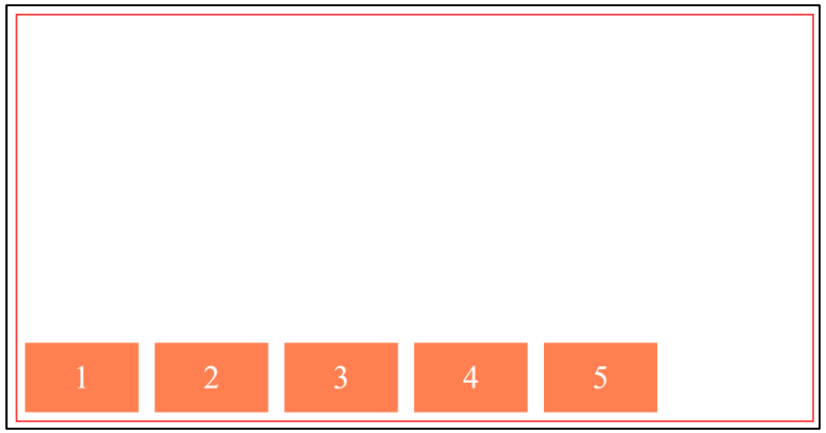
Flexbox



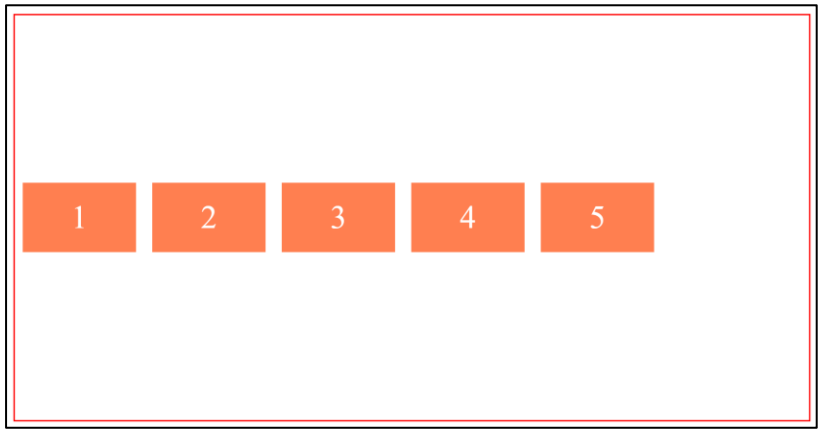
`align-items: stretch;`



`align-items: flex-start;`



`align-items: flex-end;`



`align-items: center;`

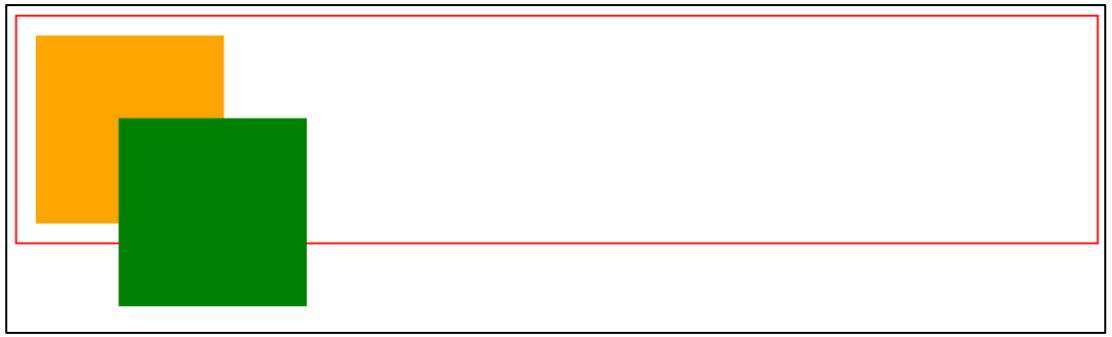
Свойство position

Устанавливает способ позиционирования элемента относительно окна браузера или других объектов на веб-странице.

absolute – указывает, что элемент абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно абсолютно позиционированного элемента и нет. Положение элемента задается свойствами **left**, **top**, **right** и **bottom**, также на положение влияет значение свойства **position** родительского элемента. Так, если у родителя значение **position** установлено как static или родителя нет, то отсчет координат ведется от края окна браузера. Если у родителя значение position задано как **fixed**, **relative** или **absolute**, то отсчет координат ведется от края родительского элемента.

```html
<body>
    <div class="container">
        <div class="blocks-wrapper">
            <div class="block absolute-block"></div>
            <div class="block"></div>
        </div>
    </div>
</body>
```

```css
.block {
    width: 200px;
    height: 200px;
    margin: 20px;
    background-color: orange
}

.absolute-block {
    background-color: green;
    position: absolute;
    left: 100px;
    right: 0;
    top: 100px;
    bottom: 0;
}
```

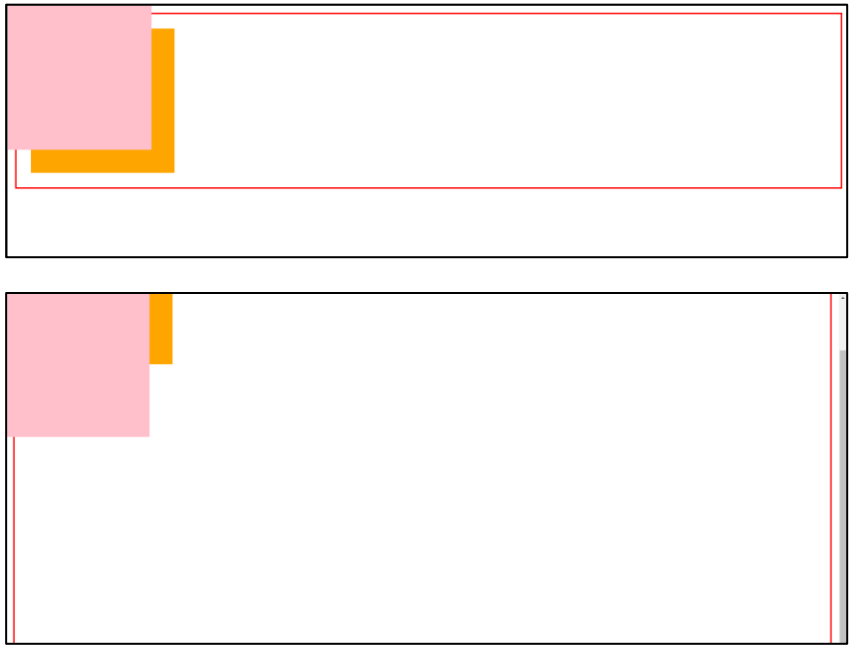


Свойство position

fixed – по своему действию это значение близко к **absolute**, но в отличие от него привязывается к указанной свойствами **left**, **top**, **right** и **bottom** точке на экране и не меняет своего положения при прокрутке веб-страницы. Браузер Firefox вообще не отображает полосы прокрутки, если положение элемента задано фиксированным, и оно не помещается целиком в окно браузера. В браузере Opera хотя и показываются полосы прокрутки, но они никак не влияют на позицию элемента.

```html
<body>
    <div class="container">
        <div class="blocks-wrapper">
            <div class="block fixed-block"></div>
            <div class="block"></div>
        </div>
    </div>
</body>
```

```css
.block {
    width: 200px;
    height: 200px;
    margin: 20px;
    background-color: orange
}

.fixed-block {
    background-color: pink;
    position: fixed;
    left: 0;
    right: 0;
    top: 0;
    bottom: 0;
    margin: 0;
}
```



Свойство position

relative – положение элемента устанавливается относительно его исходного места. Добавление свойств **left**, **top**, **right** и **bottom** изменяет позицию элемента и сдвигает его в ту или иную сторону от первоначального расположения.

```html
<body>
    <div class="container">
        <div class="blocks-wrapper">
            <div class="block relative-block"></div>
            <div class="block"></div>
        </div>
    </div>
</body>
```

```css
.block {
    width: 200px;
    height: 200px;
    margin: 20px;
    background-color: orange
}

.relative-block {
    background-color: blue;
    position: relative;
    left: 100px;
    right: 100px;
    top: 100px;
    bottom: 100px;
}
```

Свойство position

static – элементы отображаются как обычно. Использование свойств **left**, **top**, **right** и **bottom** не приводит к каким-либо результатам.

```html
<body>
    <div class="container">
        <div class="blocks-wrapper">
            <div class="block static-block"></div>
            <div class="block"></div>
        </div>
    </div>
</body>
```

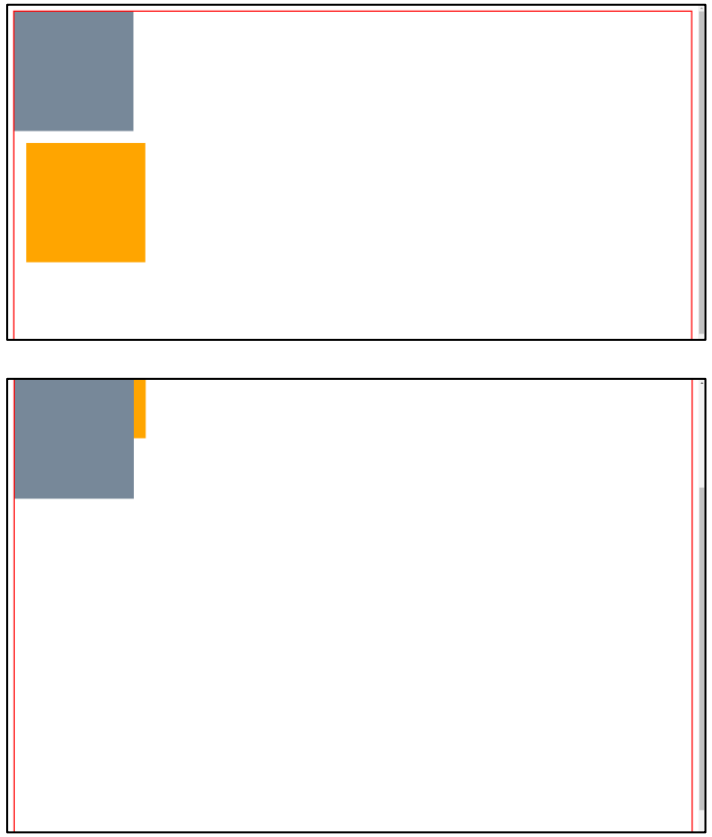
```css
.block {
    width: 200px;
    height: 200px;
    margin: 20px;
    background-color: orange
}

.static-block {
    background-color: red;
    position: static;
    left: 100px;
    right: 100px;
    top: 100px;
    bottom: 100px;
}
```

Свойство position

sticky –элемента «прилипает» к экрану при прокрутке, пока не встретит границу родительского блока. Для корректной работы необходимо указать положение «прилипающего» элемента относительно окна просмотра с помощью свойств **top**, **right**, **bottom** или **left**. Например, **top: 0;** означает, что элемент будет «прилипать» к верхней части окна просмотра

```html
<body>
    <div class="container">
        <div class="blocks-wrapper">
            <div class="block sticky-block"></div>
            <div class="block"></div>
        </div>
    </div>
</body>
```

```css
.block {
    width: 200px;
    height: 200px;
    margin: 20px;
    background-color: orange
}

.sticky-block {
    background-color: lightslategray;
    position: sticky;
    left: 0;
    right: 0;
    top: 0;
    bottom: 0;
    margin: 0;
}
```



Спасибо за внимание