

HyperText Markup Language

HTML & CSS



Урок 6.3

Позиционирование элементов

Оглавление

Позиционирование элементов.....	4
Относительное позиционирование	6
Абсолютное позиционирование.....	10
Фиксированное позиционирование	13
Липкое позиционирование	16
Использование свойства z-index.....	18
Верстка страницы с использованием свойства position	22
Общие классы и разметка.....	24
Пример блоков с относительным позиционированием	26
Пример с абсолютным позиционированием.....	32
Фиксированное позиционирование меню сайта.....	41
Использование липкого позиционирования.....	45

Использование позиционирования при наведении курсора мыши.....	48
Домашнее задание.....	54
Задание 1	54
Задание 2	55

Материалы урока прикреплены к данному PDF-файлу. Для доступа к материалам, урок необходимо открыть в программе Adobe Acrobat Reader.

Позиционирование элементов

Под позиционированием элементов подразумевается определение их свойства **position**, которое может иметь такие значения:

```
position: static | relative | absolute | fixed |  
sticky | inherit | initial
```

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Molestiae optio
consectetur quidem, aliquam error quo quam a saepe eligendi qui. Quos asperiores
sit dolorum, atque!

Iure modi rem, deserunt cum est quia fugit vel ullam blanditiis obcaecati aut
repellendus reprehenderit doloremque quibusdam magni laudantium rerum, saepe
consectetur. Suscipit, dolore, omnis.

Quidem, nulla alias libero tenetur distinctio mollitia explicabo repudiandae saepe
similique, porro consequatur itaque blanditiis qui vitae repellat! Illum repellendus
magnum, sed hic maxime dolor.

Eveniet hic dicta odit ex iusto cumque fugit nisi omnis vero cupiditate a eos rerum,
sunt incident fugiat pariatur vitae repudiandae illum dolore at ab.

Ea, doloremque ducimus. Neque repellat iusto, sapiente optio quos quam corporis
nam, dolor porro eaque asperiores adipisci quaerat fugit autem aut, nihil velit quae
quis.

Expedita ab minima neque, voluptate. Commodi obcaecati cumque eos minus velit
accusantium vel! Aperiam a ducimus, delectus voluptates beatae iste maiores sint
ab hic nostrum.

Рисунок 1

По умолчанию все элементы имеют статическую позицию, т.е. **position: static**, и располагаются друг под другом (имеются в виду блочные элементы). Это расположение называют нормальным потоком (рис. 1).

Как только вы меняете значение **position** на любое другое, отличное от **static**, элементы вырываются из нормального потока, как бы всплывая выше него, и ведут себя несколько иначе, чем обычно. Всплытие подразумевает, что позиционированные элементы находятся выше, чем статические (рис 2).

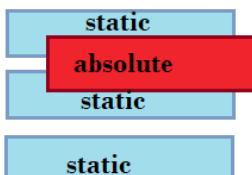


Рисунок 2

У каждого значения свойства **position** есть свои особенности, с которыми мы будем разбираться в этом уроке.

К общим особенностям следует отнести то, что любому позиционированному элементу можно назначить координаты **left**, **top**, **right**, **bottom**, которые показывают расположение его левого верхнего угла или правого нижнего. Вы можете сочетать их в любом порядке.

Необязательно задавать все 4 координаты. Можно назначить любые 2. Чаще всего это **left** и **top**, т.к. это точка начала координат любого элемента в HTML/CSS, но бывает необходимость задать **right** и **top**, **left** и **bottom** или какую-то одну, например, только **top** или только **bottom**. Для всех координат значения можно указывать

в `px`, `em`, `rem`, `%`, `vw`, `vh` и т.п., причем как положительные, так и отрицательные.

К общим свойствам для позиционированных элементов также относится `z-index`, который имеет значение в виде целого числа или `auto`.

Относительное позиционирование

Относительно позиционированными элементами являются те, которым назначено свойство `position: relative`. Визуально эти элементы ничем не отличаются от статических. Однако они занимают позицию выше тех статических элементов, которые расположены в html-разметке выше или ниже относительно позиционированных. Это становится сразу видно, если задать для таких элементов координаты. **Относительное позиционирование** — это такой вариант поведения элементов, когда их можно сместить выше/ниже и правее/левее *относительно их текущего статического положения* именно за счет координат `bottom/top` и `left/right`. То есть относительное позиционирование элемента рассматривается относительно него самого.

В папке `examples` вы найдете пример `position.html`, в котором с помощью переключателей можно назначить выделенному элементу любой вид позиционирования (рис. 3).

Выделим кликом левой кнопкой мыши второй сверху элемент и установим для него `position: relative`, щелкнув по соответствующему переключателю. Стили будут прописаны с помощью JavaScript в самом элементе.

На скриншоте видно, что элемент изменился внешне, но остался на своем прежнем месте (рис. 4).

Позиционирование элементов

position: static relative absolute fixed sticky
coords: left top right bottom

Test CSS position

 Lorem ipsum dolor sit amet, consectetur adipisicing elit. Molestiae optio
 consectetur quidem, aliquam error quo quam a saepe eligendi qui. Quos asperiores
 sit dolorum, atque!

 lure modi rem, deserunt cum est quia fugit vel ullam blanditiis obcaecati aut
 repellendus reprehenderit doloremque quibusdam magni laudantium rerum, saepe
 consectetur. Suscipit, dolore, omnis.

 Quidem, nulla alias libero tenetur distinctio mollitia explicabo repudiandae saepe
 similique, porro consequatur itaque blanditiis qui vitae repellat! Illum repellendus
 magnam, sed hic maxime dolor.

Рисунок 3

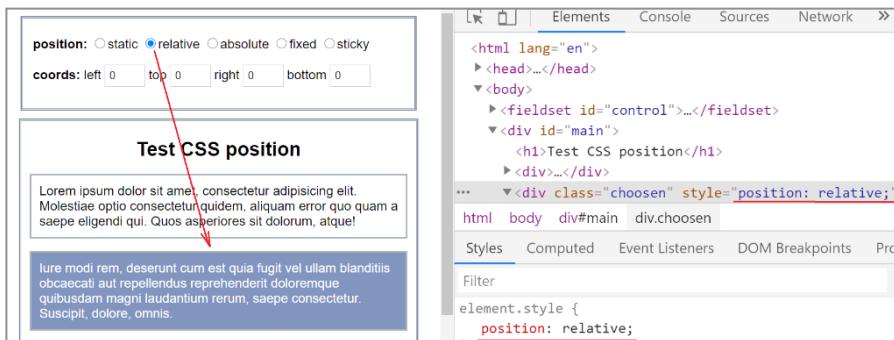


Рисунок 4

Теперь изменим координаты, задав для **left** и **top** отрицательные значения. Выделенный элемент съехал вверх и влево, оставив пустое место там, где он располагался до этого (рис. 5).

Урок 6.3. Позиционирование элементов

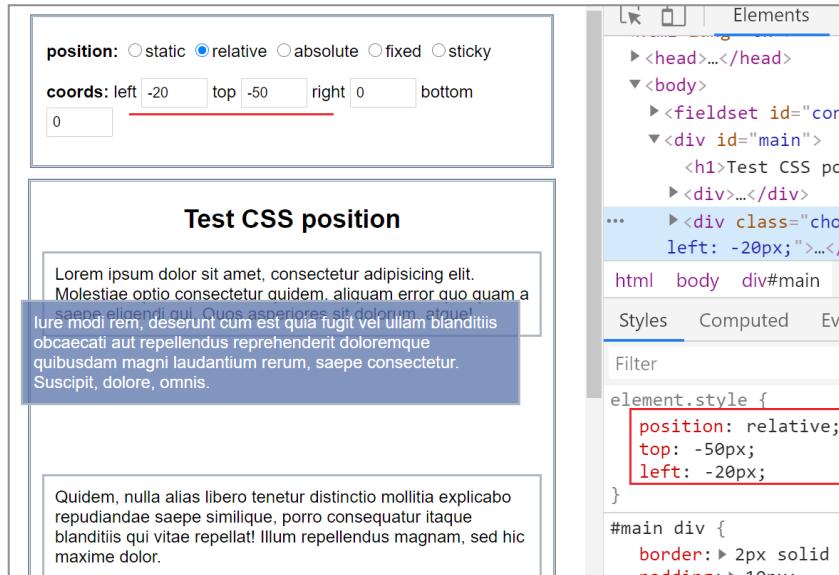


Рисунок 5

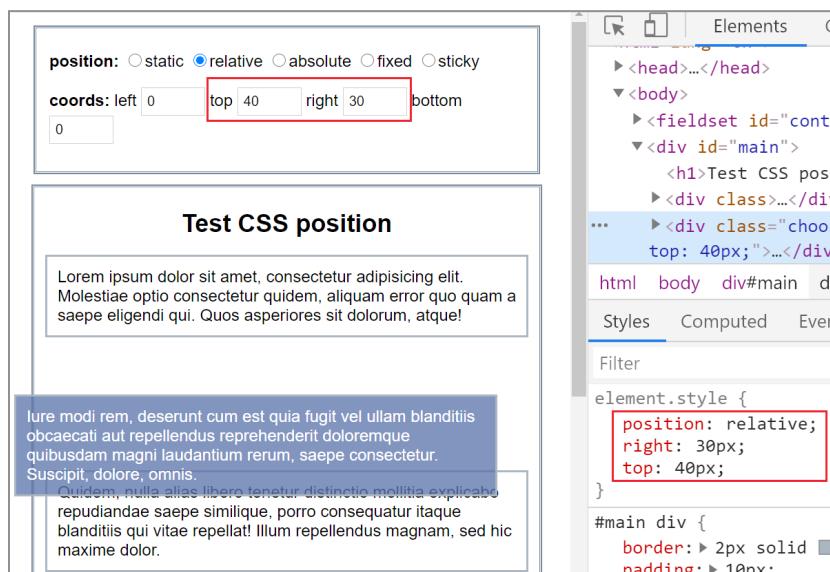


Рисунок 6

Попробуем положительные координаты для `top` и `right`: ситуация подобна. Разница состоит в том, что положительное значение координаты `top` сместило элемент вниз, а положительное значение `right` — влево (аналогично отрицательному значению `left`) (рис. 6).

Смещения происходят относительно системы координат браузера и любого элемента, расположенного внутри него. Дело в том, что начало координат окна браузера лежит в верхнем левом углу. Положительное направление оси `X` — вправо, оси `Y` — вниз. Такая же система координат характерна для каждого html-элемента. Поэтому положительные значения координаты `left` смещают позиционированный элемент вправо, вдоль положительного направления оси `X`, отрицательные — влево. Положительные значения координаты `right`, напротив, смещают элемент влево, а отрицательные вправо, т.к. отсчитываются от правой границы элемента.

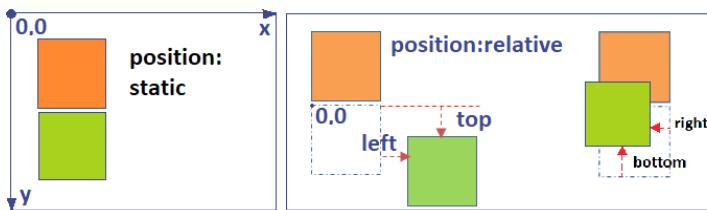


Рисунок 7

Вертикальная координата `top` с положительными значениями будет смещать элемент вниз, вдоль положительного направления оси `Y`, с отрицательными — вверх. Вертикальная координата `bottom` сдвинет элемент вверх с положительным значением и вниз — с отрицательным, т.к. отсчитывается от нижней границы элемента.

Обратите внимание на то, что если задать все координаты для относительно позиционированного элемента, то приоритет будут иметь **left** и **top**, а не **right** и **bottom**. Попробуйте сами.

Абсолютное позиционирование

Элемент с **position: absolute** называется абсолютно позиционированным. Абсолютность его расположения объясняется тем, что при отсутствии у его родительского элемента свойства **position**, отличного от **static**, он размещается относительно окна браузера, т.е. абсолютно оторвавшись от своего прежнего местоположения в html-разметке.

Абсолютное позиционирование отличается от относительного достаточно серьезно, хотя на первый взгляд вы можете ничего особенного не заметить. **Отличия** абсолютно позиционированного элемента от относительно позиционированного и от статического:

1. Элемент вырвался со своего места в нормальном потоке и всплыл выше других элементов.
2. Элемент, который в html-разметке следовал за абсолютно позиционированным, подтянулся на его место.
3. Позиционированный элемент вытянулся по ширине до края **body** справа, хотя слева остался привязанным к той точке, в которой он располагался внутри родительского элемента в нормальном потоке. Если бы текста в элементе было мало, его ширина сократилась бы до размеров контента + padding + border.

Это еще не все отличия. Новые добавятся после установки координат.

Позиционирование элементов

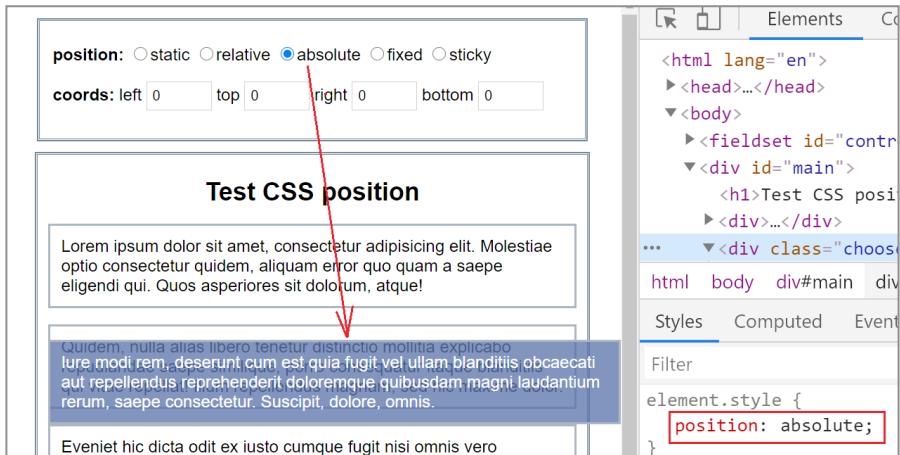


Рисунок 8

Добавьте координаты. Чаще всего используют положительные **left** и **top**. Увидим такие отличия:

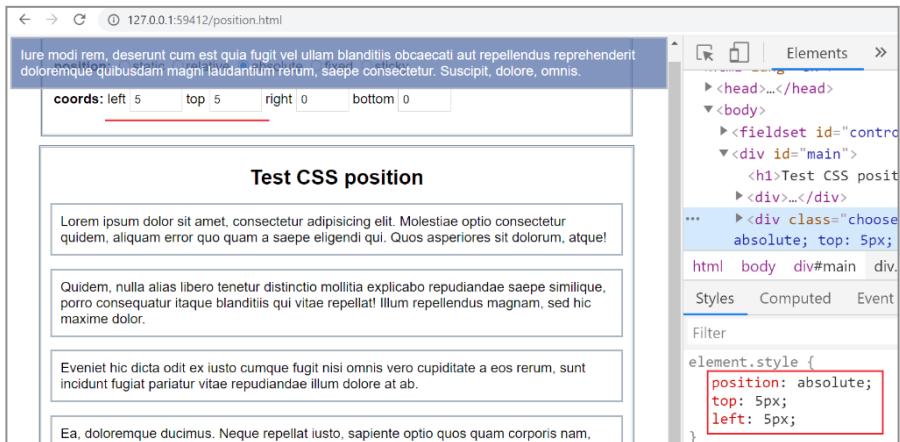


Рисунок 9

1. Элемент оторвался от своего родителя и разместился вверху документа с отступом в **5px** от его левой и верхней границы (**left: 5px; top: 5px**).

2. Ширина элемента зависит от его контента, однако при большом количестве текста она расширяется на все доступное пространство внутри **body**.
3. Еще одно отличие связано с назначением всех координат: в этом случае элемент может растянуться на экране по ширине за счет того, что **left** и **top** формируют отступы слева и сверху, а **right** и **bottom** — справа и снизу от краев html-документа.

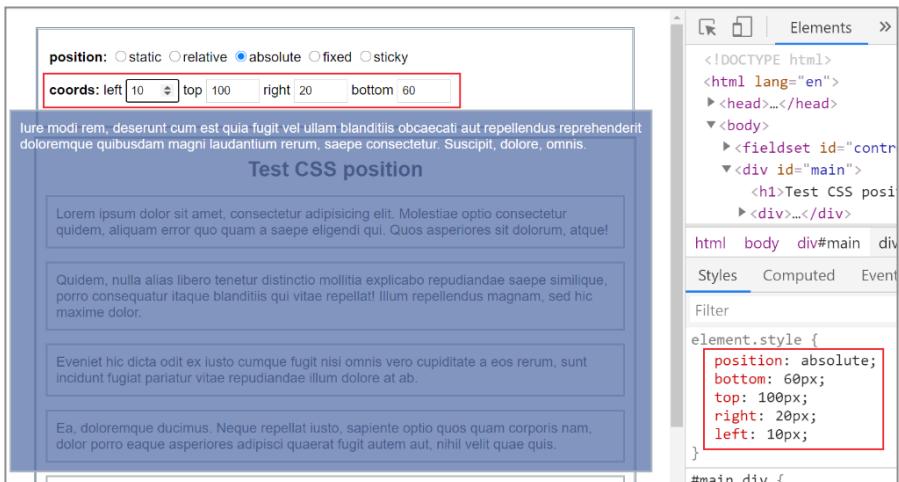


Рисунок 10

На этом еще не заканчиваются интересные факты об абсолютном позиционировании. Возможно, вы уже задались вопросом о том, а что делать, если нужно разместить абсолютно позиционированный элемент относительно его родителя? Неужели нужно высчитывать координаты относительно верха страницы? **Выход** очень прост — для родительского элемента нужно назначить позиционирование, отличное от **static**. То

есть можно использовать `position: relative`, `absolute`, `fixed` или `sticky`. На практике обычно используют `position: relative`, т.к. при относительном позиционировании без координат элемент визуально никак не отличается от нормального потока.

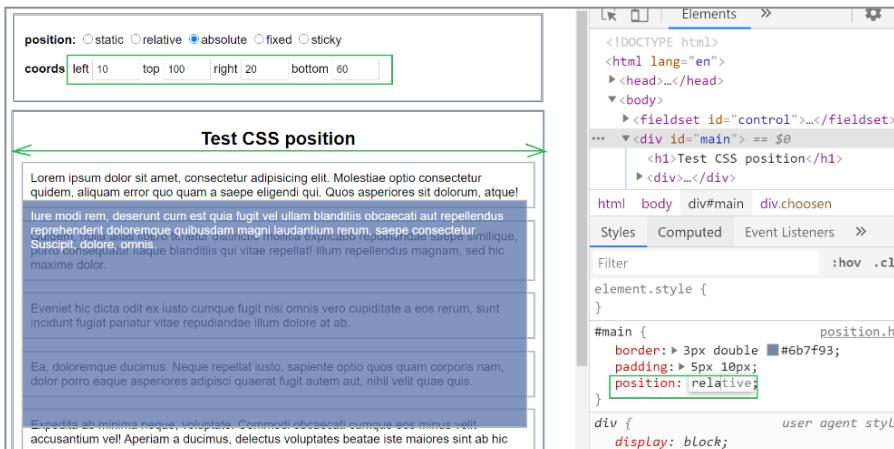


Рисунок 11

Для того чтобы увидеть эффект возвращения «блудного» абсолютно позиционированного элемента в координаты и пределы его родителя, вам будет необходимо самостоятельно написать `position: relative` в Инспекторе свойств вашего браузера для элемента с `id="main"`.

Фиксированное позиционирование

Фиксированное позиционирование в соответствии со своим названием фиксирует элемент на странице в определенном месте вне зависимости от прокрутки остального контента. Фиксированное позиционирование **похоже** по своим свойствам на абсолютное:

- Элемент вырывается со своего места в нормальном потоке и всплывает выше других элементов.
- Элемент, который в html-разметке следовал за фиксированным элементом, подтягивается на его место.
- Фиксированный элемент вытягивается по ширине до края **body** справа, оставаясь привязанным слева к границе родительского элемента. Если текста в элементе мало, его ширина сокращается до размеров контента + padding + border.

Отличия:

- Фиксированный элемент закрепляется в той точке, в которой он находится в родительском контейнере или в точке с заданными координатами, и при прокручивании остается на месте. Абсолютно позиционированный элемент прокручивается вместе с родителем или с **body**.

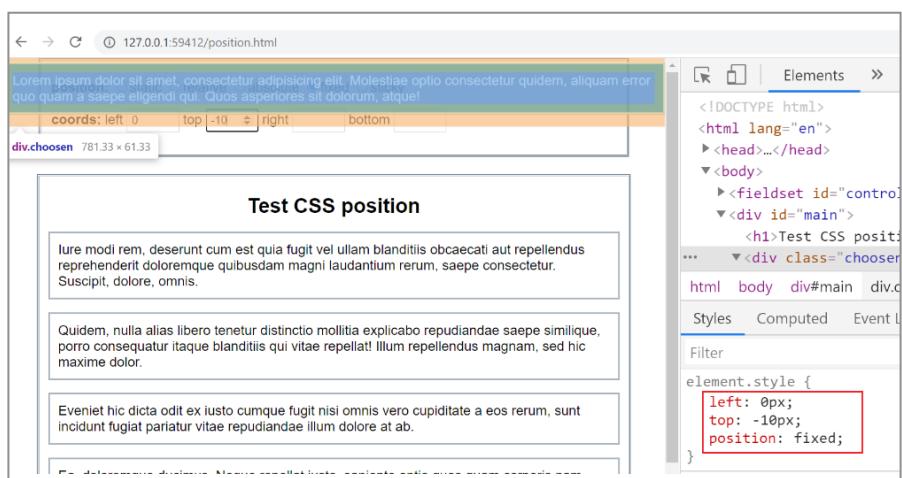


Рисунок 12

2. Если задать координаты элементу с **position:fixed**, он отрывается от родителя и постоянно находится на экране в одной точке. Этим пользуются для создания прилипшего к верху или к низу меню (рис. 12). При назначении координат важно учитывать, заданы ли внешние отступы (**margin**) для фиксированного элемента, т.к. он может быть смещен относительно заданной координаты вниз/вверх и вправо/влево, как на рис. 12.
3. Вернуть «убежавший» фиксированный элемент в пределы родителя не удастся, если вы задали ему координаты. Они будут рассчитываться относительно html-документа.

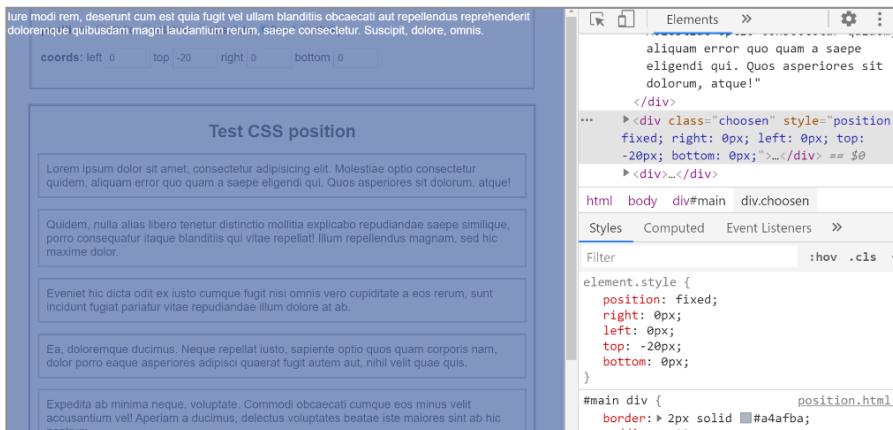


Рисунок 13

4. Если задать все координаты для фиксированного элемента, равными 0, он растягивается на весь экран и будет перекрывать контент, прокручиваемый под ним (рис. 13). Таким способом часто пользуются для

создания оверлеев — полупрозрачных слоев для-modalных окон, слайдеров, баннеров.

Липкое позиционирование

«Липкое позиционирование» — именно так переводится свойство `position: sticky` с английского. Это свойство поддерживается браузерами относительно недавно, поэтому для понимания того, будет ли ваш браузер правильно интерпретировать это свойство, нужно заглянуть на caniuse.com. Обратите внимание, что пока еще это свойство поддерживается частично и с некоторыми багами в отношении ряда элементов. Полная поддержка — это вопрос времени.

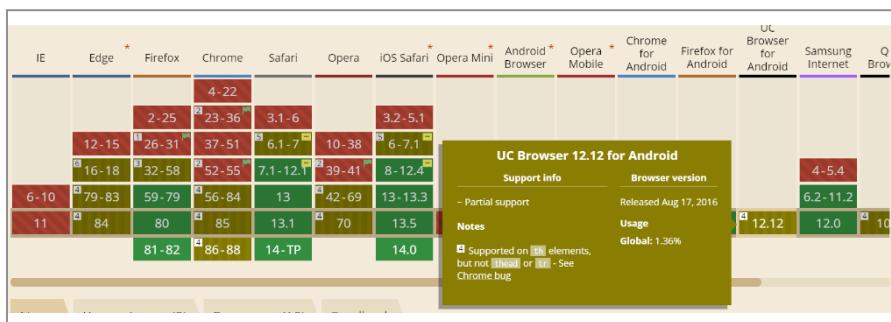


Рисунок 14

Суть работы `position: sticky` заключается в том, что элемент с таким свойством визуально ничем не отличается от соседей, как и относительно позиционированный, до тех пор, пока вы не начнете прокручивать страницу или элемент, в котором он находится. В этом случае липкий элемент закрепляется в той координате, которая указана в его свойствах, и не смешается с остальным контентом.

Для прокрутки по вертикали таким элементам указывают координату **top** или **bottom**, а для горизонтальной прокрутки — **left** или **right**. Если координату не указывать, то залипания вы не увидите. Здесь поведение sticky-элемента похоже на относительное позиционирование.

В примере ниже самый последний элемент прилип к нижнему краю за счет свойств **position: sticky** и **bottom:0**.

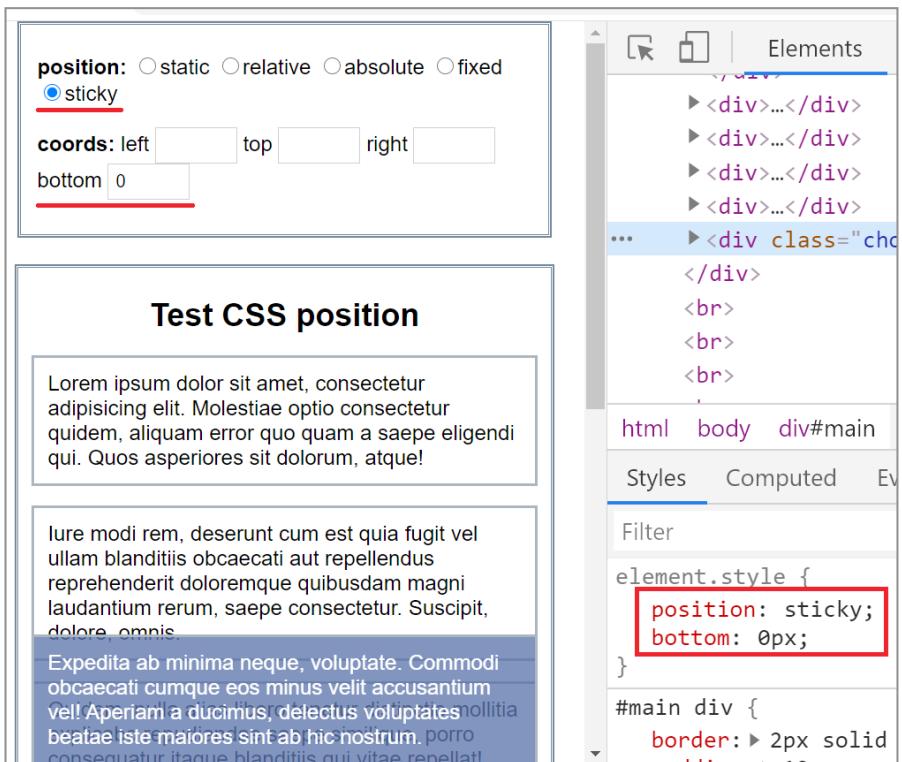


Рисунок 15

При прокручивании документа вверх этот элемент «отлипнет» от нижней границы, когда прокрутка дойдет до самого низа (рис. 16).

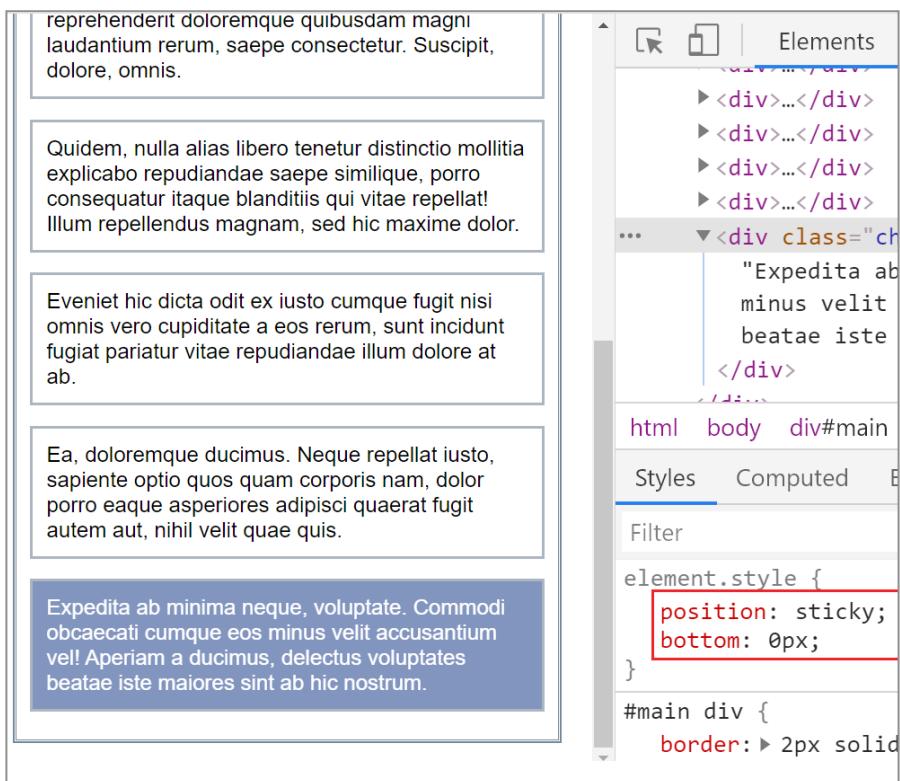


Рисунок 16

Попробуйте самостоятельно прикрепить верхний элемент к верхней границе.

Использование свойства z-index

Свойство **z-index** позволяет манипулировать позиционированными элементами в 3-м измерении (**z**), перпендикулярном экрану. Поскольку все позиционированные элементы всплывают над статическими и могут накладываться друг на друга, т.е. перекрывать частично или полностью другой элемент, при назначении координат, необходим

инструмент управления этими наложениями, которым и является **z-index**. В файле *z-index-1.html* вы найдете 4 абсолютно позиционированных элемента, расположение которых друг относительно друга по оси, перпендикулярной экрану, поменяло использование свойства **z-index**.

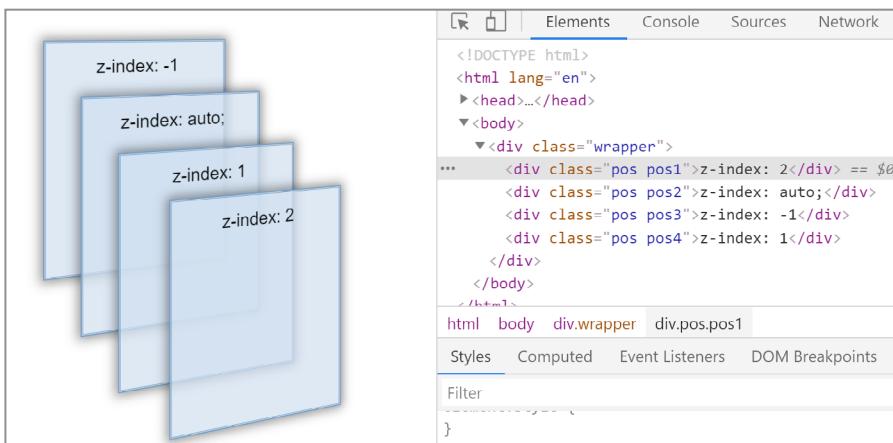


Рисунок 17

Свойство **z-index** может иметь такие значения:

```
z-index: число | auto | inherit
```

По умолчанию используется значение **auto**, которое позволяет накладываться друг на друга позиционированным элементам, идущим друг за другом в разметке. Таким образом самый нижний элемент в html-разметке всегда будет находиться выше, чем те, что были в разметке выше него.

Задавать значение **z-index** можно в виде положительных, отрицательных значений и нуля. Зависимость такова: чем больше значение **z-index**, тем выше расположено элемент.

ложен элемент по сравнению с теми позиционированными элементами, у которых это свойство имеет меньшее значение. Если у нескольких элементов значения **z-index** одинаковы, выше будет находиться тот элемент, который в html-разметке описан ниже.

Рассмотрим ситуацию, когда у нас есть ряд абсолютно позиционированных объектов, которые за счет свойств **top** и **left** накладываются друг на друга. Разметка примера такова:

```
<div class="wrapper">
    
    
    
    
</div>
```

Стили:

```
.pos { position: absolute;
    border: 3px double #0c800c;
    border-radius: 8px;
    padding: 2px;
    background-color: #fff;
    box-shadow: 0 0 12px 3px rgba(0, 0, 0, 0.45);
    cursor: pointer; }

.pos1 { left: 20%; top: 60px; }
.pos2 { left: 28%; top: 130px; }
.pos3 { left: 50%; top: 20px; }
.pos4 { left: 43%; top: 210px; }
```

Вот, что получилось:



Рисунок 18

На рисунке пронумерованы изображения в том порядке, в котором они размещены в разметке. За счет того, что свойство `z-index` негласно присутствует и по умолчанию имеет значение `auto`, каждое следующее абсолютно позиционированное изображение находится выше предыдущего и накладывается на него. Для того чтобы изменить эту последовательность, достаточно назначить `z-index: 1` при наведении на любое изображение:

```
.pos:hover { z-index: 1; }
```

Теперь при наведении изображение под курсором мыши выходит наверх.



Рисунок 19

Этим свойством можно пользоваться для того, чтобы вывести элемент выше других. Особенно это касается меню или модальных окон, которые должны быть выше других позиционированных элементов на странице.

Вы можете назначать **z-index** последовательно, указывая для нужных элементов значение 1, 2, 3, 4... или перескакивать через 5-10 единиц: 10, 20, 30, 40 и т.д. Если вы правите чужую разметку, желательно сначала узнать, какой **z-index** задан элементу, который надо перекрыть.

Верстка страницы с использованием свойства position

Для того чтобы познакомиться с различными вариантами позиционирования элементов на практике, мы рассмотрим пример страницы, содержащий все их виды.

Рабочий пример вы найдете в папке *examples/site*. Если вы захотите посмотреть на все нюансы самостоятельно,

Позиционирование элементов

создайте новый пустой css-файл и подключите его к *index.html* вместо *style.css*, а затем последовательно добавляйте скопированный в уроке код.

The screenshot displays a website layout for 'The Best Camp'. At the top, there's a navigation bar with links to 'Our Camp', 'Our Teams', 'Feedback', and 'Our Program'. Below the header is a large image of a tent interior looking out onto a forest. Overlaid on the image is the text 'Welcome to The Best Camp'. The main content area starts with a section titled 'Our Camp' featuring a short paragraph of placeholder text. Following this is a section titled 'Our Teams' containing four cards labeled A, B, C, and D, each with a title, icon, and a short description. Below 'Our Teams' is a section titled 'Feedback from our clients' with three testimonial cards. Each card includes a quote, the client's name, and a small profile picture. The bottom of the page features a footer with a 'Test Camp' section containing more placeholder text and a photo of a person raising their arms in a field of tents. The footer also includes the text 'The Best Camp is your choice in 2021'.

Рисунок 20

Общие классы и разметка

В нашем примере не все разделы и блоки будут содержать какой-либо вид позиционирования. Это и не нужно на странице, т.к. стандартный вариант размещения блоков в нормальном потоке используется не зря. Именно так организована структура книги — вы читаете абзац за абзацем внутри разделов-глав с заголовками, временами рассматривая иллюстрации.

Такому правилу подчиняется `<section id="about">`, равно как и другие разделы.

Схема html-разметки у нас будет такой:

```
<header> ... </header>
<main class="wrap">
    <section id="about">
        <h2>Our Camp</h2>
        <p>Lorem ipsum dolor sit amet, consectetur
            adipisicing elit. Repellendus vero minima,
            accusamus officiis perferendis hic soluta
            voluptas iure atque ipsum quo quisquam
            labore, nemo aliquam odit officia facere
            quae eos nisi alias quasi impedit et
            ullam. Mollitia harum asperiores
            tempora.</p>
        <p>Vitae ex tempora ipsa similique
            repudiandae unde autem necessitatibus,
            ipsam ipsum cum eos magni at deleniti
            recusandae, esse libero et eaque harum
            quas! Quae cumque itaque repudiandae
            reiciendis aperiam facere dolor
            consequuntur, reprehenderit nemo minus
            excepturi culpa debitis et quibusdam!</p>
    </section>
    <section id="teams"> ... </section>
```

```
<section id="reviews"> ... </section>
<section id="reviews"> ... </section>
<section id="program">... </section>
<footer>The Best Camp is your choice in 2021</footer>
</main>
```

Все разделы будут находиться у нас в элементе **main**, и только шапка сайта будет размещена отдельно.

Общие стили для элементов страницы представлены ниже:

```
*, *::before, *::after { box-sizing: border-box; }

body { margin: 0;
       font-family: sans-serif;
}
section { padding-bottom: 60px;
          margin-left: 5%;
          margin-right: 5%;
}
h2 { font-size: 60px;
      font-weight: 800;
      letter-spacing: 2px;
      text-decoration: none;
      color: #4caf50;
}
footer {
      width: 50%;
      margin: 20px auto 0;
      text-align: center;
      font-size: 1.2em;
      padding: 20px 0;
      border-top: 1px solid #ccc;
}
```

Внешне начальный вид нашей страницы будет таким:

Our Camp

Лorem ipsum dolor sit amet, consectetur adipisicing elit. Repellendus vero minima, accusamus officiis preferendis hic soluta voluptas iure atque ipsum quo quisquam labore, nemo aliquam edit officia facere quao eos nisi alias quasi impedit et ullam. Molitia harum asperiores tempora.

Vitae ex tempora ipsa similiique repudiandae unde autem necessitatibus, ipsam ipsum cum eos magni at dolenti recusandae, esse libero et eaque harum quas! Quae cumque itaque repudiandae reiciendis aperiam facere dolor consequuntur, reprehenderit nemo minus excepturi culpa debitis et quibusdam!

The Best Camp is your choice in 2021

Рисунок 21

Пример блоков с относительным позиционированием

Начнем со свойства **position: relative** и блока **Feedback from our clients**, т.к. именно в нем мы будем использовать относительное позиционирование для фото авторов комментариев. Разметка блока:

```
<section id="reviews">
    <h2>Feedback from our clients</h2>
    <div class="review">
        <p>Lorem ipsum dolor sit amet, consectetur
            adipisicing elit. Laboriosam ad, quas
            similiique earum quo iusto natus!</p>
        <p>Consequatur quae, numquam sint animi neque
            sequi tempore, eum facere modi rerum
            labore quod voluptate nobis. Fuga non qui
            laudantium?</p>
        <div class="author">Ben Kelan</div>
        <div class="author-img">
            
        </div>
    </div>
    <div class="review">
```

```
<p>Lorem ipsum dolor sit amet, consectetur  
adipisicing elit. Incidunt officia ratione  
quasi tenetur ut fugit voluptas, placeat  
ullam.</p>  
<p>Maxime nemo odit, voluptatem, quia  
quisquam expedita, itaque libero atque  
totam incident neque praesentium sint.  
Eos at cupiditate dicta.</p>  
<div class="author">Justin Blum</div>  
<div class="author-img">  
      
</div>  
</div>  
<div class="review">  
    <p>Lorem ipsum dolor sit amet, consectetur  
        elit. Sunt atque enim at illum doloribus  
        libero reprehenderit ipsam. </p>  
    <p>Hic voluptas esse tenetur, voluptatibus  
        debitis perferendis magnam, necessitatibus  
        eligendi. Placeat officiis, quidem nostrum  
        laboriosam!</p>  
    <div class="author">Rita Oswald</div>  
    <div class="author-img">  
          
    </div>  
</div>  
</div>
```

Стили для этого блока мы будем записывать, создавая 3 колонки с помощью свойства `display: inline-block`.

```
.review { display: inline-block;  
    width: 31%;  
    margin-right: 3%;
```

```
background: #fff;
box-shadow: 0 2px 2px 0 rgba(0, 0, 0, 0.14),
            0 3px 1px -2px rgba(0, 0, 0, 0.2),
            0 1px 5px 0 rgba(0, 0, 0, 0.12);
margin-bottom: 30px;
border-radius: 6px;
border: 1px solid #ccc;
padding: 30px 45px 0;
text-align: center;
}

.review:last-of-type { margin-right: 0; }

.review:before { font-size: 24px;
    content: url(..../images/icons/quotes.svg);
}

.author { font-weight: bold; }

.author-img { width: 100px;
    overflow: hidden;
    border-radius: 50%;
    margin: auto;
    box-shadow: 0 16px 38px -12px rgba(0, 0, 0, 0.56),
                0 4px 25px 0px rgba(0, 0, 0, 0.12),
                0 8px 10px -5px rgba(0, 0, 0, 0.2);
}

.author-img img { display: block;
    width: 100%;
    height: auto;
}
```

Собственно, больше всего нас интересует класс **.author-img**. Сейчас нижняя часть отзыва выглядит так, что круглое изображение с автором расположено сразу за его именем.

“

“

“

“

“

“

Рисунок 22

Добавим для `.author-img` свойство `position: relative`, после чего увидим, что ничего не изменилось. Происходит это потому, что без добавления координат наш относительно позиционированный блок никак себя не выдает, оставаясь в том же месте, что и при статической позиции.

Однако стоит задать координату `top: 40px` — и ситуация меняется: `div` с изображением смещается вниз, заслоняя собой рамку родительского элемента.

“

“

“

“

“

“

Рисунок 23

Теперь стили для `.author-img` выглядят так:

```
.author-img {  
    width: 100px;  
    overflow: hidden;  
    border-radius: 50%;  
    margin: auto;  
    position: relative;  
    top: 40px;  
    box-shadow: 0 16px 38px -12px rgba(0, 0, 0, 0.56),  
                0 4px 25px 0px rgba(0, 0, 0, 0.12),  
                0 8px 10px -5px rgba(0, 0, 0, 0.2);  
}
```

Примерно такой же эффект можно получить, задав отрицательный внешний отступ снизу для `.author-img`, то есть вместо закомментированных строк с позиционированием и координатой `top` мы добавляем свойство `margin-bottom: -40px`:

```
.author-img { ...;  
    margin: auto;  
    /* position: relative;  
       top: 40px;  
    */  
    margin-bottom: -40px;  
}
```

Тем не менее, разница есть: визуально блок уменьшился, т.к. отрицательный `margin-bottom` сместил нижний край изображения вниз, но у нас не добавился отступ

от имени автора отзыва так, как это произошло в случае с **position: relative** и **top: 40px**.



Рисунок 24

Такой эффект связан с особенностью относительного позиционирования — блок с **position: relative** при указании координат смещается со своего места, «наползая» на блоки ниже/выше (положительное/отрицательное значение в **top**) или справа/слева (положительное/отрицательное значение в **left**). Однако то место, которое он занимал ранее, остается пустым.

Для того чтобы повторить эффект от **position: relative** и **top: 40px** с отступами, нужно добавить не только отрицательный нижний отступ, но и такой же положительный верхний.

```
.author-img { ...;
    margin-bottom: -40px;
    margin-top: 40px;
}
```

Теперь вид блоков не отличается от того, что был с относительным позиционированием.

Урок 6.3. Позиционирование элементов



Рисунок 25

Пример с абсолютным позиционированием

Для форматирования раздела **Our teams** мы используем абсолютное позиционирование для круглых блоков с буквами A-D. Они смещены влево по отношению к темно-серому фону родительского блока. Конечный результат вы видите на скриншоте ниже.



Рисунок 26

Для начала посмотрим на разметку одного из 4-х блоков с классом **.team**:

```
<section id="teams">
  <h2>Our Teams</h2>
  <div class="team">
```

```
<div class="circle">A</div>
<div class="circle-back"></div>

<h3>Winners</h3>
<p>Lorem ipsum dolor sit amet, consectetur
    adipisicing elit. Sunt eligendi, molestiae
    optio consequuntur voluptas.</p>
</div>
...
</section>
```

Теперь запишем стили для всех элементов, которые есть в нашей разметке, но без использования `position: absolute`.

```
.team {
    display: inline-block;
    width: 22.5%;
    margin-right: 2%;
    background-color: #4e4e4e;
    color: #fff;
    padding: 40px;
    vertical-align: top;
    border-radius: 15px;
    box-shadow: 0 16px 24px 2px rgba(0, 0, 0, 0.14),
                0 6px 30px 5px rgba(0, 0, 0, 0.12),
                0 8px 10px -5px rgba(0, 0, 0, 0.2);
}
.team:last-child { margin-right: 0; }

.team h3 {    text-align: center;
    margin: 10px 40px 50px 20px;
}
```

Урок 6.3. Позиционирование элементов

```
.icon {    float: right;    margin-right: -15px;}
```

```
.circle {    width: 60px;    height: 60px;    border: 2px solid #ccc;    border-radius: 50%;    text-align: center;    color: #333;    font-weight: bold;    font-size: 1.4rem;    padding-top: 15px;    background-image: linear-gradient(#ccc, #fff);}
```

```
}
```

```
.circle-back {    width: 70px;    height: 70px;    border-radius: 50%;    background-image: linear-gradient(#ff0, #ffba00);    box-shadow: 2px 4px 2px 2px rgba(0, 0, 0, 0.14);}
```

```
}
```

```
.team:nth-of-type(2) .circle-back {    background-image: linear-gradient(#ff7c7c, #db3811);}
```

```
}
```

```
.team:nth-of-type(3) .circle-back {    background-image: linear-gradient(#21e6d1, #08a392);}
```

```
}
```

```
.team:nth-of-type(4) .circle-back {    background-image: linear-gradient(#be75f4, #7d1fc3);}
```

```
}
```

После применения стилей мы получили такой вариант:



Рисунок 27

Добавим теперь абсолютное позиционирование для классов `.circle` и `.circle-back`. Мы получили наложение цветных кругов, которыми являются div-ы с классом `.circle-back` на размещенные выше в разметке div-ы с классом `.circle`.

Кроме того, абсолютно позиционированные круги закрыли собой текст заголовка, который в разметке идет ниже `<div class="circle-back"></div>`.



Рисунок 28

Для того чтобы переместить цветные круги под светлые, воспользуемся свойством `z-index`. Поскольку по умолчанию оно имеет значение `auto`, то нам достаточно увеличить его всего до `1`.

```
.circle {...;
  position: absolute;
  z-index: 1;
}
```

Полученный результат:



Рисунок 29

Настало время для задания координат. Нам нужно сместить круги влево, поэтому для координаты **left** мы задаем отрицательное значение:

```
.circle {...;
  left: -20px;
}
```



Рисунок 30

Все светлые блоки с буквами съехали под левую границу браузера и наложились друг на друга, поэтому мы видим только блок с буквой **D**, т.к. он самый нижний в html-разметке (рис. 30).

В том случае, если мы добавим координату **left** и для цветных кругов с классом **.circle-back**, то ситуация повторится: все они разместятся в одном месте, причем виден будет только самый последний **div** в разметке с фиолетовым градиентом (рис. 31).

```
.circle-back { ...;
  left: -16px;
}
```



Рисунок 31

Давайте добавим еще координаты **top: 20px** и **17px** к обоим классам и увидим, что круги «улетели» наверх к заголовку первого блока, в котором нет никакого позиционирования (рис. 32).

The screenshot shows a web page with a header 'Our Camp'. Below it are two dark gray rectangular boxes. The left box contains the word 'Winners' and a trophy icon. The right box contains the word 'Pathfinders' and an eye icon. Both boxes contain placeholder text.

Our Camp

Winners

Placeholder text for the 'Winners' section.

Pathfinders

Placeholder text for the 'Pathfinders' section.

Рисунок 32

Такое поведение абсолютно позиционированных блоков очень характерно для них. Как уже отмечалось, до тех пор, пока элементам с `position: absolute` не заданы координаты (`left`, `top`, `right` или `bottom`), они держатся в пределах своего родителя. Как только координаты назначаются, эти элементы «отрываются» от родителя и располагаются относительно `html` или ближайшего к ним родителя со свойством `position`, отличным от `static`. Например, на этом скриншоте показано размещение круглый блоков, если для `section` добавить `position: relative` (рис. 33).



Рисунок 33

Отсюда следует вывод, которым очень часто пользуются при верстке: для того чтобы удержать абсолютно позиционированные элементы внутри родителя, необходимо задать родителю **position: relative, absolute** или **fixed**. Поскольку элементы с абсолютным и фиксированным положением имеют свои особенности, чаще всего для удержания абсолютных элементов их родителю назначают свойство **position: relative**. Такой элемент визуально ничем не отличается от статического, но отлично управляет координатами своих дочерних абсолютно позиционированных элементов внутри себя.



Рисунок 34

Поэтому мы добавляем **position: relative** в свойства класса **.team** и получаем нужный вид блоков (рис. 34).

Конечный вид свойств для рассмотренных классов:

```
.team {  
    display: inline-block;  
    width: 22.5%;  
    margin-right: 2%;  
    background-color: #4e4e4e;  
    color: #fff;  
    padding: 40px;  
    vertical-align: top;  
    border-radius: 15px;  
    box-shadow: 0 16px 24px 2px rgba(0, 0, 0, 0.14),  
                0 6px 30px 5px rgba(0, 0, 0, 0.12),  
                0 8px 10px -5px rgba(0, 0, 0, 0.2);  
    position: relative;  
}  
  
.circle {  
    width: 60px;  
    height: 60px;  
    border: 2px solid #ccc;  
    border-radius: 50%;  
    text-align: center;  
    color: #333;  
    font-weight: bold;  
    font-size: 1.4rem;  
    padding-top: 15px;  
    background-image: linear-gradient(#ccc, #fff);  
    position: absolute;  
    left: -20px;  
    top: 20px;  
    z-index: 1;  
}
```

```
.circle-back {  
    width: 70px;  
    height: 70px;  
    border-radius: 50%;  
    background-image: linear-gradient(#ff0, #ffba00);  
    box-shadow: 2px 4px 2px 2px rgba(0, 0, 0, 0.14);  
    position: absolute;  
    left: -16px;  
    top: 17px;  
}
```

Фиксированное позиционирование меню сайта

Настало время вернуться к форматированию шапки сайта. Разметка:

```
<header>  
    <nav>  
        <div class="logo">  
              
            The Best Camp  
        </div>  
        <ul>  
            <li><a href="#about">Our Camp</a></li>  
            <li><a href="#teams">Our Teams</a></li>  
            <li><a href="#reviews">Feedback</a></li>  
            <li><a href="#program">Our Program</a></li>  
        </ul>  
    </nav>  
    <h1>Welcome to The Best Camp</h1>  
</header>
```

Это довольно стандартная разметка, так что перейдём сразу к стилям:

```
header { height: 600px;
    background: url(..../images/top-header.jpg)
                no-repeat fixed center;
}

h1 { color: #fff;
    padding-top: 220px;
    text-align: center;
    font-size: 3em;
    text-shadow: 1px 1px 3px #fff;
}

.logo { float: left;
    font-size: 1.2em;
    font-weight: bold;
}

.logo img {
    vertical-align: middle;
    margin-right: 10px;
}

nav { height: 80px;
    padding: 10px 40px;
    background-color: #43B02A;
    color: #fff;
    position: fixed;
}

nav ul { float: right;
    list-style-type: none;
}

nav li {
    display: inline-block;
    margin-left: 10px;
}
```

```
nav a { color: #fff;  
    text-decoration: none;  
}  
  
nav a:hover { text-decoration: underline; }
```

После применения этих стилей в браузере мы увидим, что меню сдвинулось влево и стало меньше ширины экрана. Это произошло за счет того, что свойство `position: fixed` автоматически уменьшило ширину блока до ширины контента.

Сверху у нас появился отступ за счет `margin-top` для заголовка `h1`.

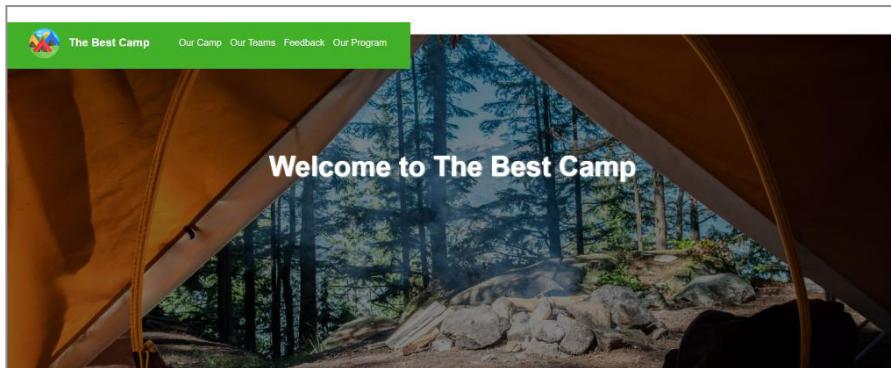


Рисунок 35

Исправим это, задав `width: 100%` для `nav` и `overflow: hidden` для `header`:

```
nav { height: 80px;  
    width: 100%;  
    ...;  
}
```

Урок 6.3. Позиционирование элементов

```
header { ...;
    overflow: hidden;
}
```

Теперь все в порядке:

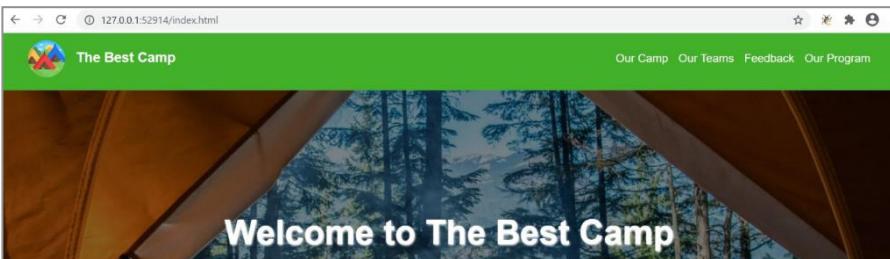


Рисунок 36

Поскольку мы задали фиксированное позиционирование для `nav`, давайте прокрутим страницу вниз. Прокрутка покажет вам, что и темно-серые блоки «Our Teams», и «фото авторов отзывов», которым мы задавали `position: relative`, наплывают на наше меню, хотя это и не входило в наши планы.



Рисунок 37



Рисунок 38

Для того чтобы это изменить, воспользуемся свойством **z-index** со значением, равным **10**:

```
nav { ...;
    position: fixed;
    z-index: 10;
}
```

Теперь все в порядке:

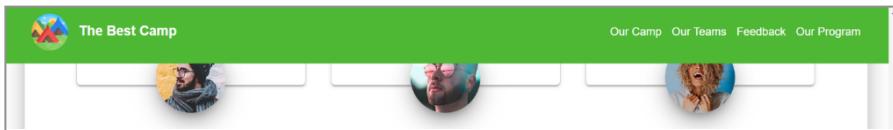


Рисунок 39

Использование липкого позиционирования

Блок «What's waiting for you» будет содержать информацию о различных мероприятиях лагеря. Его разметка включает заголовки, абзацы и изображения.

```
<section id="program">
    <h2>What's waiting for you?</h2>
    <div class="program-inner">
        <h3>Tent camp</h3>
        
        <p>Lorem ipsum dolor sit amet, ... aspernatur.
            Porro, sint.</p>
        <p>Quis, accusamus ... nemo eaque quia
            quisquam.</p>
        <p>Lorem ipsum dolor sit amet, ... earum,
            animi voluptatum.</p>
        <div class="clear"></div>
```

```
<h3>Training</h3>

<p>Lorem ipsum dolor sit amet, ... Porro,
      sint.</p>
<p>Quis, accusamus porro ... quisquam.</p>
<p>Lorem ipsum dolor sit amet, ... animi
      voluptatum.</p>
<div class="clear"></div>
<h3>Hiking</h3>

<p>Lorem ipsum dolor sit amet, ... Dolorem.</p>
<p>At iusto adipisci est perspiciatis ...
      facilis.</p>
<div class="clear"></div>
...
</div>
</section>
```

Стили для этого раздела таковы:

```
.program-inner {
    height: 380px;
    overflow-y: scroll;
}
.program-inner h3 {
    position: sticky;
    top: 0;
    background-color: #fcc068;
    color: #fff;
    margin-top: 0;
    margin-bottom: 20px;
    padding: 10px;
}
.program-inner h3:nth-of-type(3n-1) {
```

```

        background-color: #24ad64;
    }
.program-inner h3:nth-of-type(3n) {
    background-color: #8026c1;
}
.alignright {
    float: right;
    border-radius: 6px;
    margin: 0 8px 20px 15px;
    max-width: 450px;
    box-shadow: 0 16px 24px 2px rgba(0, 0, 0, 0.14),
                0 6px 30px 5px rgba(0, 0, 0, 0.12),
                0 8px 10px -5px rgba(0, 0, 0, 0.2);
}
.clear { clear: both; }

```

В стилях нужно обратить внимание на 2 блока. Первый — это **.program-inner**, для которого задана высота в **380px** и **overflow-y: scroll**, за счет чего этот блок прокручивается по вертикали. Второй, а точнее вторые — это вложенные в **.program-inner** заголовки **h3**, для которых указано свойство **position: sticky** и **top: 0**. При прокрутке каждый заголовок «прилипает» к верху **.program-inner** и остается там, пока ему не замену не придет с прокруткой следующий.

What's waiting for you?



Рисунок 40

Использование позиционирования при наведении курсора мыши

Рассмотрим пример с добавлением социальных иконок в блок с членами команды, который был сверстан в уроке, посвященном изображениям. Здесь мы используем связку `position: relative` для родительского элемента с `position: absolute` для дочернего.

Мы будем работать с файлом *StartUp-About-Section.psd* (папка *startup-page*), в котором есть только один нужный нам блок из всего макета. Для начала нам необходимо экспортировать из Adobe Photoshop иконки Facebook, Twitter и т.д. Используем для этого формат SVG и быстрый экспорт из Photoshop. Обычно в Photoshop быстрый экспорт настроен на сохранение в формате PNG или JPG. Поэтому сначала поменяем настройки в этой программе, нажав **Ctrl + K** и **Ctrl+7** или через меню **Edit > Preferences > Export**.

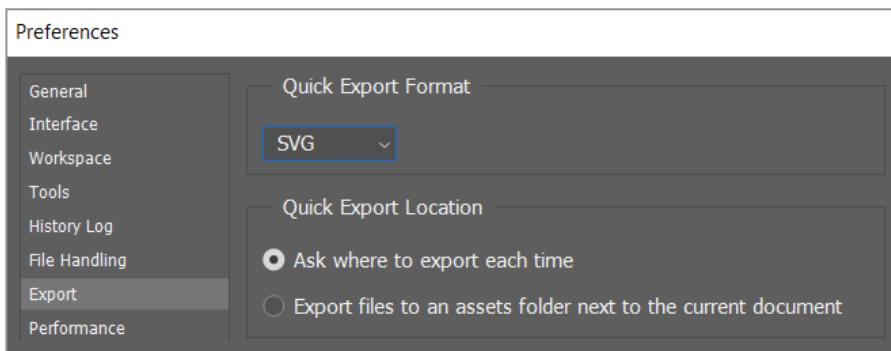


Рисунок 41

Теперь выделяем нужный слой с помощью **Move Tool**  и правым кликом на слое в палитре **Layers** экспортим каждую из иконок в SVG-формат.

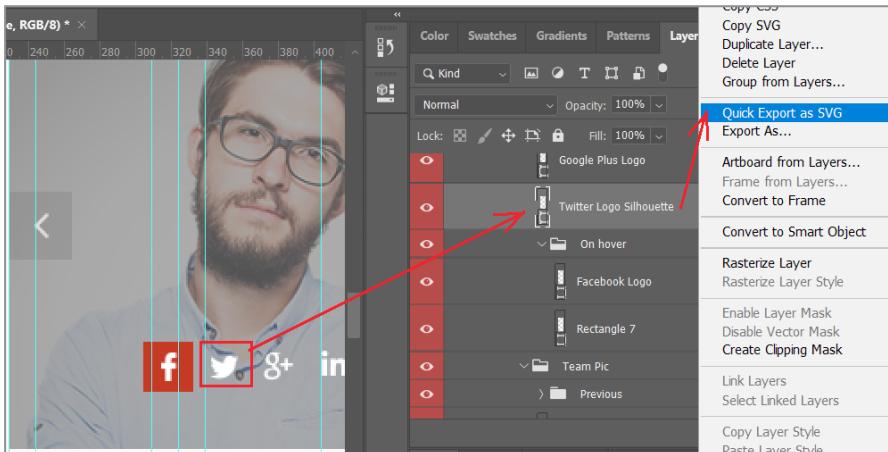


Рисунок 42

Все изображения нужно сохранить в папку *img* под именами *facebook.svg*, *twitter.svg*, *google-plus.svg* и *linkedin.svg*. Затем мы несколько изменим разметку примера из урока с изображениями, добавив в каждый из блоков `<div class="column-4 text-center team-member">` блок с иконками `<div class="social-logo">`:

```
<div class="column-4 text-center team-member">
    
    <h3 class="text-uppercase">Md. Khalil Uddin</h3>
    <p>Head of Ideas</p>
    <div class="social-logo">
        <a href="https://facebook.com">
            </a>
        <a href="https://twitter.com">
            </a>
        <a href="https://aboutme.google.com">
```

```
</a>
<a href="https://linkedin.com">
    
</a>
</div>
</div>
```

В файле *style.css* добавим такие стили:

```
.team-member {
    position: relative;
}

.social-logo {
    position: absolute;
    width: 100%;
    left: 0;
    bottom: 90px;
}

.social-logo a {
    display: inline-block;
    width: 28px;
    height: 28px;
    line-height: 32px;
}
```

Размеры иконок (**width** и **height**) возьмем из палитры **Properties** в Photoshop. Для изменения фона при наведении на каждую из иконок, добавим цвет, получив его из макета. Для этого нужно дважды щёлкнуть на слое **Rectangle** (рис. 43).

Позиционирование элементов

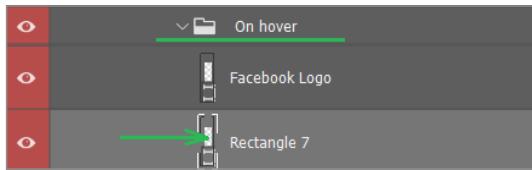


Рисунок 43

Откроется окно **Color Picker**, откуда нужно скопировать код цвета, чтобы вставить его в стиль.

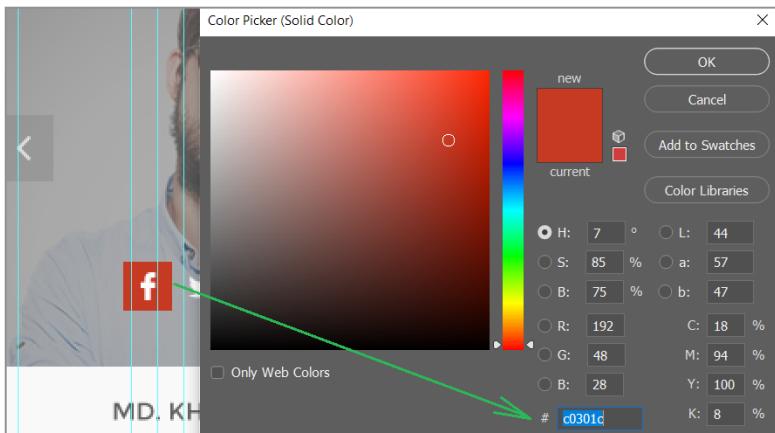


Рисунок 44

```
.social-logo a:hover {  
    background-color: #c0301c;  
}
```

Еще необходимо несколько изменить стиль для добавления фильтра при наведении на блок **team-member**:

```
.team-member:hover img {  
    filter: brightness(70%);  
}
```

Однако, при наведении на изображение-ссылку в блоке с классом `social-logo` нам нужно отменить назначение фильтра для класса `team-member`:

```
.team-member:hover .social-logo a img {filter: none;}
```

На данный момент все иконки видны сразу же для всех членов команды, а они должны появляться только при наведении.



Рисунок 45

Поэтому сейчас мы спрячем их вниз с помощью свойства `bottom:20px` и `opacity: 0`. Добавим для класса `.social-logo` свойство `transition: .6s` для плавного появления иконок снизу при наведении на блок с изображением.

```
.social-logo {  
    position: absolute;  
    width: 100%;  
    left: 0;  
    bottom: 20px;  
    opacity: 0;  
    transition: .6s;  
}
```

```
.team-member:hover .social-logo{  
    bottom: 90px;  
    opacity: 1;  
}
```

Результат лучше посмотреть в браузере.

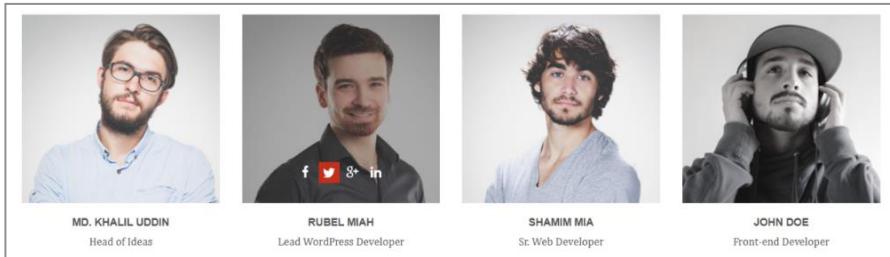


Рисунок 46

Рабочий пример вы найдете в папке *examples/site/index.html*.

Домашнее задание

Задание 1

Вам нужно будет сделать рамку вокруг картинки с поворотом на 45 градусов. Для этого вам понадобится использовать позиционирование (какое именно, решайте сами), `z-index` и свойство `transform: rotate(45deg)`, позволяющее осуществить поворот любого html-элемента.

Изображение вы найдете в папке `HW/images`. Вам необходимо будет центрировать его на странице (тег `<center>` применять нельзя).

Подсказка: добавьте в разметку 2 div-а для рамки или используйте для этого псевдоэлементы `::before` и `::after`.

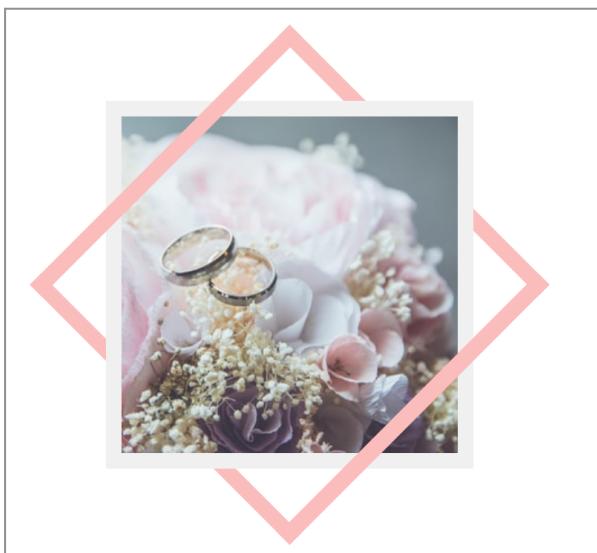


Рисунок 47

Задание 2

Вам нужно будет сверстать блок с проектами, который состоит из 3-х колонок:

OUR PROJECTS



Cobb County, Georgia, United States

Project 1

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!



Southwestern, United States

Project 2

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!



Dubai, UAE

Project 3

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!

Рисунок 48

В каждой колонке у вас находится изображение и текстовый блок, содержащий описание проекта, его заголовок и текст из 2-х абзацев. При этом изначально мы видим только первый абзац из 2-х.

При наведении на весь блок с проектом, текстовый блок поднимается вверх, закрывая своим полупрозрачным черным фоном картинку, и мы видим оба абзаца.



Cobb County, Georgia, United States

Project 1

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!



Southwestern, United States

Project 2

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!



Dubai, UAE

Project 3

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!

Est libero ipsa repellendus accusantium, illum? Magnam accusantium, possimus temporibus. Possimus, commodi!

Рисунок 49

Кроме того, меняется выравнивание и цвет текста, заголовок проекта выделяется оранжевым фоном. Низ блока оформляется утолщенной оранжевой рамкой.

Подсказка: используйте свойство `position` со значениями `relative` и `absolute`.

Фото для блока вы найдете в папке `HW/images`. Текст абзацев сгенерируйте с помощью Emmet. Скриншоты задания лежат в папке `HW/screenshots`.

Для того чтобы при наведении у вас было плавное изменение позиции текстового блока, используйте свойство `transition: .5s`.

Домашнее задание



Урок 6.3. Позиционирование элементов

© Слуцкая Елена.
© STEP IT Academy, www.itstep.org.

All rights to protected pictures, audio, and video belong to their authors or legal owners.

Fragments of works are used exclusively in illustration purposes to the extent justified by the purpose as part of an educational process and for educational purposes in accordance with Article 1273 Sec. 4 of the Civil Code of the Russian Federation and Articles 21 and 23 of the Law of Ukraine "On Copyright and Related Rights". The extent and method of cited works are in conformity with the standards, do not conflict with a normal exploitation of the work, and do not prejudice the legitimate interests of the authors and rightholders. Cited fragments of works can be replaced with alternative, non-protected analogs, and as such correspond the criteria of fair use.

All rights reserved. Any reproduction, in whole or in part, is prohibited. Agreement of the use of works and their fragments is carried out with the authors and other right owners. Materials from this document can be used only with resource link.

Liability for unauthorized copying and commercial use of materials is defined according to the current legislation of Ukraine.