**Тема 6. Позиционирование блоков в CSS.**

**Занятие 1**

**16.06.20**

Позиционированием называется положение элемента в системе координат. Различают четыре типа позиционирования: нормальное, абсолютное, фиксированное и относительное. В зависимости от типа, который устанавливается через свойство **position**, изменяется и система координат.

Благодаря комбинации свойств **position**, **left**, **top**, **right** и **bottom** элемент можно накладывать один на другой, выводить в точке с определёнными координатами, фиксировать в указанном месте, определить положение одного элемента относительно другого и др. Подобно другим свойствам CSS управление позиционированием доступно через скрипты. Таким образом, можно динамически изменять положение элементов без перезагрузки страницы, создавая анимацию и различные эффекты.

**Нормальное позиционирование**. Если для элемента свойство **position** не задано или его значение **static**, элемент выводится в потоке документа как обычно. Иными словами, элементы отображаются на странице в том порядке, как они идут в исходном коде HTML.

Свойства **left**, **top**, **right**, **bottom,** если определены, игнорируются.

**Абсолютное позиционирование.** При абсолютном позиционировании элемент не существует в потоке документа и его положение задаётся относительно краёв браузера. Задать этот тип можно через значение **absolute** свойства **position**. Координаты указываются относительно краёв окна браузера, называемого «видимой областью» (рис.6.1).



Рис. 6.1. Значения свойств left, right, top и bottom при абсолютном позиционировании

Для режима характерны следующие особенности.

* Ширина слоя, если она не задана явно, равна ширине контента плюс значения полей, границ и отступов.
* Слой не меняет своё исходное положение, если у него нет свойств **right**, **left**, **top** и **bottom**.
* Свойства **left** и **top** имеют более высокий приоритет по сравнению с **right** и **bottom**. Если **left** и **right** противоречат друг другу, то значение **right** игнорируется. То же самое касается и **bottom**.
* Если **left** задать отрицательное значение, то слой уйдёт за левый край браузера, полосы прокрутки при этом не возникнет. Это один из способов спрятать элемент от просмотра. То же относится и к свойству **top**, только слой уйдёт за верхний край.
* Если **left** задать значение больше ширины видимой области или указать **right** с отрицательным значением, появится горизонтальная полоса прокрутки. Подобное правило работает и с **top**, только речь пойдёт о вертикальной полосе прокрутки.
* Одновременно указанные свойства **left** и **right** формируют ширину слоя, но только если **width** не указано. Стоит добавить свойство **width** и значение **right** будет проигнорировано. Аналогично произойдёт и с высотой слоя, только уже участвуют свойства **top**, **bottom** и **height**.
* Элемент с абсолютным позиционированием перемещается вместе с документом при его прокрутке.

Свойство **position** со значением **absolute** можно использовать для создания эффекта фреймов. Кроме абсолютного позиционирования для элементов необходимо назначить свойство **overflow** со значением **auto**. Тогда при превышении контентом высоты видимой области появится полоса прокрутки. Высота и ширина «фреймов» формируется автоматически путём одновременного использования свойств **left**, **right** для ширины и **top**, **bottom** для высоты.

**Практическая работа. Применение абсолютного позиционирования.**

1. Откройте редактор (Breckets или Блокнот)
2. Наберите следующий код.

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

**<html** xmlns="http://www.w3.org/1999/xhtml"**>**

 **<head>**

 **<title>**Абсолютное позиционирование**</title>**

 **<style** type="text/css"**>**

 **body** { margin: 0; }

 #sidebar, #content { position: absolute; }

 #sidebar, #content { overflow: auto; padding: 10px; }

 #header {

 height: 80px; /\* Высота слоя \*/

 background: #FEDFC0; border-bottom: 2px solid #7B5427;

 }

 #header **h1** { padding: 20px; margin: 0; }

 #sidebar {

 width: 150px; background: #ECF5E4; border-right: 1px solid #231F20;

 top: 82px; /\* Расстояние от верхнего края \*/

 bottom: 0; /\* Расстояние снизу \*/

 }

 #content {

 top: 82px; /\* Расстояние от верхнего края \*/

 left: 170px; /\* Расстояние от левого края \*/

 bottom: 0; right: 0;

 }

 **</style>**

 **</head>**

 **<body>**

 **<div** id="header"**><h1>**Плов народов мира**</h1></div>**

 **<div** id="sidebar"**>**

 **<p>**Плов по-фергански**</p><p>**Плов узбекский**</p>**

 **<p>**Плов сибирский**</p><p>**Плов итальянский**</p>**

 **<p>**Плов эстонский**</p><p>**Плов по-американски**</p>**

 **<p>**Плов по-индейски**</p>**

 **</div>**

 **<div** id="content"**>**

 **<h2>**Плов по-фергански**</h2>**

 **<p>**Положить в казан нарезанное кусочками мясо и поджарить его до

 образования корочки. Нашинкованный кольцами лук жарить вместе

 с мясом до красноватого цвета, затем добавить морковь, нарезанную

 соломкой. Положить половину соли, всё перемешать и жарить, пока

 морковь не приобретёт золотисто-коричневый цвет. После этого налить

 половину необходимого количества воды и дать закипеть.**</p>**

 **<p>**Засыпать ровным слоем рис, усилить огонь и тотчас налить воду,

 чтобы она накрыла рис на 1–1,5 см. Как только вода выпарится, плов

 при помощи шумовки собрать к середине горкой, проколоть палочкой в

 нескольких местах так, чтобы вода, находящаяся на поверхности,

 прошла на дно. Затем накрыть плов и дать ему упреть 20–25 мин.**</p>**

 **<p>**Тщательно перемешайте готовый плов, переложите в большое блюдо,

 сверху разложите мясо.**</p>**

 **</div>**

 **</body>**

**</html>**

1. Сохраните его в файле 6\_CSS.html
2. Проанализируйте результат. Слой **header** (шапка) выводится в потоке как обычно, а для слоёв **sidebar** (боковая колонка сайта) и **content** (содержимое) установлено абсолютное позиционирование.

**Занятие 2.**

**17.06.20**

**Фиксированное положение** слоя задаётся значением **fixed** свойства **position** и по своему действию похоже на абсолютное позиционирование. 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. Ещё одна разница от **absolute** заключается в том, что при выходе фиксированного слоя за пределы видимой области справа или снизу от неё, не возникает полос прокрутки.

Применяется такой тип позиционирования для создания меню, вкладок, заголовков, в общем, любых элементов, которые должны быть закреплены на странице и всегда видны посетителю. В примере показано добавление подвала, который остаётся на одном месте независимо от объёма информации на сайте.

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

**<html** xmlns="http://www.w3.org/1999/xhtml"**>**

 **<head>**

 **<title>**Фиксированный подвал**</title>**

 **<style** type="text/css"**>**

 **BODY** { margin-bottom: 50px; }

 #footer {

 position: fixed; /\* Фиксированное положение \*/

 left: 0; bottom: 0; /\* Левый нижний угол \*/

 padding: 10px; /\* Поля вокруг текста \*/

 background: #39b54a; /\* Цвет фона \*/

 color: #fff; /\* Цвет текста \*/

 width: 100%; /\* Ширина слоя \*/

 }

 **</style>**

 **</head>**

 **<body>**

 **<div** id="content"**>**

 Все перечисленные на сайте методы ловли льва являются теоретическими

 и базируются на вычислительных методах. Автор не гарантируют

 вашей безопасности при их использовании и снимает с себя всякую

 ответственность за результат.

 Помните, лев это хищник и опасное животное!

 **</div>**

 **<div** id="footer"**>**&copy; Влад Мержевич**</div>**

 **</body>**

**</html>**

Поскольку фиксированный подвал **(footer)** накладывается на текст и скрывает его, добавлен отступ снизу для селектора **BODY**. Браузер IE6 не поддерживает значение **fixed**, поэтому в нём данный пример будет работать некорректно.

**Относительное позиционирование.** Если задать значение **relative** свойства **position**, то 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. Положительное значение **left** определяет сдвиг вправо от левой границы элемента, отрицательное — сдвиг влево. Положительное значение **top** задаёт сдвиг элемента, отрицательное — сдвиг вверх.



Свойства **bottom** и **right** производят обратный эффект. При положительном значении **right** сдвигает элемент влево от его правого края, при отрицательном — сдвигает вправо. При положительном значении **bottom** элемент поднимается вверх, при отрицательном опускается вниз.



Для относительного позиционирования характерны следующие особенности.

* Этот тип позиционирования не применим к элементам таблицы вроде ячеек, строк, колонок и др.
* При смещении элемента относительно исходного положения, место, которое занимал элемент, остаётся пустым и не заполняется ниже или вышележащими элементами.

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"

 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

**<html** xmlns="http://www.w3.org/1999/xhtml"**>**

 **<head>**

 **<title>**Заголовок**</title>**

 **<style** type="text/css"**>**

 **H1** {

 font: bold 2em Arial, Tahome, sans-serif; /\* Параметры шрифта \*/

 color: #fff; background: #375D4C;

 padding: 0 10px;

 }

 **H1** **SPAN** {

 position: relative; /\* Относительное позиционирование \*/

 top: 0.3em; /\* Сдвигаем вниз \*/

 }

 **</style>**

 **</head>**

 **<body>**

 **<h1><span>**Аз и буки шрифтовой науки**</span></h1>**

 **<p>**Шрифт это средство выражения дизайна, а не какого-то

 банального чтения.**</p>**

 **</body>**

**</html>**

**Практическая работа.**

1. Попробуйте выполнить примеры с относительным и фиксированным позиционированием блоков.
2. Дополните файл 6\_CSS.html блоками с относительным и фиксированным позиционированием.

**Тема 7. Свойства блоков CSS.**

**Занятие 1**

**18.06.20**

**Наложение элементов** При применении свойств позиционирования элементы могут накладываться друг на друга. Свойство **z-index** позволяет установить какой элемент в случае наложения будет сверху, а какой снизу.

Элементы с большим значением свойства **z-index** располагаются выше других.

<html>

<head>

<style type='text/css'>

#pos1

{

position:absolute;

top:0px;

left:100px;

border:1px solid;

width:80px;

height:80px;

background-color:red;

font-size:4em;

z-index:10;

}

#pos2

{

position:absolute;

top:36px;

left:130px;

border:1px solid;

width:80px;

height:80px;

background-color:blue;

font-size:4em;

z-index:5;

}

#pos3

{

position:absolute;

top:10px;

left:170px;

border:1px solid;

width:80px;

height:80px;

background-color:pink;

font-size:4em;

z-index:-1;

}

</style>

</head>

<body>

<p id='pos1'>10</p>

<p id='pos2'>5</p>

<p id='pos3'>-1</p>

<p>Элемент с <i>z-index:10</i> - красный, с <i>z-index:5</i> - синий, а с <i>z-index:-1</i> - розовый. </b></p>

</body>

</html>

**Свойства связанные с размещением.**

|  |  |  |
| --- | --- | --- |
| Свойства | Описание | Значения |
| clip | Обрезает элемент размещенный абсолютно. Свойство clip не будет работать если установлено overflow:visible | rect (отступ сверху, справа, снизу, слева)auto (не применять свойство clip) |
| cursor | Задает вид, который будет принимать курсор при наведении на элемент. | autocrosshairdefaultpointermovee-resizene-resizenw-resizen-resizese-resizesw-resizes-resizew-resizetextwaithelp |
| overflow | Устанавливает как должно быть отображено содержимое вышедшее за границы элемента. | аuto (прокрутка при необходимости)hidden (скрыть)scroll (прокрутка всегда)visible (показать) |

**Занятие 2. Практическая работа.**

**19.06.20**

1. Создайте файл 7\_CSS.html.
2. Скопируйте в него предложенный код.
3. Дополните его так, чтобы геометрические фигуры переместились в соответствующие им рамки с помощью свойств размещения.

<html>

<head>

<style type='text/css'>

gimg.gif

gimg2.gif

gimg3.gif

gimg4.gif

#geom1

{

/\* Пишите код здесь \*/

}

#geom2

{

/\* Пишите код здесь \*/

}

#geom3

{

/\* Пишите код здесь \*/

подсказка

#geom1

{

position:absolute;

bottom:100px;

left:170px;

}

#geom2

{

position:absolute;

top:50px;

right:220px;

}

#geom3

{

position:absolute;

bottom:30px;

right:30px;

}

#geom4

{

position:absolute;

top:30px;

left:50px;

}

}

#geom4

{

/\* Пишите код здесь \*/

}

#cont1

{

color:green;

position:absolute;

top:30px;

left:50px;

border:1px solid green;

width:150px;

height:130px;

}

#cont1 p

{

padding-left:8px;

padding-top:13px;

}

#cont2

{

color:#c50080;

position:absolute;

top:50px;

right:220px;

border:1px solid #c50080;

width:170px;

height:130px;

}

#cont2 p

{

padding-left:11px;

padding-top:16px;

}

#cont3

{

color:blue;

position:absolute;

bottom:100px;

left:170px;

border:1px solid blue;

width:170px;

height:130px;

}

#cont3 p

{

padding-left:25px;

padding-top:16px;

}

#cont4

{

color:#bfbc30;

position:absolute;

bottom:30px;

right:30px;

border:1px solid #bfbc30;

width:160px;

height:130px;

}

#cont4 p

{

padding-left:4px;

padding-top:17px;

}

</style>

</head>

<body>

<img id='geom1' src='gimg.gif' />

<img id='geom2' src='gimg2.gif' />

<img id='geom3' src='gimg3.gif' />

<img id='geom4' src='gimg4.gif' />

<div id='cont1'>

<p><b>Поместите в меня зеленый квадрат.</b></p>

</div>

<div id='cont2'>

<p><b>Поместите в меня розовый прямоугольник.</b></p>

</div>

<div id='cont3'>

<p><b>Поместите в меня синий овал.</b></p>

</div>

<div id='cont4'>

<p><b>Поместите в меня желтый многоугольник.</b></p>

</div>

</body>

</html>

**Тема 8. Специальные стили для списков CSS.**

**Занятие 1**

**22.06.20**

Списки это очень важная тема в HTML и CSS. Благодаря спискам мы можем делать не только перечень чего-либо, но также создавать меню сайта.

Как сделать меню сайта на CSS? Разберем это на примере.

Сначала рассмотрим сам HTML:

<ul id="navbar">

 <li><a href="#">Главная</a></li>

 <li><a href="#">Новости</a></li>

 <li><a href="#">Контакты</a>

 <ul>

 <li><a href="#">Адрес</a></li>

 <li><a href="#">Телефон</a></li>

 <li><a href="#">Email</a></li>

 </ul>

 </li>

 <li><a href="#">О нас</a></li>

</ul>

Здесь все просто. Есть один список и в нем вложен еще один список, который будет раскрываться при наведении мыши на нужную ячейку меню. Теперь давайте рассмотрим весь CSS код:

<style type='text/css'>

#navbar ul{

 display: none;

 background-color: #f90;

 position: absolute;

 top: 100%;

}

#navbar li:hover ul {

 display: block;

}

#navbar, #navbar ul{

 margin: 0;

 padding: 0;

 list-style-type: none;

}

#navbar {

 height: 30px;

 background-color: #666;

 padding-left: 25px;

 min-width: 470px;

}

#navbar li {

 float: left;

 position: relative;

 height: 100%;

}

#navbar li a {

 display: block;

 padding: 6px;

 width: 100px;

 color: #fff;

 text-decoration: none;

 text-align: center;

}

#navbar ul li { float: none; }

#navbar li:hover { background-color: #f90; }

#navbar ul li:hover { background-color: #666; }

</style>

Основным моментом здесь является то, что при наведении мыши мы делаем вложенный список блоком: **#navbar li:hover ul {display: block}**. При помощи дополнительных стилей мы указываем чтобы список был в форме блоков и чтобы у него не было значков списка и прочих стилей. Таким образом у нас получается работающее меню для сайта.

**Занятие 2. Практическая работа.**

**23.06.20**

1. Соберите фрагменты кода, размещенные выше, в одном файле.
2. Сохраните его с именем 8\_CSS.html.
3. Попробуйте изменить получившееся меню.

**Тема 9. Слои в языке CSS.**

**Занятие 1**

**25.06.20**

Элемент **<div>** является блочным элементом и предназначен для выделения фрагмента документа с целью изменения вида содержимого. Как правило, вид блока управляется с помощью стилей. Чтобы не описывать каждый раз стиль внутри тега, можно выделить стиль во внешнюю таблицу стилей, а для тега добавить атрибут **class** или **id** с именем селектора.

Как и при использовании других блочных элементов, содержимое тега **<div>** всегда начинается с новой строки. После него также добавляется перенос строки.

Также для этого тега доступны универсальные атрибуты и события.

<!DOCTYPE HTML>

**<html>**

 **<head>**

 **<meta** charset="utf-8"**>**

 **<title>**Тег DIV**</title>**

 **<style** type="text/css"**>**

 .block1 {

 width: 200px;

 background: #ccc;

 padding: 5px;

 padding-right: 20px;

 border: solid 1px black;

 float: left;

 }

 .block2 {

 width: 200px;

 background: #fc0;

 padding: 5px;

 border: solid 1px black;

 float: left;

 position: relative;

 top: 40px;

 left: -70px;

 }

 **</style>**

 **</head>**

 **<body>**

 **<div** class="block1"**>**Lorem ipsum dolor sit amet, consectetuer

 adipiscing elit, sed diem nonummy nibh euismod tincidunt ut lacreet

 dolore magna aliguam erat volutpat.**</div>**

 **<div** class="block2"**>**Ut wisis enim ad minim veniam, quis nostrud

 exerci tution ullamcorper suscipit lobortis nisl ut aliquip ex

 ea commodo consequat.**</div>**

 **</body>**

**</html>**

**Занятие 2**

**26.06.20**

**CSS3 Анимация. Обратите внимание:** свойства анимации поддерживаются браузерами IE10+, Firefox и Opera. Для браузеров Chrome и Safari перед свойством требуется добавить префикс **-webkit**.

Для создания анимации в CSS3 используется свойство **@keyframes**.

Данное свойство представляет собой контейнер, в который должны помещаться различные свойства оформления.

Синтаксис:

@keyframes имяАнимации

{

from {CSS свойства} /\* Оформление элемента перед началом анимации \*/

to {CSS свойства} /\* Оформление элемента после завершения анимации \*/

}

После того, как анимация была создана необходимо добавить к элементу CSS3 свойство **animation** и указать в нем имя анимации (1 значение) и время (2 значение), в течении которого она будет выполняться.

Можно устанавливать количество повторов анимации (3 значение).

@keyframes anim {

from {margin-left:3px;}

to {margin-left:500px;}

}

#wrap1 {

animation:anim 4s 3;

}

Например:

<style type='text/css'>

@keyframes anim{

from {margin-left:3px;}

to {margin-left:500px;}

}

@-moz-keyframes anim{

from {margin-left:3px;}

to {margin-left:500px;}

}

@-webkit-keyframes anim{

from {margin-left:3px;}

to {margin-left:500px;}

}

#wrap1{

border:2px #000 solid;

background-color:#7F0055;

height:100px;

width:100px;

font-size:2em;

animation:anim 4s 3;

-webkit-animation:anim 4s 3;

}

</style>

<body>

<div id="wrap1"></div>

<p><b>Обратите внимание:</b> данная анимация будет повторяться 3 раза.</p>

</body>

</html>

**Ход выполнения анимации.** Можно определять ход выполнения анимации не только с помощью ключевых слов **from** и **to** (которые использовались в предыдущем примере), но и с помощью **%**. Например, можно указать, что определенный элемент в начале анимации (0%) должен быть белым к середине (50%) должен окрашиваться в оранжевый цвет, а к концу (100%) становиться черным.

CSS3 свойства анимации

|  |  |
| --- | --- |
| Свойство | Описание |
| @keyframes | Контейнер для определения анимации. |
| animation | Позволяет задать все значения для настройки выполнения анимации за одно определение. |
| animation-name | Позволяет указать имя анимации. |
| animation-duration | Позволяет задать скорость выполнения анимации в секундах (по умолчанию имеет значение 0). |
| animation-timing-function | Позволяет задать функцию смягчения отвечающую за плавность выполнения анимации (по умолчанию имеет значение ease). |
| animation-delay | Позволяет задать задержку перед началом выполнения анимации (по умолчанию имеет значение 0). |
| animation-iteration-count | Позволяет задать количество повторов анимации (по умолчанию имеет значение 1). |
| animation-direction | При значении alternate в нечетные разы (*1,3,5 ...*) анимация будет проигрываться в нормальном, а в четные (*2,4,6 ...*) в обратном порядке. По умолчанию данное свойство имеет значение normal, при данном значении анимация всегда проигрывается в нормальном порядке. |

**Практическая работа.**

1. Соберите фрагменты кода этого занятия в одном файле.
2. Сохраните его с именем 9\_CSS.html.
3. Попробуйте изменить получившуюся анимацию.

<html>

<style type='text/css'>

@keyframes anim {

0% {margin-left:3px;margin-top:3px;background-color:#7F0055;}

30% {margin-left:3px;margin-top:250px;background-color:#7F0055;}

60% {margin-left:500px;margin-top:250px;background-color:black;}

100% {margin-left:3px;margin-top:3px;background-color:#7F0055;}

}

@-moz-keyframes anim {

0% {margin-left:3px;margin-top:3px;background-color:#7F0055;}

30% {margin-left:3px;margin-top:250px;background-color:#7F0055;}

60% {margin-left:500px;margin-top:250px;background-color:black;}

100% {margin-left:3px;margin-top:3px;background-color:#7F0055;}

}

@-webkit-keyframes anim {

0% {margin-left:3px;margin-top:3px;background-color:#7F0055;}

30% {margin-left:3px;margin-top:250px;background-color:#7F0055;}

60% {margin-left:500px;margin-top:250px;background-color:black;}

100% {margin-left:3px;margin-top:3px;background-color:#7F0055;}

}

#wrap1 {

border:2px #000 solid;

background-color:#7F0055;

height:100px;

width:100px;

font-size:2em;

animation:anim 6s 3;

-webkit-animation:anim 6s 3;

}

</style>

<body>

<div id="wrap1"></div>

</body>

</html>