**Терминология**

Чтобы не было двусмысленностей, договоримся о терминах.  
**Float** (плавающий элемент) – под *float* или плавающим элементом в статье подразумевается блочный элемент с установленным CSS свойством *float* — *right* или *left*.  
**Блок** – под термином блок подразумевается воображаемый прямоугольник, в который заключен каждый элемент HTML разметки.

**Виды позиционирования HTML элементов**

Для ясного представления изложенного надо также знать основные принципы отображения элементов в HTML документах.

Согласно [w3.org](http://www.w3.org/TR/CSS2/visuren.html#positioning-scheme) в WEB разметке используется три схемы размещения элементов в Web документах.

**Нормальный поток**  - нормальный поток включает обычные элементы верстки, у которых не установлены *float*, равный *left/right*, или *position*, равный *absolute/fixed* .  
**Float** – плавающий элемент вначале размещается в соответствии с правилами размещения в нормальном потоке, потом выводится из этого потока и смещается в своей строке влево или вправо, насколько это возможно.

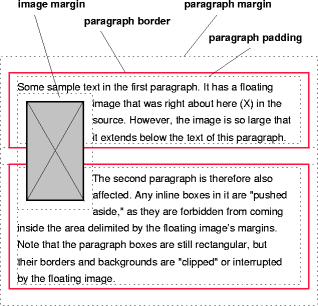
Четко усвойте следующее важное правило. Так как плавающие элементы выводятся из нормального потока документа, то блочные боксы нормального потока их не видят, но строчные боксы нормального потока их видят и обтекают по одной из сторон, к строчным боксам, в частности, относится обычный текст блочных боксов. А это означает, что если при своем размещении блочные боксы нормального потока не обращают никакого внимания на присутствие плавающих боксов, то содержимое этих боксов (строчные элементы) обтекают такие элементы.

**Абсолютное позиционирование** – при абсолютном позиционировании бокс полностью удаляется из нормального потока (не влияет на последующие ) и размещается относительно позиции своего контейнера, т.е элемента, в котором он размещен.

**Различия между плавающими(float) и позиционированными(position) элементами**

В чем разница между плавающим (*float*) и позиционированным (*position*) элементами? Относительно позиционированный элемент со значением *position:relative* остается в нормальном потоке документа, но сдвигается относительно своего положения, другие элементы при этом не затрагиваются и остаются там, где они были. Это дает возможность элементам с относительным позиционированием перекрывать окружающие его элементы на странице. Если у элемента установлено *position:absolute;* или *position:fixed;*, то он удаляется из нормального потока и размещается в соответствии с правилами абсолютного позиционирования.  
Плавающий элемент всегда удаляется из нормального потока документа и влияет на размещение окружающих элементов.

**Что такое float?**

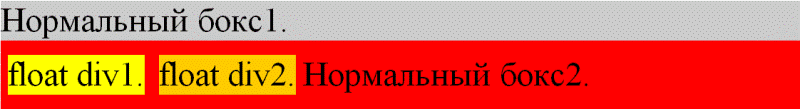
  
«Float» — одно из CSS свойств блочного элемента разметки, благодаря которому HTML элемент смещается в крайнее, доступное для размещения, левое или правое положение внутри содержащего этот элемент контейнера. Если в текущей строке контейнера по ширине не хватает места для его размещения, он смещается на следующую строку до тех пор, пока не сможет разместиться. Замечу, что ширина *float* элемента определяется его содержимым. В зависимости от установленного значения *float*, строчные элементы могут обтекать такой блочный элемент по одной из его горизонтальных сторон.  
Свойство *float* может принимать следующие значения: *none*(по умолчанию), *left, right и inherit*.

**Правила отрисовки float элементов**

Что происходит, когда браузер встречает элемент, у которого свойство *float* отлично от *none*?  
Вначале такой бокс размещается, как обычный элемент, потом извлекается из нормального потока и сдвигается в левую или правую сторону до тех пор, пока не встретит такой же *float* элемент или границу своего родительского контейнера. В случае, если оставшегося места по горизонтали для элемента не хватает, браузер смещает его на следующую строку своего контейнера.  
Поскольку *float* элемент извлечен из нормального потока, обычные блочные элементы без установленного свойства *float* не знают о его существовании и размещаются обычным способом, каждый на новой строке. Строчные же элементы разметки «видят» *float* бокс и обтекают его по одной из сторон в зависимости от значения — *left* или *right*.  
пример.

<div style="background:#CCCCCC;">Нормальный бокс1.</div>  
<div style="float:left;background:#FFFF00;margin:5px;">float div1.</div>  
<div style="float:left;background:#FFCC00;margin:5px;">float div2.</div>  
<div style="background:#FF0000;padding:10px;">Нормальный бокс2.</div>

Вот что при этом отобразится браузером.



**Демонстрация отрисовки float элементов**

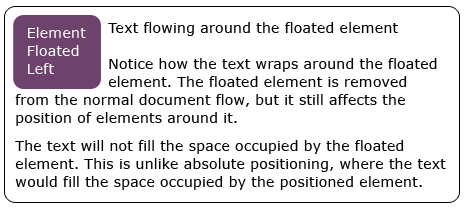
А теперь разберемся, что получилось. В примере приведено 4 блока, бокс1 и бокс2 отрисовываются браузером по умолчанию, т.е в нормальном потоке, а у внутренних элементов стоит *float = left*. Чтобы пример получился нагляднее, я добавил блокам фоновый цвет и отступы. Итак, в итоге блоки по умолчанию (1-й и 4-й) отобразились, как обычно, каждый на новой строке и занимают всю ширину(поскольку приведен screenshot разметки, ширина боксов ограничена шириной картинки). Плавающие же (2-й и 3-й) были извлечены из нормального потока (т.е, 1-й и 4-й боксы их не видят, поэтому «нормальный бокс2» начинается под 1-м с начала строки) и смещены к левой границе контейнера. И хотя нормальные блочные боксы не видят плавающих, но строчный бокс последнего бокса(а это его текст) видит плавающие боксы и обтекает их с правой стороны.  
Как видите, все в соответствии с вышеприведенным алгоритмом отрисовки *float* элементов.

**Как используют float**

Существует два основных способа использования свойства *float*.  
Рассмотрим их.

***Обтекание изображения текстом.***

В HTML разметке часто используются изображения, да и просто какие-либо прямоугольные элементы. Свойство float широко используется для того, чтобы текст располагался вокруг них, а не со следующей за этими прямоугольниками строки.



**Текст обтекает изображение с float:left**

Пример верстки страницы с текстом, обтекающим изображение.

HTML  
‹p›  
‹img class="alignleft" src="examp.gif " alt="" width="" height="" /›  
Здесь размещается текст, обтекающий изображение examp.gif.  
‹/p›

CSS  
img.alignleft {float: left; margin: 0 10px 10px 0}

В коде сверху и плавающий элемент *img* и обтекающий текст находятся внутри одного и того же контейнера **p**.  
Обратите внимание, что в CSS стилях слева и справа плавающему элементу добавлены отступы. Это обычная практика, когда изображению добавляется пространство между ним и окружающим текстом.

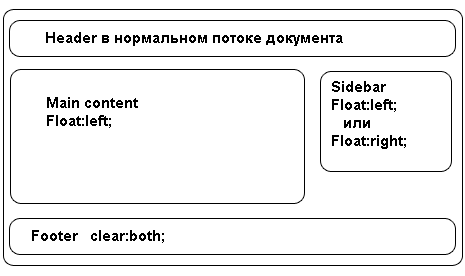
***Макеты с колонками***

Свойство *Float* позволяет два блочных элемента располагать рядом друг с другом, а не начинать с новой строки. Это свойство плавающих элементов очень удобно использовать для создания колонок в документе. Чтобы получить макет с 2 колонками, достаточно задать обтекание(т.е. задать свойcтво float) двум DIVам.  
Пример создания 2-х колоночного макета с помощью float.

HTML  
‹div id="wrapper"›  
‹div id="left-column"›‹div›  
‹div id="right-column"›‹/div›  
‹/div›

CSS  
#left-column {float: left}  
#right-column {float: left}

В примере оба внутренних DIVа получили обтекание слева, в итоге имеем 2 колонки. Вы также должны убедиться, что общая ширина двух внутренних столбцов (div) не превышает ширину родительского бокса, иначе вторая колонка перескочит на новую позицию, ниже первой колонки.  
Добавив div заголовка над 2-мя колонками и div подвала с использованием свойства *clear,* вы получите простой и красивый 2-х колоночный макет.

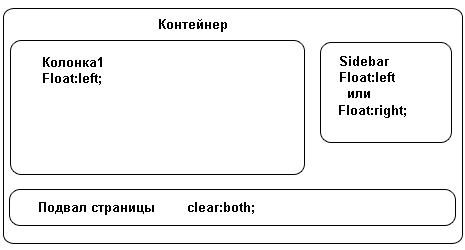


**Пример использования float для создания 2-х колоночного макета**

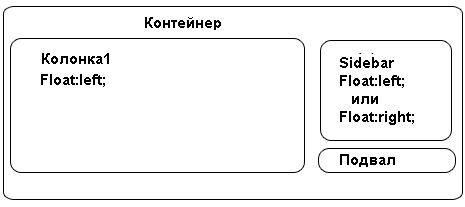
И еще один совет. Всегда устанавливайте ширину плавающего элемента, при ее отсутствии вы можете получить непредвиденные последствия.

**Отмена свойства float**

Нередко при работе с *float* бывает необходимо, чтобы последующие элементы разметки располагались не рядом с плавающим текущим, а как обычно, на следующей строке, т.е требуется вернуться к размещению элементов в соответствии с нормальным потоком.  
Часто этого добиваются использованием свойства *clear*. Свойство *clear* принимает значения *left, right, both, none* (по умолчанию), и *inherit*. На практике используется первые 3 значения, причем в подавляющем большинстве случаев вы будете использовать *clear: both.*  
Чаще всего, пожалуй, *clear* используется для того, чтобы div подвала страницы установить ниже ваших 2-х или 3-х плавающих колонок.



**Макет документа с «очищенным» подвалом**

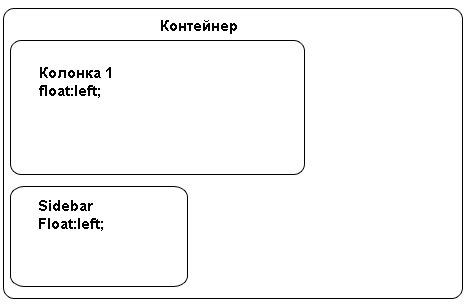


**Макет того же документа, где в подвале не установлено свойство clear**

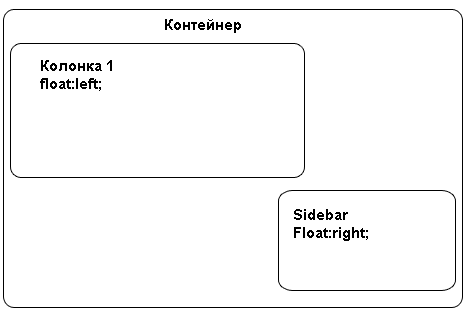
Будте внимательны. При использовании после плавающего элемента в новом боксе правила с *clear*, вы возвращаете этот бокс в нормальный поток, но при этом его свойство *margin-top* перестает работать. Зачастую эту ситуацию исправляют помещая перед требуемым в нормальном потоке боксом пустой div со свойством *clear* —  <div style=»clear:both;»></div>. Справедливости ради надо заметить, что такой прием считается технически правильным, но семантичеки не корректным, т.к. в документе появляется пустой *div* контейнер.

**Контейнер с float элементами**

Первое, что нужно сделать при использовании контейнера для плавающих элементов, это быть уверенным, что ширины контейнера достаточно для установки элементов рядом друг с другом. Если вы установите у двух элементов свойство *float:left* для создания 2-х колоночной компоновки, а общая ширина элементов окажется больше ширины контейнера, то один из элементов перескочит на следующую строку. Это произойдет из-за того, что второй колонке не будет хватать места стоять в родительском боксе рядом с 1-й колонкой. Ниже приведены рисунки, иллюстрирующие подобную ситуацию.



**Общая ширина плавающих элементов превышает ширину своего контейнера, использовано свойство float:left**



**Общая ширина плавающих элементов превышает ширину своего контейнера, колонке sidebar’а установлено свойство float:right**

Ранее упоминалось, что плавающие элементы убираются из нормального потока документа. Это приводит к интересной и запутанной проблеме, когда все элементы внутри контейнера плавающие. Обычно такое наблюдается, например, когда в div контейнере заголовка Web страницы находится изображение логотипа со свойством *float:left*. В заголовке также, как правило, с помощью свойства *background-image*приводится какое-либо фоновое изображение. В сумме имеем контейнер в нормальном потоке, который содержит плавающий элемент и пустой элемент в нормальном потоке, поскольку изображение выведено фоном. Проблема в том, что div заголовка содержит единственный не пустой элемент – плавающее изображение логотипа, которое выпало из нормального потока. Поэтому браузер считает div заголовка пустым и устанавливает его высоту нулевой. Часто при этом говорят, что контейнер схлопнулся. Существует несколько путей решения этой проблемы. Вы можете явно задать высоту div контейнера не меньше высоты изображения. Это будет хорошо работать в случае с заголовком, как в приведенном примере, но такое по ряду причин не всегда возможно. Иногда добавляют пустой div с установленным свойством *clear*, чтобы следующие элементы не были плавающими.

HTML  
‹div class="clear"›‹/div›

CSS  
.clear { clear:both; }

Это удобно, если допускается иметь некоторое пустое пространство после плавающих элементов, т.к. некоторые браузеры для пустых div могут иметь по умолчанию конечные величины для *height, margin, padding* и .т.д.  
Существует также метод использования свойства *overflow* в родительском элементе. Обычно используют *overflow: hidden*, но в зависимости от ситуации вы можете использовать *overflow: auto* или *overflow-y* вместо использования этого свойства в обоих направлениях. Использование свойства *overflow* защищает родительский контейнер от схлопывания.

**Проблемы с float**

**Итоги по float**

При изучении CSS начинающие тяготеют к абсолютному и относительному позиционированию, так как они кажутся проще. Но обычно лучшая практика использовать float. Если вы запомните несколько ключевых моментов, то будете хорошо понимать и управлять плавающими элементами в своих макетах.

* Плавающие элементы выкладываются в нормальный поток документа, затем сдвигаются влево или вправо внутри родительского элемента в соответствии со значением float и удаляются из нормального потока документа.
* Теги с установленными значениями *float* автоматически становятся блочными элементами.
* Блочные боксы с установленным *float* располагаются рядом друг с другом, пока  для них  достаточно места. Если места в контейнере  не хватает, они будут отображаться ниже плавающих элементов.
* Ширина бокса с установленным свойством *float* определяется шириной его содержимого. Рекомендуется при разметке устанавливать ширину float элементам, иначе результат может быть непредсказуем, рекомендация не касается изображений, поскольку для них ширина подразумевается.
* Плавающие и позиционированные элементы различны и ведут себя по разному. Вы не можете использовать свойства*float* и *position* в одном элементе.
* Для сброса свойства обтекания для последующих элементов используйте CSS свойство *clear*.
* *Float* бокс выводится из основного потока, поэтому блочные элементы в нормальном потоке размещаются так, как будто его нет, но строчные боксы этих элементов освобождают место для плавающего блока.
* Контейнер с исключительно плавающими элементами, имеет нулевую высоту, т.к. float элементы выведены из нормального потока, поэтому родитель считает, что он пустой. Существуют способы борьбы с этим эффектом без дополнительной разметки, ознакомиться с ними можно, например, [здесь](http://robertnyman.com/2007/04/12/how-to-clear-css-floats-without-extra-markup-different-techniques-explained/).
* В отличие от элементов основного потока, вертикальные границы *float* бокcов не объединяются с границами соседних блоков.