Практика 2 FLOAT

Последовательность отображения элементов HTML

По умолчанию HTML-документ формируется сверху вниз, от верхнего края окна браузера по направлению к нижнему краю. Такова и очередность появления элементов веб-страницы. Порядок вывода называется потоком документа.

На скриншоте ниже — пример стандартного порядка вывода элементов:

[](https://idg.net.ua/blog/wp-content/uploads/float-01.png)

В CSS существует несколько инструментов для управления этим потоком. Например, вы можете задавать обтекание элементов, а также вытаскивать их из общего потока. Об одном из этих инструментов — свойстве **float** — пойдет речь далее.

CSS float

Элемент, к которому применено свойство float, прижимается к левому или к правому краю родительского элемента (в зависимости от заданного значения), а все элементы, которые расположены ниже, поднимаются и словно обтекают данный элемент. Для лучшего понимания представьте себе камень, который обтекает вода. Собственно, именно поэтому такие элементы еще называют «плавающими».

Данное свойство широко применяется в верстке, поэтому очень важно понять его работу. Float используется для создания макетов с несколькими колонками, навигационных меню, галерей и многого другого.

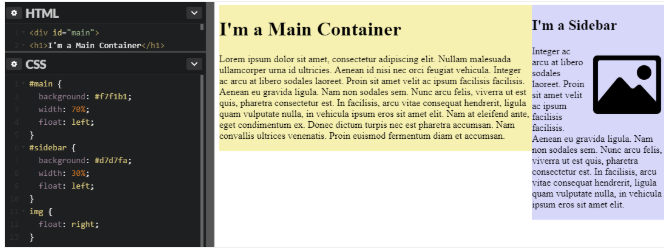
Давайте рассмотрим значения, которые может принимать свойство float. Их всего три:

* **left** — элемент выравнивается по левой стороне. Элементы, находящиеся ниже в потоке, обтекают его с правой стороны.
* **right** — значение, противоположное предыдущему. Элемент выравнивается по правой стороне, а элементы, находящиеся ниже в потоке, обтекают его слева по левому краю.
* **none** — элемент не обтекается и находится в своей обычной позиции.

Как уже упоминалось, элемент со свойством float прижимается к левой либо правой стороне родительского элемента. Это может быть как основной контейнер веб-страницы, так и небольшой блок, скажем, сайдбар.

Применять данное свойство можно как к строчным, так и к блочным элементам. Но поскольку блочный элемент по умолчанию занимает всю ширину контейнера, то при использовании **float** с блоком рекомендуется задать ему ширину **width** и таким образом оставить место для содержимого, которое должно обходить элемент. Иными словами, необходимо дать воде возможность обтекать камень, и не перекрывать поток.

Ниже показан наглядный пример работы свойства float со значениями **left** и **right**:

[](https://idg.net.ua/blog/wp-content/uploads/float-02.png)

Как видите, CSS позволяет легко и быстро создать макет из нескольких колонок, используя всего лишь два свойства в паре — **float** и **width**. А задать обтекание картинки текстом еще легче — мы просто применили к ней свойство **float:right**.

Отмена обтекания: свойство clear

В верстке практически всегда существует необходимость сбросить обтекание после определенного элемента. Например, в случае, когда основной блок веб-страницы обтекается боковой колонкой (сайдбаром), скорее всего, нам понадобится отменить обтекание после сайдбара. Ведь это определенно не последний элемент на странице, и, как минимум, есть еще футер — нижняя часть страницы, «подвал», который должен располагаться внизу, никуда не съезжать и ничего не обтекать.

Сейчас на скриншоте обтекание после сайдбара не отменено, и блок-футер ведет себя странным образом, пытаясь обтекать элемент выше (но ничего не выходит, поскольку ширина как минимум одного элемента этого сделать не позволяет — элемент «застряет»):

Как сделать сброс обтекания? В этом нам поможет свойство clear и его значения:

* **left** — элемент, к которому применяется свойство clear:left, перестает обтекать элемент со свойством float:left, однако правое обтекание сохраняется.
* **right** — элемент, к которому применяется свойство clear:right, перестает обтекать элемент со свойством float:right, однако левое обтекание сохраняется.
* **both** — полностью отменяет обтекание со всех сторон. Элемент смещается вниз и образовывает стандартный поток. На следующие за ним элементы обтекание также перестает влиять.
* **none** — отменяет очистку clear. Элемент будет вести себя, как обычно, и на него будут влиять настройки float.

Воспользуемся полученными знаниями и зададим нашему футеру свойство **clear:both**:

[](https://idg.net.ua/blog/wp-content/uploads/float-04.png)

В итоге футер выровнялся и разместился на своем месте. Кстати, также стал виден результат работы свойства **margin-bottom:10px**, которое было ранее применено к сайдбару.

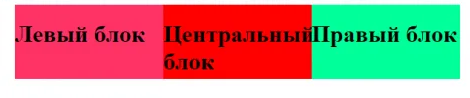


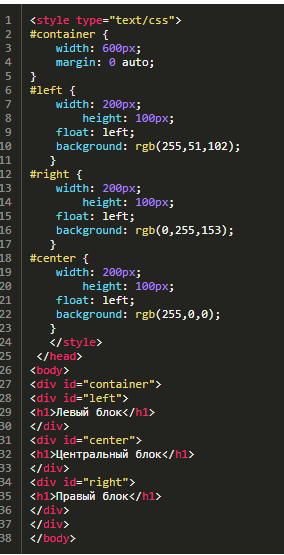
Рассмотрим еще примеры

Пример 1

|  |  |
| --- | --- |
|  |  |

Центрирование слоев

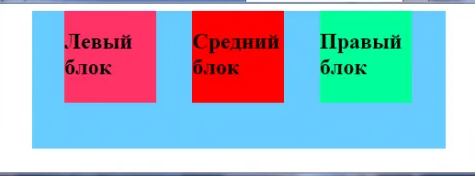


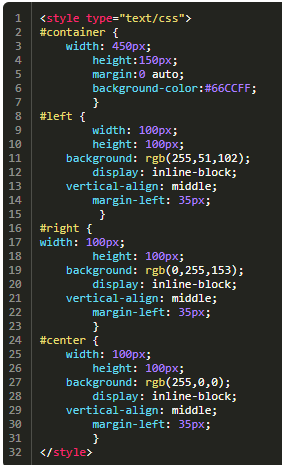


Выравнивание по вертикали

|  |  |
| --- | --- |
|  |  |

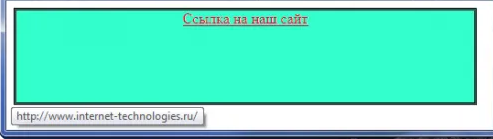
 центрирования слоев внутри контейнера

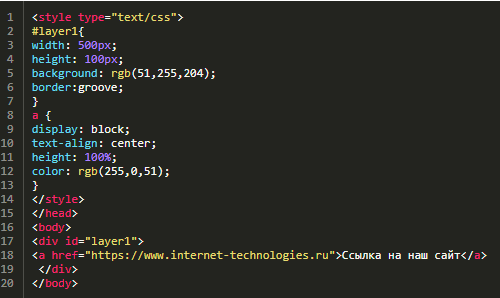




* **display: inline-block** – выравнивает блочный элемент в строку и обеспечивает его обтекание другим элементом;
* **vertical-align: middle** – выравнивает элемент посредине относительно родительского;
* **margin-left** – устанавливает отступ слева.

### Как из слоя сделать ссылку





### Скрытие и отображение блочных элементов



В данном примере расположение **div** блоков никак не меняется. Здесь используется простейшая функция **JavaScript**, меняющая значение свойства **css display** после нажатия на кнопку (***событие onclick***).

Синтаксис **display**:  
**display: block | inline | inline-block | inline-table | list-item | none | run-in | table | table-caption | table-cell | table-column-group | table-column | table-footer-group | table-header-group | table-row | table-row-group**

Как видите, данное свойство может принимать множество значений. Поэтому является очень полезным и может использоваться для позиционирования элементов. В одном из предыдущих примеров с помощью одного из его значений (***inline-block***) мы реализовали выравнивание **div** внутри **div** по центру.

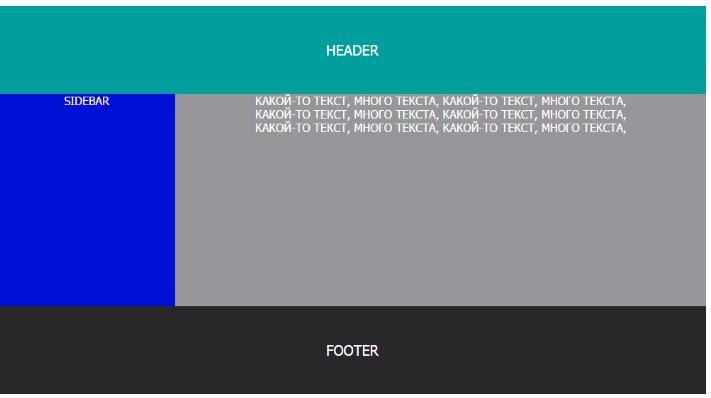
Для скрытия и показа слоя мы использовали два значения свойства **display**:

* **none** – временно удаляет блочный элемент со страницы;
* **block** – показывает элемент как блочный.

Еще одним средством **css**, позволяющим скрывать и показывать элемент, является свойство **visibility**.

Задание

1. **Самостоятельно разработать следующий макет страницы:**



1. **Разработать сайт по следующему макету**

