Вёрстка

По материалам Ивана Сагалакова
http://softwaremaniacs.org/blog/2005/08/03/css-layout-positioning/

Типы вёрстки

- Табличная
- Фреймовая
- Блочная (слоями)

Табличная

- Преимущества
  - Создание колонок
  - «Резиновый» макет
  - «Склейка» изображений
  - Фоновые рисунки
  - Выравнивание элементов
  - Особенности браузеров
- Недостатки
  - Долгая загрузка
  - Громоздкий код
  - Плохая индексация поисковиков
  - Разделение содержимого и оформления затруднено
  - Несоответствие стандартам
  - Невозможность наложения слоёв
  - Трудности при вёрстке сложных страниц

Фреймы

- Преимущества
  - Разбиение на блоки
  - Подгрузка из отдельных файлов
  - Чёткая структура
- Недостатки
  - Проблемы с навигацией
  - Проблемы с индексацией

Блочная вёрстка

- Преимущества
  - Связь со стилями
  - Разделение содержимого от оформления
  - Компактный код
  - Лучшая индексация поисковыми системами
  - Возможность создавать слои (облегчение расположения элементов)
  - Быстрая загрузка страниц
  - Компоновка слоёв по значимости в коде
- Недостатки
  - Разные браузеры по разному отображают (не соответствие спецификациям)

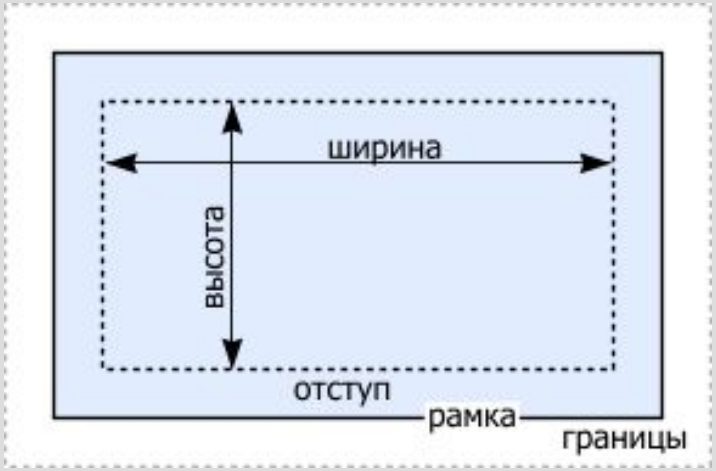
Структура страницы

- Типы
  - Вертикальная
  - Горизонтальная
  - Смешанная

- Блоки
  - Заголовок
  - Контент
  - Подвал

CSS -вёрстка

- Базовая экранная единица — блок (box)
  - Прямоугольная область
  - Могут вкладываться друг в друга
  - Имеют размер




```
<body>
<h1>Заголовок</h1>
<p>Абзац</p>
</body>
```

Части блока

Область содержимого	Content area	Содержимое блока
ширина	width	
высота	height	
отступы	padding	Отступ от невидимых границ содержимого до рамки (фон)
рамка	border	Рамка вокруг содержимого (имеет толщину)
границы	margin	Отступы от рамки до других внешних боксов

Важно!

- В CSS принято, что при задании размеров областей бокса отступы, толщина рамки и границы добавляются к размерам области содержимого

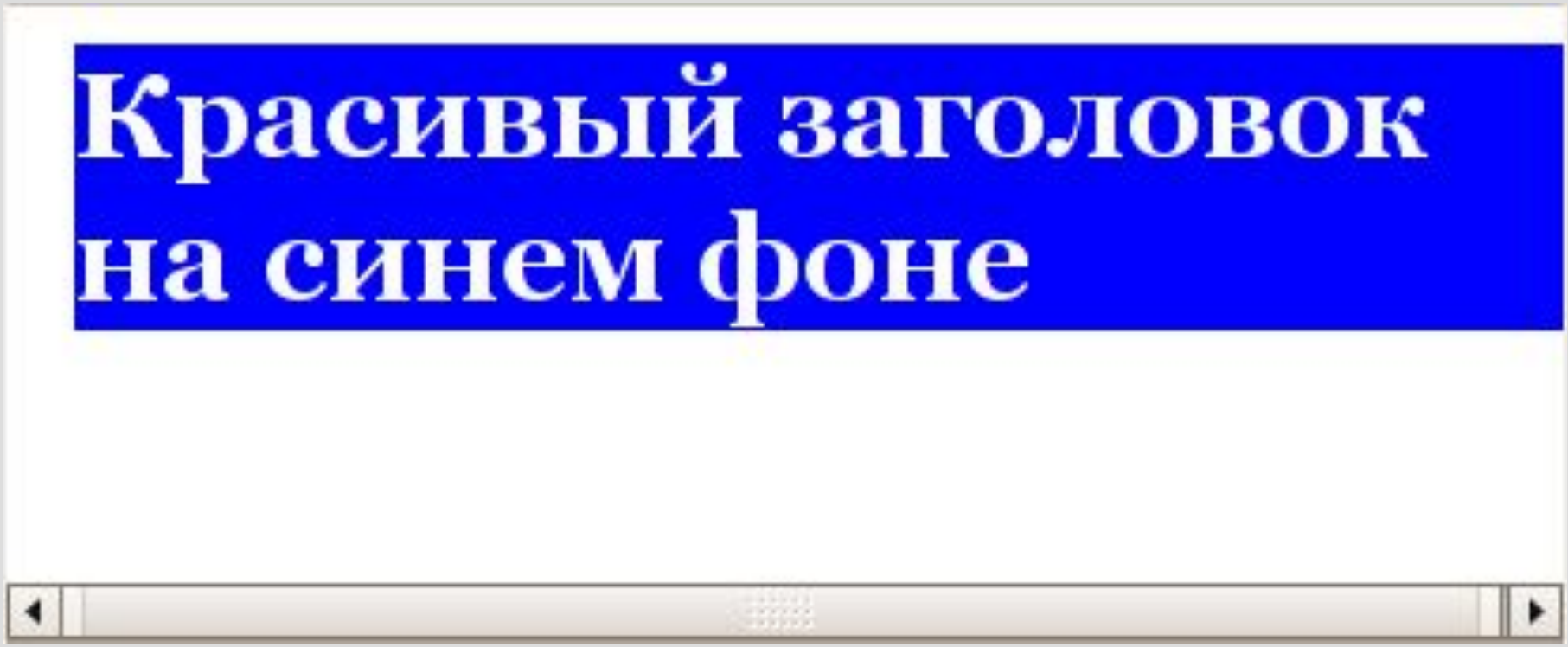
Задача

заголовок на синем фоне, занимает по ширине все доступное пространство, а вокруг него белое пространство в 10 пикселов

```
<body>
  <h1>Красивый заголовок на синем фоне</h1>

h1 {
  width:100%;
  margin:10px;
  color:white; background-color:blue;
}
```

Но ...



Почему?

- границы — по 10 пикселов с каждой стороны — добавились к той ширине, которую мы указали — 100%.
- Общий размер всего бокса будет по ширине на 20 пикселов больше того, в котором он лежит ("контейнера").

Раскладка в CSS

- прямой поток
- позиционирование
- float'ы
- таблицы

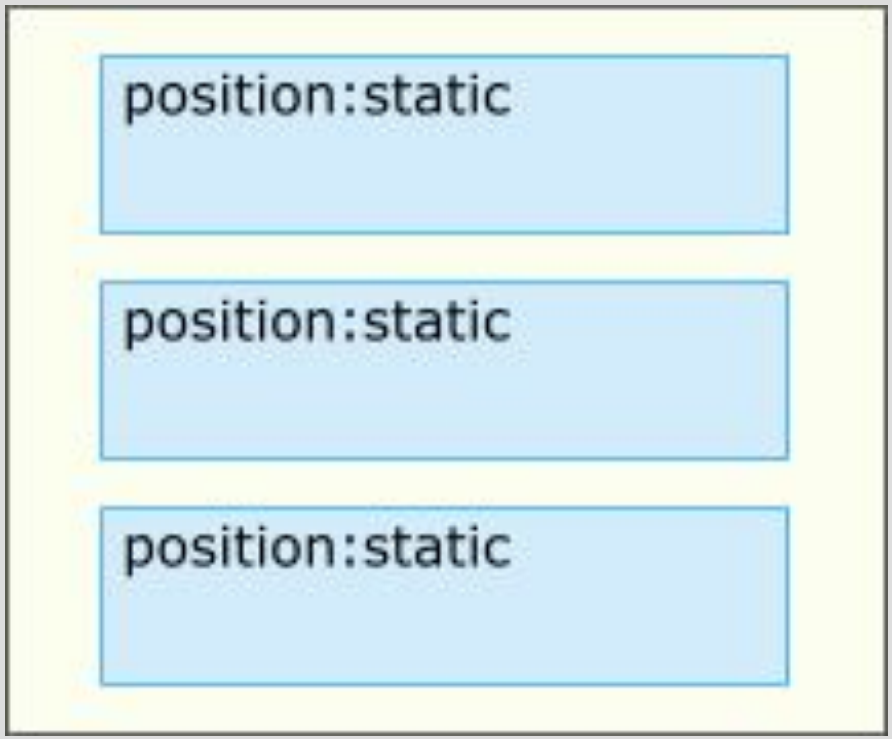
Позиционирование

- четыре способа позиционирования боксов:
  - Static
  - Absolute
  - Fixed
  - Relative

STATIC

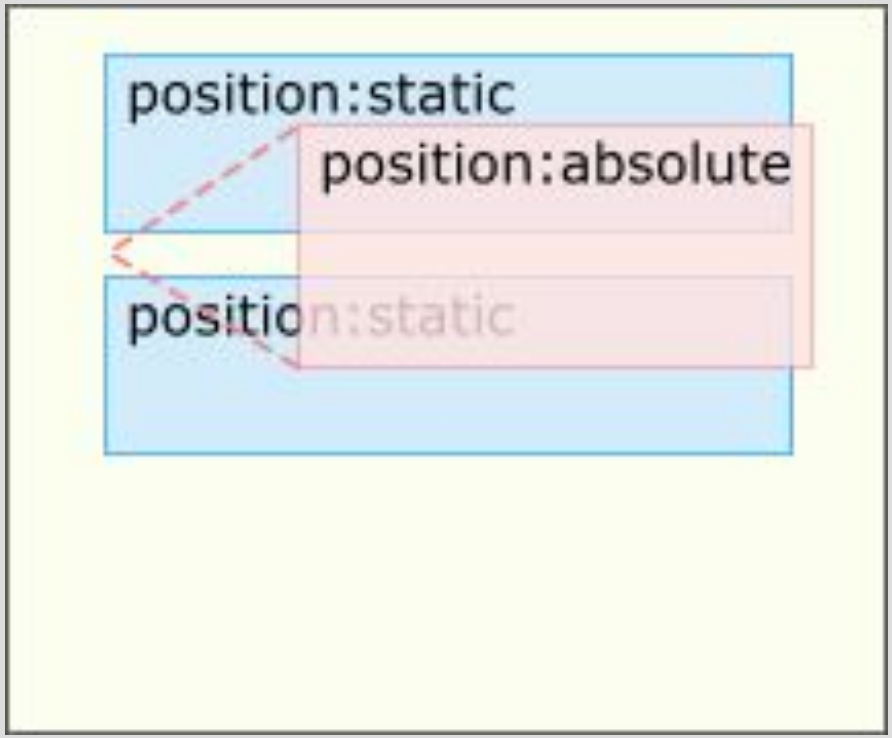
- Это способ по умолчанию или отсутствие какого бы то ни было специального позиционирования
- просто выкладывание боксов одного за другим сверху вниз.

Этот порядок как раз и есть прямой поток.



ABSOLUTE

- Бокс с абсолютным позиционированием располагается по заданным координатам, а из того места, где он должен был бы быть, он удаляется, и в этом месте сразу начинают раскладываться следующие боксы.
- Говорят, что он "исключается из потока".

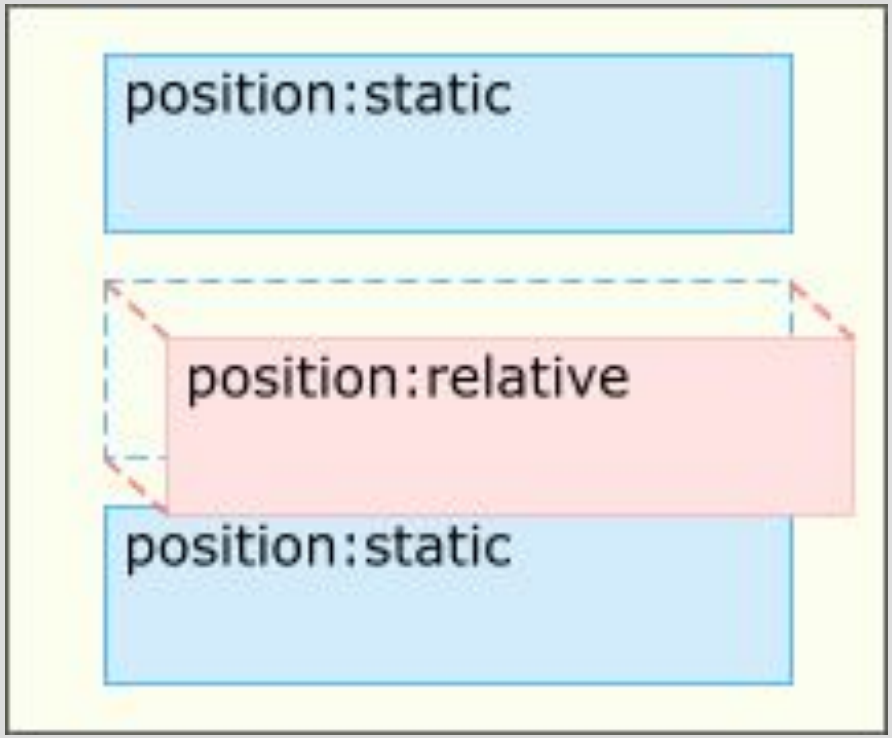


FIXED

- Ведет себя так же, как absolute
- Не скролится вместе с остальной страницей.

RELATIVE

- Такой бокс можно сдвинуть относительно того места, где он был бы в потоке, но при этом из потока он не исключается, а продолжает занимать там свое место. То есть сдвигается со своего места он только визуально, а положение всех боксов вокруг него никак не меняется.



СТАКАН

- Страница начинает раскладываться в своеобразный перевернутый "стакан", начинающийся от верха окна, ограниченный с боков и бесконечно продолжающийся вниз. Если все блоки статические, то они так в этот стакан и раскладываются один за другим. Если в этом потоке появляется позиционированный бокс, то его координаты вычисляются от сторон этого самого стакана.

- Позиционированный бокс внутри себя создает такой же стакан, и все его дети (боксы, находящиеся у него внутри) позиционируются уже относительно него, а не относительно окна. И внутри него происходит то же самое: любой позиционированный (не static) бокс создает внутри себя такой стакан.

- Такой стакан называется "содержащим блоком" (containing block).

Абсолютное позиционирование

```
#somebox {
  position:absolute;
  left:100px; top:100px;
  bottom:100px; right:100px;
}
```

- Координаты означают расстояние бокса от краев: top:0 - бокс прижат к верхнему краю,
- right:10px - отстоит на 10 пикселов от правого края
- Любая из координат необязательна.
- В случае, если координаты не задают вертикального или горизонтального положения, то оно остается таким же, какое было бы без позиционирования.

FIXED

- Бокс с position:fixed - разновидность того же абсолютного позиционирования.
- При скроллинге окна боксы остаются на месте.
- Используется на страницах веб-приложений для всяческих прилипающих блоков меню и тулбаров.

- Для обычных абсолютных боксов браузер всегда сделает достаточно скроллбаров, чтобы их можно было просмотреть.
- Если фиксированный бокс не влезет в окно, то доскролиться до него будет уже нельзя.

АВТОМАТИЧЕСКИЕ РАЗМЕРЫ

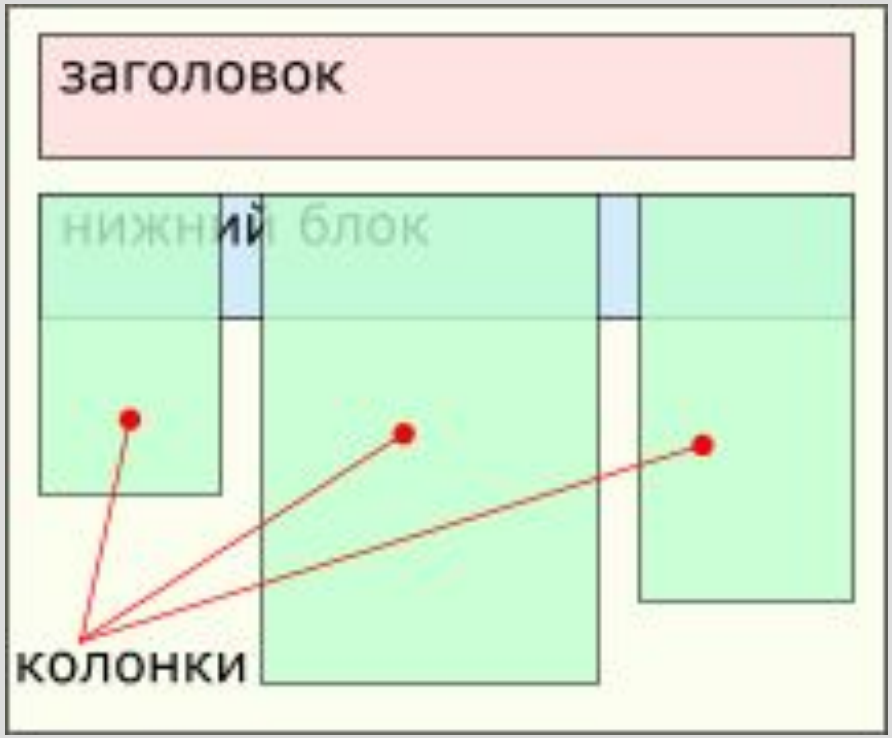
- можно задавать размеры бокса по их внешним границам

```
#somebox {
  position:absolute;
  top:0; left:0; right:0;
  margin:20px; padding:20px;
}
```

- касается боковых сторон своего стакана, какой бы ширины тот ни был,
- margin'ы и padding'и откладываются внутрь бокса
- для неперекрывающиеся боксов на весь экран по определенной сетке

ПРОБЛЕМА

- Абсолютное позиционирование - не универсальное средство.
- координаты и размеры бокса можно задать только относительно содержащего блока, в котором он лежит.
- пример - абсолютным позиционированием нельзя сделать самую традиционную раскладку: заголовок, содержимое любой высоты в несколько колонок и нижний блок.



ПРОБЛЕМЫ

- две проблемы:
- Колонки не получается выровнять по высоте, потому что колонки друг в друге не лежат, и в CSS нет средств сказать "высота как вот у того другого бокса".
- Нижний блок проваливается за колонки, так как они изымаются из потока, его нельзя абсолютно позиционировать под самой высокой колонкой, потому что в CSS нет средств сказать "верх под тем другим боксом".

ОТНОСИТЕЛЬНОЕ ПОЗИЦИОНИРОВАНИЕ

- Относительное позиционирование - похоже на абсолютное, но бокс продолжает занимать место в потоке.

- Чаще же всего position:relative используют вообще без задания смещений. В этом случае он ведет себя как обычный статический бокс, но поскольку он все таки не статический, то он создает внутри себя содержащий блок, тот самый, относительно которого будут позиционироваться боксы внутри него.

ПРИМЕР

```
<div id="header">
</div>
<div id="contents">
  <div id="author">
  </div>
...
</div>
<div id="footer">
</div>
```

- пусть высота заголовка не известна.
- Боксы статические, идут один за другим, и какая бы высота у заголовка ни была, содержимое будет начинаться прямо под ним.
- расположить «об авторе» в правом верхнем углу содержимого

ПРИМЕР

```
#author {
  position:absolute;
  top:0; right:0;
}
```

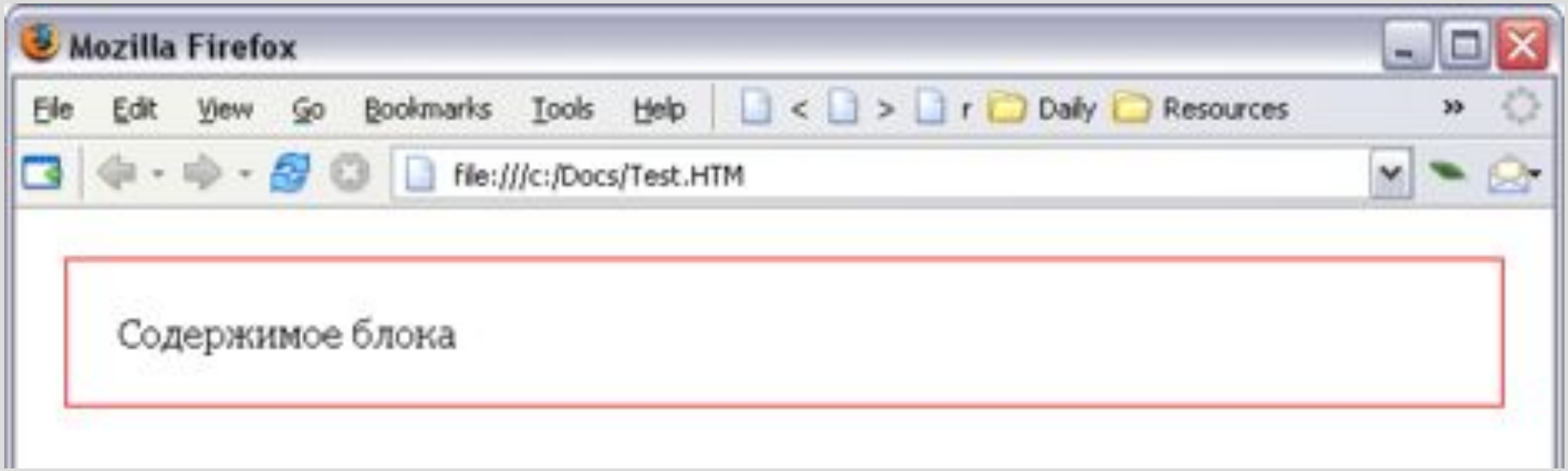
Недостаточно, так как содержащим блоком является все окно, и блок об авторе уедет поверх заголовка. Точно поставить ему расстояние от верха тоже нельзя, потому что размер заголовка у нас может быть разный.

Если сделать блок содержимого тоже абсолютным, тогда он станет содержащим блоком. Но тогда он сам выдернется из потока и низ прижмется прямо к заголовку.

Решение: position:relative.
Блок не исчезнет из потока, но в то же время станет содержащим блоком, и "об авторе" расположится в его правом верхнем углу.

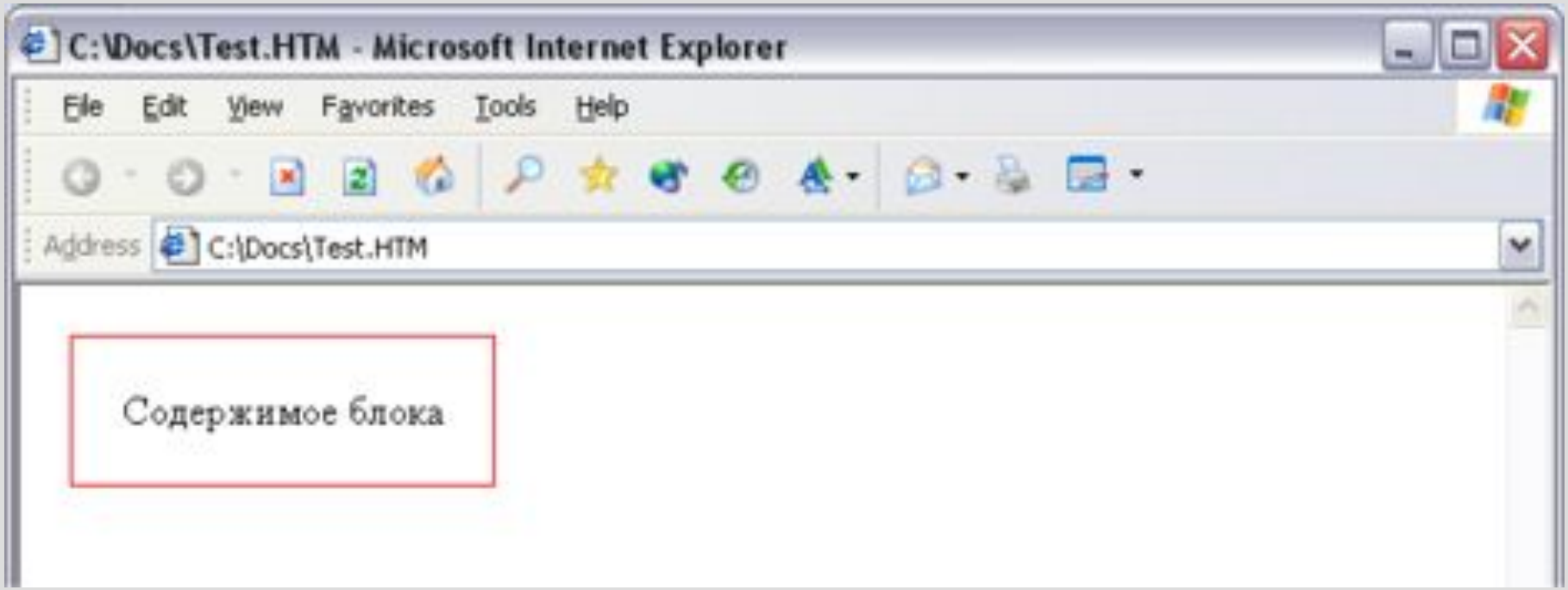
АТОМАТИЧЕСКИЕ РАЗМЕРЫ

- В WinIE не работает задание размеров боксов через одновременную установку координат противоположных сторон. Это, однако, довольно легко обходится использованием интересной способности IE рассчитывать значения свойства для CSS с помощью Javascript. Возьмем тот же пример с боксом во всю ширину и margin'ами и padding'ами по 20 пикселов:


Mozilla Firefox
File Edit View Go Bookmarks Tools Help
Daily Resources
file:///c:/Docs/Test.HTM
Содержимое блока

- Код для этого выглядит так:

- #somebox {
- position:absolute;
- top:0; left:0; right:0;
- margin:20px; padding:20px;
- border:solid red 1px;
- }
- Здесь IE просто проигнорирует right:0 и бокс останется у левого края, обтягивая свое содержимое:


C:\Docs\Test.HTM - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Address C:\Docs\Test.HTM
Содержимое блока

- Чтобы такого не было, ему надо задать какую-то конкретную ширину. Но поскольку указать ее точной цифрой мы не можем, так как неизвестно, какая ширина будет у родительского бокса, то мы заставим IE высчитывать эту ширину:

- #somebox {
- position:absolute;
- top:0; left:0; right:0;
- margin:20px; padding:20px;
- border:solid red 1px;
- width:expression(document.body.offsetWidth-82);
- }
- document.body.offsetWidth - это текущая ширина бокса страницы. А число 82 - это margin'ы+padding'и+border'ы с двух сторон бокса.

- Если нужно взять ширину не "body", а какого-то произвольного бокса, то ему надо дать id (пусть будет "mainbox") и тогда его ширина будет браться так: mainbox.offsetWidth.

•FIXED

- Если не работает position:fixed.
- Существует способ сымитировать поведение фиксированного позиционирования (чтобы бокс не скролился) с помощью абсолютного. Возьмем такую структуру:

```
<div id="contents">
</div>

<div id="menu">
</div>
```

- Мы хотим, чтобы блок меню висел фиксировано в каком-нибудь месте, а содержимое бы свободно скролилось в окне. Если поставить для меню position:absolute, то оно будет скролиться вместе с окном, потому что оно лежит внутри скролящегося окна. Но мы хотим заставить скролиться не окно, а только бокс содержимого:

```
html {
  width:100%; height:100%;
  overflow:hidden;
}

body {
  width:100%; height:100%;
  margin:0; padding:0;
  overflow:auto;
}

#menu {
  position:absolute;
  top:20px; left:20px;
}
```

- Поведение элемента "html" во многом отвечает за поведение окна.
- Мы говорим ему занять все окно целиком и отключаем скроллинг (overflow:hidden). Зато элементу "body" говорим по размеру занять, опять-таки, все окно, а скроллинг включаем (overflow:auto). И теперь абсолютно подвешенный бокс "menu" не двигается, потому что скролится не его содержащий блок "html", а совсем другой - "body".

ЗАКЛЮЧЕНИЕ

- Позиционирование обладает своими плюсами и минусами.
- Самый большой минус - это невозможность завязать размеры и положение произвольных боксов друг на друга, что вынуждает использовать только жестко заданные размеры боксов.
- Самым большим плюсом является полная независимость визуального расположения от порядка элементов в HTML. Это позволяет кардинально менять основную сетку раскладки страницы с минимальными усилиями.