Задание для студентов гр. ИС 2 на период с 06.05 – 15.05.2020

Дисциплина «Математика»

Преподаватель Токарская М.С.
Почта для обратной связи: maya_tok@mail.ru
Тел. 89147174421 – WhatsApp – если есть вопросы.

Все задания отправлять на почту!!!!!

Дисциплина: МДК 05.02 Разработка кода информационных систем

Выполнить тест.

1. Как называется процесс создания web - страниц с помощью специальных приемов:
    a. Верстка сайта
    b. Редактирование кода страниц
    c. Программирование сайта
2. Перечислите требования, выдвигаемые при верстке сайтов?
3. Как называется низко детализированное представление дизайна будущего сайта:
    a. Проект сайта
    b. Вайрфрейм
    c. Фрейм
    d. Макет сайта
4. Как называется совокупность стиля, цветовой палитры, формы и дизайна всех компонентов сайта:
    a. Проект сайта
    b. Вайрфрейм
    c. Фрейм
    d. Макет сайта
5. На какие особенности документа сайта следует обращать внимание в первую очередь:
    a. Цветовая палитра, тип шрифта, дизайн элементов
    b. Ширина документа, высота документа
    c. Наличие места для рекламы, макет сайта.
6. Фиксированный макет сайта используется при решении проблемы:
    a. Ширины документа
    b. Высоты документа
    c. Визуального восприятия документа
7. При формировании какого макета в качестве единицы измерения выступают проценты?
    a. Фиксированного макета
    b. Резинового макета
    c. Трехколонного макета

8. Размещение макета сайта «по центру» помогает визуально сгладить проблему пустого пространства при:
    a. Фиксированном макете сайта
    b. Резиновом макете сайта
    c. Одноколонный макет сайта
9. Какой макет используется для публикации больших статей:
    a. Фиксированном макете сайта
    b. Одноколонный макет сайта
    c. Двухколонный макет сайта
10. Какой макет сайта предпочтительней для вас и почему?

Тема лекции: Варианты верстки сайтов

Задание: составить конспект по теме.

При создании сайта пользуются несколькими вариантами верстки:

a) Верстка с использованием фреймов
b) Табличная верстка
c) Верстка с помощью блоков

Каждый из способов имеет свои преимущества и недостатки, которые мы разберем.

1. Использование фреймов.

Фрейм это законченный отдельный HTML-документ, входящую в общую структуру страницы, и может быть отражен совместно с другими HTML-документами, в браузере

Достоинства фреймов:

- *простота* — web-страница, созданная с помощью фреймов, обычно содержит две изолированных области с навигацией и контентом. Это позволяет, выбрав ссылку на документ в одном фрейме, открыть его в другом. Такая схема работы интуитивно понятна и удобна для пользователей;
- *быстрота* — использование фреймов позволяет увеличить объем информации на каждой странице сайта. Учитывая, что при вызове нового документа перезагружается только один фрейм, а не вся страница в целом, можно ожидать, что скорость работы с сайтом увеличится;
- *размещение* — разместив фрейм в окне браузера, можно не волноваться, что его положение на странице сайта или размеры изменятся в зависимости от объема информации;
- *изменение размеров областей* — при необходимости размеры фреймов можно изменять «на лету» с помощью мыши. Никакие другие инструменты HTML не позволяют это сделать;
- *загрузка* — управление фреймами обеспечивает загрузку контента строго в указанное окно, в то время как остальные остаются неизменными.

Недостатки фреймов:

- *навигация* — при использовании поисковых систем, нажав на ссылку, пользователь перемещается на возможно незнакомый ему сайт. Традиционно для удобства ориентации пользователя на странице сайта размещают его название, заголовок страницы и меню. При использовании фреймов этот принцип нарушается, так как в этом случае заголовок страницы сайта обычно не коррелироваться ни с содержанием, ни с навигацией;

- *плохая индексация поисковыми системами* — связана с тем, что в составе сайта с фреймовой структурой входят файлы с контентом, которые не имеют ссылок на другие документы. При этом главный файл сайта оказывается абсолютно безликим, так как содержит только описание его структуры;

- *внутренние страницы нельзя добавить в «Избранное»* — их адреса скрыты во фреймах и пользователю виден только адрес самого сайта. В результате механизм браузера, позволяющий добавлять страницу в «Избранное», не работает;

- *несовместимость с разными браузерами* — к сожалению, сайты, написанные на базе фреймов, по-разному отображаются в разных браузерах. Это связано с тем, что у каждого из них своя интерпретация, одни и те же параметров.

Пример верстки сайта с помощью фреймов

Пример оформления главной страницы ***ЭОР (электронный образовательный ресурс) «Все о капусте»*** с помощью фреймов.

При разбиении экрана на фреймы (кадры) были заданы две строки **rows="110,*"** и три столбца **cols="20%,*,20%"**.

Использование * дает возможность задать резиновую верстку.

При «резиновой» вёрстке размер страницы задается в процентах к ширине окна браузера, что позволяет (почти) устранить зависимость от разрешения пользовательского монитора.

На нашей странице одновременно отображается четыре html-документа: рис.html , sod.html, istor.html, sod1.html.

Весь блок описания фреймовой структуры должен стоять выше тега **BODY**, иначе в браузере фреймовая структура не просматривается. Тег **BODY** в файле можно удалять.

```
<html>
<head>
        <title>Главная</title>
</head>
<!-- frames -->
<frameset  rows="110,*">
   <frame src="рис.html" marginwidth="10" marginheight="10" scrolling="auto" frameborder="0" target="_top">
   <frameset  cols="20%,*,20%">
      <frame name="" src="sod.html" marginwidth="10" marginheight="10" scrolling="auto" frameborder="0">
      <frame name="текст" src="istor.html" marginwidth="10" marginheight="10" scrolling="auto" frameborder="0">
      <frame name="текст1" src="sod1.html"" marginwidth="10" marginheight="10" scrolling="auto" frameborder="0">
   </frameset>
</frameset>
<body>
</body>
</html>
```

Атрибут *scrolling="auto"* позволяет автоматически появляться полосе прокрутки.

Чтобы загрузить документ в другое окно было проделано следующее. Самому большому кадру из нашего примера присвоено имя **«текст»** - ***frame name="текст"*** (имя, конечно, может быть любым).

А ссылку ставим так:

 target="текст" белокочанная капуста.

Файл beloc.html будет грузиться в целевой кадр, помеченный нами как *текст*

Но если фрейм содержит много ссылок, нет необходимости добавлять атрибуты *target* ко всем тегам <A>. Атрибут *target* задается по умолчанию, так что браузер будет отправлять каждую загружаемую по ссылке страницу в заданный вами фрейм.Чтобы задать атрибут *target* по умолчанию, добавьте следующий тег раздел заглавия (то есть между тегами <head> и </head> **<base target="текст">**

Файлы *sod.html, istor.html, sod1.html* загружаемые в это окно содержат тег <base target="текст">. Параметр *target* (цель) указывает на эту метку.

Чтобы страница грузилась не в кадр, а целиком заняла бы окно браузера, надо поставить в гиперссылке атрибут *target*, указывающий на специальную метку *_top* (вверх).

2. Табличная верстка сайта

Преимущества таблиц:

- *удобство и широкие возможности верстки* — таблица дает возможность удобно формировать web-страницу, используя ячейки в роли модульной сетки, тем более что границы ячеек можно задать невидимыми (нулевой толщины);
- *создание колонок* — формируемый таблицей многоколонный макет сайта удобен в использовании, так как отдельные ячейки в отличии от слоев сохраняют свой вид и не накладываться друг на друга при изменении размера окна браузера. С другой стороны, высота разных колонок, расположенных в одной строке таблицы, остается одинаковой, независимо от объема содержимого каждой из них. Добиться такого же эффекта при использовании блоков достаточно сложно;
- *«резиновый» макет* — благодаря тому, что размер таблицы можно задавать в процентах, они хорошо подходят для «резинового» макета, ширина которого привязана к ширине окна браузера. С другой стороны, в таблицах удобно регулировать и высоту колонок, позволяя отрегулировать ее так, что и при небольшом тексте «подвал» страницы будет плотно прилегать к нижнему краю окна браузера, независимо от размеров окна;
- *«склейка» изображений* — большие изображения удобно разрезать на несколько небольших фрагментов. Собрать изображение в единое целое помогает таблица. При этом некоторые фрагменты могут быть заменены, для них могут быть созданы эффекты перекатывания, анимации и др. При правильной обработке фрагментов изображения можно добиться уменьшения их суммарного объема. С другой стороны, загрузка изображения, состоящего из множества фрагментов, психологически воспринимается пользователем более быстрым;

• *фоновые рисунки* — возможность использовать в ячейках фоновые изображения создает дополнительные возможности при оформлении дизайна сайта. Например, можно создавать декоративные рамки, линии и тени;

• *выравнивание элементов страницы* — контекст, размещаемый в ячейке таблицы, можно выравнивать как по горизонтали, так и по вертикали. Эта придает дополнительную гибкость размещению элементов страницы;

• *особенности браузеров* — в отличие от вольного толкования стилей слоев и при их отображениия, таблицы во всех браузерах выглядят практически одинаково.

Недостатки таблиц:

• *долгая загрузка* — существенный недостаток при использовании таблиц для верстки сайтов. Дело в том, что таблица не отображается на экране, пока не будет полностью загружена. Это делается для того, чтобы собрать всю доступную информацию о таблице для ее правильного отображения. Особенно сильно такое торможение загрузки сайта будет заметно при использовании для верстки большой по высоте таблицы. Обойти этот недостаток можно, если вместо одной большой таблицы использовать несколько маленьких;

• *громоздкий код* — формирование таблицы на языке HTML требует использование сложной структуры вложенных тегов (таблица описывается тегом `<table>`, в который вкладывается тег описания строк `<tr>`, содержащие, в свою очередь, теги ячеек `<td>`). Это не только увеличивает объем программного кода, но и сложность отладки сайта. Проблемы нарастают по экспоненте, если требуется одну таблицу вложить внутрь другой;

• *плохая индексация поисковиками* — при использовании табличного макета основной контекст страницы оказывается раздробленным по отдельным ячейкам таблицы. При этом между фрагментами контекста будут располагаться теги программного кода. В результате контекст отодвинется от начала программного кода, а его прочтение станет затруднено. В результате будет затруднено индексирование страниц сайта поисковиками;

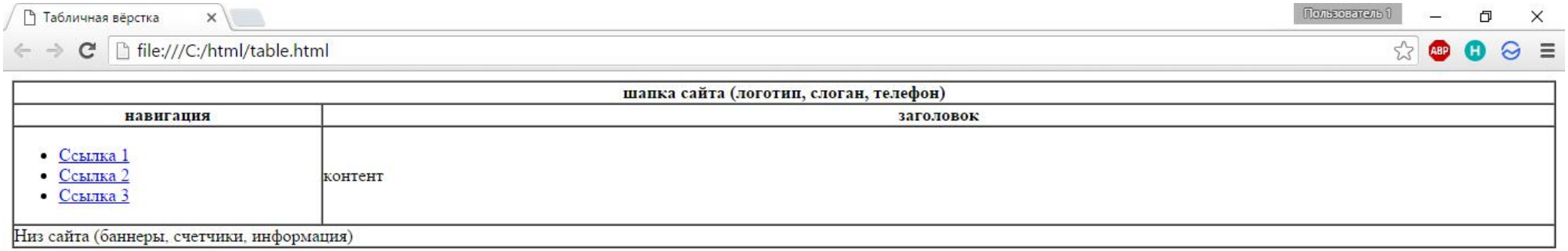
• *нет разделения содержимого и оформления* — обычно при разработке web-сайта разработчики стараются выносить все особенности форматирования в каскадные таблицы, хранящиеся в виде CSS-файлов. Такой подход оправдан, так как позволяет легко переформатировать отдельные элементы сайта, не затрагивая основ его программного кода. При использовании табличного макета далеко не все параметры можно форматировать с помощью стилевых операторов. Это потребует при необходимости изменить внешний вид элементов сайта напрямую корректировать программный код страницы. Учитывая сложную вложенную структуру описания таблиц в HTML (смотри выше), такая корректировка может оказаться далеко нетривиальной;

- *несоответствие стандартам* — на современном этапе разработчики широко используют метаязыки XHTML и XML, которые дают разработчикам большие возможности, чем HTML и CSS. Использование современных метаязыков дает максимальную эффективность при использовании верстки слоями и блоками.

Пример табличной вёрстки:

```
<!DOCTYPE html>
<html>
<head>
   <title>Табличная вёрстка</title>
</head>
<body>
<table border="1" cellpadding="0" cellspacing="0" width="100%">
<tr>
<th colspan=2>шапка сайта (логотип, слоган, телефон)</th>
</tr>
<tr>
<th width="20%">навигация</th>
<th width="80%">заголовок</th>
</tr>
<tr>
<td width="20%">
<ul>
<li><a href="index.html" title="Ссылка 1">Ссылка 1</a></li>
<li><a href="index.html" title="Ссылка 2">Ссылка 2</a></li>
<li><a href="index.html" title="Ссылка 3">Ссылка 3</a></li>
</ul>
</td>
<td width="80%">контент</td>
</tr>
<tr>
<td colspan=2>Низ сайта (баннеры, счетчики, информация)</td>
</tr>
</table>
</body>
</html>
```

Если поместить этот код в тело HTML-документа и открыть получившийся файл в браузере, то страница будет выглядеть так:



Вот и вся разметка. Для создания страниц вам остаётся только сделать копию файла и вместо отображаемых слов вставить свой контент. Так, копируя файлы и редактируя содержащийся в ячейках текст, вы можете создать сколь угодно большой веб-сайт с табличной вёрсткой.

Пояснения к коду:.

```
<table border="1" cellpadding="0" cellspacing="0" width="100%">
```

Тег <table> открывает таблицу.

Атрибут border задаёт толщину табличных рамок.

Cellspacing устанавливает расстояние между ячейками. В данном случае оно сделано нулевым, чтобы таблица не расползалась.

```
<th colspan=2>шапка сайта (логотип, слоган, телефон)</th>
```

<th> — открывающий тег ячейки заголовка таблицы. В отличие от других, текст в этой ячейке будет выровнен по центру и выделен полужирным.

Colspan — атрибут, значение которого определяет, сколько ячеек по горизонтали относительно других строк текущий элемент будет занимать.

</th> закрывает ячейку.

Текст между тегами <th> и </th> — это и есть содержимое ячейки, ради него всё и делалось.

</tr> — конец строки.

```
<th width="20%">навигация</th> <th width="80%">заголовок</th>
```

100% ширины страницы разделены на две части: 20% отдал под блок навигации, 80% — под основной контент.

```
<td width="20%">
<ul>
<li><a href="index.html" title="Ссылка 1">Ссылка 1</a></li>
<li><a href="index.html" title="Ссылка 2">Ссылка 2</a></li>
<li><a href="index.html" title="Ссылка 3">Ссылка 3</a></li>
</ul>
```

Вёрстка блока навигации. Создаю ячейку, занимающую 20% ширины таблицы. Внутри тегов <ul> расположен список ссылок. На его основе можно сделать меню сайта.

<tr><td colspan=2>Низ сайта (баннеры, счетчики, информация)</td></tr>

В следующей строке настраиваю подвал (низ) сайта. Для этого использую уже знакомые теги. При помощи colspan=2 делаю, чтобы низ сайта занимал по горизонтали сразу две ячейки, как шапка — эти части, в отличие от тела страницы, на блоки в моём случае делить не нужно.

Таким образом, для создания табличной вёрстки сайта, содержащего все основные элементы (шапку, низ, меню, контент страницы) понадобилась таблица, состоящая из четырёх строк и шести ячеек (строки головы и ног сайта содержат по одному элементу).

На данный момент фреймовая и табличная вёрстка сайта считаются устаревшими.

На современных сайтах практикуется блочная верстка.

3. Верстка сайта с помощью блоков

При использовании блочной верстки каждая область принятого к разработке макета страницы помещается в свой блок, а каждый блок наполняется содержимым средствами HTML. Блоки размещаются, позиционируются и оформляются с помощью CSS-разметки.

Таким образом, конечный web-документ представляет собой набор блоков с контентом внутри, а оформление зачастую находится в отдельном CSS-файле, подключенном к странице.

Преимущества блоков:

- *меньший объем кода* — код страницы уменьшается и становится гораздо логичнее и понятнее. Учитывая, что блоки отображаются по мере своей загрузки, уменьшается время загрузки страницы и нагрузка на сервер;
- *удобство изменения элементов* — практически любой из элементов внешнего вида сайта можно изменить путем правки файла стилей, не трогая код самой web-страницы;
- *преимущества в плане SEO* — при табличной верстке контент разбросан по разным ячейкам, что усложняет индексацию поисковыми роботами. При блочной верстке такой проблемы не существует, поскольку в коде страницы основное место занимает контент. Кроме того, и на странице он оказывается размещенным семантически правильно;

- *реализация сложных задач* — с помощью блоков легко реализуются различные задачи по нестандартному расположению, вложению и оформлению элементов сайта. Посредством нескольких инструкций можно реализовать такие расширенные возможности, как выпадающие списки меню, эффекты скругленных углов и теней, а также другие визуальные эффекты, которые делают сайт более выразительным и понятным для посетителя;
- *возможность создания адаптивного дизайна* — в виде создания специальных сайтов, суть которых состоит в создании макета, который одинаково корректно отображается на устройствах с различным разрешением экрана.

Недостатки блоков:

- *сложность в освоении* — если сверстать web-сайт таблицами может и новичок в сайтостроении, то освоить блочную верстку сложнее, так как управление блоками требует хороших знаний не только HTML, но и CSS;
- *кроссбраузерность* — определенные инструкции web-страницы, сверстанной блоками, по-разному воспринимаются разными браузерами, что может привести к изменению ее внешнего вида;
- *проблемы масштабирования* — с уменьшением разрешения блоки спадают или наезжают друг на друга, что разрушает планируемый вид web-страницы.

Пример блочной вёрстки

Допустим, у нас есть макет (рисунок ниже).

Согласно макету, страница сайта будет содержать пять блоков: «шапку», навигационное меню, боковую панель, основной блок с контентом и «ноги».

Сначала создадим HTML-страницу: обозначим структуру, разметим её. HTML-код будет таким:

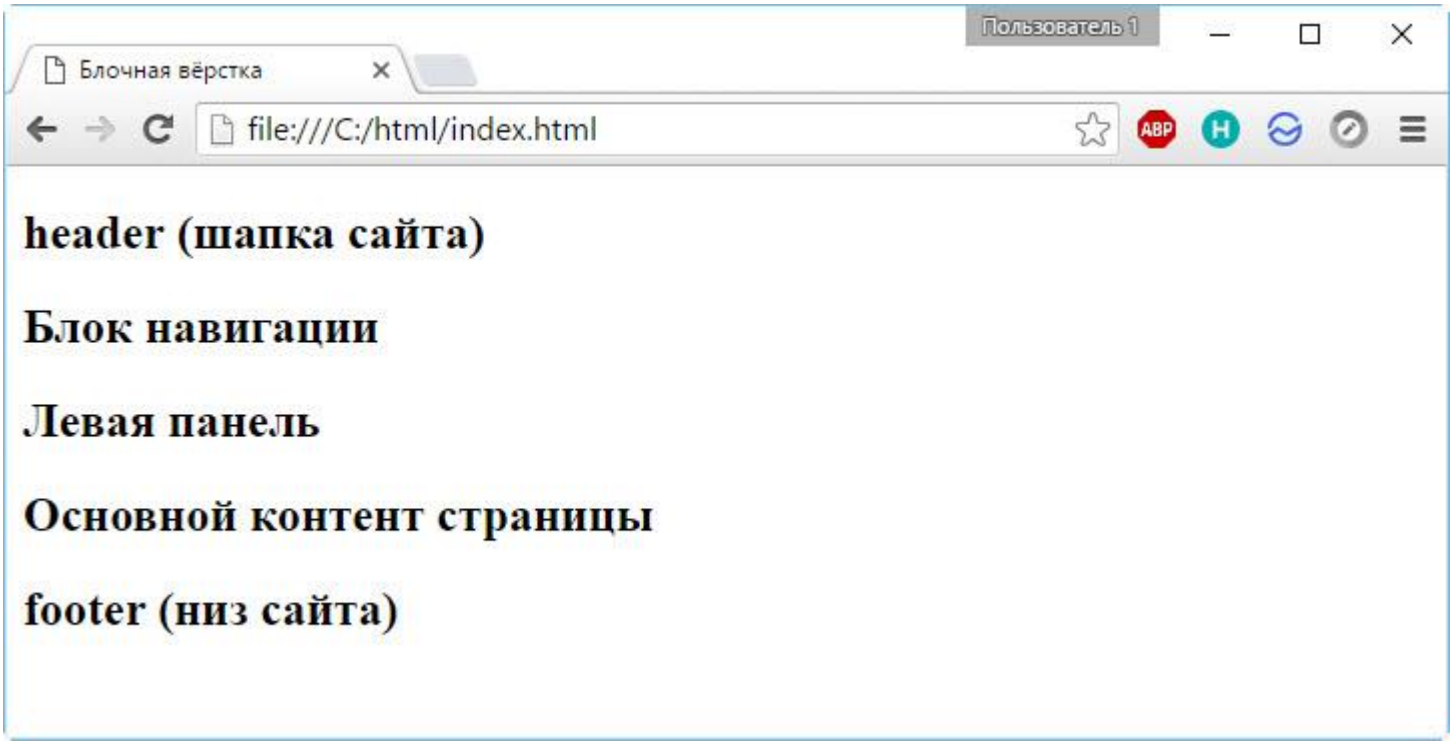
```
<!DOCTYPE html>
<html>
<head>
   <title>Блочная вёрстка</title>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
<body>
<div id="container">
        <div id="header">
        <h2>header (шапка сайта)</h2>
        </div>
        <div id="navigation">
        <h2>Блок навигации</h2>
        </div>
        <div id="sidebar">
        <h2>Левая панель</h2>
        </div>
        <div id="content">
        <h2>Основной контент страницы</h2>
        </div>
        <div id="clear">
        </div>
        <div id="footer">
        <h2>footer (низ сайта)</h2>
        </div>
</div>
</body>
</html>
```

Разберём некоторые моменты.

 — это блок-родитель, внутри которого расположились все остальные блоки. Как ячейки таблицы внутри <table>. Назначение дочерних контей-

неров должно быть понятно, за исключением разве что блока . Это вспомогательный слой, его смысл будет понятен, когда вы увидите код CSS.

Если открыть HTML-файл в браузере, не подключая таблицу стилей, страница будет выглядеть так.



Теперь добавим файл CSS, код которого приведён ниже.

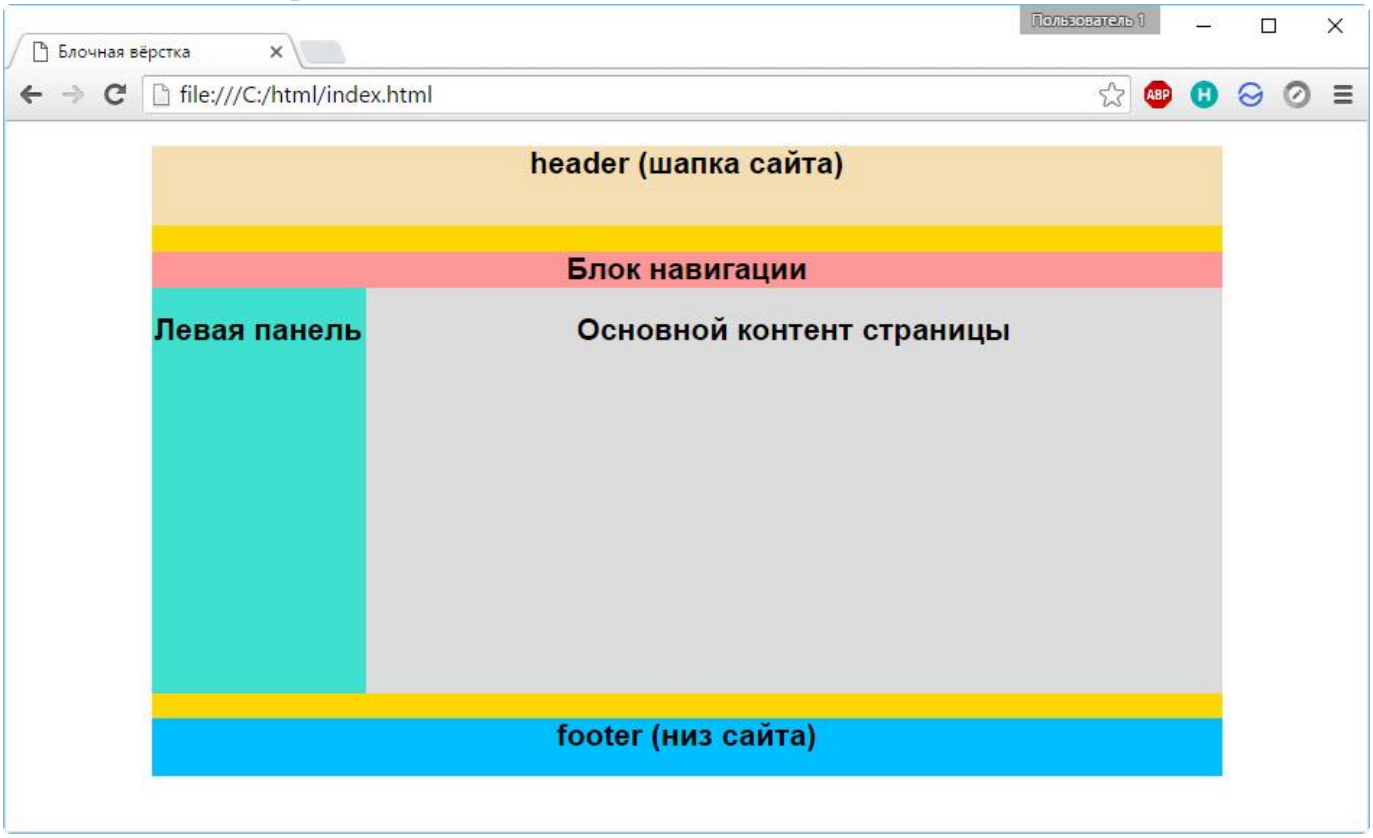
```
body {
    background: #FFF;
    color: #000;
    font-family: Arial, sans-serif;
    font-size: 14px;
}
#header {
    background: #F5DEB3;
    width: 100%;
    height: 55px;
}
#container {
    background: #FFD700;
    margin: auto auto;
    text-align: center;
```

```
    width: 80%;
    height: 400px;
}
#navigation {
    background: #FE9798;
    width: 100%;
    height: 25px;
}
#sidebar {
    background: #40E0D0;
    float: left;
    width: 20%;
    height: 280px;
}
#content {
    background: #DCDCDC;
    float: right;
    width: 80%;
    height: 280px;
}
#clear {
    clear: both;
}
#footer {
   background: #00BFFF;
        width: 100%;
        height: 40px;
}
```

С помощью стилей мы последовательно оформляем содержимое тега <body> и всех находящихся внутри контейнеров с помощью ранее изученных правил.

#clear { clear:both; } запрещает обтекание элемента слева и справа. Если убрать это правило, вёрстка «поедет» и низ сайта перестанет корректно отображаться.

Вид сайта после применения CSS:



Вот и весь смысл блочной структуры. Дальше можно только наполнять сайт содержимым и усложнять оформление, но делаться это будет всё равно по изложенному выше принципу.

Практическая часть.

Лабораторная работа № 1 «Макет сайта с одноколонной сеткой»

Цель: научиться разрабатывать сайты с одноколонной сеткой, используя различные способы верстки сайта.

План работы:

1. Повторить материал лекции
2. Ознакомиться с полученным материалом
3. Создать сайт по своему варианту с использованием фреймов (набирать текст можно в любом текстовом редакторе: Блокнот, Notepad++ и т.д.).
4. Составить отчет по работе: документ Word с темой, целью ЛР. Кроме того необходимо сделать скрин получившегося сайта и прикрепить ссылку на него.
5. К присланному отчету прикрепляем файлы текстового редактора с набранными html – документами (все файлы собираем в архив и даем ему название – Сказка_фреймы).
6. Перед тем, как выполнить работу – найдите текст сказки (допускается ее краткое изложение):

№	ФИО		№	ФИО	
1.	Антипенко А.	Колобок	10	Русяев Р.	Сказка о глупом мышонке
2.	Вылегжанина Т.	Сказка о золотом петушке	11	Сабецкий С.	Сказка об умном мышонке
3.	Грушенец А.	Гуси-лебеди	12	Табула П.	У страха глаза велики
4.	Корниенко Р.	Заюшкина избушка	13	Тишкин В.	Рукавичка
5.	Красикова А.	Теремок	14	Хохрин Е.	Лиса, заяц и петух
6.	Куринный М.	Репка	15	Шевченко Д.	Лиса и рак
7.	Лаврихин Д.	Волк и семеро козлят	16	Шеина В.	Лиса и журавель
8.	Лобякова В.	Курочка Ряба	17	Щерба Е.	Петушок и бобовое зернышко
9.	Проць В.	Сказка о рыбаке и рыбке (краткое изложение)		http://dlya-detey.com/skazki/ - детские сказки	

Задание:

На оценку «3» - набрать готовый текст примера сайта.

На оценку «4» - набрать готовый текст примера, изменив только сказку по своему варианту

На оценку «5» - набрать готовый текст, изменив сказку по своему варианту, добавив строки в меню (минимум 6), в «подвале» сайта под надписью – «Файл разработан, как пример использования фреймов» добавить ФИО разработчика и название организации.

Макет с одноколонной сеткой.

Разработаем пример сайта на базе фреймов.

index.html

```
<html>
 <head>
  <title> Макет одна колонка </title>
 </head>
 <frameset rows="20%,*%,7%">
```

```
    <frameset cols="30%,*">
    <frame src="menu.html" name="MENU">
    <frame src="top.html" name="TOP">
   </frameset>
   <frame src="content.html" name="CONTENT">
   <frame src="boot.html" name="BOOT">
 </frameset>
</html>
```

top.html

```
<html>
 <head>
  <title>Шапка</title>
 </head>
 <body>
  <br>
  <font color="Mediumvioletred">
  <h1 align="center">Сказки нашего детства</h1>
 </body>
</html>
```

menu.html

```
<html>
 <head>
  <title>Меню</title>
 </head>
 <body>
  <font size="4">
  <p align="center"><b>Выбери сказку:</b></p><br>
  <a href="1.html"> Золотой петушок</a><br>
  <a href="2.html"> По щучьему велению</a><br>
  <a href="content.html"> Три медведя</a>
 </body>
</html>
```

content.html

```
<html>
 <head>
  <title>Контент</title>
 </head>
 <body>
  <h1 align="center">Сказка "Три медведя"</h1>
  <img src="Tri.jpg" height="200" align="left">
  <font size="5">
    Одна девочка ушла из дома …<br>
    Дверь была отворена… <br>
    В домике было две комнаты… <br>
    Девочка взяла самую большую…
 </body>
</html>
```

boot.html

```
<html>
 <head>
  <title>Подвал</title>
 </head>
 <body>
  <font size="4"> &copy; Файл разработан как пример
                  использования фреймов.
 </body>
</html>
```