**Лабораторная работа №2**

***Создание структуры проекта. Выбор текстового редактора. Базовая разметка страницы***

***Создание структуры проекта***

Создание любого проекта начинается с создания базовой структуры папок. Эта структура является однотипной:

В удобном для вас месте создайте папку с вашим будущем проектом. В корне этой папке создайте папку “static”(здесь будут хранится статические файлы: изображения, стили) и документ “index.html”(это базовый файл с которым нам доведётся работать). В папке “static” создайте папку “css” и “images”. Последним штрихом добавляем файл “style.css” в папку “css”.

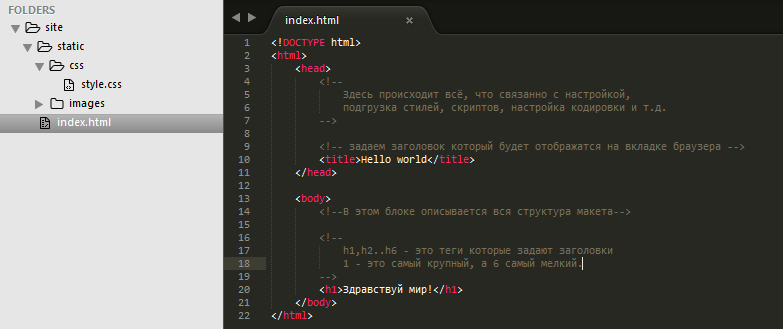
***Примечание***: Кроме того более динамические сайты используют для интерактивные элементы которые создаются с помощью JavaScript. И файлы со скриптами хранятся еще в папке “js”, которая находится в папке “static”.

***Выбор текстового редактора***

HTML - cтраницу можно сверстать используя Блокнот или Notepad++, но на данный момент существуют редакторы кода, имеющие огромный функционал в виде подсветки синтаксиса, открытие целых директорий, множества файлов, разбитие на несколько окон и т.д. Одни из самых популярных это SublimeText, Brackets, Atom. Весь код представленный в дальнейшем в описании лабораторных работ будет оформлен в SublimeText3.

***Базовая разметка страницы***

Откройте файл“index.html” и перепишите вручную текст с картинки. Всё, кроме того что заключено в теги вида <!-- какой-то коментарий -->(текст в данных тегах не будет отображаться в веб браузере). И посмотрите, что в браузере получилось.

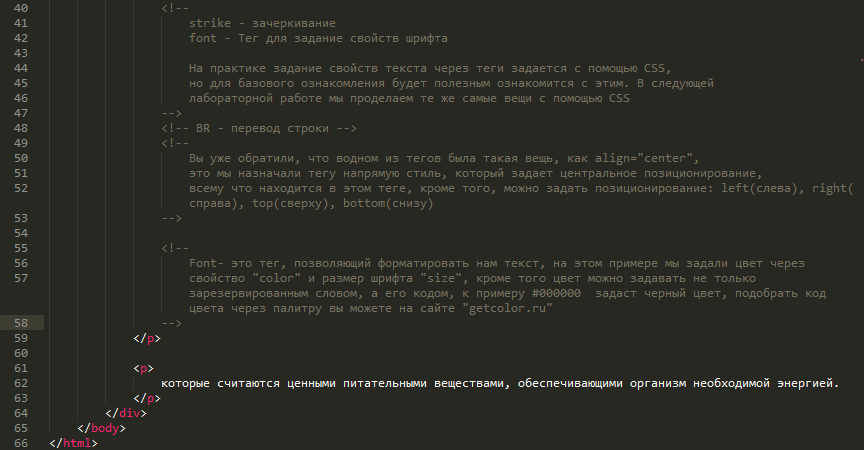
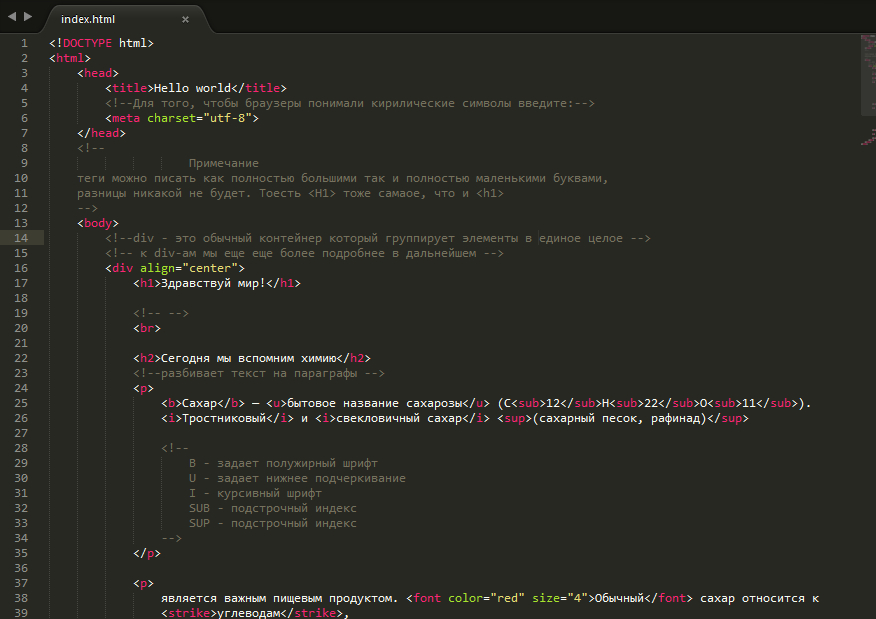


В результате вы написали базовое приложение типа “Hello world”. Самым главным, что стоит запомнить из этого примера, что теги - это элементы языка вида <имя тега> и они обязательно должны закрываться: </имя тега>. В дальнейшем содержимое тегов может форматироваться. Указанная структура документа является базовой и обязательной.

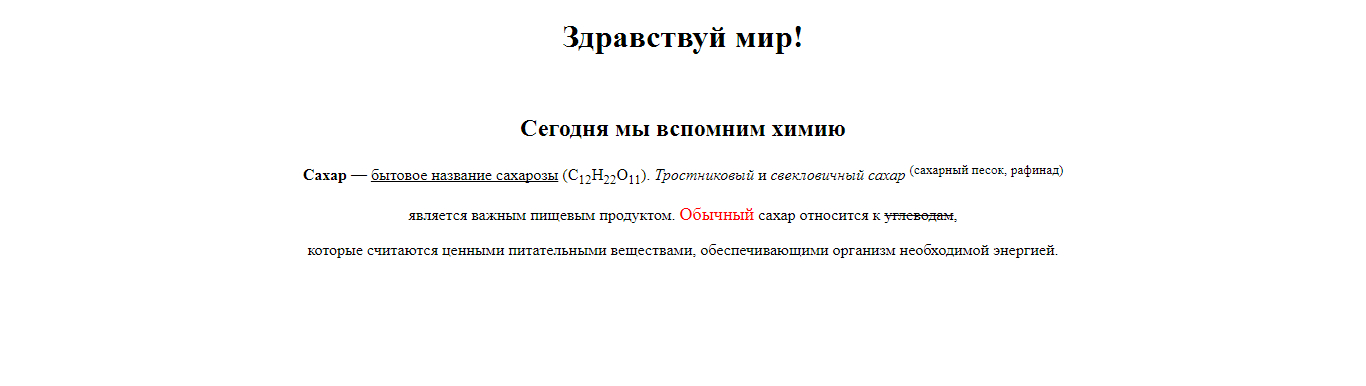
**Лабораторная работа №3**

***Форматирование. Работа с текстом.***

Дальнейшую работу продолжим в документе, созданном в лабораторной работе №2.

****

В комментариях вы можете увидеть описание каждого нового тега. Открывая страницу в браузере, вы получите такой результат:

****

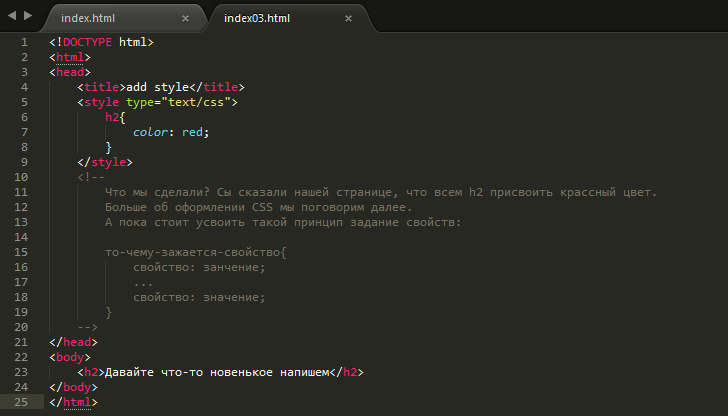
**Лабораторная работа №4**

***Обьявление стилей. Базовые свойства стилей стилей.***

Рассмотрим форматирование текста в HTML-документе с помощью таблицы каскадных стилей (далее CSS - CascalSheetStyles).

Существуют различные способы объявление стилей: в самих тегах, в теге <style></style>, и в отдельном файле. Мы же рассмотрим последние два метода.

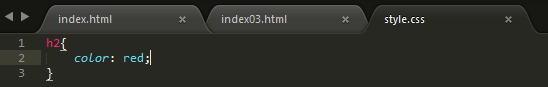
Для начала создайте новый документ «index03.html» и задайте ему базовую структуру, как в лабораторной работе № 2 и новое содержимое блока <style> как в примере:



Чтобы ещё меньше нагружать наш документ стилями, мы подключим внешние стили. Для этого вместо блока <style> вставьте эту строчку:

OnPaste.20170908-012227

Затем открываем файл “style.css” из “static/css” и добавляем следующий код:



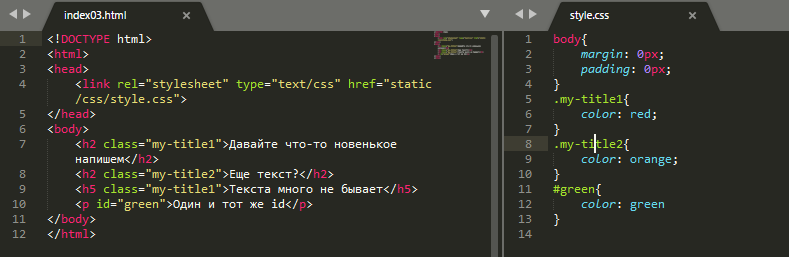
Обновив страницу, вы обнаружите, что заголовок так и останется красным. Так и должно быть, так как вы логику описания стилей перенесли в отдельный файл.

С объявлением мы разобрались, но что если мы не хотим делать так, чтобы все элементы h2 принимали красный цвет, или даже чтобы только один элемент имел собственный стиль? К нам на помощь приходят классы и идентификаторы.

Как работают классы? Для того чтобы тегу присвоить класс в самом теге вам надо написать class=”именна\_классов\_через\_пробел”(имена должны писаться латинскими символами и использовать из сторонних символов только дефис и нижнее подчеркивание),

В самом же файле стилей, стилю задаются свойства таким образом .имя\_класса{свойства как обычно}. С идентификатором(id) дела обстоят почти также. В теге объявляется через id=”имя\_id” а в стилях #имя\_id{свойства}. А в чем же собственно разница? Разница в том, что id это уникальное имя которое следует объявлять только лишь раз один раз на странице, а класс сколько угодно раз.

Для большего понимания повторите следующий код и посмотрите результат:



Полученный результат для вас должен стать очевидным и понятным.

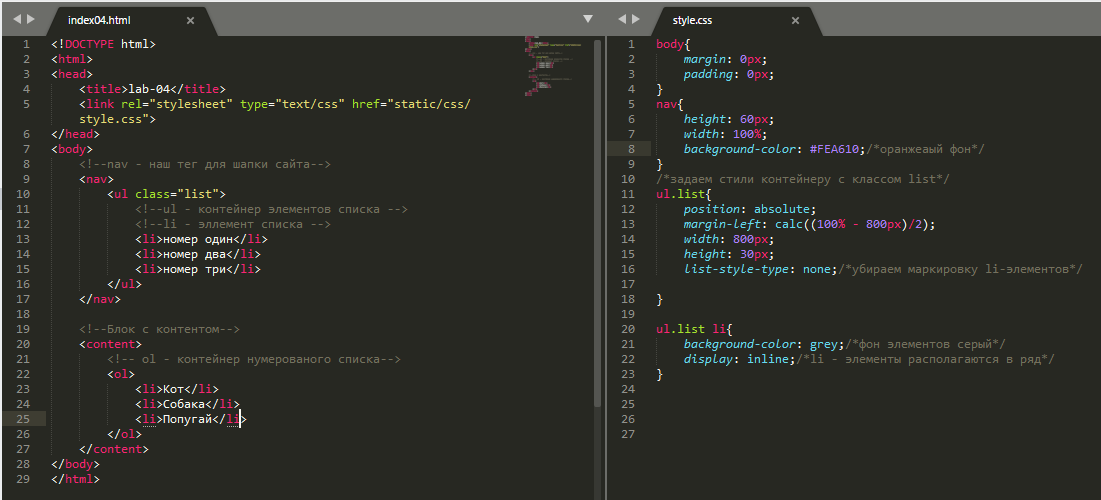
P.S. Если в дальнейшем css вызывает трудности то можно воспользоваться сайтом <http://htmlbook.ru/css>. Здесь можно найти как описание тегов с подробным описанием поддержки тегов в различных браузерах так и различные миниуроки.

**Лабораторная работа №5**

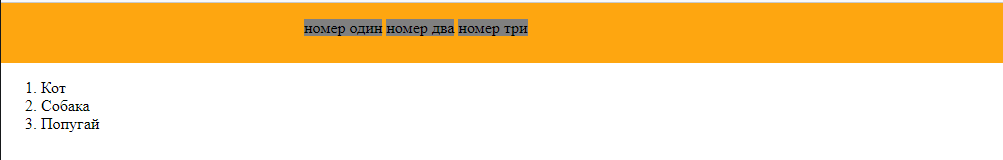
***Списки. Типы списков. Ссылки. Hover-эффекты.***

Для отображения различных элементов в связанной структуре используют такой элемент, как списки. На следующем рисунке представлен код по созданию простой шапки сайта с использованием немаркированных списков.

Прежде чем повторять код, примите к сведению, что у элементов <li> (элемент самого списка), есть html-атрибут «type». Этот атрибут может принимать различные значения, например «disc» (элементы списка будут иметь круглый черный маркер), «circle»(маркеры будут в виде кольца) и «square»(квадратные маркеры).

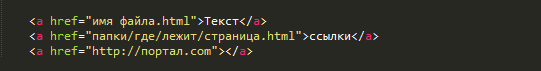


В результате вы будете иметь такой результат:



Давайте теперь создадим на элементах списка ссылки и добавим к ней простую анимацию.

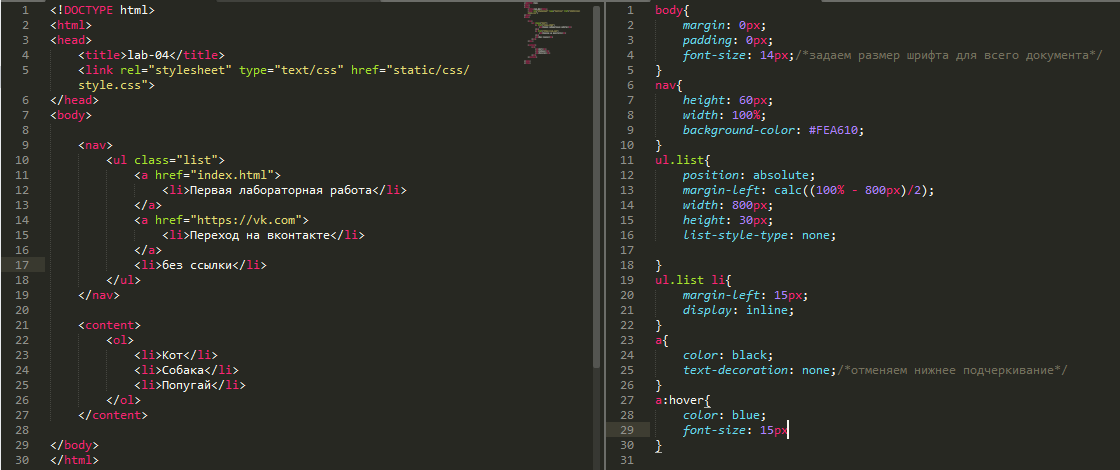
Ссылки описывает тег <a>. Основной атрибут этого тега является «href», href – это ссылка на страницу. Ссылки бывают такого типа:



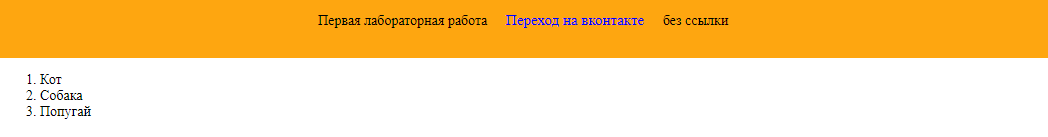
Кроме того в ссылку можно оборачивать любой объект (тег), что либо: текст, изображение, блок и т.д.

Стандартно HTML задает тексту в ссылке подчеркивание и голубой цвет шрифта и изменение цвета шрифта ссылки на фиолетовый, после перехода по этой ссылке. Но с помощью css или тегов html атрибутов можно изменить эти стандартные свойства.

Кроме того в CSS есть такое понятие, как псевдокласс. **Псевдокласс** в **CSS** - это ключевое слово, добавленное к селектору, которое определяет его особое состояние. Например, :hover применит стиль, когда пользователь наводит курсор на элемент, указанный селектором. Как раз этот псевдокласс и применим к ссылке и посмотрим, что получится.



Результат:

****

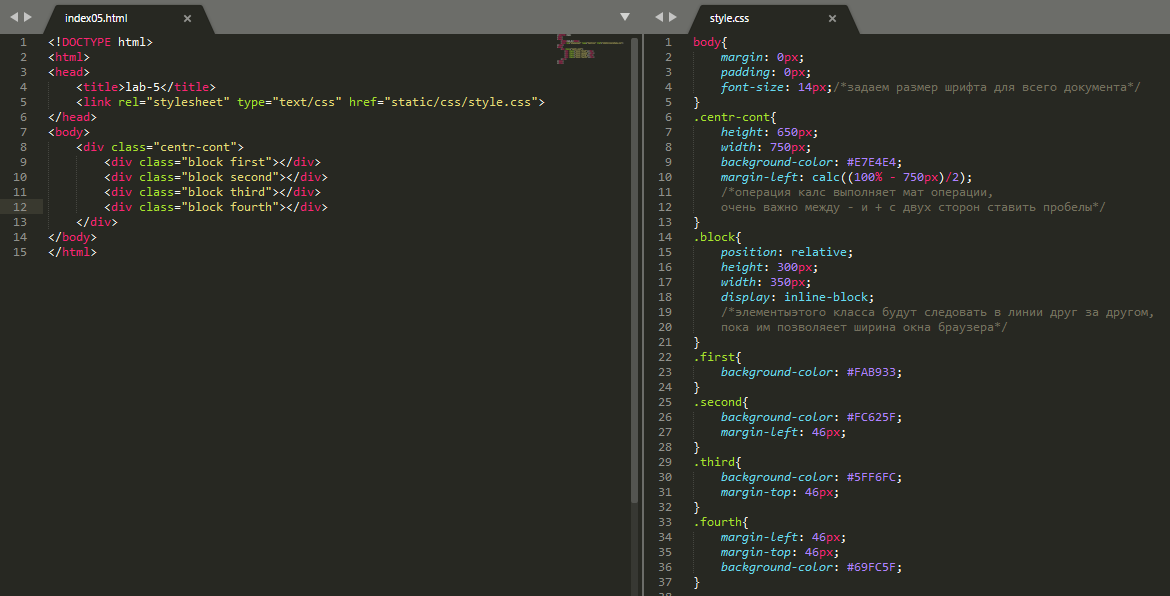
Пример совсем не совершенен, но позволяет опробовать новые изученные свойства.

**Лабораторная работа №6**

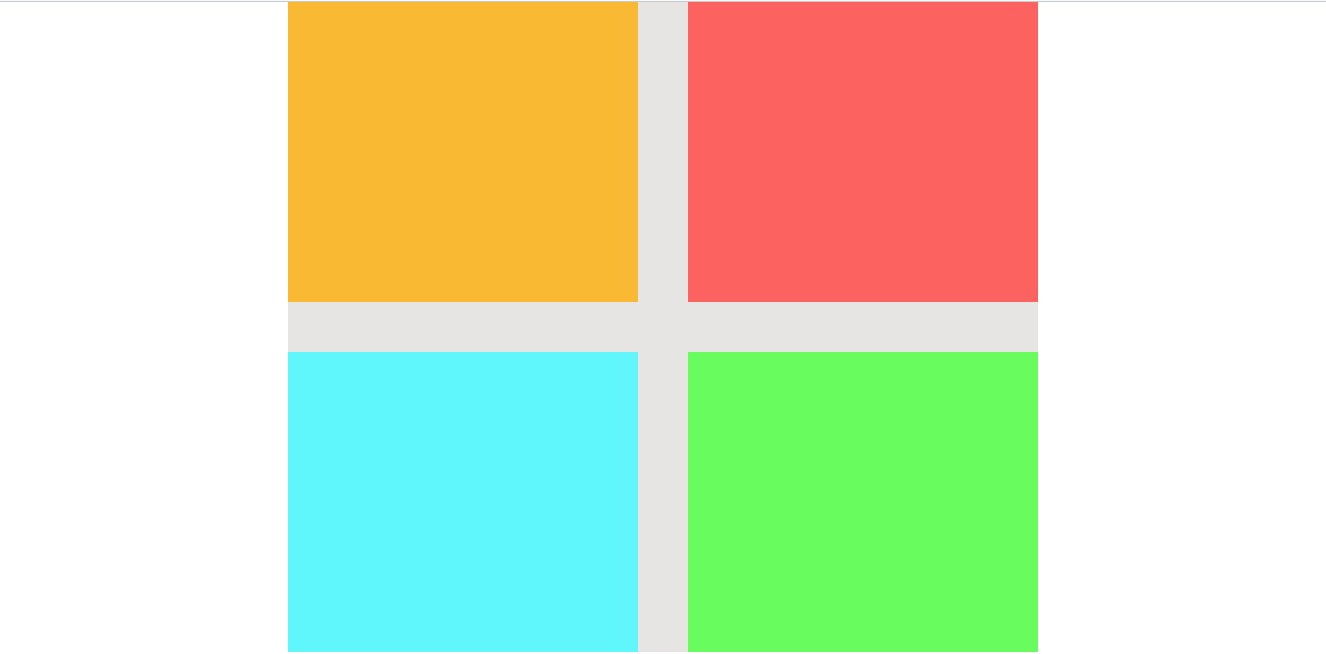
***Блоки. Изображения. Позиционирование. Отступы.***

Блок (атрибут <div>), это сущность которая напоминает контейнер (коробку), которая хранит в себе различные элементы с другими свойствами (размер, цвет и т.д.). Вложенные элементы в этот контейнер наследуют многие из свойств (такие, как шрифт, цвет, текстовое форматирование). В CSS есть свойства, которые применяются в основном к блоку и отвечают за способы расположения блоков.

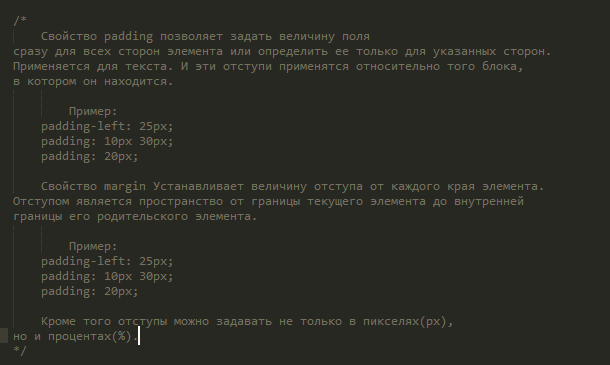
Следующий пример покажет, как создать плитку цветных блоков, которой мы воспользуемся в дальнейшем.



Результат:



**Теория**



**Позиционирование(css свойство- position)**

Position - устанавливает способ позиционирования элемента относительно окна браузера или других

объектов на веб-странице.

Способ обьявления: position: absolute | fixed | relative | static | inherit

***absolute***

Указывает, что элемент абсолютно позиционирован, при этом другие элементы отображаются на веб-странице словно

абсолютно позиционированного элемента и нет. Положение элемента задается свойствами left, top, right и bottom,

также на положение влияет значение свойства position родительского элемента. Так, если у родителя значение

position установлено как static или родителя нет, то отсчет координат ведется от края окна браузера.

Если у родителя значение position задано как fixed, relative или absolute, то отсчет координат ведется от края родительского элемента.

***fixed***

По своему действию это значение близко к absolute, 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. Браузер Firefox вообще не отображает полосы прокрутки, если положение элемента задано фиксированным, и оно не помещается целиком в окно браузера. В браузере Opera хотя и показываются полосы прокрутки, но они никак не влияют на позицию элемента.

***relative***

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.

***static***

Элементы отображаются как обычно. Использование свойств left, top, right и bottom не приводит к каким-либо результатам.

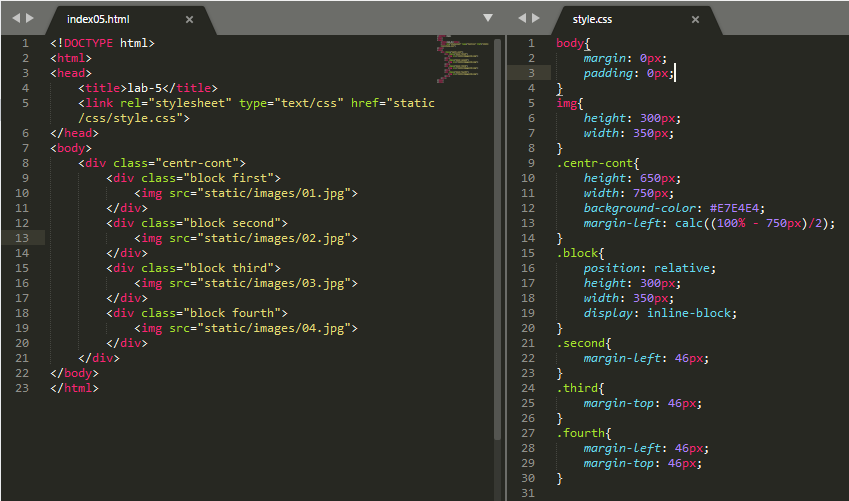
***inherit***

Наследует значение родителя.

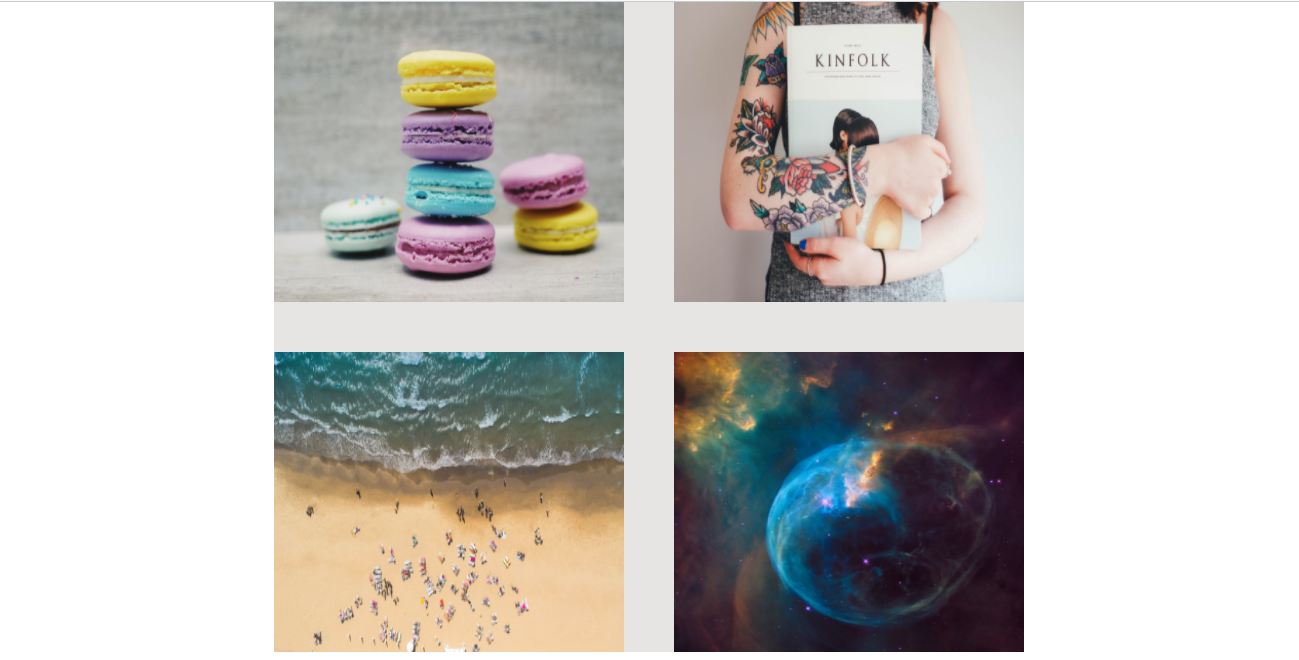
Чтобы сделать еще красивее, добавим несколько изображений. Изображения очень похожи на ссылки, но имеют некоторые различия. Тег - <img>(не закрывающийся), атрибут определяющий путь к файлу – src=”путь”. Путь указывается также как и у ссылки, только является важным разрешение файла (например .jpg)

Скачаем 4 изображения из интернета. Можно воспользоваться гугл, яндекс картинками. Или же сервисом со стоковыми изображениями, к примеру <https://unsplash.com>. Стоит иметь ввиду, что изображения с таких сервисов имееют огромный размер (до 15мб за изображение) и для быстрого отображения таких изображений на странице следует сжать изображение до некоторого минимума.

Скачанные изображения поместим в раннее созданную папку «images», а для большей простоты изображения переименуем в по образцу «01.расширение\_изображения»



Результат:



П.С: Существует способ задания фона блоку через css-свойство «background-img», для ознакомления рекомендуем изучить его самостоятельно и опробовать на одном из блоков.

**Лабораторная работа №7**

***Таблицы.***

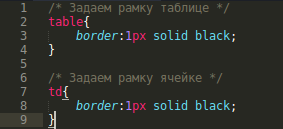
Прежде чем приступить к выполнению лабораторной создайте файл «index0?.html» и файл стилей для него «style0?.css»

**1.** Создание простейшей таблицы

Создание таблицы начинается с тега **<table></table>**, далее, внутри него располагается все содержимое, а именно строки **<tr>** и уже внутри строк ячейки **<td>.** Чтобы понять, как все это работает обратите внимание на пример ниже:

Примечание: заголовочную ячейку можно создать тегом **<th>**, текст в такой ячейке располагается посередине и выделяется жирным шрифтом.

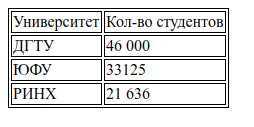
1. Для начала зададим базовые стили



не забудьте подключить файл стилей к вашему HTML файлу.

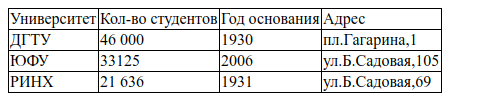
2. Создадим простейшую таблицу и посмотрим на результат в браузере





***Задание 1***

Дополнить таблицу как показано ниже:



Самостоятельно освойте свойство **border-collapse**, в выполнении задания оно обязательно понадобится.

**2.** Работа с рамками, отступами внутри и между ячейками, ячейки-заголовки

Иногда бывают ситуации, когда нужно чтобы отображалась только нижняя рамка ячейки и т.д, такие эффекты достигаются за счет свойств представленных ниже:

**border-right,**

**border-left,**

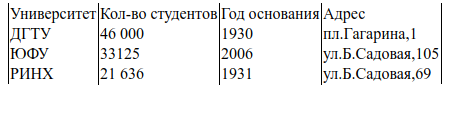
**border-top,**

**border-bottom**

Примечание: для того чтобы убрать рамку в значении свойства напишите **none**

***Задание 2***

В созданной таблице из Задания 1 оставьте рамки только справа и слева как на примере ниже:



Отступы внутри ячеек задаются с помощью свойства **padding:**

padding: 5px - одно значение задает отступ всем сторонам,

padding: 5px 10px - первое значение задает отступ сверху-снизу, второе

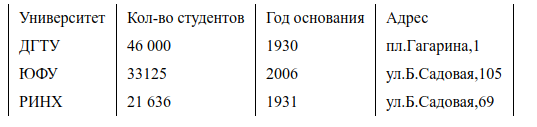
справа-слева.

padding: 5px 10px 5px - первое значение задает отступ сверху, второе одновременно справа-слева, третье снизу.

padding: 5px 10px 5px 10px - первое сверху, второе справа, третье снизу, четвертое слева.

***Задание 3***

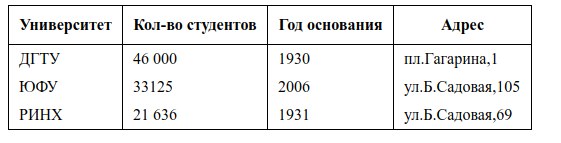
Задайте ячейкам отступы сверху-снизу (5px), справа-слева (10px) как на примере ниже:



***Задание 4***

**1.**Замените тег **<td>** в категориях на **<th>.**

**2.** В стилях задайте тегу <th> рамку и внутренний отступ (10px)

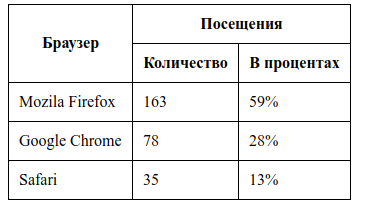
**3.** Объединение ячеек в строках-столбцах

Для объединения двух и более ячеек используют атрибуты colspan и rowspan для тега **<td>**

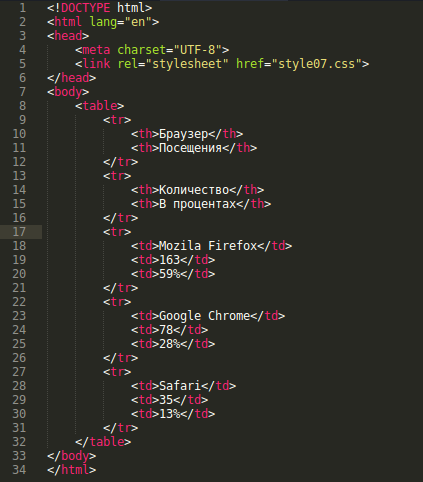
colspan — устанавливает число объединяемых ячеек по горизонтали

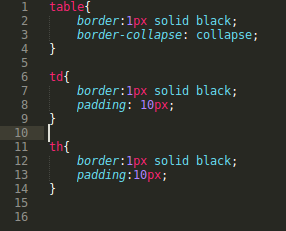
rowspan — аналогично, только по вертикали

Следуя инструкциям, создайте таблицу:



1. Задайте базовую структуру таблицы и стили





2. Начинаем работать с объединением ячеек

Первый тег <th> занимает две ячейки по горизонтали и две ячейки по вертикали, поэтому задаем:

**<th colspan = “2“ rowspan = “2“></th>**

Второй занимает две ячейки по горизонтали:

**<th colspan = “2“></th>**

Тег <th> с названиями браузеров точно так же как и первый занимает две ячейки по горизонтали:

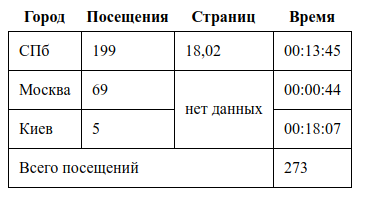
**<th colspan = “2“></th>**

Проделав все действия выше, вы должны получить такой код и результат изображенный выше:



***Задание 5***

Создайте таблицу изображенную ниже (размеры отступов кратны 5):



**Лабораторная работа №8**

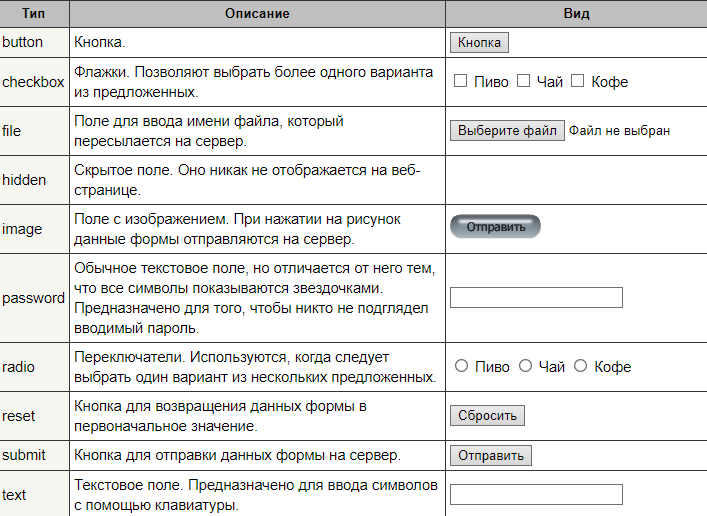
***Формы***

Тег **<form>** устанавливает форму на веб-странице. Форма предназначена для обмена данными между пользователем и сервером. Область применения форм не ограничена отправкой данных на сервер, с помощью клиентских скриптов можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.

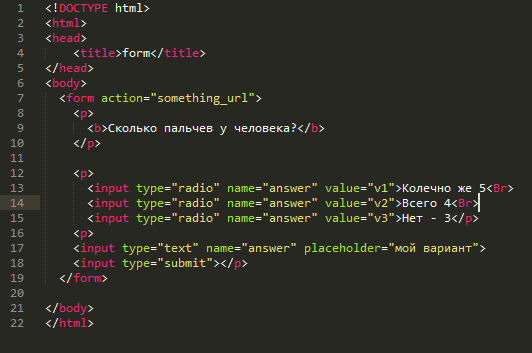
Для отправки формы на сервер используется кнопка Submit, того же можно добиться, если нажать клавишу  Enter  в пределах формы. Если кнопка Submit отсутствует в форме, клавиша  Enter  имитирует ее использование.

Атрибут action в форме принимает ссылку(URL) который ведет либо на локальный скрипт, либо на сервер которые примут на себя обработку данных формы.

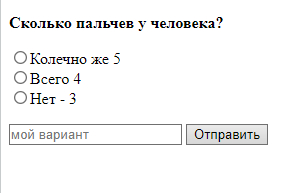
Поля в форме именуются тегом <input> имеют один важный атрибут – type, который определяет тип поля, который будет использован. Type может принимать значения:



Чтобы понять данный материал выполните следующий код (так как у нас нет сервера, то никаких действий не выполнится после подтверждения формы):



В результате должно получится:



*Задание:*

1) Изучите тег, который также относится к форме “SELECT”

2) Напишите форму, которая спрашивает у пользователя запрашивает персональные данные(такие как Имя, Фамилия, телефон, возраст(с помощью <select>), адрес и секретный вопрос(например: «Любимы цвет»)). Форма должна содержаться в блоке шириной 400px автоматической высотой, цветом на усмотрение, с отступом сверху в 75px и располагаться по центру экрана.