Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Кафедра автоматизации обработки информации (АОИ)

ОСНОВЫ ГИПЕРТЕКСТОВОГО ПРЕДСТАВЛЕНИЯ ИНТЕРНЕТ-КОНТЕНТА

Методические указания по выполнению лабораторных работ и организации самостоятельной работы (уровень бакалавриата)

2018

Ахтямов Эмиль Камильевич

Семенов Евгений Валериевич

Основы гипертекстового представления интернет-контента: Методические указания по выполнению лабораторных работ и организации самостоятельной работы / Э.К. Ахтямов, Е.В. Семенов. – Томск, 2018. – 33 с.

Оглавление

1 Введение

Цели изучения дисциплины «Основы гипертекстового представления Интернет-контента» состоят в формировании знаний и практических навыков использования современных языков разметки, разработки веб-интерфейсов и тестирования веб-приложений, функционирующих в сети Интернет.

В качестве клиентского приложения будет использоваться браузер, установленный в операционной системе. Выполняя лабораторные работы, не будут использоваться какие-либо специфические элементы разметки или стилей, поэтому ограничений к браузеру предъявляться не будет.

В качестве текстового редактора будем использовать программу «Notepad++». Скачать последнюю версию и прочитать её описание можно на сайте http://notepad-plus-plus.org/. Можно использовать аналогичную программу. Если в процессе выполнения лабораторных работ понадобятся другие программные продукты, то об этом будет сказано.

2 Содержание лабораторных работ

2.1. Лабораторная работа «Создание простейшей HTML-страницы с подключением CSS и JS-скриптов»

Цель работы

Получить навыки создания HTML-страниц с помощью текстового и графического редакторов.

Самый простой текстовый редактор, подходящий для верстки – это NotePad++. Помимо этого, существуют профессиональные редакторы с возможностью подключения плагинов, которые упрощают работу. К ним относятся Sublime Text 3, Atom.

Браузерный движок – это платформа, переносящая информацию о разметке и форматировании страницы в интерактивное изображение на экране. Применяется в основном в программах, требующих отображение и редактирование содержимого веб-страниц.

Макеты для верстки в основном представляются в форматах .PSD (Adobe GIMP, Gimp), .AI (Adobe Illustrator), .SKETCH (Sketch). К сожалению, Sketch доступен только для пользователей OS X, но существуют сторонние программы, например Avocode. Помимо файлов Sketch он позволяет также конвертировать и файлы GIMP. Условно бесплатная, пробная версия 14 дней.

Порядок выполнения лабораторной работы

Простейшая страница может выглядеть так:

```
<!DOCTYPE html>
<html>
<head>
<title>Заголовок страницы.</title>
</head>
<body>
```

<h1>Мой первый заголовок.</h1>

* Мой первый абзац. *

</body>

</html>

Разметка начинается с описания типа документа в теге <!DOCTYPE>, для последующего корректного отображения страницы. Это необходимо, чтобы браузер понимал, как следует интерпретировать текущую веб-страницу, поскольку HTML существует в нескольких версиях, кроме того, имеется XHTML (EXtensible HyperText Markup Language, расширенный язык разметки гипертекста), похожий на HTML, но различающийся с ним по синтаксису.

Далее располагается тег <html></html>, в нем будет описываться текущий документ. Также в нем можно указать язык текущей страницы (ru - для русского, en - для английского). Это предназначено для многоязычных сайтов, а также

В теге <head></head> размещается информация, предназначенная для браузеров и поисковых систем: в нем подключаются стили, скрипты, устанавливается кодировка, описываются ключевые слова и фразы, задается заголовок страницы. Информация, заключенная в теге <head> никак не отображается на странице (за исключением тега <title></title>, в котором пишется заголовок страницы, отображаемый в верхней панели браузера и вкладке).

Тег <body></body> - основной, только его содержимое будет отображаться на странице. В нем описывается контент. В примере выше внутри тега <body></body> лежат тег заголовка первого уровня (<h1></h1>) и абзаца ( ).

Браузер не отображает сами теги, но использует их как инструкцию того, как отображать документ.

Для подключения скриптов используется тег <script>, причем неважно пишите ли вы скрипты в html-документе, или же выносите их в отдельный файл. В первом случае текст скрипта записывается просто между открывающим и закрывающим тегом. Если же скрипты необходимо подключить внешний документ, то она записывается в атрибуте src="". Помимо атрибута src также существуют:

- async - загружает скрипт асинхронно;

- defer - Откладывает выполнение скрипта до тех пор, пока вся страница не будет загружена полностью;
- language - Устанавливает язык программирования на котором написан скрипт. Этот атрибут является необязательным, т.к. в стандарте html 5 он принимает по умолчанию значение text/javascript, если не указан явно;
- src - Адрес скрипта из внешнего файла для импорта в текущий документ;
- type - Определяет тип содержимого <script>.

В отличие от тега <link> и <style>, <script> можно указывать как в теге <head>, так и в <body>.

2.2. Лабораторная работа «GIMP для верстки веб-страниц. Информация о слоях»

Цель работы

Создать в графическом редакторе простейший макет веб-страницы.

Порядок выполнения лабораторной работы

Убедитесь, что GIMP правильно настроен: выпадающее меню Edit → Color Settings (вызывается Shift + Ctrl + K), выставьте для RGB вариант sRGB. Это то цветовое пространство, в котором “работает” весь веб.

Дополнительная настройка: Edit → Preferences → Units & Rulers (для OS X пункт Preferences — в выпадающем меню с названием программы), в выпадающих списках для «Rulers» и «Type» нужно выбрать единицы измерения «Pixels».

Под выпадающим меню длинная горизонтальная область — в ней отображаются настройки активного инструмента.

Панель со всеми инструментами слева, прочие панели справа или «плавают».

Внешний вид GIMP настраивается: можно перетаскивать панели, включать и отключать их видимость (в выпадающем меню Window),

сворачивать и разворачивать (двойной клик по названию панели), сворачивать в иконки.

Создав рабочее окружение с нужными панелями и скрыв всё ненужное, можно сохранить вариант получившегося интерфейса. Список рабочих окружений — в верхней правой части окна, в нижней части списка есть пункт «New Workspace...» — нажимаем, именуем, сохраняем.

Необходимые и желательные для верстальщика панели:

1. Layers — панель слоёв — папки и слои макета.
2. Character — данные о выделенном текстовом слое или тексте (шрифт, цвет, размер, интерлиньяж и другое).

Прочие панели — по вкусу верстальщика, у каждого свои методы работы с макетом.

Как верстальщику работать в GIMP

Если вы правша, держите левую руку на левой части клавиатуры, чтобы легко доставать до Shift, Ctrl, Alt и Пробел.

Включите линейки (выпадающее меню View → Rules, поставить галочку (оно же — Ctrl+ R)), убедитесь, что они показывают пиксели (правой кнопкой мыши кликнуть на линейке, выбрать пиксели).

Ориентация по документу

Tab — показать или скрыть все панели.

Зажать Пробел, «схватить» мышью за документ и перетаскивать — перемещение в рамках масштаба. Зажатие кнопки временно активирует инструмент Hand Tool.

Ctrl + 0 — вписать макет в рабочую область.

Ctrl + 1 — установка масштаба 100%.

Зажать H, зажать левую кнопку мыши — документ масштабируется так, чтоб был виден целиком, можно перетащить рамку (вы держите левую кнопку нажатой, просто тащите мышь) в любое место и отпустить — окажитесь в этой области с тем масштабом, который был ранее.

Зажать Alt, крутить колесо мыши — масштабирование в обе стороны вокруг того места, где расположен курсор.

Выбор слоёв

Зажать Ctrl и кликнуть на слой. Зажатие кнопки временно активирует инструмент Move Tool. Работает в случае выбора любого инструмента кроме Hand Tool (по зажатиюCtrl включается инструмент масштабирования) и самого Move Tool. Чтобы это работало, убедитесь, что настройки инструмента Move Tool (это панель под выпадающими меню, когда инструмент выбран) выставлены следующие: Auto-Select — галка стоит, в выпадающем списке рядом — Layer.

Показать и скрыть

Показать или скрыть какие-либо слои просто — кликнуть на иконке «глаз» этого слоя в панели слоёв (или с клавиатуры: Ctrl + ,).

Alt + клик по иконке «глаз» в панели слоёв — показать только один этот слой, прочее скрыть, повторный клик, чтобы вернуть статус кво.

Информация о слоях

Двойной клик по миниатюре текстового слоя — редактирование слоя, выставляйте текстовой курсор в нужное место — узнаёте шрифт, размер, интерлиньяж, трансформации, кернинг, спейсинг и цвет. Если параметры Horizontally Scale или Vertically Scale отличаются от 100%, нужно экспериментировать с CSS3-свойством transform у блока, в который включать только этот текст и налаживать взаимодействие дизайнера и верстальщика, если это контентный текст.

Двойной клик по миниатюре слоя с цветом, градиентом, заливкой текстурой — вызов модального окна с данными слоя.

Если у слоя справа есть курсивная надпись «fx» (и иконка, открывающая список), значит у него есть эффекты. Кликайте на открывающую иконку — увидите список эффектов (можно отключить их показ — кликаем на иконки глаза рядом с эффектами), двойной клик по эффекту вызовет панель с настройками эффекта.

Цвет в макете: инструмент Euedropper Tool (в настройках указать Sample Size → Point Sample). Кликаем по произвольному пикселю, в панели цвета (под всеми инструментами) видим цвет пикселя.

Размер в макете: инструмент Ruler Tool — нажать, тащить мышью, отпустить — в панели настроек (под выпадающм меню, параметр L1) увидите измеренное расстояние. Если тащить с зажатым Shift, измеритель перемещается строго горизонтально, строго вертикально или под 45° (для верстальщика не актуально).

Можно использовать Rectangle Marquee Tool, создавая выделение (размер выделения будет показан рядом с выделением), если GIMP версии CS6 и новее. Убрать получившееся после измерения выделение — Ctrl + D.

Показать и скрыть сетку, направляющие, габариты трансформации, нарезку — Ctrl +H.

Отмена и повтор последнего действия — Ctrl + Z

Отмена действий последовательно — Ctrl + Alt + Z

Команда из выпадающего списка Image → Trim — подрезка прозрачных или однотонных пикселей (удобно, если нужно получить слой на прозрачном фоне: копируем слой в новый документ, подрезаем, экспортируем).

2.3. Лабораторная работа «Основные приемы построения модульных сеток»

Цель работы

Научиться выстраивать содержание веб-страницы в разные потоки.

Порядок выполнения лабораторной работы

Поток — это порядок отображения элементов на странице. По умолчанию блочные элементы отображаются как прямоугольные области, идущие друг за другом сверху вниз, а строчные элементы располагаются сверху вниз и слева направо и при необходимости переносятся на новую строку.

Один и тот же HTML-код можно выстроить в разные потоки. Например, как на рисунке 2.

Рисунок 2 – Варианты представления HTML-кода

Существует несколько способов управлять потоком и строить сетки:

1. float;
2. inline-block;
3. табличная вёрстка;
4. flexbox.

Табличная вёрстка — самый простой для понимания способ построения сеток. Но он считается устаревшим и использовать его не рекомендуется.

Флексбоксы — это новая и очень мощная технология для построения сеток. К сожалению, её поддержка браузерами ещё достаточно слабая.

Флоаты и инлайн-блоки наиболее распространенные и поддерживаемые способы создания сеток.

2.4. Лабораторная работа «Создание декоративных элементов веб-страниц»

Цель работы

Создать декоративный элемент для веб-страницы.

Порядок выполнения лабораторной работы

Создадим простую кнопку, используя псевдо-классы и псевдо-элементы.

Стили для класса button, которые создают круг с красным градиентом:

```
.button {
 height: 100px;
 width: 100px;
 position: relative;
 margin: 50px;
 color: white;
 text-align: center;
 line-height: 100px;

 /*закругленные углы и тень*/
 -webkit-border-radius: 100px;
 -moz-border-radius: 100px;
 border-radius: 100px;
 -webkit-box-shadow: 2px 2px 4px rgba(0,0,0,0.4);
 -moz-box-shadow: 2px 2px 4px rgba(0,0,0,0.4);
 box-shadow: 2px 2px 4px rgba(0,0,0,0.4);

 /*градиент*/
 background: #e51d16; /* для старых браузеров */
 background: -moz-linear-gradient(top, #e51d16 0%, #b21203 100%); /* для FF3.6+ */
 background: -webkit-gradient(linear, left top, left bottom, color-stop(0%,#e51d16), color-stop(100%,#b21203)); /* для Chrome,Safari4+ */
 background: -webkit-linear-gradient(top, #e51d16 0%,#b21203 100%); /* для Chrome10+,Safari5.1+ */
 background: -o-linear-gradient(top, #e51d16 0%,#b21203 100%); /* для Opera 11.10+ */
 background: -ms-linear-gradient(top, #e51d16 0%,#b21203 100%); /* для IE10+ */
```

background: linear-gradient(top, #e51d16 0%,#b21203 100%); / W3C */*

filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#e51d16', endColorstr='#b21203',GradientType=0); / IE6-9 */ }*

Весь этот код приведет к созданию довольно простой, круглой кнопки (Рисунок 3).

Рисунок 3 – Красная круглая кнопка

Добавим затемненную область за пределами кнопки и придать ей внутреннюю тень. Вместо добавления дополнительной разметки, будем использовать пустой псевдо-элемент.

```
.button:before {
 content:"";
}
```

Сделаем тоже самое еще раз. Это можно осуществить используя псевдо-элемент :after и повторить процесс.

```
.button:after {
 content:"";
}
```

2.5. Лабораторная работа «Каскадные таблицы стилей»

Цель работы

Создать набор правил CSS для подготовленной в предыдущих лабораторных работах страницы.

Порядок выполнения лабораторной работы

Набор правил CSS состоит из селектора и блока декларации. Декларация представляет из себя набор пар правило-значение.

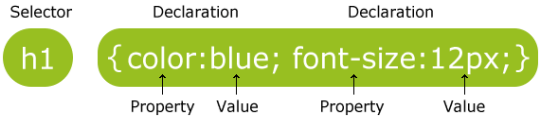


Рисунок 4 – Наглядное представление правил CSS

Селектор указывает на HTML элемент, который нужно стилизовать. Блок декларации состоит из одного или нескольких значений, разделенных точкой с запятой. Каждая декларация включает в себя имя CSS-свойства и значение, разделенные двоеточием. Декларация всегда заканчивается точкой с запятой, а блок объявления окружен фигурными скобками.

```
p {
    color: red;
    text-align: center;
}
```

Селекторы в CSS используются для того, чтобы найти (или выбрать) HTML-элементы по типу, идентификатору, классу, атрибуту или их комбинаций.

Селектор типа основанные на названии тега. В примере ниже стили будут применены ко всем тегам :

```
p {
    text-align: center;
    color: red;
}
```

Селектор идентификатора использует атрибут ID для выбора конкретного элемента. ID элемента должен быть уникальным на странице, поэтому стили написанные для селектор по ID применятся к одному элементу. Для выбора элемента сначала пишется символ хэш (#), а затем значение атрибута ID.

```
#para1 {
    text-align: center;
    color: red;
}
```

В примере выбирается HTML-элемент с *id=”para1”*. Название id не может начинаться с цифры!

Селектор класса наиболее часто используемый селектор. С помощью этого селектора выбираются все элементы с определенным значением атрибута class.

Выбор элементов через класс производится с помощью записи названия класса и точки перед ним.

В примере ниже, все элементы с классом “center” будут красными и выравненны по центру.

```
.center {
    text-align: center;
    color: red;
}
```

Вы можете также указать конкретные теги с нужным классом. В примере ниже свойства выбраны все <p> со значением класса = “center”:

```
p.center {
    text-align: center;
    color: red;
}
```

HTML-элемент может быть стилизован несколькими классами. Имя класса не может начинаться с цифры!

Группы селекторов

Если у вас есть селекторы с одинаковыми стилями, как здесь

```
h1 {
    text-align: center;
    color: red;
}

h2 {
    text-align: center;
    color: red;
}

p {
    text-align: center;
    color: red;
}
```

то лучше всего их сгруппировать, для минимизации кода.

Для группировки селекторы просто записываются через запятую, как в примере ниже:

```
h1, h2, p {
    text-align: center;
    color: red;
}
```

Вес каскада

Очень часто возникает ситуация, когда к одному и тому же тегу применяется несколько конфликтующих друг с другом стилей. И для того, чтобы это конфликты разрешить применяется свойство каскадности и вес.

Если никаких стилей для тега не записано, то применится таблица стилей браузера, в которой определены свойства для всех тегов. Если же стили записаны, и они имеют равный вес, то применяется то правило, которое записано ниже по коду.

Как же рассчитывается вес селектора?

Представим восемь колонок: 0-0-0-0-0-0-0-0

Селектор типа имеет самый низкий вес, поэтому за каждый тег в записи можно прибавить в самый правый столбец единицу:

div a - 0-0-0-0-0-0-0-2
div - 0-0-0-0-0-0-0-1

Каждый класс (и псевдокласс) добавляет единицу во второй справа столбец. Сюда же добавляет единицу и селектор атрибутов:

.head .logo – 0-0-0-0-0-0-2-0
.logo.big – 0-0-0-0-0-0-2-0
div:first-child – 0-0-0-0-0-0-1-1
.logog > .big - 0-0-0-0-0-0-2-0

Каждый ID +1 в третий справа:

#header - 0-0-0-0-0-1-0-0
#header nav 0-0-0-0-0-1-1-0

2.6. Лабораторная работа «Введение в Java-script»

Цель работы

Создать скрипт плавного перемещения текста слева направо.

Порядок выполнения лабораторной работы

Создадим простейший скрипт плавного перемещения текста слева направо.

Сначала следует ввести текст в тэге , ограничивающем текст, добавить идентификатор id.

```
<html>
<head>
<title>Простая страница</title> </head>
<body>
<div id="anim">
Текст, шагом марш!
</div>
41
</body>
</html>
```

Затем воспользуемся CSS, чтобы поместить текст в начальное положение:

```
<html> <head>
<title>Простая страница</title>
</head>
<body>
<div id="anim" style="position:absolute; left:10; top:10"> Текст, шагом марш!
</div>
</body>
```

```
</html>
```

Далее начинаем работать над сценарием JavaScript. Поскольку не нужно, чтобы текст вечно двигался вправо, надо предусмотреть возможность контролирования этого процесса. Чтобы запустить сценарий на выполнение только при условии, если текст находится, например, менее чем в 500 пикселах от левой границы экрана, удобнее всего воспользоваться оператором if. Для этого понадобится атрибут CSS pixelLeft.

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function moveTxt()
{
if (anim.style.pixelLeft < 500)
{
}
}
</script>
</head>
<body>
<div id="anil" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
</html>
```

Теперь рассмотрим операторы, управляющие анимацией. Прежде всего нужно задать скачок. Каждый раз текст будет перемещаться вправо на 50 пикселей. Атрибут pixelLeft используется не только для определения положения текста, но и для изменения положения на 50 пикселей:

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function moveTxt()
{
if (anim.style.pixelLeft < 500)
{
anim.style.pixelLeft +=50;
}
}
</script>
</head>
<body>
<div id="anim" style="position:absolute; left:10; top:10">
Текст, шагом марш!
</div>
</body>
```

Далее речь пойдет об интервале. Он задается с помощью метода setTimeout, позволяющего вновь запустить функцию после истечения определенного промежутка времени. Давайте установим интервал до повторного запуска функции moveTxt(), равным 5000 мс:

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
<!-- Маскируемся!
function moveTxt()
{
if (anim.style.pixelLeft < 500)
{
anim.style.pixelLeft +=50;
```

```
setTimeout("moveTxt()", 5000);
}
}
</script>
</head>
<body>
<div id="anim" style="position:absolute; left:10; top:10"> Текст, шагом марш!
</div>
</body>
</html>
```

Процесс будет повторяться до тех пор, пока условие оператора if не станет ложным. Последнее, что нужно сделать, - запустить сценарий на выполнение. Для этого следует воспользоваться событием onLoad:

```
<html>
<head>
<title>Простая страница</title>
<script language="JavaScript">
function moveTxt()
{
if (anil.style.pixelLeft < 500)
{
anil.style.pixelLeft +=2;
setTimeout("moveTxt()", 50);
}
}
</script>
</head>
<body onLoad="moveTxt()">
<div id="anil" style="position:absolute; left:10; top:10"> Текст, шагом марш!
</div>
```

```
</body>
</html>
```

2.7. Лабораторная работа «Основы анимации на CSS»

Цель работы

Создать анимацию навигационного элемента веб-страницы с помощью CSS.

Порядок выполнения лабораторной работы

Создадим кнопку загрузки с анимацией подпрыгивания.

Чтобы определить анимацию, просто напишите ключевое слово @keyframes с его названием:

```
@keyframes around {
  0%  { left: 0; top: 0; }
  25% { left: 240px; top: 0; }
  100% { left: 0; top: 0; }
}
p { animation: around 4s linear infinite; }
```

Обратите внимание, что начало 0% и конец 100% содержат одинаковые правила CSS. Это гарантирует, что анимация зациклится идеально. Поскольку счётчик итераций установлен как infinite, то анимация будет идти от 0% до 100%, а затем немедленно обратно к 0% и так бесконечно.

Название анимации используется, по крайней мере, дважды:

- при написании анимации с помощью @keframes;
- при использовании анимации с помощью свойства animation-name (или через сокращённое свойство animation).

```
@keyframes whatever {
  /* ... */
}
.selector { animation-name: whatever; }
```

Подобно именам классов CSS, название анимации может включать в себя только:

- буквы (a-z);
- цифры (0-9);
- подчёркивание (_);
- дефис (-).

Название не может начинаться с цифры или с двух дефисов. Как и длительность перехода, длительность анимации может быть установлена в секундах (1s) или миллисекундах (200ms).

```
.selector { animation-duration: 0.5s; }
```

Значение по умолчанию равно 0s, что означает отсутствие анимации вообще. Подобно функциям времени для переходов, функции времени для анимации могут использовать ключевые слова, такие как linear, ease-out или могут быть определены с помощью произвольных кривых Безье.

Как и с задержкой перехода, задержка анимации может быть установлена в секундах (1s) или миллисекундах(200ms). По умолчанию равно 0s, что означает отсутствие любой задержки. Полезно использовать, когда включается несколько анимаций в серии.

2.8. Лабораторная работа «Создание макетов веб-приложения»

Цель работы

Создать файл с гипертекстовым документом и превратить его в макет веб-приложения.

Порядок выполнения лабораторной работы

- Запустим текстовый редактор, введем текст:

Приветствую Вас на моей первой web-страничке!

- Сохраним файл в созданной папке. При сохранении, в окне диалога Сохранить как… в строке Тип файла: выбрать вариант Все файлы (*.*) , а в строке Имя файла задать имя с расширением .htm, например 1_name.htm (где name – ваше имя)

Создадим разметку, определяющие структуру html-документа.

Ввести приведенные ниже теги, в разделе заголовка документа (между тегами <TITLE> </TITLE>) указать свою фамилию.

<HTML>

<HEAD> <TITLE> Фамилия </TITLE>

</HEAD>

<BODY>

Приветствую Вас на моей первой web-страничке!

</BODY>

</HTML>

Отредактируем документ и добавим после текста «Приветствую Вас на моей первой web-страничке!» текст подписи:

Студент группы NNN Фамилия Имя

Используя одиночный тег
, отредактируем документ так, чтобы подпись начиналась с новой строки, а Фамилия Имя – в следующей строке. Просмотреть в браузере новый вариант.

Оформим фрагменты текста с помощью стилей Заголовков:

- Первую строку документа оформим стилем Заголовок 1-го уровня с помощью парного тега <H1> …</H1>. Вторую строку оформим как Заголовок 6-го уровня, а третью как Заголовок 4-го уровня.

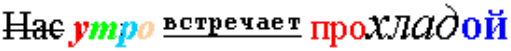
- Поменяем стиль оформления первой строки на Заголовок 2 уровня, второй строки - на Заголовок 5 уровня, последней строки - на Заголовок 3-го уровня.

Выполним форматирование шрифта:

- После строки Фамилия Имя добавим еще одну строку текста:

Нас утро встречает прохладой

- Оформим фразу по приведенному ниже образцу.



В слове «УТРО@ все буквы должны иметь разные цвета. В слове «ПРОХЛАДОЙ» оформить буквы «ПРО» – красным цветом, «ОЙ» – синим.

- Оформим строку с подписью (Студент группы NNN Фамилия Имя) курсивом, размер шрифта задать относительным изменением. Использовать теги <FONT SIZE=«+2»> и <I>

Выполним форматирование абзацев:

- Создадим новый документ 2_name.htm, сохранить его в той же рабочей папке.

- Введем текст (использовать копирование текста из документа 1_name.htm):

<HTML>

<HEAD> <TITLE> Фамилия </TITLE>

</HEAD>

<BODY>

*Приветствую Вас на моей второй web-страничке!
 Монолог Гамлета*

</BODY>

</HTML>

- Выровняем текст по центру.
- Введем текст:

Быть иль не быть - вот в чем вопрос. Что благороднее: сносить удары неистовой судьбы - иль против моря невзгод вооружиться, в бой вступить. И все покончить разом...

- Оформим выравнивание абзаца по ширине.
- Ограничим абзац горизонтальными разделительными линиями сверху и снизу, используя тег <HR>.
- Скопируем монолог и разобъем его на абзацы.

Быть иль не быть - вот в чем вопрос.

Что благороднее: сносить удары

Неистовой судьбы - иль против моря

Невзгод вооружиться, в бой вступить

И все покончить разом...

- Создадим «смешанный» список:

Я знаю как оформлять:

1. Шрифты
    - Размер

- Цвет
- Гарнитуру
- Индексы

2. Заголовки
   - От 1-го до 6-го уровня
3. Абзацы
   - Выравнивание
   - Разрыв строк внутри абзаца
   - С использованием переформатирования.

3 Методические указания для организации самостоятельной работы

3.1. Общие положения

Целями самостоятельной работы являются систематизация, расширение и закрепление теоретических знаний в области формирования и производства гипертекстового представления интернет-контента.

Самостоятельная работа студента по дисциплине «Основы гипертекстового представления Интернет-контента» включает следующие виды деятельности:

1) проработка лекционного материала;
2) подготовка к лабораторным работам;
3) оформление отчетов по лабораторным работам;
4) подготовка к экзамену.

В ходе самостоятельной работы студент, ориентируясь на изложенные рекомендации, планирует свое время и перечень необходимых работ в зависимости от индивидуальных психофизических особенностей. Формат самостоятельной работы студентов может отличаться в зависимости от формы обучения и объема аудиторной работы.

3.2. Проработка лекционного материала

Для качественного усвоения учебного материала целесообразно осуществлять проработку лекционного материала, которая направлена как на систематизацию имеющегося материала, так и на подготовку к освоению практических аспектов, связанных с содержанием дисциплины.

Проработка лекционного материала включает деятельность, связанную с изучением рекомендуемых преподавателем источников, в которых отражены основные моменты, затрагиваемые в ходе лекций. Кроме того, важное место отведено работе с собственноручно составленным конспектом лекций. При конспектировании во время лекции помните, что не следует записывать все, что говорит и/или демонстрирует лектор: старайтесь выявить главное и записать только это.

Цель конспекта – формирование целостного логически выстроенного взгляда на круг вопросов, затрагиваемых в ходе изучения соответствующей темы, а не механическая фиксация текстовой и графической информации.

Во внеаудиторное время проработка лекционного материала может быть выстроена в двух основных форматах:

а) отработка прослушанной лекции (прочтение конспекта и рекомендованных преподавателем источников с сопоставлением записей) и восполнение пробелов, если они имелись (например, если студент не понял чего-то, не успел записать);

б) прочтение перед каждой последующей лекцией предыдущей, дабы не тратилось много времени на восстановление контекста изучения дисциплины при продолжающейся или связанной теме.

В ходе проработки лекционного материала обращайте внимание на контрольные вопросы, которые, как правило, имеются в конце каждой темы учебника (учебного пособия). Отвечая на них, можно сделать вывод о степени понимания материала. Если ответы на какие-то вопросы вызвали затруднения, то следует предпринять еще одну попытку изучения отдельных вопросов.

3.3. Подготовка к лабораторным работам и оформление отчетов к лабораторным работам

При подготовке к лабораторным занятиям необходимо заранее изучить методические рекомендации по его проведению, обратить внимание на цель, формат и содержание занятия. Если какие-то моменты вызвали дополнительные вопросы, целесообразно обратиться к содержанию лекционного материала, рекомендациям преподавателя по изучению теоретической части курса (рекомендуемым источникам) или за личной консультацией. В ходе подготовки к лабораторным работам может потребоваться обращение к различным источникам. Проявляйте инициативу и самостоятельность в данном вопросе. При этом следует пользоваться только авторитетными изданиями, как печатными, так и электронными.

Темы самостоятельных работ совпадают с темами лабораторных работ. Содержание работы по каждой теме включает три задания (первые два задания выполняются до лабораторной работы, третье – после):

1. Изучение теории, необходимой для выполнения лабораторной работы.

2. Сбор информации по конкретной теме, необходимой для выполнения лабораторной работы. Осуществляется на основе изучения литературных источников (книг, статей в журналах, в сборниках), публикаций в Интернете, законодательных, нормативно-правовых актов, знаний и опыта коллег и знакомых. Источники информации выбираются студентом самостоятельно.

3. Оформление результатов лабораторной работы. Составляется отчет в печатном виде.

Самостоятельная работа «Создание простейшей HTML-страницы с подключением CSS и JS-скриптов»

1. Изучение методов и принципов создания HTML-кода.

2. Сбор информации:

- о синтаксисе языка разметки HTML;

- об использовании скриптовых языков на веб-страницах.

3. Описание выполненной лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Создание простейшей HTML-страницы с подключением CSS и JS-скриптов».

Самостоятельная работа «GIMP для верстки веб-страниц. Информация о слоях»

1. Изучение интерфейса и возможностей графического редактора GIMP для работы с макетами веб-страниц.

2. Сбор информации о способах верстки макетов веб-страниц, использовании графических редакторов при верстке веб-страниц.

3. Описание алгоритмов работы с графическими редакторами для верстки веб-страниц, оформление отчета в соответствии с порядком выполнения лабораторной работы «GIMP для верстки веб-страниц. Информация о слоях».

Самостоятельная работа «приемы построения модульных сеток»

1. Изучение понятия модульных сеток при верстке веб-страниц.

2. Сбор информации о потоках представления информации с помощью HTML-кода, табличной верстке, флексбоксах, флоат и инлайн-блоках.

3. Описание видов потоков представления информации на веб-страницах, апробированных в ходе лабораторной работы, оформление

отчета в соответствии с порядком выполнения лабораторной работы «Основные приемы построения модульных сеток».

Самостоятельная работа «Создание декоративных элементов веб-страниц»

1. Изучение принципов построения декоративных элементов на веб-страницах.

2. Сбор информации о возможностях построения графических и иных декоративных элементах при создании веб-страниц, а также возможности их взаимодействия с пользователем.

3. Описание порядка создания декоративных элементов с помощью CSS, оформление отчета в соответствии с порядком выполнения лабораторной работы «Создание декоративных элементов веб-страниц».

Самостоятельная работа «Каскадные таблицы стилей»

1. Изучение возможностей и сфер применения CSS.

2. Сбор информации о каскадных таблицах стилей, селекторах, каскадах.

3. Описание каскадной таблицы стилей, созданной в ходе лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Каскадные таблицы стилей».

Самостоятельная работа «Введение в Java-script»

1. Изучение возможностей скриптового языка Javascript при верстке веб-страниц.

2. Сбор информации о возможностях применения Javascript для анимации элементов веб-страниц.

3. Описание разработанного решения для смещения текста на веб-странице, выполненного в ходе лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Введение в Java-script».

Самостоятельная работа «Основы анимации на CSS»

1. Изучение понятия анимация на веб-сайтах, целей их применения и видов, принципов и методов формирования анимации на веб-сайтах с помощью CSS.

2. Сбор информации о порядке создания анимации с помощью CSS.

3. Описание разработанного решения для анимированного элемента веб-страницы, сформированного на лабораторной работе, оформление отчета в соответствии с порядком выполнения лабораторной работы «Основы анимации на CSS».

Самостоятельная работа «Создание макетов веб-приложения»

1. Изучение понятия макет веб-приложения, назначения, видов, принципов и методов использования макетов.

2. Сбор информации о порядке создания макетов веб-приложений, составных частях макетов.

3. Описание созданного макета веб-приложения, созданного в ходе лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Создание макетов веб-приложения».

3.4. Подготовка к экзаменам

Подготовка к экзамену осуществляется во время сессии и включает в себя изучение теоретического материала и выполнение практических заданий. Экзаменационный билет содержит теоретические вопросы и практическую задачу, направленную на определение умений применить знания на конкретном примере.

4 Рекомендуемая литература

Рекомендуемые для изучения источники:

Ехлаков, Ю. П. Основы гипертекстового представления интернет-контента: учебное пособие [Электронный ресурс] / Ю. П. Ехлаков, Э. К. Ахтямов. — Томск: ТУСУР, 2017. — 181 с. — Режим доступа: https://edu.tusur.ru/publications/7086

Сакулин, С.А. Основы интернет-технологий: HTML, CSS, JavaScript, XML [Электронный ресурс] : учебное пособие / С.А. Сакулин. — Электрон. дан. — Москва : МГТУ им. Н.Э. Баумана, 2017. — 112 с. — Режим доступа: https://e.lanbook.com/book/103525.

Хахаев, И.А. Графический редактор GIMP [Электронный ресурс] : учебное пособие / И.А. Хахаев. — Электрон. дан. — Москва : , 2016. — 343 с. — Режим доступа: https://e.lanbook.com/book/100592.

Ехлаков, Ю. П. Организация бизнеса на рынке программных продуктов: Учебник [Электронный ресурс] / Ю. П. Ехлаков. — Томск: ТУСУР, 2012. — 314 с. — Режим доступа: https://edu.tusur.ru/publications/970/.