Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Томский государственный университет систем управления и радиоэлектроники». (ТУСУР)

УТВЕРЖДАЮ

Заведующий кафедрой
«Управление инновациями»

_____ /*А.Ф.Уваров*
(подпись) *(ФИО)*
"_____" _____ 2011 г.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ К ПРАКТИЧЕСКИМ ЗАНЯТИЯМ

по дисциплине

Программирование

Составлена кафедрой «Управление инновациями»

Для студентов, обучающихся
по направлению подготовки 220600.62 «Инноватика»
по специальности 220601.65 «Управление инновациями»

Форма обучения очная

Составитель
к.т.н., Титков Антон Вячеславович
"_20_" _сентября_ 2011 г

Томск 2011 г.

Введение

Изучение дисциплины «Программирование» имеет важное значение в специальной подготовке студентов по направлению «Инноватика» и специальности «Управление инновациями» и предоставляет им инструментарий презентации в интернет своих разработок, своей работы, деятельности инновационной фирмы путем размещения информации на веб-сайтах.

Цель данного пособия состоит в выработке навыков в программировании веб-сайтов и их разработке. Для полноценного понимания и усвоения материала необходимо предварительно изучить дисциплину "Информатика".

Для углубленного изучения и освоения материала целесообразно применение различных форм обучения студентов: тесты, задачи, упражнения, которые используются при проведении практических занятий в университете, выполнении контрольных и аудиторных работ, а также при самостоятельном изучении данных дисциплин.

Одним из наиболее интенсивных способов изучения дисциплины является самостоятельное решение практических задач. При этом вырабатываются навыки оценки качества построения веб-сайтов.

Предлагаемые задания позволяют глубже освоить теоретические и практические вопросы, понять принципы программирования веб-сайтов и научиться создавать свои интернет приложения.

Практическое занятие №1. Создание html-документа.

Цель занятия: освоить язык гипертекстовой разметки HTML, научиться создавать html-документы.

Задание:

Необходимо создать валидный html-документ. Документ может содержать произвольный контент и должен содержать следующий набор элементов:

- изображение (тег img);
- список (теги ul, ol, dl);
- заголовок (теги h1, h2, …, h6);
- ссылка и якорь (тег a);
- таблица (тег table);
- параграф (тег p);
- блок (тег div);
- встроенный элемент (тег span);
- форматированный текст (теги pre, code);
- жирный текст (тег b);
- курсивный текст (тег i);
- подчеркивание (тег u);
- перевод строки (тег br).

Ход работы.

Задание необходимо начать с изучения основ языка разметки гипертекста и понятия валидации документов. Самоучитель HTML - http://htmlbook.ru/samhtml. Валидация документов - http://htmlbook.ru/samhtml/validatsiya-dokumentov. В качестве справочника по веб-технологиям предлагается использовать сайт http://htmlbook.ru/, где представлено множество полезной информации по HTML и CSS.
Для проверки валидности документов необходимо использовать дополнение HTML Validator для Firefox. Инструкция по установке и использованию - http://htmlbook.ru/samhtml/validatsiya-dokumentov/proverka-dannykh-na-validnost. Кроме этого для работы с html-документом потребуется дополнение Firebug. Видео-инструкция по установке и использованию - http://zuzle.name/lessons/firebug/.
В папке X:\WebServers\home\localhost\www\, где X:\WebServers\ - диск и папка, в которую установлен Denwer, создайте папку lab1. В ней создайте файл index.html со следующим содержимым:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Моя первая веб-страница</title>
</head>
<body>
<h1>Заголовок страницы</h1>
<p>Основной текст.</p>
</body>
</html>
```

Для просмотра в браузере созданного документа введите в адресной строке X:\WebServers\home\localhost\www\lab1\index.html. В таком случае браузер загружает файл index.html непосредственно с диска компьютера. В нашем случае необходимо, чтобы браузеру данный файл передавал веб-сервер. Для этого введите в адресной строке браузера адрес http://localhost/lab1/index.html. В этом случае браузер обращается по адресу http://localhost, где localhost означает текущий компьютер, вместо localhost можно ввести IP-адрес 127.0.0.1. Веб-сервер по данному адресу (т.е. Ваш Denwer) вернет Вашему браузеру содержимое файла index.html, расположенного в директории (папке) lab1. Если все сделано правильно, то в браузере Вы увидите результат, как показано на рисунке 1.

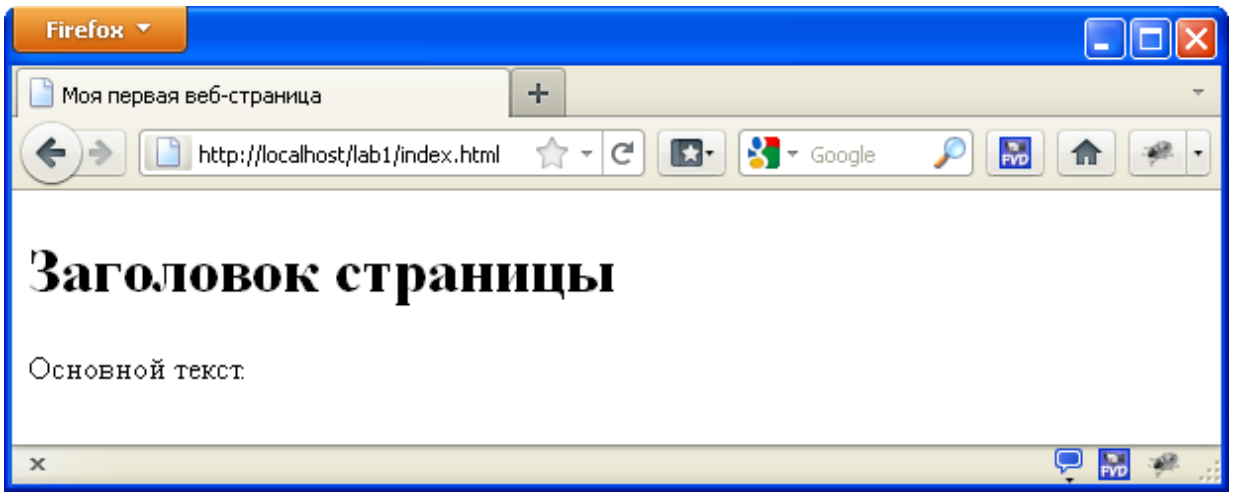

Рисунок 1. Вид веб-страницы в окне браузера.

Если браузер выдаст сообщение, что файл не найден, значит, произошла ошибка, возможно, не запущен веб-сервер.
Если в адресной строке вводить адрес без указания конкретного файла, например, http://localhost/lab1/, то веб-сервер вернет содержимое файла index.html или результат выполнения файла index.php (если конечно в директории такие файлы существуют).
Заполните созданный документ произвольным содержимым. Используйте все теги, указанные в задании. Описание структуры html-документа - http://htmlbook.ru/samhtml/struktura-html-koda. Справочник тегов HTML - http://htmlbook.ru/html. Заголовок документа head должен содержать теги meta с различными атрибутами (http://htmlbook.ru/html/meta).
При помощи Firebug изучите DOM (Document Object Model) созданного Вами документа (окно Firebug открывается клавишей F12). Что такое DOM - http://javascript.ru/tutorial/dom/intro.

Практическое занятие №2. PHP. Создание страницы авторизации. POST и GET запросы.

Цель занятия: изучить основы php, создать страницу авторизации на основе GET и POST запросов.

Задание:

При помощи php создать страницу авторизации.

Ход работы.

Для разработки страницы авторизации для гостевой книги, создайте папку в каталоге «C:\WebServers\home\localhost\www» с именем вашего сайта. Далее поместите в вашу папку файл с именем index.php – файлы index.php и index.html запускаются автоматически при переходе на ваш сайт.
index.php будет содержать в себе текст с приветствием, где имя пользователя будет передаваться GET запросом. GET запрос – это самый распространенный вид HTTP запроса. С его помощью происходит запрос любого файла на сервере и передача параметров. Параметры GET запроса передаются в адресной строке «`resource?param1=value1&param2=value2`», здесь

resource – адрес сайта, затем идет знак вопроса, далее параметр и его значение. Несколько параметров разделяются знаком “&”.
Код index.php содержит в себе html с php вставками. Php код начинается с «<?php» и заканчивается «?>». Внутри будет находиться проверка GET запроса. Если параметр username существует и равен какому-либо значению, то выведем на экран «username», иначе «Гость». Работа с параметрами запросов осуществляется через массивы $_GET и $_POST.

```
if ($_GET['username']) {
     echo $_GET['username'];
} else {
     echo 'Гость';
}
```

Затем разместите в index.php ссылку на файл login.php, который будет содержать html форму для авторизации и проверки логина и пароля, вшитого в php. Форма создается при помощи тега <form>. В него поместите три тега input с типами text, password и submit. Для тега form задайте атрибут method с параметром POST. То есть отправка формы произойдет при помощи метода POST. Применяется для передачи пользовательских данных заданному ресурсу. Например, в блогах посетители обычно могут вводить свои комментарии к записям в HTML-форму, после чего они передаются серверу методом `POST` и он помещает их на страницу. При этом передаваемые данные (в примере с блогами — текст комментария) включаются в тело запроса. Аналогично с помощью метода `POST` обычно загружаются файлы на сервер. Атрибут action оставим пустым, это означает, что после отправки формы, будет происходить перенаправление на эту же страницу.
Как правило, при разработке приложения требуется отладка, т.е. процесс нахождения, локализации и устранения ошибок. Для этого в php существует функция var_dump(), которая возвращается строку - структуру объекта, который был передан ей. Но все данные в этой строке не отформатированы, поэтому для правильного отображения необходимо разместить это строку внутри тегов pre. Поэтому нужно создать новый файл debug.php и создать там функцию для отладки.

```
function debug($obj){
     echo '<pre>';
     var_dump($obj);
     echo '</pre>';
}
```

В файле login.php для подключения файла debug.php используем функцию require_once(path_to_file), где path_to_file – путь к файлу, который нужно подключить. Теперь написав debug($_POST), на экране появятся все параметры POST запроса.
Для проверки логина и пароля зададим две переменные:
$username = 'user';
$password = '1234';
И проверка, если человек вошел, то будет отображаться надпись «Привет, username», иначе форма для входа:

```
<?php
if ($_POST['username'] == $username && $_POST['password'] == $pass-
word) {
      echo 'Привет '.$_POST['username'];
} else {
?>
      <h3>Войти</h3>
      <form method="POST" action="">
            <p>Имя пользователя: <input type="text" name="username"></p>
            <p>Пароль: <input type="password" name="password"></p>
            <p><input type="submit"></p>
      </form>
<?php
      }
?>
```

Практическое занятие №3. Фреймворк jQuery для JavaScript.

Цель занятия: изучить работу с jQuery и понять
Задание:
При помощи jQuery сформировать часть html-документа.
Ход работы.
jQuery – это библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими.
Скачайте библиотеку jQuery с http://jquery.com/ и подключите ее в новом html файле. Затем создайте переменные для счетчика (counter) и массив, хранящий список группы (arr). Добавьте в body следующий код:

```
<p onclick="add_to_list()"> Добавить в список</p>
<ul id="list">
</ul>
```

Определите функцию add_to_list() и сделайте проверку массива на существование в массиве записи с номером, хранящемся в переменной counter.
Вся работа с jQuery ведется с помощью функции $. Получить элемент в jQuery можно по его любому параметру. Для этого используются селекторы.

```
$("div") – получим все элементы div.
$("#my_id") – получим все элементы с атрибутом id равным my_id.
$(".my_class") – получим все элементы с атрибутом class равным
my_class.
```

Это основные типы селекторов. Так же можно указывать связи между ними. Подробно об этом можно прочитать на сайте http://jquery.com/.
Следующая строчка получает список и добавляет в конец новую запись из массива.

```
$("#list").append("<li>"+arr[counter]+"</li>");
```

Инкрементируйте счетчик.
Все готово, теперь можно увидеть, на сколько jQuery облегчает работу web-разработчикам. Так же в jQuery входят функции для создания различных ви-

зуальных эффектов, такие как анимация, появление и затухание, скольжение и прочие. О них так же можно почитать на официальном сайте jQuery.
Добавьте в body две строчки:

```
<p id="fade" onclick="fade()">Скрыть список</p>
<p id="show" onclick="show()" style="display:none">Показать спи-
сок</p>
```

Причем вторая строка будет по умолчанию скрыта. Затем создайте функцию fade() с кодом:

```
$("#list").fadeOut('slow', function(){
        $("#fade").css('display', 'none');
        $("#show").css('display', '');    });
```

Здесь мы применяем к списку “list” функцию fadeOut() – это стандартная функция из библиотеки jQuery для затухания объекта. Она принимает два значения:

1. Скорость затухания (fast, slow).
2. Функция, которая будет вызвана по завершению.

Дальше скрываем элемент с id равным fade и показываем элемент с id равным show.

Создайте функцию show по аналогии с fade, только сделайте появление быстрым и при ее вызове будет скрыта строка «Показать список» и, наоборот, показана строка «Скрыть список».

Рекомендуемая литература

Основная литература:

1. О.Н. Рева. HTML. Просто как дважды два : / О. Н. Рева. - М. : ЭКСМО, 2007. – 240 с. (3 экз. в библиотеке ТУСУР)
2. А. Кириленко. Самоучитель HTML / А. Кириленко. - СПб.: Питер, 2006 ; Киев : BHV, 2006. - 271 с. (10 экз. в библиотеке ТУСУР)
3. Д. Н. Колисниченко. Самоучитель PHP 5 : самоучитель / Д. Н. Колисниченко ; ред. М. В. Финков. - 3-е изд. - СПб. : Наука и техника, 2006. - 567 с. (3 экз. в библиотеке ТУСУР)

Дополнительная литература:

1. Д. Скляр, А. Трахтенберг PHP. «Рецепты программирования PHP». Издательства: Русская Редакция, БХВ-Петербург, 2007 г., 736 стр.
2. М.Браун,Д.Ханикатт «HTML в подлиннике», издательство: BHV, 2002, 1024 стр., перевод с английского
3. Муссиано, Кеннеди «HTML и XHTML. Подробное руководство», издательство: Символ-Плюс, 2008г., 752 стр, перевод с английского.
4. http://php.net – PHP: hypertext preprocessor
5. Флэнаган Д. «JavaScript. Подробное руководство» Издательство: Символ-Плюс, 2008г., 992 стр, перевод с английского.