

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ
ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ**

Б1.В.18 Разработка WEB-приложений

Направление подготовки *09.03.01 Информатика и вычислительная техника*

Профиль образовательной программы *Автоматизированные системы обработки информации и управления*

Форма обучения *очная*

СОДЕРЖАНИЕ

1 Конспект лекций

1.1 Лекция № 1 Архитектура WWW. Обзор Web- технологий

1.2 Лекция № 2 Интерактивные формы HTML. Фреймы.

2. Методические материалы по проведению практических занятий

2.1 Лабораторная работа № 1, 2 Введение в HTML. Структура HTML документа.

2.2 Лабораторная работа № 3, 4 Слои. CSS верстка. Синтаксис. Свойства. Селекторы. Позиционирование элементов.

2.3 Лабораторная работа № 5 Введение в JavaScript. Синтаксис языка. Объектная модель JavaScript.

2.4 Лабораторная работа № 6 Обработка событий. Примеры эффективного программирования на JavaScript.

1. КОНСПЕКТ ЛЕКЦИЙ

1. 1 Лекция № 1 (2 часа).

Тема: Архитектура WWW. Обзор Web- технологий

1. Основные понятия и определения
2. . Сдерживающие факторы развития web-технологий.
3. Клиентские Web-технологии.

1.1.2 Краткое содержание вопросов:

1. Web-программирование - это отдельное направление в программировании, используемое для создания web-приложений. В связи с ускоренными темпами развития Интернет технологий постоянно появляются новые направления в этой области. Поэтому Web-программирование не сформировавшаяся наука, а скорее набор существующих программных технологий (клиентских и серверных), используемых для организации работы пользователя в сети Интернет. Сервер – компьютер, на котором находятся документы и приложения. Клиент – рабочая станция, используемая для просмотра документов и приложений, хранящихся на сервере.

2. Сдерживающие факторы развития web-технологий^
- ограничения, связанные с сетью Интернет (скорость);
 - отсутствие централизованного управления Интернет (конкуренция браузеров);
 - ограничения, связанные с технологиями программирования.

3 Интерпретатор команд обрабатывает код приложения и выдает браузеру сформированный документ. Преимуществом такой архитектуры являются возможность работы с динамическими документами (появляется интерактивность), а также возможность однажды загрузив приложение работать в нем, не обращаясь к серверу.

1. 2Лекция № 2 (2 часа).

Тема: Интерактивные формы HTML. Фреймы.

1. Фреймы.
2. Создание HTML форм
- 3Элементы управления

1.2.2 Краткое содержание вопросов:

1. Фреймы делят окно браузера на несколько областей, каждая из которых функционирует независимо от других и может отображать отдельный HTML-документ
2. HTML форма представляет собой раздел HTML документа, содержащего элементы управления: текстовые поля, флажки, кнопки с зависимой фиксацией, меню и т.д. HTML форма задается тегом
3. Элементы управления Пользователи взаимодействуют с формами с помощью управляющих элементов. Для задания большинства управляющих элементов используется тег.

2. МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ПО ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

2.1 Лабораторная работа № 1, 2 (4 часа).

Тема: «Введение в HTML. Структура HTML документа.»

2.1.1 Цель работы: Ознакомиться с представлением HTML-документов.

2.1.2 Задачи работы:

1. Структура документа;
2. Структура и параметры тегов.

2.1.3 Описание (ход) работы:

1. HTML-документ заключается в теги **<html>** и **</html>**. Между этими тегами располагаются два раздела: раздел заголовка (элемент **head**) и раздел тела документа (элемент **body** для простого документа либо элемент **frameset**, задающий набор кадров). Все указанные элементы имеют начальный и конечный тег.

2. Секция заголовка содержит описание параметров, используемых при отображении документа, но не отражающихся непосредственно в окне браузера. Секция тела документа содержит текст, предназначенный для отображения браузером и элементы, указывающие на способ форматирования текста, определяющие графическое оформление документа, задающие параметры гиперссылок и так далее.

2.2 Лабораторная работа № 3, 4 (4 часа).

Тема: Слои. CSS верстка. Синтаксис. Свойства. Селекторы.

Позиционирование элементов.

2.2.1 Цель работы: Ознакомиться с областью применения слоев и с построением модульной сетки при помощи слоев.

2.8.2 Задачи работы:

1. Слои: основные понятия;
2. Область применения; параметры слоев;
3. CSS верстка: принципы верстки при помощи слоев;
4. Построение модульной сетки при помощи слоев.

2.8.3 Описание (ход) работы:

1. Слои документа в Photoshop можно перемещать для изменения положения и видимости отдельных фрагментов изображения аналогично перекладыванию листов бумаги в стопке, менять их порядок. Также есть возможность изменять уровень непрозрачности любого слоя, чтобы сделать его содержимое прозрачным или полупрозрачным.

Основными параметрами слоя являются режимы наложения, степень непрозрачности и уровень непрозрачности заливки, которые находятся в верхней части палитры СЛОИ. Для слоев работают те же режимы наложения, что и для инструментов рисования и применяются они ко всему содержимому слоя. Также на все пиксели слоя действует степень непрозрачности. В отличие от параметра НЕПРОЗРАЧНОСТЬ, параметр ЗАЛИВКА воздействует только на нанесенные на слой пиксели или нарисованные на нем формы, не оказывая влияния на прозрачность слоевых эффектов слоя. Например применяем к слою с текстом "Стиль слоя" ОБВОДКА и меняем непрозрачность слоя до 20% и видим, что и текст и обводка одинаково сменили непрозрачность.

2. Первоначально слои ввела компания Netscape, включив в свой браузер поддержку тега **<layer>**. Этот тег позволял прятать/показывать текущее содержимое, устанавливать положение относительно окна браузера, накладывать один слой поверх других и загружать данные в содержимое слоя из файла. Что характерно, все эти

параметры легко менялись с помощью JavaScript и это расширяло возможности по созданию действительно динамического контента на странице. Несмотря на столь впечатляющий набор возможностей, тег `<layer>` не был включен в спецификацию HTML и так и остался лишь расширением браузера Netscape.

3. **Модульная сетка** – хорошее дополнительное «приспособление» для выравнивания и позиционирования элементов, размещённых на будущей веб-странице. Перед началом работы над макетом страницы я всегда делаю модульную сетку «персонально» для каждого проекта исходя из ширины будущей страницы. В интернете существует множество различных сервисов по созданию модульных сеток. Так же достаточно плагинов для фотошоп благодаря которым можно сделать такие сетки. Однако я не вижу смысла в их использовании так как задача эта решается буквально в два действия

2.3 Лабораторная работа № 5 (2 часа).

Тема: Введение в JavaScript. Синтаксис языка. Объектная модель JavaScript.

2.3.1 Цель работы: Ознакомиться с синтаксисом языка JavaScript.

2.3.2 Задачи работы:

1. Объектная модель: модель DOM. Синтаксис языка JavaScript.

2.3.3 Описание (ход) работы:

Большинство действий в javascript выполняется с HTML-страницей. В javascript страница представлена в виде объектной модели DOM (DocumentObjectModel).

Любые действия со страницей требуют вызова соответствующего метода DOM.

Понимание, как работать с документом в модели DOM - краеугольный камень в javascript-программировании.

- Введение. DOM в примерах.

Основным инструментом работы и динамических изменений на странице является DOM (DocumentObjectModel) - объектная модель, используемая для XML/HTML-документов.

Согласно DOM-модели, документ является иерархией. Каждый HTML-тег образует отдельный элемент-узел, каждый фрагмент текста - текстовый элемент, и т.п.

2.4 Лабораторная работа № 6 (2 часа).

Тема: Обработка событий. Примеры эффективного программирования на JavaScript.

2.4.1 Цель работы: Ознакомиться с примерами эффективного программирования.

2.4.2 Задачи работы:

1. Примеры эффективного программирования на JS.

2.4.3 Описание (ход) работы:

JavaScript - объектный скриптовый язык программирования, разработанный в 1995 году.

JavaScript обычно используется как встраиваемый язык для программного доступа к объектам приложений. Наиболее широкое применение находит в браузерах как язык сценариев для придания интерактивности веб-страницам (встраивается в HTML)

Основные архитектурные черты JavaScript: динамическая типизация, слабая типизация, автоматическое управление памятью, прототипное программирование, функции как объекты первого класса.