федеральное государственное бюджетное образовательное учреждение высшего образования «Мордовский государственный педагогический университет имени М.Е. Евсевьева»

Факультет среднего профессионального образования

РАБОЧАЯ ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

Наименование дисциплины: ОП.18 Web-программирование
Специальность: 09.02.07 Информационные системы и программирование
Форма обучения: очная

Разработчик: Базаркин А. Ф., преподаватель факультета среднего профессионального образования.

Программа рассмотрена и утверждена на заседании предметно-цикловой комиссии профессионального цикла по специальности 09.02.07 Информационные системы и программирование от 30.03.2020 г., протокол №9.

Программа с обновлениями рассмотрена и утверждена на заседании предметно-цикловой комиссии профессионального цикла по специальности 09.02.07 Информационные системы и программирование от 01.09.2020 г., протокол № 1.

СОДЕРЖАНИЕ

1 Общая характеристика рабочей программы учебной дисциплины 2
2 Структура и содержание учебной дисциплины 3
3 Условия реализации учебной дисциплины 11
4 Контроль и оценка результатов освоения учебной дисциплины 12
5 Методические рекомендации по организации самостоятельной работы обучающихся 13

1 ОБЩАЯ ХАРАКТЕРИСТИКА РАБОЧЕЙ ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ «ОП.18 WEB-ПРОГРАММИРОВАНИЕ»

1.1. Область применения программы

Программа учебной дисциплины «ОП.18 Web-программирование» является частью программы подготовки специалистов среднего звена в соответствии с ФГОС по специальности среднего профессионального образования 09.02.07 Информационные системы и программирование углубленной подготовки укрупненной группы специальностей 09.00.00 Информатика и вычислительная техника.

1.2. Место учебной дисциплины в структуре программы подготовки специалиста среднего звена:

Дисциплина «ОП.18 Web-программирование» входит в вариативную часть общепрофессионального цикла образовательной программы среднего профессионального образования – программы подготовки специалистов среднего звена – по специальности 09.02.07 Информационные системы и программирование.

Изучение данного учебного курса является необходимой основой для последующего изучения дисциплин профессиональной подготовки, а также для прохождения учебной и производственной практик, подготовки студентов к государственной итоговой аттестации.

1.3. Цели и задачи учебной дисциплины – требования к результатам освоения учебной дисциплины:

1. Цели и задачи дисциплины

Целью дисциплины является формирование умений и навыков разработки Web-сайтов.

Задачи дисциплины:

– сформировать систему знаний и навыков основ web-программирования;

– изучить основы технологий проектирования сайтов различными программными средствами.

Компетенции, на формирование которых направлен процесс изучения дисциплины:

– выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам (ОК 01);

– осуществлять поиск, анализ и интерпретацию информации, необходимой для выполнения задач профессиональной деятельности. (ОК 02);

– использовать информационные технологии в профессиональной деятельности (ОК 09);

– формировать алгоритмы разработки программных модулей в соответствии с техническим заданием (1.1);

– разрабатывать программные модули в соответствии с техническим заданием (1.2);

– разрабатывать модули программного обеспечения для мобильных платформ (1.6).

В результате освоения учебной дисциплины обучающийся должен

уметь:

– разрабатывать Web-сайты, используя технологии проектирования сайтов и web-программирования, и использовать их на практике;

знать:

– основы web-программирования;

– основы проектирования сайтов и технологии проектирования;

– основы программирования сайтов различными программными средствами.

1.4. Рекомендуемое количество часов на освоение примерной программы учебной дисциплины:

максимальной учебной нагрузки обучающегося 109 часов, в том числе:
обязательной аудиторной учебной нагрузки обучающегося 91 час;
самостоятельная работа 18 часов.

2 СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

2.1. Объем учебной дисциплины и виды учебной работы

Вид учебной работы	**Объем часов**
Максимальная учебная нагрузка (всего)	**109**
Обязательная аудиторная учебная нагрузка (всего)	**91**
в том числе:	
лекции	44
практические занятия	47
Самостоятельная работа	**18**
Итоговая аттестация в форме зачета	

2.2. Тематический план и содержание учебной дисциплины «ОП.18 Web-программирование»

Наименование разделов и тем	Содержание учебного материала, лабораторные работы и практические занятия, самостоятельная работа обучающихся, курсовая работа (проект)	Объем часов	Уровень освоения
1	2	3	4
Раздел I	**Введение в HTML**	52	
Тема 1.1 Введение	Содержание учебного материала	1	*1*
	1 Локальные и глобальные сети..	1	
	2 Гипертекст. Мультимедиа ресурсы		
	3 Web-сервисы.		
Тема 1.2 Язык разметки гипертекста	Содержание учебного материала	1	*1,2*
	1 Структура web-страницы. Начало и конец страницы.	1	
	2 Описание страницы. Имя страницы. Содержание страницы.		
Тема 1.3. Форматирование текста в HTML.	Содержание учебного материала	5	*2*
	1 Заголовок. Атрибуты тега. Абзац. Атрибут абзаца..	1	
	2 Форматирование шрифта. Атрибуты. Полужирный шрифт. Курсив. Размер шрифта		
	3 Цвет шрифта Гарнитура шрифта		
	Практические занятия Форматирование текста в HTML.	4	
Тема 1.4. Атрибуты изображений в HTML.	Содержание учебного материала	4	*2*
	1 Вставка изображения. Вывод альтернативного текста вместо изображения.	1	
	2 Размещение изображения в тексте.		
	Практические занятия Вставка изображений на Web-страницу.	3	
Тема 1.5. Организация гиперссылок в HTML.	Содержание учебного материала	4	*2*
	1 Ссылка на другую страницу (в качестве ссылки выступает текст).). Цвет ссылки.	1	
	2 Ссылка на другую страницу (в качестве ссылки выступает рисунок		
	3 Цвет пройденной ссылки. Цвет активной ссылки		

	Практические занятия Организация гиперссылок на страницы и файлы.		3	
Тема 1.6. Организация карт изображений в HTML.	Содержание учебного материала		5	*2*
	1	Карта изображений. Прямоугольная область. Овальная область. Многоугольная область.	1	
	2	Определение координат базовых точек.		
	Практические занятия Создание карты ссылок на Web-странице.		4	
Тема 1.7. Гиперссылки в пределах одной страницы. Якоря.	Содержание учебного материала		5	*2*
	1	Якорь. Гиперссылка.	1	
	2	Атрибуты тега <A>.		
	Практические занятия Организация гиперссылок в пределах одной страницы.		4	
Тема 1.8. Создание таблиц в HTML.	Содержание учебного материала		5	*2*
	1	Атрибуты команды TABLE. Строка таблицы.	1	
	2	Атрибуты команды TR Ячейка внутри строки таблицы. Ячейка – заголовок. Атрибуты команды TD (TH).		
	Практические занятия Вставка таблиц в HTML-страницу.		4	
Тема 1.9. Объединение строк и столбцов в HTML-таблицах.	Содержание учебного материала		5	*2*
	1	Объединение по вертикали. Объединение по горизонтали.	1	
	2	Атрибуты colspan и rowspan.		
	Практические занятия Вставка таблиц в HTML-страницу.		4	
Тема 1.10. Списки в HTML.	Содержание учебного материала		6	*2*
	1	Вставка маркированного списка. Вставка нумерованного списка.	1	
	2	Элемент маркированного или нумерованного списка.		
	Практические занятия Организация списков в HTML.		5	
Тема 1.11. Фреймы в HTML.	Содержание учебного материала		6	*2*
	1	Фрейм. Область применения фреймов.	2	
	2	Преимущества и недостатки фреймовой структуры.		

	Практические занятия Разработка многофреймовой страницы.		4	
Тема 1.12. Атрибуты фреймов.	Содержание учебного материала		1	
				2
	1	Атрибуты тега <frameset>. Атрибуты тега <frame>.	1	
	2	Управление полосами прокрутки. Управление размерами и местоположением фреймов.		
Тема 1.13. «Плавающий» фрейм.	Содержание учебного материала		4	*2*
	1	Применение плавающего фрейма. Атрибуты тега <*iframe*>.	1	
	2	Загрузка содержимого фрейма по ссылке.		
	Практические занятия Создание фреймовой структуры страницы.		3	
Раздел II	**Формы и стили в HTML**		14	
Тема 2.1. Формы в HTML	Содержание учебного материала		2	*2*
	1	Форма HTML. Назначением формы.	2	
	2	Виды форм. Заполнение и обработка формы.		
Тема 2.2. Атрибуты форм.	Содержание учебного материала		2	*2,3*
	1	Методы. Действия.		
	2	Свойства объектов. Атрибуты форм.		
Тема 2.3. Каскадные таблицы стилей.	Содержание учебного материала		1	*2,3*
	1	Понятие стиля. Атрибуты текста и абзаца.	1	
	2	Назначение стилей. Преимущества использования таблиц стилей.		
Тема 2.4. Назначение стилей отдельным элементам страницы.	Содержание учебного материала		1	*2*
	1	Создание таблицы стилей/style sheet..	1	
	2	Базовая модель CSS. Коды для использования для CSS в HTML-документе		
Тема 2.5. Структура CSS- файла.	Содержание учебного материала		8	*2*
	1	Текст. Ссылки.	1	
	2	Идентификация и группировка элементов.		

	Практические занятия Форматирование страницы с помощью CSS.		4	
Раздел III	**Основы Java Script.**		21	
Введение в Java Script.	Содержание учебного материала		1	*2*
	1	История возникновения JavaScript. Возможности JavaScript .		
	2	Тенденции развития JavaScript.		
Тема 3.2. Основные синтаксические конструкции JavaScript.	Содержание учебного материала		1	*2*
	1	Переменные. Типы данных.	1	
	2	Ветвление.		
	3	Циклы. Сравнение.		
Тема 3.3. Функции даты-времени в JavaScript.	Содержание учебного материала		1	*2,3*
	1	Функции определения даты.	1	
	2	Функции определения времени.		
	3	Обновление содержимого по расписанию.		
	Самостоятельная работа: Создание динамических скриптов на странице сайта		4	
Тема 3.4. Обработка событий в JavaScript.	Содержание учебного материала		1	*2*
	1	События. Виды. событий..	1	
	2	Обработчики событий. Примеры событий.		
	3	Реализация обработки событий на JavaScript		
Тема 3.5. Способы размещения скриптов.	Содержание учебного материала		1	*2*
	1	Размещение скриптов в html-файлах..	1	
	2	Размещение скриптов в js-файлах		
	3	Структура js-файла.		
Тема 3.6. Встроенные функции JavaScript.	Содержание учебного материала		1	*2*
	1	Математические функции.	1	
	2	Логические функции.		
	3	Функции обработки событий.		
	Содержание учебного материала		1	

Тема 3.7. Пользовательские функции в JavaScript.	**1**	Объявление функций. Локальные и глобальные переменные.	1	
	2	Параметры.		
	3	Аргументы по умолчанию.		
Тема 3.8. Массивы в JavaScript.	Содержание учебного материала		1	*1,2*
	1	Объявление массивов. Методы обработки массивов.	1	
	2	Перебор элементов.		
	3	Многомерные массивы.		
Тема 3.9. Работа с формами в JavaScript.	Содержание учебного материала		1	*1,2*
	1	Иерархия объектов в JavaScript.	1	
	2	Атрибуты и пользовательские свойства форм и объектов.		
	3	Методы и свойства объекта location.		
Тема 3.10. Работа с окнами браузера в JavaScript.		Содержание учебного материала	7	*2*
	1	Объект Window..	1	
	2	Методы и свойства объекта Window		
	3	Управление положением и размером окон браузера.		
	Практические занятия Управление окнами браузера средствами JavaScript.		3	
Тема 3.11. Простая галерея на JavaScript.	Содержание учебного материала		1	*2,3*
	1	Массив графических объектов. Свойства изображений	1	
	2	. Масштабирование изображений.		
	3	Практическое применение циклов.		
	Самостоятельная работа: Создание простейшей галереи на Java Script		2	
Тема 3.14. Динамический вывод текста.	Содержание учебного материала		1	*2,3*
	1	Динамические эффекты на Web-странице.	1	
	2	Свойства текстовых величин.		
	3	Практическое применение условных циклов.		
	Самостоятельная работа: Создать таблицу на установка атрибутов динамического текста		2	
Тема 3.15. Применение к	Содержание учебного материала		1	*2*
	1	Управление яркостью и цветом фона страниц.	1	

тексту визуальных эффектов.	**2**	Цветовая таблица.		
	3	Реализация эффекта прозрачности.		
	Самостоятельная работа: Проработка теоретического материала		2	
Тема 3.16. Вывод служебной информации на Web-страницу.	Содержание учебного материала		1	*2,3*
	1	Отладка JavaScript.-кода.	1	
	2	Информация о Web-странице.		
	3	Получение служебной информации средствами JavaScript.		
	Самостоятельная работа: Написание скрипта, выводящего подробные метаданные		2	
Тема 3.17. Защита Web-страницы.	Содержание учебного материала		1	*2,3*
	1	Ограничение доступа к Web-странице.	1	
	2	Запрет выделения фрагментов страницы.		
	Самостоятельная работа: Изучение работы протокола HTTPS		2	
Раздел IV	**Общие принципы разработки WEB-приложений.**		12	*2*
Тема 4.1. Дизайнерские требования к оформлению Web-страницы.	Содержание учебного материала		6	
	1	Эргономика.	2	
	2	Особенности визуального восприятия.		
	3	Стандартные элементы интерфейса.		
	Практические занятия Анализ Web-страниц с точки зрения дизайна и эргономики.		2	
Тема 4.2. Анализ и редактировани е скриптов.	Содержание учебного материала		1	*2,3*
	1	Объектная модель JavaScript.	1	
	2	. Синтаксис и алгоритмические конструкции JavaScript.		
	3	Ошибки применения методов и свойств. Синтаксические ошибки.		
	Самостоятельная работа: Синтаксический анализ скриптов		2	
Тема 4.3. Подбор цветовой схемы сайта.	Содержание учебного материала		1	*2*
	1	Цветовые сочетание.	1	
	2	Цветовой круг.		
	3	Устоявшиеся цветовые стереотипы.		

	Практические занятия			
Тема 4.4. Основные принципы выбора провайдера.	Содержание учебного материала		1	*2*
	1	Web-сервисы. Хостинг. Провайдер.	1	
	2	Услуги провайдера. Спам.		
	3	Фишинг.		
	Практические занятия			
Тема 4.5. Визуальные методы акцентирования внимания пользователя.	Содержание учебного материала		1	*2,3*
	1	Структура Web-страницы..	1	
	2	Группировка информации по различным признакам. Эргономика восприятия		
	3	Информационная насыщенность контента.		
	Самостоятельная работа: Создание макета рекламной страницы		2	
Тема 4.6. Распространенные ошибки начинающих Web-разработчиков.	Содержание учебного материала		1	*2*
	1	Планирование работы по созданию сайта. Ошибки дизайна.	1	
	2	Этапы разработки сайта. Функции и обязанности разработчиков сайта.		
	3	Ошибки интерфейса. Ошибки публикации.		
Тема 4.7. Перспективные пути развития Web-технологий.	Содержание учебного материала		1	*2*
	1	Web-сервисы нового поколения.	1	
	2	Мультимедийные ресурсы.		
	3	Перспективы развития средств связи.		
Итого:			**109**	

Для характеристики уровня освоения учебного материала используются следующие обозначения:

1. – ознакомительный (узнавание ранее изученных объектов, свойств);
2. – репродуктивный (выполнение деятельности по образцу, инструкции или под руководством)
3. – продуктивный (планирование и самостоятельное выполнение деятельности, решение проблемных задач)

3 УСЛОВИЯ РЕАЛИЗАЦИИ УЧЕБНОЙ ДИСЦИПЛИНЫ

3.1. Материально-техническое обеспечение

Для реализации программы учебной дисциплины предусмотрена лаборатория разработки веб-приложений, оснащенная необходимым оборудованием:

Основное оборудование:

Автоматизированное рабочее место преподавателя (персональный компьютер (процессор Core i7, оперативная память 8 Гб; дискретная видеокарта 8GB ОЗУ монитор 24”), проектор мультимедийный, интерактивная доска, маркерная доска, видеокамера Hikvision, принтер А3, цветной, многофункциональное устройство формата А4); автоматизированное рабочие места для обучающихся (персональный компьютер (процессор Core i7, оперативная память 8 Гб, дискретная видеокарта 8GB ОЗУ, монитор 24”) – 14 шт., эргономичная мебель для работы за компьютером – 14 шт.); офисный мольберт (флипчарт).

Учебно-наглядные пособия:

Презентации.

Лицензионное программное обеспечение:

– Microsoft Windows 7 Pro;

– Microsoft Office Professional Plus 2010.

3.2. Информационное обеспечение обучения

Перечень рекомендуемых учебных изданий, Интернет-ресурсов, дополнительной литературы

Основная литература

1. Малашкевич, В. Б. Интернет-программирование : лабораторный практикум / В. Б. Малашкевич ; Поволжский государственный технологический университет. – Йошкар-Ола : Издательство ПГТУ, 2017. – 96 с. – URL : http://biblioclub.ru/index.php?page=book&id=476400 – ISBN 978-5-8158-1854-5. – Текст : электронный.

2. Тузовский, А. Ф. Проектирование и разработка web-приложений : учебное пособие для среднего профессионального образования / А. Ф. Тузовский. – Москва : Издательство Юрайт, 2020. – 218 с. – URL: http://biblio-online.ru/bcode/456394 – ISBN 978-5-534-10017-4. – Текст : электронный.

Дополнительная литература

1. Сысолетин, Е. Г. Разработка интернет-приложений : учебное пособие для среднего профессионального образования / Е. Г. Сысолетин, С. Д. Ростунцев. – Москва : Издательство Юрайт, 2020. – 90 с. – URL: http://biblio-online.ru/bcode/456393 – ISBN 978-5-534-10015-0. – Текст : электронный.

4 КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

Контроль и оценка результатов освоения учебной дисциплины осуществляется преподавателем в процессе проведения практических занятий, тестирования, а также выполнения обучающимися индивидуальных заданий.

Результаты обучения	***Критерии оценки***	***Формы и методы оценки***
Перечень знаний, осваиваемых в рамках дисциплины – основы web-дизайна и программирования; – основы проектирования сайтов и технологии проектирования; – основы программирования сайтов различными программными средствами. *Перечень умений, осваиваемых в рамках дисциплины:* – разрабатывать Web-сайты, используя технологии проектирования сайтов и web-программирования, и использовать их на практике;	*«Отлично»* – теоретическое содержание курса освоено полностью, без пробелов, умения сформированы, все предусмотренные программой учебные задания выполнены, качество их выполнения оценено высоко. *«Хорошо»* – теоретическое содержание курса освоено полностью, без пробелов, некоторые умения сформированы недостаточно, все предусмотренные программой учебные задания выполнены, некоторые виды заданий выполнены с ошибками. *«Удовлетворительно»* – теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые умения работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий содержат ошибки. *«Неудовлетворительно»* – теоретическое содержание курса не освоено, необходимые умения не сформированы, выполненные учебные задания содержат грубые ошибки.	– Компьютерное тестирование на знание терминологии по теме. – Тестирование. – Контрольная работа. – Самостоятельная работа. – Защита реферата. – Семинар. – Наблюдение за выполнением практического задания (деятельностью студента). – Оценка выполнения практического задания (работы). – Подготовка и выступление с докладом, сообщением, презентацией. – Решение ситуационной задачи.

5 МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ОРГАНИЗАЦИИ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

5.1. Выполнение самостоятельных работ

Самостоятельная работа студентов по дисциплине проводится с целью:

– систематизации и закрепления полученных теоретических знаний и практических умений по дисциплине;

– углубления и расширения теоретических знаний;

– формирования умений использовать полученные знания в новых условиях;

– развития познавательных и творческих способностей;

– формирования самостоятельности мышления, способности к саморазвитию, самореализации.

Перед выполнением обучающимися внеаудиторной самостоятельной работы преподаватель проводит инструктаж по выполнению заданий, которые включают цель задания, его содержание, сроки выполнения, объем работы, основные требования к результатам работы, критерии оценки результатов внеаудиторной самостоятельной работы.

В качестве форм контроля внеаудиторной самостоятельной работы обучающихся используется тестирование, самоотчеты, контрольные работы.

Для выполнения заданий самостоятельной работы необходимо тщательно изучить теоретический материал и практические занятия. Самостоятельные работы выполняются на любом компьютере с установленной операционной системой и наличием актуальной версии браузера.

Самостоятельная работа студента предполагает написание программного кода на языках веб-программирования, демонстрация полученных знаний, умений.

5.2 Методические рекомендации по написанию реферата

Работа студента над рефератом состоит из следующих этапов:

1. Выбор темы на основе предложенной тематики;

2. Подбор материала (посещение библиотеки, просмотр информационных программ, изучение научных работ, статистических данных, материалов периодической печати);

3. Подготовка и написание реферата;

4. Защита реферата на практическом занятии.

Реферат должен иметь следующую структуру:

– план;

– введение;

– изложение основного содержания темы;

– заключение;

– список используемой литературы.

Предварительный план реферата состоит обычно из трех – четырех вопросов, в процессе работы он уточняется и конкретизируется.

При работе над рефератом необходимо внимательно изучить соответствующую теме литературу.

Основному тексту в реферате предшествует введение. В нем необходимо показать значение, актуальность рассматриваемой проблемы, обоснованность причины выбора темы, кроме того, следует отметить, в каких произведениях известных авторов рассматривается изучаемая проблема, сформировать основную задачу, которая ставиться в реферате.

В основной части работы большое внимание необходимо уделить глубокому теоретическому освещению как темы в целом, так и отдельным ее вопросам, правильно связать теоретические положения с практикой, конкретным фактическим материалом. Изложение должно осуществляться в соответствии с составленным планом.

Реферат должен быть написан ясным языком, без повторений, сокращений, противоречий между отдельными положениями.

Большое значение имеет правильное оформление реферата. Страницы текста, включенные в реферат приложения, таблицы и распечатки должны соответствовать формату А4. Титульный лист должен содержать реквизиты: название учебного заведения, по какой дисциплине написан реферат, тема, кто выполнил работу (фамилия, инициалы, номер группы) и кто проверил работу (фамилия, инициалы преподавателя). Реферат должен быть выполнен машинописным способом на одной стороне листа белой бумаги через полтора интервала, 14 шрифтом (допускается написание реферата от руки пастой синего или черного цвета).

Текст реферата следует печатать, соблюдая следующие размеры полей: левое – 30 мм, правое – 15 мм, верхнее и нижнее – 20 мм.

Все линии, буквы, цифры и знаки должны быть одинаково черными по всему реферату.

Заголовки разделов основной части следует располагать в середине строки без точки в конце и печатать прописными буквами, не подчеркивая.

Страницы реферата следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию по всему тексту отчета. Номер страницы проставляют посередине листа в верхнем поле без точки в конце. Титульный лист включают в общую нумерацию страниц реферата. Номера страниц на титульном листе и в оглавлении не проставляют.

Приводимые в тексте цитаты из литературы, а также статистические данные должны быть снабжены соответствующими ссылками на источники, из которых они взяты, с указанием авторов, названия работы, тома, страницы. Объем реферата 10 – 15 листов.

В конце реферата приводится список использованной литературы. Использованные в реферате источники указываются в алфавитном порядке фамилии авторов.

Примерная тематика рефератов

1. Язык разметки гипертекста HTML-5, его возможности и отличия от предыдущих версий.

2. Web-сервер Apache. Назначение, применение в современной web-разработке.

3. Каскадные таблицы стилей CSS 3. Назначение, применение в современной web-разработке.

4. Фреймворк Bootstrap 4. Назначение, применение в современной web-разработке.

5. Клиентский скриптовый язык JavaScript. Назначение, применение в современной web-разработке.

6. Библиотека jQuery. Назначение, применение в современной web-разработке.

7. Серверный скриптовый язык PHP. Назначение, применение в современной web-разработке.

8. Технология AJAX. Назначение, применение в современной web-разработке.

9. Язык MySQL. Назначение, применение в современной web-разработке.

10. HTML-редактор Dreamweaver. Назначение, применение в современной web-разработке.

11. Система управления содержимым сайта WordPress. Назначение, применение в современной web-разработке.

12. Язык программирования Python и его использование в современной web-разработке.

13. Фреймворк Django. Назначение, применение в современной web-разработке.

14. Node.js – серверная реализация языка JavaScript. Назначение, применение в современной web-разработке.

5.3 Методические рекомендации по созданию динамических скриптов на странице сайта

Современные динамические интерфейсы web-страниц подразумевают не только изменение содержимого различных тегов, но и динамическую подгрузку скриптов JavaScript с их последующим выполнением. Например, получение скриптов или данных для них через AJAX. Способы передачи данных я тут рассматривать не буду, а расскажу о том, как динамически добавить скрипт на сформированную web-страницу и затем выполнить его. Это можно сделать как минимум двумя способами. Первый способ – добавление скрипта средствами JavaScript с использованием стандартной функции eval. Она получает в качестве аргумента строку и, рассматривая ее содержимое как код JavaScript, пытается выполнить. Например:

```
1. <script type="text/javascript">
2. eval('function do_my_job(txt) { alert(txt); }');
3. do_my_job('ok');
```

```
</script>
```

Второй способ, более корректный, – это добавление скриптов через DOM. При этом создается новый объект script, заполняется его тип и текст, а затем созданный объект добавляется в качестве дочернего элемента к элементу head. В этом случае добавленный скрипт будет сразу же выполнен. Для удобства я написал небольшую функцию, получающую в качестве аргумента текст скрипта, и добавляющую его на страницу.

```
">
function add_script(txt) {
    var newScript = document.createElement("script");
    newScript.type = "text/javascript";
    newScript.text = txt;
    document.getElementsByTagName('head')[0].appendChild(newScript);
}
</script>
```

Во всех тестовых браузерах (IE 5.5-8, Opera 7-10, Firefox 2-3, Chrome, Safari и т.д.) скрипт выполнялся также при добавлении его к элементу body. Опытным путем установлено, что скрипт срабатывает при добавлении и к другим элементам страницы, но лучше, наверное, так не делать для сохранения кроссбраузерности и совместимости.

5.4 Методические рекомендации по созданию таблицы на установку атрибутов динамического текста

Для установки атрибутов динамического текста панель инспектора свойств содержит следующие элементы:

– текстовое поле *Instance Name* (Имя образца), в котором указывается имя текстового поля; несмотря на то, что имя поля выводится (непосредственно в нем) символами серого цвета, которые обычно обозначают в Windows-приложениях заблокированный элемент интерфейса, в данном случае ввод разрешен;

– раскрывающийся список *Line type* (Тип строки) форматов текстового поля:

- *Single Line* (Однострочное);
- *Multiline* (Многострочное);
- *Multiline no wrap* (Многострочное без переносов);

– кнопка *Render text as HTML;* если она нажата, то заданные параметры форматирования текста (такие как размер, стиль, использование в качестве гиперссылки) при публикации фильма будут сохранены в виде соответствующих HTML-тэгов;

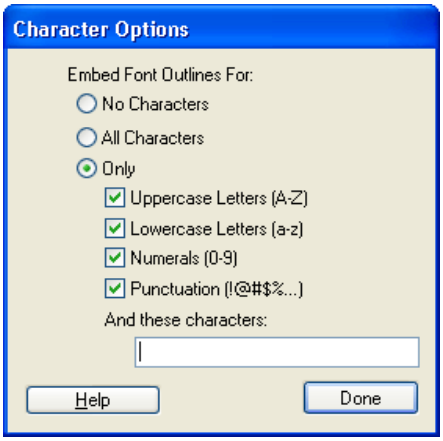
– кнопка *Show Border* (Показать рамку); если она нажата, то текстовое поле будет окружено рамкой;

– флажок *Selectable* (Выбираемый); его назначение аналогично рассмотренному для статического текста;

– текстовое поле *Var* (от Variable – Переменная) предназначено для ввода имени переменной, связанной с создаваемым текстовым полем; об использовании переменных для управления элементами фильма будет рассказано в подразделе «Создание интерактивных элементов фильма»;

– кнопка *Characters* (Символы), щелчок на которой открывает дополнительное диалоговое окно *Character Options*, позволяющее установить параметры использования встроенного шрифта, используемого для текстового поля; окно содержит три переключателя:

- *No Characters* (Никакие символы) — информация об используемом шрифте не экспортируется в Flash-фильм при его публикации;
- *All Characters* (Все символы) - - в Flash-фильм включается информация о всех символах шрифта;
- *Only* (Избранные) — в Flash-фильм включается информация о только о тех символах шрифта, которые указаны с помощью расположенных ниже флажков.



По поводу установки параметров встроенного шрифта следует сделать следующее пояснение.

Когда вы используете в Flash-фильме шрифт, установленный на вашем компьютере, Flash внедряет информацию о шрифте в SWF-файл, гарантируя тем самым, что текст будет корректно отображен Flash-плеером. Однако не все шрифты, используемые в Flash, могут экспортироваться в SWF-файл. Поэтому

предварительно рекомендуется проверять, может ли экспортироваться данный шрифт. Для этого необходимо в меню *View* выбрать команду *Antialias Text* (Сглаживание текста) и оценить результат сглаживания. Если сглаживание не выполнено, это означает, что Flash не распознает такой шрифт и не будет его экспортировать.

Альтернативный способ – использование физических шрифтов (Device Font).

Информация о физическом шрифте не экспортируется в SWF-файл. Вместо этого Flash-плеер использует любой шрифт из числа установленных на компьютере, наиболее близкий к физическому шрифту.

Поскольку информация о физическом шрифте не включается в SWF-файл, такой вариант обеспечивает несколько меньший размер файла Flash-фильма. Кроме того, физический шрифт может быть более четким по сравнению с внедренным шрифтом для мелких символов (менее 10 пунктов). Однако, если на компьютере пользователя отсутствует подходящий шрифт, текст может выглядеть совсем не так, как ожидал автор фильма.

Flash содержит три вида физических шрифтов: _sans (близкий к шрифтам Helvetica и Arial); _serif (близкий к Times Roman); _typewriter (близкий к шрифту Courier).

Чтобы указать используемый в данном текстовом поле физический шрифт, необходимо выбрать его в списке шрифтов, имеющемся на панели инспектора свойств текста.

5.5 Методические рекомендации по созданию простейшей галереи на Java Script

1. Создадим html документ и назовем его: gallery.html. В нем напишем следующее:

```
<script src = "gal.js"></script>
<html>
    <head>
      <title> Gallery </title>
    </head>
    <body>
```

```
    <center>
      <img src = "" width = "640" height = "480"  id = "img" > </img> <br />
      <img src = "left_arrow.jpg"   onClick = "javascript: left_arrow()" />
    <img src = "right_arrow.jpg"  onClick = "javascript: right_arrow()" />
      </center>
  </body>
</html>
```

Мы создаем картинку с размером «640x480». Фиксированный размер задаем для того, чтобы изображение не смещалось, если оно будет иметь больший размер. Затем добавляем 2 изображения стрелок, на которые накладываем 2 события(onClick).

2. Создадим файл gal.js. В нем напишем несколько функций:

```
var mas = ["1.jpg","2.jpg","3.jpg","4.jpg","5.jpg"] // массив картинок
var to = -1;  // Счетчик, указывающий на текущую картинки

function right_arrow() // Открытие следующей картинки(движение вправо)
{
 var obj = document.getElementById("img");
  if (to < mas.length-1)  to++
   else
     to = 0;
     obj.src = mas[to];
```

```
}
```

```
function left_arrow() // Открытие предыдущей картинки(движение влево)
{
 var obj = document.getElementById("img");
if (to > 0) to--;
  else
    to = mas.length-1;
    obj.src = mas[to];
}
```

Вот мы и создали простую галерею.
Теперь немного усложним ее. Пусть при обновлении страницы последняя открытая картинка будет отображаться. Воспользуемся cookie (в нем будет хранится № последней картинки).

3. Улучшим файл gal.js

```
var mas = ["1.jpg","2.jpg","3.jpg","4.jpg","5.jpg"]
var to = -1;  // Счетчик, указывающий на текущую картинки
```

```
function right_arrow() // Открытие следующей картинки(движение вправо)
{
 var obj = document.getElementById("img");
```

```
  if (to < mas.length-1)  to++
   else
     to = 0;
     obj.src = mas[to];
     setCookie("foo", mas[to] , "", "/");  // запоминаем текущую картинк
y
}

function left_arrow()
{
 var obj = document.getElementById("img");
if (to > 0) to--;
  else
    to = mas.length-1;
    obj.src = mas[to];
    setCookie("foo", mas[to] , "", "/");   // запоминаем текущую картинк
y
}
```

```
function setCookie(name, value, expires, path, domain, secure) // Ф-ция
создания куки
{
 if (!name || !value) return false;
    var str = name + '=' + encodeURIComponent(value);
        if (expires) str += '; expires=' + expires.toGMTString();
        if (path)    str += '; path=' + path;
        if (domain)  str += '; domain=' + domain;
        if (secure)  str += '; secure';
  document.cookie = str;
  return true;
}

function getCookie(name)   // Ф-ция получения куки
{
  var pattern = "(?:; )?" + name + "=([^;]*);?";
  var regexp  = new RegExp(pattern);
    if (regexp.test(document.cookie))
        return decodeURIComponent(RegExp["$1"]);
```

```
        return false;
}

function Load()    // Ф-ция загрузки "сохраненной" картинки
{
var cook_val = getCookie("foo");  // Получаем значение куки по имени
 for (var i = 0 ; i < mas.length; i++)
 {
    if (mas[i] == cook_val)    // Как только встретилась
     {
      document.getElementById("img").src = mas[i];    // Загружаем картин
ку
      to = i;  // Задаем текущее значение счетчику
      break // выходим
     }
  }
}
```

```
<i>
```

Важно: Затем в тег body gallery.html добавим событие onLoad = "Load()".

```
</i>
```