Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет им. Г.И. Носова»

Многопрофильный колледж

УТВЕРЖДАЮ
Директор
_____ С.А. Махновский
«24» февраля 2021 г.

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ ПРАКТИЧЕСКИХ РАБОТ

ПМ.03 Техническое обслуживание и ремонт компьютерных систем и комплексов
МДК.03.01 «Техническое обслуживание и ремонт компьютерных систем и комплексов»
Раздел 3 Программное обеспечение компьютерных сетей и WEB-серверов
для студентов специальности

09.02.01 Компьютерные системы и комплексы

(базовой подготовки)

Магнитогорск, 2021

ОДОБРЕНО:
Предметно-цикловой комиссией
Информатика и вычислительная техника
Председатель И.Г.Зорина

Протокол № 6 от 17 февраля 2021 г.

Методической комиссией МпК
Протокол №3 от «24» февраля 2021г

Составитель:
преподаватель ФГБОУ ВО МГТУ МпК Денис Дмитриевич Тутаров

Методические указания по выполнению практических работ разработаны на основе рабочей программы ПМ. 03 Техническое обслуживание и ремонт компьютерных систем и комплексов

Содержание практических работ ориентировано на формирование общих и профессиональных компетенций по основной профессиональной образовательной программе по специальности 09.02.01 Компьютерные системы и комплексы: МДК.03.01. Техническое обслуживание и ремонт компьютерных систем и комплексов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки студентов составляют практические занятия.

Состав и содержание практических работ направлены на реализацию действующего федерального государственного образовательного стандарта среднего профессионального образования по специальности 09.02.01 Компьютерные системы и комплексы.

Ведущей дидактической целью практических занятий является формирование практических умений - профессиональных (умений выполнять определенные действия, операции, необходимые в последующем в профессиональной деятельности), необходимых в последующей учебной деятельности по профессиональным модулям.

В соответствии с рабочей программой ПМ.03 Техническое обслуживание и ремонт компьютерных систем и комплексов, МДК.03.01. Техническое обслуживание и ремонт компьютерных систем и комплексов предусмотрено проведение практических работ.

В результате их выполнения, обучающийся должен:

уметь:

- проводить контроль, диагностику и восстановление работоспособности компьютерных систем и комплексов;
- проводить системотехническое обслуживание компьютерных систем и комплексов;
- принимать участие в отладке и технических испытаниях компьютерных систем и комплексов;
- инсталляции, конфигурировании и настройке операционной системы, драйверов, резидентных программ;
- выполнять регламенты техники безопасности.

Содержание практических работ ориентировано на формирование общих компетенций по профессиональному модулю основной профессиональной образовательной программы по специальности:

ОК 1 Понимать сущность и социальную значимость своей будущей профессии, проявлять к ней устойчивый интерес.

ОК 2 Организовывать собственную деятельность, выбирать типовые методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.

ОК 3 Принимать решения в стандартных и нестандартных ситуациях и нести за них ответственность.

ОК 4 Осуществлять поиск и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.

ОК 5 Использовать информационно-коммуникационные технологии в профессиональной деятельности.

ОК 6 Работать в коллективе и команде, эффективно общаться с коллегами, руководством, потребителями.

ОК 7 Брать на себя ответственность за работу членов команды (подчиненных), результат выполнения заданий.

ОК 8 Самостоятельно определять задачи профессионального и личностного развития, заниматься самообразованием, осознанно планировать повышение квалификации.

ОК 9 Ориентироваться в условиях частой смены технологий в профессиональной деятельности.

И овладению профессиональными компетенциями:

ПК 3.1. Проводить контроль параметров, диагностику и восстановление работоспособности компьютерных систем и комплексов.

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Выполнение студентами практических работ по ПМ. 03 Техническое обслуживание и ремонт компьютерных систем и комплексов, МДК.03.01 Техническое обслуживание и ремонт компьютерных систем и комплексов направлено на:

- обобщение, систематизацию, углубление, закрепление, развитие и детализацию полученных теоретических знаний по конкретным темам междисциплинарных курсов;
- формирование умений применять полученные знания на практике, реализацию единства интеллектуальной и практической деятельности;
- формирование и развитие умений: наблюдать, сравнивать, сопоставлять, анализировать, делать выводы и обобщения, самостоятельно вести исследования, оформлять результаты в виде таблиц, схем, графиков;
- развитие интеллектуальных умений у будущих специалистов: аналитических, проектировочных, конструктивных и др.;
- выработку при решении поставленных задач профессионально значимых качеств, таких как самостоятельность, ответственность, точность, творческая инициатива.

Продолжительность выполнения практической работы составляет не менее двух академических часов и проводится после соответствующего занятия, которое обеспечивает наличие знаний, необходимых для ее выполнения.

1 МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Тема 3.2. Серверы приложений, протоколы

Практическое занятие № 1

Установка Web-сервера

Формируемая компетенция:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Изучить способы настройки и установки web-сервера

Выполнив работу, Вы будете:

уметь:

- настраивать сервера в локальной сети.

Материальное обеспечение:

рабочее место, оснащенное ПК;
инструкция для работы.

Задание:

1 Установить и настроить web-сервер.

Порядок выполнения работы:

1 Выполните установку Denver.
2 Настройте файл config_php.ini.
3 Подготовьте отчет.

Форма представления результата:

отчет.

Тема 3.3. Развитие языков разметки. HTML. CSS

Практическое занятие № 2

Форматирование текста

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Уметь форматировать текст в html страницах

Выполнив работу, Вы будете:

уметь:

- работать с текстами;

Материальное обеспечение:
рабочее место, оснащенное ПК;
инструкция для работы.

Задание:
Форматировать в html странице следующий параграф, используя тэги <font><h1><hr>
«Santa Fe – вседорожник, который изменит ваш взгляд на мир. Дерзкий и элегантный – идеальное сочетание. Уникальный образ Santa Fe: аэродинамичный профиль позволяет снизить расход топлива и шум в салоне. Мощь и передовые технологии: невероятное сочетание силы и плавности хода. Мощный и экономичный дизельный двигатель объемом 2,2 л. Запуск двигателя кнопкой и бесконтактный ключ. Запустить двигатель, а также открыть и закрыть двери можно, не вынимая ключ из кармана. Автоматическая 6-ступенчатая коробка передач помогает оптимизировать динамику и расход топлива автомобиля. Рычаг коробки передач – один из ярких элементов интерьера.»

Порядок выполнения работы:
1 Изучить справочник по тэгам html.
2 Подготовить страницу.

Форма представления результата:
созданная html страница.

Практическое занятие № 3

Работа с таблицами и списками

Формируемые компетенции:
ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.
ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

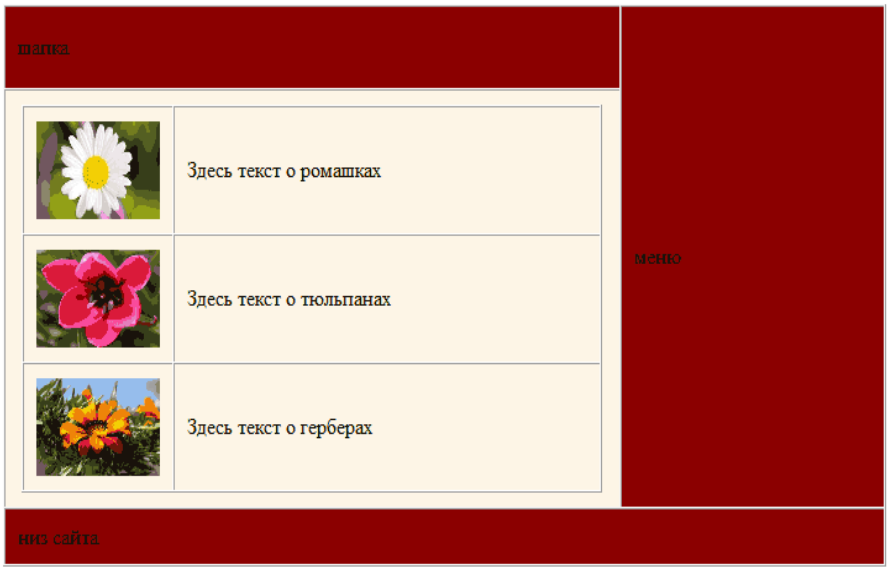
Цель работы: Научиться работать с таблицами и списками
Выполнив работу, Вы будете:
уметь:
- использовать таблицы.

Материальное обеспечение:
рабочее место, оснащенное ПК;
инструкция для работы.
Порядок выполнения работы:
1. Создайте таблицу по образцу.

Название таблицы		
1	**2**	**3**
11	12	13
21	22	23

31	32	33

2. Создайте таблицу сложной структуры



Форма представления результата:
созданная html страница.

Практическое занятие № 4

Создание панели навигации (меню)

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Научиться создавать панель меню
Выполнив работу, Вы будете:
уметь:
- создавать панель меню.

Материальное обеспечение:
рабочее место, оснащенное ПК;
инструкция для работы.
Порядок выполнения работы:

Напишите сценарий, который будет рассчитывать площадь прямоугольника по введенным пользователем длине и ширине. Для этого сначала разместите на html-странице нужные элементы формы:

```
<html>
  <head>
    <title>Расчет площади прямоугольника</title>
    <link rel=”stylesheet” type=”text/css” href=”style.css”>
    <script language=”javascript” src=”script.js”>  </script>
  </head>
```

```
  <body>
   <form name=”forma1”>
   Введите длину прямоугольника <input type=”text” name=”t1” size=”10”><br>
   <br>
   Введите ширину прямоугольника <input type=”text” name=”t2” size=”10”><br>
   <br>
   <input type=”button” name=”button” value=”Вычислить”><br><br>
    Площадь прямоугольника равна <input type=”text” name=”res” size=”10”>
   </form>
  </body>
</html>
```

Пользователь вводит значения ширины и длины и нажимает на кнопку "Вычислить". После чего, в поле площадь должен появиться результат. Таким образом, событие наступает при нажатии на кнопку "Вычислить", значит именно к ней мы и привяжем обработчик события. Функцию вычисления площади назовем "areaRectangle":

```
<input type=”button” name=”button” value=”Вычислить”
        onClick=”areaRectangle();”><br><br>
```

Теперь напишите саму функцию "areaRectangle". Для этого откроем страницу script.js и напишите тело функции. Для начала объявите три переменные: *a* - значение длины прямоугольника, *b* - значение ширины прямоугольника, *s* - площадь прямоугольника:

```
function areaRectangle(){
  var a;
  var b;
  var s;
}
```

Значение (value) *a* должно браться из текущей страницы (document), из формы с именем "forma1",из текстового поля с именем "t1". Так это и записывается *document.forma1.t1.value*, т.е. перечисляются через точку имена объектов от родительского до нужного(иерархическую структуру объектов мы обсуждали в предыдущем уроке). Последним указывается необходимое свойство объекта (value).

Аналогично и для значения *b* - *document.forma1.t2.value*.

Переменная *s* - есть произведение *a* на *b*. Запишите это в тело функции:

```
function areaRectangle(){
  var a=document.forma1.t1.value;
  var b=document.forma1.t2.value;
  var s=a*b;
}
```

Осталось только написать инструкцию записи вычисленной площади в текстовое поле с именем "res" нашей формы. Т.е надо, чтобы в текущую страницу, в форму с именем "forma1", в текстовое поле с именем "res", в качестве значения (value) было присвоено значение *s*.Так и запишем:

```
function areaRectangle(){
  var a=document.forma1.t1.value;
  var b=document.forma1.t2.value;
  var s=a*b;
  document.forma1.res.value=s;
}
```

Иными словами, мы сначала присвоили нашим переменным *a* и *b* значения из формы, затем произвели необходимые расчеты, а после этого присвоили некоторому элементу формы полученное значение *s*.

Форма представления результата:

созданная html страница.

Тема 3.4. Основы Web-программирования. Javascript. PHP.
Практическое занятие № 5

Создание анимированной галереи картинок

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Научиться создавать анимированную галерею картинок

Выполнив работу, Вы будете:

уметь:

- создавать анимированную галерею картинок.

Материальное обеспечение:

рабочее место, оснащенное ПК;
инструкция для работы.

Порядок выполнения работы:

Пусть у нас есть список ягод, а при наведении мышкой на название ягоды, ее описание будет появляться в текстовом поле.

- Малина
- Черника
- Ежевика

Такая конструкция может пригодиться не на одной странице, поэтому потребуется два параметра: один с именем объекта (формы),второй - с описанием ягод. Параметры в таком случае записываются через запятую. Функция будет выглядеть так:

```
function showDesc(obj, n){
  obj.desc.value=n;
}
```

где desc - имя текстового поля для вывода описаний. Эта функция будет срабатывать, когда на название ягоды наведен курсор, но понадобится еще одна функция, которая будет очищать текстовое поле, когда курсор выйдет за пределы названия. Назовем ее "delete":

```
function delet(obj){
  obj.desc.value=' ';
}
```

где пробел в одинарных кавычках означает пустую строку.

Форма представления результата:

созданная html страница.

Практическое занятие № 6

Создание формы для регистрации и входа

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Научиться создавать анимированную галерею картинок

Выполнив работу, Вы будете:

уметь:

- создавать форму для регистрации и входа.

Материальное обеспечение:

рабочее место, оснащенное ПК;

инструкция для работы.

Порядок выполнения работы:

Напишите код самой страницы. Обработчик события, когда указатель мыши помещается над элементом, называется*onMouseOver*, а обработчик события, когда указатель мыши выходит за пределы элемента - *onMouseOut*.

```
<html>
  <head>
    <title>javascript параметры</title>
    <link rel=”stylesheet” type=”text/css” href=”style1.css”>
    <script type=”text/javascipt” src=”script.js”></script>
  </head>
  <body>
    <form name=”forma2”>
      <textarea name=”desc” cols=45 rows=4></textarea>
    </form>
    <ul>
      <li onMouseOver=”showDesc(forma2, ‘Малина обыкновенная – кустарник с многолетним корневищем, из которого развиваются двухгодичные надземные стебли высотой до полутора метров.');”
 onMouseOut=”delet(forma2);”>Малина</li>

      <li onMouseOver=”showDesc(forma2, ‘Черника – кустарничек высотой
 15-30 см. Ветви отходят от главного стволика под острыми углами.');”
 onMouseOut=”delet(forma2);”>Черника</li>

      <li onMouseOver=”showDesc(forma2, ‘Ежевика – название нескольких
 видов растений из рода Rubus семейства Розовые.');”
 onMouseOut=”delet(forma2);”>Ежевика</li>
    </ul>
  </body>
</html>
```

Проверьте работу сценария и убедитесь, что вы поняли, как все это работает.

Самостоятельно напишите сценарии подсчета площади круга по заданному радиусу; средне годовой температуры воздуха по заданным месячным температурам.

Форма представления результата:
созданная html страница.

Тема 3.5. Расширяемый язык разметки XML
Практическое занятие № 7

Синтаксис XML

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: изучить язык PHP
Выполнив работу, Вы будете:
уметь:
- создавать ссылки;

Материальное обеспечение:
рабочее место, оснащенное ПК;
инструкция для работы.
Порядок выполнения работы:
Задание 1

Переменные в php начинаются со знака $, и могут состоять из латинских букв, цифр, и знака подчеркивания.

Для работы переменными напишите небольшой скрипт:

```
<?php
$a = 'Строковая переменная';
echo $a;
?>
```

Помимо записи и вывода переменных на экран, можно производить с ними математические операции:

```
<?php
$a = 5;
$b = 10;
$c = 2;
$result = (($a + $b) * $c) / $c;
echo "Ответ: $result";
?>
```

На экране появится число 15. При выводе переменной на экран использовались двойные кавычки ("). Если использовать одинарные кавычки ('), то на экран будет выведено не значение переменной, а имя переменной. Например:

```
<?php
$result = 15;
echo 'Ответ: $result';
?>
```

На экране вы увидите не ***Ответ: 15***, а «***Ответ: $result***». ***Работа с переменными*** одна из самых необходимых вещей в программировании на любом языке. Невозможно написать скрипт без использования переменных.

Задание 2:

Для создания массива напишите небольшой скрипт:

```
<?php
```

```
$arr[0] = 5;
$arr[1] = 3;
$arr[2] = 2;
echo ($arr[0] * $arr[1] + $arr[2]);
?>
```

На экране будет выведено 17 (5 * 3 + 2).

Если в массиве не указывать индекс, то он будет назначен автоматически.

```
<?php
$arr[] = 5;
$arr[] = 3;
$arr[] = 2;
echo ($arr[0] * $arr[1] + $arr[2]);
?>
```

На экране получим 17.

Ассоциативные массивы в php

Помимо обычных массивов, существуют еще и ассоциативные массивы, в принципе тоже самое, но ключ массива будет не число а строка.

```
<?php
$arr["asb"] = 2;
$arr["asd"] = 3;
$arr["asf"] = 4;
echo ($arr["asb"] * $arr["asd"] + $arr["asf"]);
?>
```

На выходе получаем 10.

Двумерные массивы в PHP

Массивы в php могут быть и многомерными. Рассмотрим небольшой пример:

```
<?php
$arr["abc"][0] = 3;
$arr["abc"][1] = 6;
$arr["xxx"][0] = 4;
echo ($arr["abc"][0] * $arr["abc"][1] + $arr["xxx"][0]);
?>
```

На экране получим 22.

Создание массивов при помощи функции array()

Массивы можно создавать при помощи ***функции array()***.

```
<?php
$arr = array(0 => 5, 1 => 3, 2 => 2);
echo ($arr[0] * $arr[1] + $arr[2]); //Получаем 17
?>
```

Первое значение — ключ, потом => значение.

Форма представления результата:

работающий скрипт.

Практическое занятие № 8

Применение языка описания типа документа к XML документу

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: изучить язык PHP
Выполнив работу, Вы будете:
уметь:
- создавать ссылки;

Материальное обеспечение:
рабочее место, оснащенное ПК;
инструкция для работы.

Порядок выполнения работы:

Условный оператор используется для сравнения. Условный оператор имеет следующую конструкцию:

```
<?php
if (Условие) { //Если условие верно
}
else { //Если условие не верно
}
?>
```

Если условие верно, то выполняется то, что находится в фигурных скобках оператора ***if***, если условие не верно, то выполняется то что находится в ***else***.

Так же есть ***более сложная конструкция***, с использованием ***elseif(Условие)***. Сравнение происходит при помощи следующих знаков:

1. == - равно
2. ***!***= - не равно
3. >= - больше или равно (если числа)
4. =< - меньше или равно (если числа)

Так же ***условия можно объединять*** при помощи логических операторов ***And (можно писать &&)*** и ***Or (можно писать ||)***.

Напишите пример. Скрипт будет сравнивать 3 переменные, и в зависимости от значений переменных выводить ту или иную информацию:

```
<?php
$a = 5;
$b = 8;
$c = 40;

if (($a * $b) == $c) {
echo "Условие 1 верно!";
}
elseif ($a * $b == 25) {
echo "Условие 2 верно!";
}
elseif (($a * $b == 50) And ($a == 10)) {
echo "Условие 3 верно!";
}
else {
echo "Все условия не верны";
}
?>
```

Так же есть ***логические операторы с жесткой привязкой к типу***:>

1. === - равно по типу
2. **!**=== - не равно по типу

То есть сравнивать можно только цифру с цифрой, строку со строкой.
Например:

```
<?php
if (5 === '5') {
echo 'верно';
}
else {
echo 'неверно';
}
?>
```

На экране будет выведено «неверно», т.к. сравниваем 5 со строкой '5'(в ковычнах). Если в условие написать 5 === 5, то условие будет верно (т.е. Число с числом)

Если перед ***условием поставить восклицательный знак (!)***, то будет все наоборот, например:

```
<?php
if (!(4 == 4)) {
echo 'верно';
}
else {
echo 'неверно';
}
?>
```

На экране появится «неверно».

Форма представления результата:
работающий скрипт.

Тема 3.6. Почтовые и клиентские серверы и их сервисы
Практическое занятие № 9

Отправка писем с помощью SMTP

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: изучить язык PHP
Выполнив работу, Вы будете:
уметь:
- отправлять письма с помощью SMTP;

Материальное обеспечение:
рабочее место, оснащенное ПК;
инструкция для работы.

Порядок выполнения работы:

Очень часто при написании скриптов требуется выполнить одно и тоже действие несколько раз. Для этого и нужны циклы. ***Циклы в php***, как и в других языках, делятся на несколько типов:

1. ***Цикл со счетчиком for***
2. ***Цикл с условием while, do-while***

3. ***Цикл для обхода массивов foreach***

Цикл For в PHP

Цикл со счетчиком for - выполняется определенное количество раз. Рассмотрим пример:

```
<?php
for ($i = 0; $i <= 10; $i++) {
echo "Циклы в PHP - написано: $i раз<br/>";
}
?>
```

В этом примере цикл будет выполняться 11 раз. От 0 (т.к. переменная $i = 0) до 10 (т.к. $i <= 10). Каждую итерацию $i будет увеличено на 1 ($i++). Чтобы было понятней, сделаем еще один пример:

```
<?php
for ($i = 5; $i < 10; $i++) {
echo "Номер итерации: $i<br/>";
}
?>
```

Цикл будет выполняться от 5 и до 9 ($i < 10 (в предыдущем примере было <= 10)).

Так же цикл можно выполнять в обратном порядке:

```
<?php
for ($i = 10; $i > 5; $i--) {
echo "Номер итерации: $i<br/>";
}
?>
```

Цикл будет выполнен от 10 и до 5.

Так же цикл можно выполнять с определенным шагом, рассмотрим пример:

```
<?php
for ($i = 0; $i <= 10; $i = $i + 5) {
echo "Номер итерации: $i<br/>";
}
?>
```

В цикле будет выполнено 3 итерации (0, 5, 10) с шагом 5. Каждую итерацию, счетчик цикла будет увеличен на 5.

Цикл foreach в PHP

Цикл foreach - самый распространенный цикл. Требуется почти во всех скриптах, особенно если php скрипт работает с базами данных. Используется для обхода массивов.

Например, рассмотрим небольшой пример:

```
<?php
$arr[0] = "red";
$arr[1] = "blue";
$arr[2] = "green";
$arr["color"] = "yellow";
$arr["test"] = "Изучение PHP";
foreach ($arr as $key => $value) {
echo "Ключ: $key, Значение: $value<br/>";
}
?>
```

При запуске скрипта вы увидите:

Ключ: 0, Значение: red
Ключ: 1, Значение: blue
Ключ: 2, Значение: green
Ключ: color, Значение: yellow
Ключ: test, Значение: Изучение PHP

Цикл while в PHP

Цикл while используется для выполнения цикла до тех пор, пока выполняется условие. Если условие ни когда не будет выполнено, то цикл зациклится.

Рассмотрим пример:

```
<?php
$i = 0;
while ($i < 20) {
$i = $i + 1;
echo $i . "<br/>";
}
?>
```

На экране увидим числа от 1 до 19

Цикл do-while в PHP:

Цикл ***do-while*** - работает точно так же как и ***цикл while***, единственное отличие что условие выполняется после итерации. Напишите пример:

```
<?php

$i = 0;
do {
$i = $i + 1;
echo "$i<br/>";
}
while ($i < 20);
?>
```

На экране увидите числа от 1 до 20. Обратите внимание, что в предыдущем примере с ***циклом while*** было от 1 до 19, т.к. условие выполнялось до итерации цикла.

Для того чтобы ***прервать цикл*** существует ***функция break***, она позволяет ***выйти из цикла***, не зависимо от того сколько итераций осталось до окончания цикла.

Для того чтобы пропустить итерацию, и перейти к следующей итерации - есть ***функция continue.***

Форма представления результата:

работающий скрипт.

Практическое занятие № 10

Работа с почтовым сервером POP3

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: изучить язык PHP

Выполнив работу, Вы будете:

уметь:

- работать с почтовым сервером;

Материальное обеспечение:

рабочее место, оснащенное ПК;
инструкция для работы.

Порядок выполнения работы:

Формы обозначаются тегом form:

```
<form method="post">

</form>
```

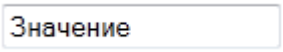
Атрибут method указывает каким методом будет производиться отправка формы. ***Существуют 2 метода – GET и POST.***

Текстовое поле - Input text

Код:

```
<input type="text" value="Значение" name="name" />
```

Текстовое поле, в которое можно вводить информацию. ***Атрибут name*** – имя текстового поля, ***value*** – значение. Выглядят вот так:

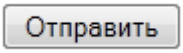


Кнопка - Input Submit

Код:

```
<input type="submit" value="Отправить" name="send"/>
```

Кнопка, используется для отправки форм. Выглядит вот так:



Текстовой блок - Textarea

Код:

```
<textarea rows="5" cols="40" name="message">Работа с формами- элемент textarea</textarea>
```

Текстовой блок, обычно используется для написания сообщений большого размера. Атрибут rows, cols – длина и ширина.

Метод GET и POST при отправке форм

Методы GET и POST указываются в атрибуте method формы при ее инициализации:

```
<form method="post">

</form>
```

Метод GET передает информацию в адресе страницы, ***метод POST*** – в заголовках. Чтобы лучше понять что такое ***метод GET и POST*** напишем скрипт, который будет отправлять наше имя и сообщение на сервер, а сервер выводить сообщение на экран.

В PHP данные ***полученные методом GET*** хранятся в переменной ***$_GET***. Данные ***переданные методом POST*** хранятся в переменной ***$_POST.***

Напишите код для работы с методом GET, потом методом POST.

```
  <html>
    <head>
      <title>Формы - PHP</title>
    </head>
    <body>
      <pre> <!-- Тег <Pre> нужен для удобного вывода на экран -->
<?php
print_r($_GET);
?>
      </pre>

      <form method="get">
        Вашем имя: <input type="text" value="Иван Иванов" name="name" /><br/>
        Сообщение: <br/>
        <textarea rows="5" cols="40" name="message">Метод GET. PHP – используется для
        создания динамических страниц сайта.
```

```
</textarea><br/>
            <input type="submit" value="Отправить" name="send"/>
          </form>
        </body>
    </html>
```

Теперь переделайте скрипт, для отправки данных методом POST.

```
    <html>
      <head>
        <title> Формы - PHP </title>
      </head>
      <body>
        <pre> <!-- Тег <Pre> нужен для удобного вывода на экран -->
<?php
print_r($_POST);
?>
        </pre>

        <form method="post">
          Вашем имя: <input type="text" value=" Иван Иванов " name="name" /><br/>
           Сообщение: <br/>
           <textarea rows="5" cols="40" name="message">Метод POST. PHP – используется для
           создания динамических страниц сайта.
</textarea><br/>
            <input type="submit" value="Отправить" name="send"/>
          </form>
        </body>
    </html>
```

Очень часто требуется обрабатывать введенные пользователем данные в текстовые поля. Для этого и существуют ***строковые функции в PHP***. С их помощью можно обрабатывать текстовую информацию так, как нужно.

Вот список основных, и самых используемых строковых функций в php:

1. ***srelen ("текст")*** — считает количество символов в строке.

```
<?php
echo strlen("Изучение PHP"); //На экране появится 12
?>
```

2. ***str_replace("что заменять", "на что заменять", "текст");*** – функция нужна для замены подстроки в строке. Например, нужно заменить слово *настройка* на «ля-ля-ля», в предложении: *«Установка и настройка web-сервера в локальной сети»*

```
<?php
echo str_replace("настройка ", "ля-ля-ля", "Установка и настройка web-сервера в локальной
сети");
?>
```

На экране увидим: *«Установка и ля-ля-ля web-сервера в локальной сети»*

3. ***trim ("текст", "символы")*** — удаляет символы по краям. Например, нам нужно удалить по-краям пробелы и запятые в строке - *« , , , Установка и настройка web-сервера, , ", ", »*. Для этого напишем код:

```
<?php
echo trim(" , , , Установка и настройка web-сервера, , ", ", ");
?>
```

Первый параметр строка, второй — символы, которые нужно удалить по краям (если не указывать второй параметр, то обрезаться будут только пробелы). На экране увидите *«Уста-*

новка и настройка web-сервера», без пробелов и запятых. Есть аналогичные ***функции ltrim*** — удаляет символы слева, и ***rtrim*** — справа. ***Функция очень часто используется*** при создании интернет-магазинов, например при авторизации, чтобы пользователь при копировании пароля из блокнота ***случайно не вставил лишние пробелы*** или другие знаки.

4. ***substr("Строка", "Начальная позиция", "Конечная позиция");*** - возвращает часть строки. Например, нужно в строке *" Установка и настройка web-сервера в локальной сети "* обрезать все лишнее, и оставить только *«web-сервера»*, для этого напишем код:

```
<?php
echo substr("Установка и настройка web-сервера в локальной сети ", 23, 33);
?>
```

Вырезаем подстроку, начинаю с 23-ого символа, и заканчивая 33. На экране увидим *«web-сервера»*. Если последний параметр (33) не указывать, то текст будет вырезан с 14-ой позиции до конца строки.

5. ***strpos("Строка", "подстрока", позиция начального символа);*** — возвращает позицию найденной подстроки в строке. Например, нам нужно узнать позицию слова *" web-сервера "* в строке *" Установка и настройка web-сервера в локальной сети "*, пишем код:

```
<?php
echo strpos("Установка и настройка web-сервера в локальной сети.",
 "web-сервера");
?>
```

На экране увидим 23, т.к. с 23-ого символа начинается первое вхождение слова *web-сервера*. Если указать 3-ий параметр то поиск вхождения будет с этой позиции. Так же есть ***функция strrpos***, она ищет справа налево. Если в этом примере указать вместо strpos – strrpos, то на экране увидим 29, т.к. справа первое вхождение начинается с 29-ого символа.

Теперь напишете пример, для закрепления материала. В скрипте ***будете обрабатывать данные из текстового поля*** «Адрес сайта:». Адрес сайта можно ввести так: «http://primer.ru/», можно так «www. primer.ru», можно так: «primer.ru/», и т.д. Вариантов достаточно много. А для нашего интернет-магазина обязательное условие, это хранение всех данных о пользователе в едином формате («primer.ru», без http, www, пробелов и слэшей по краям!).

```
<?php
$string = " http://www. primer.ru/ ";

$string = trim($string, "/ ");
if (strpos($string, "http://") !== false) {
$string = substr($string, strpos($string, "http://") + strlen("http://"));
}
if (strpos($string, "www.") !== false) {
$string = substr($string, strpos($string, "www.") + strlen("www."));
}
echo "\"$string\"";

?>
```

Конечно это не самый удачный вариант, использовать такое количество строковых функций для обрезания лишнего, можно воспользоваться той же ***функцией str_replace***:

```
<?php
echo str_replace("www.", "", str_replace("http://", "", trim(" http://www. primer.ru/ ", "/ ")));
?>
```

Форма представления результата:
работающий скрипт.

Тема 3.7. Сетевые ОС и файл системы
Практическое занятие № 11

Сетевые ОС семейства Windows

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомление с базовыми понятиями локальных вычислительных сетей. Ознакомление с сетевыми утилитами, предназначенных для ОС семейства Windows.

Выполнив работу, Вы будете:

уметь:

- пользоваться сетевыми ОС семейства Windows;

Материальное обеспечение:

рабочее место, оснащенное ПК;
инструкция для работы.

Порядок выполнения работы:

Запуск командной строки

Нажать комбинацию клавиш [win]+[r] В поле «открыть» написать cmd. Подтвердить запуск программы [enter]

Cmd.exe – интерпретатор командной строки (англ. command line interpreter) для операционных систем семейства Microsoft Windows NT

Получение справки по командам – запуск команды с ключом /? Пример ping /?

Просмотр конфигурации с помощью команды ipconfig /all

Устраняя неполадки сетевых соединений TCP/IP, начинайте с проверки конфигурации TCP/IP на компьютере, на котором возникают эти неполадки. Для получения сведений о конфигурации компьютера, включая его IP-адрес, маску подсети и основной шлюз, можно использовать программу ipconfig.

Примечание

Когда команда ipconfig выполняется с параметром /all, она выдает подробный отчет о конфигурации всех интерфейсов, включая все настроенные последовательные порты. Результаты выполнения команды ipconfig /all можно перенаправить в файл и вставить в другие документы. Можно также использовать эти результаты для проверки конфигурации TCP/IP на всех компьютерах сети и для выявления причин неполадок TCP/IP-сети.

Например, если компьютер имеет IP-адрес, который уже присвоен другому компьютеру, то маска подсети будет иметь значение 0.0.0.0.

Обновление конфигурации с помощью команды ipconfig /renew

Устраняя неполадки сетевых соединений TCP/IP, начинайте с проверки конфигурации TCP/IP на компьютере, на котором возникли эти неполадки. Если компьютер настроен на использование DHCP и получает конфигурацию от DHCP-сервера, можно инициировать обновление аренды, выполнив команду

ipconfig /renew

Когда выполняется команда ipconfig /renew, все сетевые адаптеры компьютера, на котором используется DHCP (за исключением тех, которые настроены вручную), пытаются связаться с DHCP-сервером и обновить имеющиеся или получить новые конфигурации.

Можно также выполнить команду ipconfig с параметром **/release**, чтобы немедленно освободить текущую конфигурацию DHCP для узла.

Устранение неполадок аппаратных адресов с помощью программы arp

Протокол ARP (Address Resolution Protocol) позволяет узлам определять аппаратные адреса сетевых интерфейсов других узлов, расположенных в той же физической сети, по IP-адресам этих узлов. Для более эффективного использования ARP каждый компьютер кэширует сопоставления IP-адресов с аппаратными адресами, устраняя тем самым повторяющиеся широковещательные запросы ARP.

Для просмотра и изменения таблицы ARP на локальном компьютере можно использовать команду arp. Команда arp служит для просмотра кэша ARP

и устранения неполадок с разрешением адресов. Чтобы просмотреть кэш протокола ARP

1.Откройте окно «Командная строка».

2.В командной строке выполните команду **arp –a.**

Использование сетевых утилит, входящих в состав операционных систем семейства Microsoft Windows NT 5.X/6.0.

1.Открыть интерпретатор командной строки

2.Просмотреть конфигурации TCP/IP с помощью команды **ipconfig**

3.Просмотреть полную информацию о конфигурации TCP/IP с помощью команды **ipconfig /all**

4.Вынести полную информацию о конфигурации TCP/IP в файл при помощи команды

ipconfig /all > c:\ipconfig.txt

5.при помощи команды **arp –a** вывести информацию об аппаратных адресах сетевых интерфейсов

6.Вывести информацию об аппаратных адресах сетевых интерфейсов в

файл:

[win]+[r] cmd

arp –a

Щелкнуть правой клавишей мыши по окну cmd.exe, в контекстном меню выбрать пункт “Пометить”

Выделить вывод команды arp –a

Нажать [enter] [win] + [r]

notepad c:\arplist.txt

[ctrl] + [v]

Сохранить файл как **c:\arplist.txt**

7.при помощи утилиты nslookup узнать ip адреса сайта ya.ru **nslookup ya.ru**

8.полученные результат вынести в файл nslookupyaru.txt в корне

диска C:

nslookup ya.ru > c:\nslookupyaru.txt

9.Используя команду **ping**, следует выполнить перечисленные ниже действия.

1. Используйте адрес замыкания на себя, чтобы проверить правильность настройки TCP/IP на локальном компьютере. **ping** 127.0.0.1

2. Обратитесь по IP-адресу локального компьютера, чтобы убедиться в том, что он был правильно добавлен к сети. **ping** *IP_адрес_локального_узла*

3.Обратитесь по IP-адресу основного шлюза, чтобы проверить работоспособность основного шлюза и возможность связи с локальным узлом локальной сети.

ping IP_адрес_основного_шлюза

4. Обратитесь по IP-адресу удаленного узла, чтобы проверить возможность связи через маршрутизатор. **ping** *IP_адрес_удаленного_узла*

Использование сторонних сетевых приложений

Скачать с сайта **http://www.isc.org/software/bind** файл последней версии программы BIND.

Извлечь из скачанного архива следующие файлы: **dig.exe nslookup.exe liblwres.dll libbind9.dll libisccfg.dll libdns.dll libisc.dll** в **c:\bind**

1.Открыть интерпретатор командной строки

2.Ввести команду

c:&& cd c:\bind

3.Для того, чтобы просто получить IP-адрес по имени хоста достаточно выполнить

dig ya.ru

после выполнения утилита **dig** выводит секции DNS сообщения и его заголовок. Вывод содержит значение “Query time” - время запроса dns сервера

4.Вывод данных, полученных от заданного dns сервера **dig ya.ru @208.67.222.222**

где 208.67.222.222 – адрес dns сервера opendns (http://www.opendns.com)

5.Сравнить значения Query time

при помощи dig определить версию удаленного dns сервера, выполнив команду

dig -t txt -c chaos VERSION.BIND @208.67.222.222

команда

dig -t txt -c chaos VERSION.BIND

выведет ответ, содержащий версию dns сервера провайдера.

6.По результатам проверки сделать вывод, настроены ли проверенные dns серверы на сокрытие версии программного обеспечения

7.При помощи утилиты nslookup, входящей в комплект поставки ПО bind, узнать ip адреса сайта ya.ru

nslookup ya.ru

Сравнить выводы nslookup, входящей в комплект поставки ПО bind, и nslookup, входящий в состав ОС семейства Microsoft Windows NT.

Форма представления результата:

Выполненная работа, отчет

Практическое занятие № 12

Прочие сетевые ОС

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомление с базовыми понятиями локальных вычислительных сетей. Ознакомление с сетевыми утилитами, предназначенных для ОС семейства Windows.

Выполнив работу, Вы будете:

уметь:

- пользоваться сетевыми ОС;

Материальное обеспечение:

рабочее место, оснащенное ПК;

инструкция для работы.

Порядок выполнения работы:

Операционная система UNIX - многопользовательская, многозадачная операционная система, способная функционировать на различных аппаратных платформах. В микроядро ОС UNIX встроен модуль, выполняющий протокол управления передачей/межсетевой протокол (протокол TCP/IP).

Операционная система Linux - сетевая операционная система, ядро которой разработано на базе операционной системы Unix. Linux распространяется с открытыми исходными кодами и применяется для создания серверов в вычислительных сетях и в Интернете.

Сетевая операционная система NetWare - разработанная корпорацией Novell сетевая операционная система, которая использует одноранговую архитектуру или архитектуру клиент-сервер.

Первые версии ОС NetWare появились на рынке информационных технологий в 1985 году и, поскольку у нее тогда не было конкурентов, очень быстрыми темпами наращивала сбыт своих сетевых ОС.

Сетевая ОС NetWare имеет асимметричную структуру из основных управляющих программ, которые объединяются в ядро операционной системы и функционируют на файл-сервере. На рабочих станциях загружено программное обеспечение обеспечивающее связь с сервером и называемое оболочкой (shell) или клиентской частью.

NetWare является типичным представителем сетевых ОС, работающих по технологии файл-сервер. Это означает, что все сетевые операции реализуются на базе специально определенноймашины в сети - выделенного сервера. Этот сервер предоставляет всем остальным пользователям доступ к своим ресурсам, через сеть, поддерживая работу сетевых принтеров и обеспечивая защиту данных в сети.

Для файл серверной технологии характерно то, что сам сервер только предоставляет дисковое пространство, а все программы, даже если они запускаются с сервера, всегда работают на клиентских станциях.

Самвыделенный сервер работает в режиме консоли, с которой можно управлять работой сервера и сети. С консоли можно загружать специальные программные модули NLM (NetWare Loadable Module), которые и осуществляют контроль за работой сети, регистрацией пользователей, выделения для пользователей ресурсов, управления печатью и т.п.

ОС NetWare имеет свою собственную уровневую архитектуру коммуникационных протоколов, которая лишь частично совпадает с 7 уровневой моделью ISO.

В NetWare выделяются следующие уровни:

- Open Link Protocol (Interface) протокол, открытый для подключения драйверов сетей различных стандартов.
- Internet Packet Exchange (IPX) - протокол, соответствующий сетевому уровню, но имеющий ряд функций канального уровня.
- Sequenced Packed Exchange (SPX) - протокол транспортного уровня, соответствующий ISO.
- NetWare Core Protocol (NCP) выполняет функции сеансового и представительского уровня модели ISO.
- NetWare Application and Utilities соответствует прикладному уровню OSI.

В состав общей структуры сетевого программного обеспечения NetWare входят:

- ядро ОС, резидентное на файл-сервере.
- утилиты ОС (NLM) работающие на файл-сервере.
- сетевые утилиты, находящиеся на файл-сервере и работающие на клиентских машинах.
- клиентская часть ОС (оболочка рабочих станций), представляющая собой загружаемую задачу DOS на рабочих станциях.
- Программное обеспечение для работы в качестве моста в сети(локального или удаленного доступа).
- Программное обеспечение для работы в качестве шлюза в сети.

Итак, к функциям файл-сервера относятся:

- контроль запросов к сетевым операциям.
- управление доступам к сетевым ресурсам в виде файлов, программ, принтеров и т.д.
- контроль и анализ корректности данных.
- защита информации
- восстановление и сохранение данных (архивирование).
- дополнительные услуги в виде поддержки языков баз данных SQL и т.д.

Правила именования объектов nds.

Дерево каталога состоит из именованных объектов NDS, представляющих организацию сети. Такой именованный объект может принадлежать к одной из следующих четырех категорий:

C - название страны (например, RU);

O - название организации (например, MGIEM);
OU - название подразделения (например, AVT);
CN - общее имя (например, GOSTEV).

Выводится дерево каталога на экран в графическом виде с помощью административной утилиты. Затем его можно просматривать, усекать его части, выполнять поиск и управлять объектами.

В NetWare ссылаться на объекты нужно по их именам в дереве каталога. Имя точно определяет, где в структуре каталога находится объект. По умолчанию используется следующая последовательность имен, отделенная друг от друга точкой:

общее_имя.название_подразделения.название_организации.название_сраны

Если использовать сокращения, имя будет выглядеть так:

CN.OU.O.C

Например: CN=GOSTEV.OU=AVT.O=MGIEM.C=RU

В некоторых случаях можно использовать такой сокращенный вариант:

GOSTEV.AVT.MGIEM.RU

Иногда удобно использовать и неполные имена, например, GIM.

Используемые имена могут быть как типизированные, так и бес типовые. К примеру, имя GOSTEV.AVT.MGIEM бес типовое, а CN=GOSTEV.OU=AVT.O=RU содержит типы.

Форма представления результата:

Отчет

Тема 3.8. Защита и просмотр трафика
Практическое занятие № 13

Настройка прокси-сервера Squid

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: научиться создавать бэкап сайта

Выполнив работу, Вы будете:

уметь:

- выполнять резервное копирование сайта;

Материальное обеспечение:

рабочее место, оснащенное ПК;
инструкция для работы.

Задание:

1 Создать резервное копирование сайта

Порядок выполнения работы:

1 Наименование задания

Введите произвольное название задания резервного копирования, например, "**бэкап сайта по FTP**" и нажмите кнопку "Далее ..."

2 Тип бэкапа

Оцените примерный объем и скорость выкачиваемых данных с сайта по FTP.Если объем исходных данных не слишком велик, можете выбрать Полный тип бэкапа. Если же объем исходных данных исчисляется десятками или сотнями мегабайт, выберите Добавочный тип бэкапа (Incremental).

На 4-м шаге (параметры сжатия ZIP) укажите, нужно ли упаковывать резервную копию сайта в один ZIP-файл или же создавать резервную копию простым копированием файлов. При необходимости можно установить пароль на ZIP-архив и указать метод шифрования.

3 Исходные файлы и папки

Для добавления папок и файлов, необходимых для резервирования, нажмите кнопку "Добавить" -> "Файлы и папки на FTP". Появится окно выбора папок сайта.

В появившемся окне необходимо один раз указать FTP настройки доступа к сайту (кнопка "Указать" или "Изменить"). Появится окно с настройками параметров доступа по FTP.

Введите:

- FTP-сервер: имя или IP адрес сайта (без "http://" и "www");
- порт;
- логин и пароль, если требуется авторизация;
- таймаут и число попыток соединения (рекомендуется оставить по-умолчанию);
- режим "Активный" или "Пассивный";
- параметры Proxy (если необходимо).

После заполнения необходимых настроек нажмите OK - окно закроется, а в выпадающем списке настроек появится указанный вами набор параметров **для доступа к сайту по FTP** в виде одной строки.

Нажмите кнопку "Подключиться" - произойдет подключение к FTP-серверу, и, в случае успеха, отобразится список папок сайта, а в случае проблем с доступом в журнал ниже (лог) напишется текст ошибки. Выберите одну или несколько папок, например папку "www", резервную копию которой хотим создать и нажмите OK. После выбора исходных данных вы можете указать свойства каждой выбранной папки (Exiland Backup позволяет гибко настраивать параметры для каждой исходной папки), например, указать исключающие подпапки, то есть подпапки, которые не следует включать в резервную копию.

4 Расписание резервного копирования

Укажите периодичность выполнения задания, например один раз в сутки ночью в 4:30. Даже если указано периодическое выполнение бэкапа вы всегда можете запустить задание вручную, не дожидаясь указанного времени.

5 Запуск резервного копирования сайта

Задание создано. Нажмите кнопку "Настройки" вверху главного окна программы и убедитесь, что включена опция "Загружать Exiland Backup при старте Windows" и нажмите OK. Нажмите кнопку "Выполнить" и понаблюдайте как будет выполняться резервное копирование файлов (бэкап) сайта, чтобы убедиться, что всё настроено верно. Можете закрыть программу по крестику, но главное, не выгружайте, а сверните программу в область уведомлений (System Tray).

Во время резервного копирования по каждому заданию ведется подробный и понятный лог (журнал), информирующий об этапах и результатах выполнения задания. Также в области уведомлений (System Tray) отображается панель индикации процесса, способ отображение которой можно настроить в общих настройках программы.

Также, не маловажно отметить, что программа Exiland Backup является устойчивой к разрыву соединения при передачи данных по протоколу FTP. В случае потери соединения программа попытается восстановить соединение с FTP сервером и докачать файл, затем продолжить выполнение задания. Количество попыток соединения и таймаут, как уже было сказано ранее, указываются в настройках доступа к FTP-серверу.

Форма представления результата:
созданный файл бэкап.

Практическое занятие № 14

Работа с программой WireShark

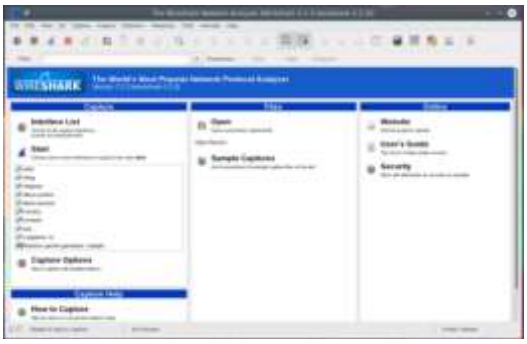
Формируемые компетенции:
ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.
ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомиться с возможностями программы WireShark, выполнит анализ сетевого трафика

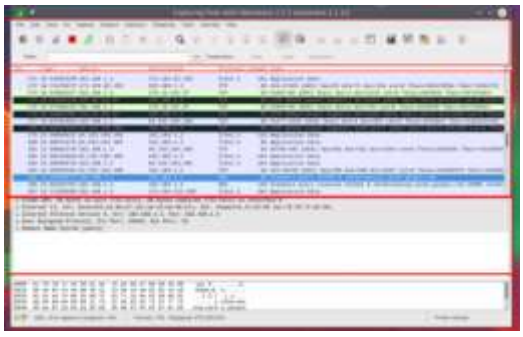
Выполнив работу, Вы будете:
уметь:
- пользоваться программой WireShark для анализа сетевого трафика;

Материальное обеспечение:
рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:
Главное окно программы WireShark разделено на три части: первая колонка содержит список доступных для анализа сетевых интерфейсов, вторая - опции для открытия файлов, а третья - помощь.



Для начала анализа сетевого трафика выберите сетевой интерфейс, например eth0, и нажмите кнопку Start.

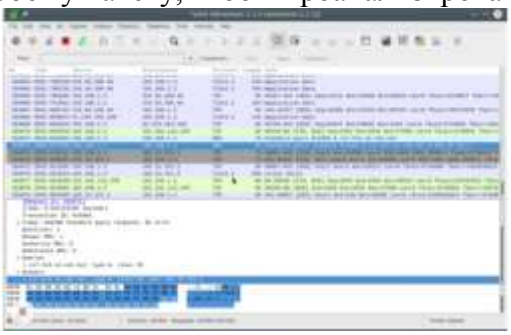


После этого откроется следующее окно, уже с потоком пакетов, которые проходят через интерфейс. Это окно тоже разделено на несколько частей:

- **Верхняя часть** - это меню и панели с различными кнопками;
- **Список пакетов** - дальше отображается поток сетевых пакетов, которые вы будете анализировать;
- **Содержимое пакета** - чуть ниже расположено содержимое выбранного пакета, оно разбито по категориям в зависимости от транспортного уровня;

- **Реальное представление** - в самом низу отображается содержимое пакета в реальном виде, а также в виде HEX.

Вы можете кликнуть по любому пакету, чтобы проанализировать его содержимое:



Здесь мы видим пакет запроса к DNS, чтобы получить IP-адрес сайта, в самом запросе отправляется домен, а в пакете ответа мы получаем наш вопрос, а также ответ.

Для более удобного просмотра можно открыть пакет в новом окне, выполнив двойной клик по записи:

ФИЛЬТРЫ WIRESHARK

Перебирать пакеты вручную, чтобы найти нужные, очень неудобно, особенно при активном потоке. Поэтому для такой задачи лучше использовать фильтры. Для ввода фильтров под меню есть специальная строка. Вы можете нажать **Expression**, чтобы открыть конструктор фильтров, но там их очень много, поэтому мы рассмотрим самые основные:

- **ip.dst** - целевой IP-адрес;
- **ip.src** - IP-адрес отправителя;
- **ip.addr** - IP отправителя или получателя;
- **ip.proto** - протокол;
- **tcp.dstport** - порт назначения;
- **tcp.srcport** - порт отправителя;
- **ip.ttl** - фильтр по ttl, определяет сетевое расстояние;
- **http.request_uri** - запрашиваемый адрес сайта.

Для указания отношения между полем и значением в фильтре можно использовать такие операторы:

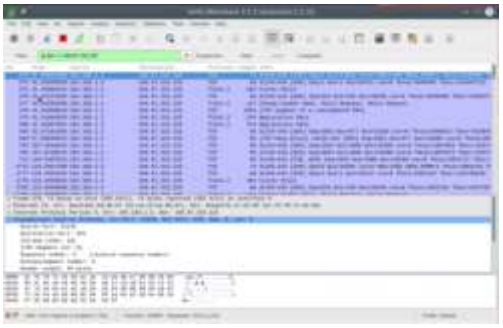
- **==** - равно;
- **!=** - не равно;
- **<** - меньше;
- **>** - больше;
- **<=** - меньше или равно;
- **>=** - больше или равно;
- **matches** - регулярное выражение;
- **contains** - содержит.

Для объединения нескольких выражений можно применять:

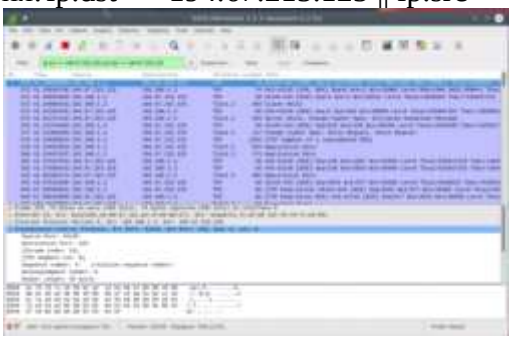
- **&&** - оба выражения должны быть верными для пакета;
- **||** - может быть верным одно из выражений.

Теперь рассмотрим подробнее на примерах несколько фильтров и попытаемся понять все знаки отношений.

Сначала отфильтруем все пакеты, отправленные на 194.67.215.125 (losst.ru). Наберите строку в поле фильтра ip.dst == 194.67.215.125 и нажмите **Apply**. Для удобства фильтры Wireshark можно сохранять с помощью кнопки **Save**:



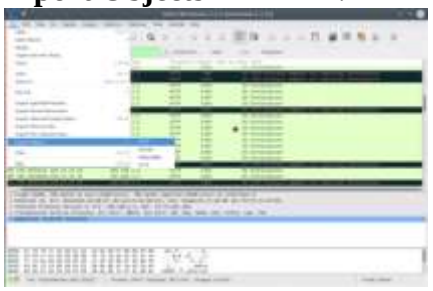
А чтобы получить не только отправленные пакеты, но и полученные в ответ от этого узла, можно объединить два условия: ip.dst == 194.67.215.125 || ip.src == 194.67.215.125



АНАЛИЗ ТРАФИКА WIRESHARK

Вы можете очень просто понять, что именно скачивали пользователи и какие файлы они смотрели, если соединение не было зашифровано. Программа очень хорошо справляется с извлечением контента.

Для этого сначала нужно остановить захват трафика с помощью красного квадрата на панели. Затем откройте меню **File** -> **Export Objects** -> **HTTP**:



Далее в открывшемся окне вы увидите все доступные перехваченные объекты. Вам достаточно экспортировать их в файловую систему. Вы можете сохранять как картинки, так и музыку.

Дальше вы можете выполнить анализ сетевого трафика Wireshark или сразу открыть полученный файл другой программой, например плеером.

Форма представления результата:

Выполненная работа, отчет

Тема 3.9. Сетевые сервисы и программы для установки соединений
Практическое занятие № 15

Программа Putty

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомиться с возможностями программы Putty

Выполнив работу, Вы будете:

уметь:

- пользоваться программой Putty для передачи команд;

Материальное обеспечение:

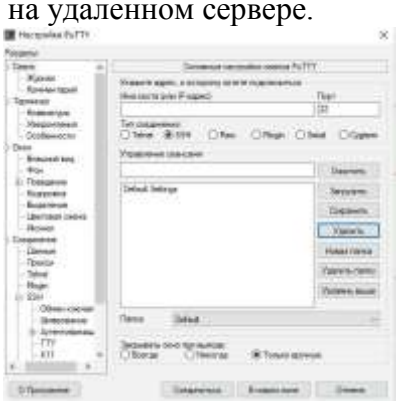
рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

Программа Putty является одним из самых популярных инструментов под OS Windows. Основное назначение – передача команд подключенным устройствам (хостам) по протоколу SSH, Telnet и Rlogin, а также настройка устройств с помощью COM-портов. Доступны версии для смартфонов, а само приложение имеет открытый код, который можно дорабатывать под свои нужды и требования.

При первом запуске приложения нужно внести некоторые настройки для корректной работы:

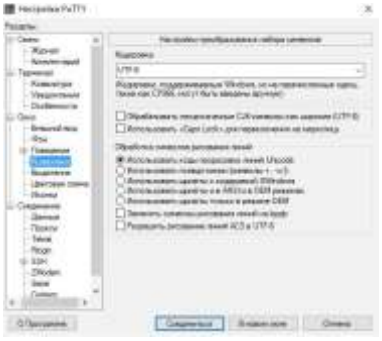
1. Запустить программу.
2. Заполнить поле «Имя хоста». Нажать «Соединиться». Стоит указать соответствующий порт, который не блокирован на удаленном сервере.



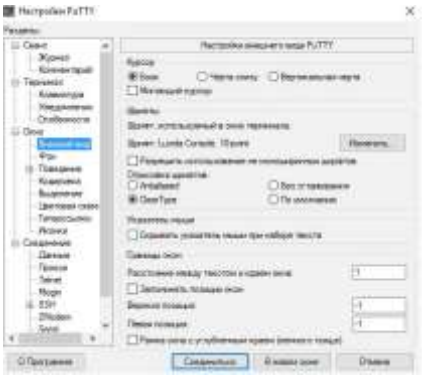
3. При правильном вводе данных хоста будет предложено указать информацию об учетной записи (имя пользователя и пароль).

4. После этого авторизированному пользователю будет предоставлена возможность ввода команд, разрешенных удаленным сервером.

5. В случае надобности, можно провести настройку кодировки. Для этого нужно перейти в главное меню, пункт «Окно», подпункт «Кодировка». При неправильно установленной кодировке будут отображены непечатные символы.



6. В этом же меню «Окно» можно выбрать начертание шрифта. Для этого нужно выбрать подпункт «Внешний вид».



В отличие от своих аналогов, Putty имеет больше возможностей, и несмотря на свой сложный интерфейс – настройки, выставленные по умолчанию, дают возможность подключиться к удаленному серверу даже неопытному пользователю ПК.

Форма представления результата:

Выполненная работа, отчет

Практическое занятие № 16

Серверы терминалов

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомиться с возможностями терминальных серверов

Выполнив работу, Вы будете:

уметь:

- настраивать терминальный сервер;

Материальное обеспечение:

рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

Терминальный сервер, сервер терминалов - сервер, предоставляющий клиентам вычислительные ресурсы (процессорное время, память, дисковое пространство) для решения задач. Технически терминальный сервер представляет собой очень мощный компьютер (либо кластер), соединенный по сети с терминальными клиентами — которые, как правило, представля-

ют собой маломощные или устаревшие рабочие станции, либо специализированные решения для доступа к терминальному серверу. Терминальный сервер служит для удалённого обслуживания пользователя с предоставлением рабочего стола.

Терминальный клиент после установления связи с терминальным сервером пересылает на последний вводимые данные (нажатия клавиш, перемещения мыши) и, возможно, предоставляет доступ к локальным ресурсам (например, принтер, дисковые ресурсы, устройство чтения смарт-карт, локальные порты (COM/LPT)). Терминальный сервер предоставляет среду для работы (терминальная сессия), в которой исполняются приложения пользователя. Результат работы сервера передается на клиента, как правило, это изображение для монитора и звук (при его наличии).

Преимущества терминального сервера

- Снижение временных расходов на администрирование
- Повышение безопасности — снижение риска инсайдерских взломов
- Снижение затрат на программное и аппаратное обеспечения
- Снижение расхода электроэнергии

Недостатки

- Концентрация всей функциональности в рамках одного (нескольких) серверов — выход из строя любого элемента между приложением и клиентами (сервер, коммутаторы, СКС) приводит к простою многих пользователей.
- Усиливаются негативные последствия ошибок конфигурации и работы ПО (последствия ошибок сказываются не на отдельных пользователях, а на всех пользователях сервера сразу же)
- Проблемы с лицензированием (некоторое ПО не предусматривает ситуации работы нескольких пользователей на одном компьютере или требует использования более дорогих версий).

В условиях использования свободнораспространяемого ПО (такого, как X Window System) проблема лицензирования не возникает. Для ПО, предусматривающего в лицензионном соглашении ограничение на количество копий/пользователей, возникают затруднения.

В условиях терминального сервера могут использоваться следующие модели лицензирования:

- Per seat (per device — на рабочее место) — для каждого устройства (тонкого клиента или рабочей станции) требуется отдельная лицензия, вне зависимости от количества пользователей. Подобная схема используется при лицензировании Terminal Services в составе Windows Server.
- Per user (на пользователя) — для каждого пользователя (вне зависимости от числа одновременно работающих пользователей) требуется отдельная лицензия.
- Per connection (конкурентная лицензия) — для каждого соединения требуется отдельная лицензия, при этом количество пользователей/рабочих мест не играет роли — важно количество одновременно обслуживаемых пользователей. Такую систему лицензирования использует Citrix Metaframe. В этом случае существует *пул лицензий*, каждое новое соединение забирает одну лицензию из пула. Лицензия возвращается в пул при окончании соединения.

Во многих крупных пакетах ПО предусматривается особый сервис — сервер лицензий (приложение, занимающееся учётом, выдачей и приёмом лицензий). В условиях крупных сетей рекомендуется выделение под сервер лицензий отдельного компьютера (или нескольких — для резервирования).

Виды терминальных серверов

- Microsoft Windows Terminal Server (поставляется в Microsoft Windows Server)
- Citrix Metaframe
- X Window System
- NX NoMachine
- FreeNX
- RX@Etersoft

Форма представления результата:
Отчет

Практическое занятие № 17

Файловые серверы

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомиться с возможностями файловых серверов, настроить файловый сервер

Выполнив работу, Вы будете:

уметь:

- настраивать файловый сервер;

Материальное обеспечение:

рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

Файловый сервер – это сервер, предназначенный для хранения данных и предоставления к ним общего доступа.

Позволяет организовывать совместную работу над корпоративными документами.

Файловые серверы иногда используются для создания резервных копий важнейших данных.

Файловые серверы настраиваются только для отправки и получения файлов, и не выполняют никаких активных процессов для пользователя. Они также могут быть настроены для распространения данных через Интернет с использованием FTP (передача файлов по протоколу) или HTTP (гипертекстовый протокол передачи).

В качестве сервера, желательно, выбрать профессиональное оборудование. Системные требования для файлового сервера не высокие:

- Процессор может быть самый простой;
- Оперативная память также не сильно используется;
- Дисковая система — самый основной компонент. Ее объем зависит от специфики бизнеса. Примерная формула — не менее 15 Гб на пользователя и не менее 1 Тб на сервер. До 50 пользователей можно рассматривать диски SATA, после — SAS или SSD.

Например, для компании в 300 пользователей подойдет сервер с процессором Xeon E3, 8 Гб ОЗУ и 5 Тб дискового пространства на дисках SAS 10K.

Дополнительные требования

1. Для обеспечения сохранности информации при выходе из строя жесткого диска, необходим RAID-контроллер. Настройка последнего выполняется из специального встроенного программного обеспечения, которое запускается при загрузке сервера;
2. Сервер должен быть подключен к источнику бесперебойного питания;
3. Необходимо предусмотреть резервное копирование. Для этого нужен дисковый накопитель (внешний жесткий диск) или другой сервер.

Установка Windows и настройка системы

Установка системы

На этом шаге все стандартно, за исключением одного нюанса: разбивая во время установки Windows жесткий диск, стараемся выделить небольшую часть (70 - 120 Гб) для системы и все остальное под данные. Если выделить много дискового пространства для системного раз-

дела, увеличится время его обслуживания и фрагментация, что негативно скажется на производительности и надежности системы в целом.

Настройка системы

1. Проверяем правильность настройки времени и часового пояса;
2. Задаем понятное имя для сервера и, при необходимости, вводим его в домен;
3. Если сервер не подключен напрямую к сети Интернет, стоит отключить брандмауэр;
4. Для удаленного администрирования, включаем удаленный рабочий стол;
5. Устанавливаем все обновления системы.

Базовые настройки файлового сервера

Это стандартные действия, которые выполняются при настройке обычного файлового сервера.

Установка роли и вспомогательных компонентов

Как правило, данная роль устанавливается вместе с Windows. Остается только это проверить и доустановить компоненты, которые нужны для полноценной эксплуатации сервиса.

Открываем *Диспетчер серверов*. Он может быть запущен из панели быстрого запуска.

Нажимаем *Управление* - *Добавить роли и компоненты*.

В открывшемся окне оставляем *Установка ролей и компонентов* и нажимаем *Далее*.

В следующем окне выбираем нужный сервер (выбран по умолчанию, если работаем на сервере, а не через удаленную консоль) и нажимаем *Далее*.

Среди ролей находим *Файловые службы и службы хранилища*, раскрываем ее и проверяем, что установлены галочки напротив следующих компонентов:

- Службы хранения;
- Файловый сервер;

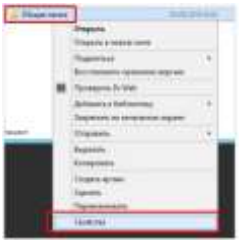
Если данные службы не установлены, выбираем их и нажимаем *Далее*.

В окне *Выбор компонентов* просто нажимаем *Далее*.

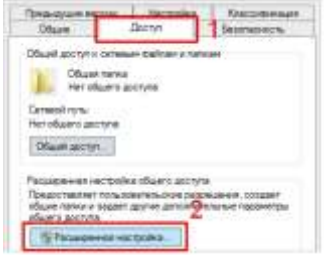
Откроется окно *Подтверждение установки компонентов*. Нажимаем *Установить* и после окончания процесса перезагружаем сервер.

Настройка общей папки

Создаем первую папку, которую хотим предоставить в общее использование. Затем кликаем по ней правой кнопкой мыши и нажимаем *Свойства*:



В открывшемся окне переходим на вкладку *Доступ* и нажимаем *Расширенная настройка*:



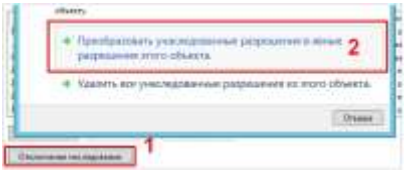
Ставим галочку *Открыть общий доступ к этой папке* и нажимаем кнопку *Разрешения*:

Предоставляем полный доступ всем пользователям:

Нажимаем *OK* и еще раз *OK*.

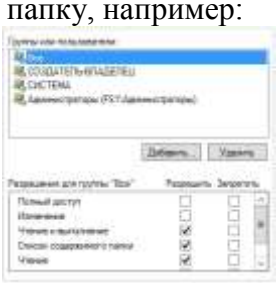
Теперь переходим на вкладку *Безопасность* и нажимаем *Дополнительно*:

В открывшемся окне нажимаем *Отключение наследования* и *Преобразовать унаследованные разрешения в явные разрешения этого объекта*.



Нажимаем *OK* и *Изменить*.

Выставляем необходимые права на папку, например:



Совет: старайтесь управлять правами на ресурсы только при помощи групп. Даже если доступ необходимо предоставить только одному человеку!

Теперь нажимаем *OK* два раза. Папка настроена для общего использования и в нашем примере доступна по сетевому пути *\\fs1\Общая папка*.

Форма представления результата:

Выполненная работа, отчет

Тема 3.10. Создание собственных серверов
Практическое занятие № 18

Языки, позволяющие создавать собственные серверы. Семантика.

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомиться с возможностью создания собственных серверов

Выполнив работу, Вы будете:

уметь:

- создавать сервер;

Материальное обеспечение:
рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

ИИС (intelligent information system) – *это информационная система, которая основана на концепции использования базы знаний для генерации алгоритмов решения задач различных классов в зависимости от конкретных информационных потребностей пользователей.*

Особенности и признаки интеллектуальности ИС

Любая информационная система (ИС) выполняет следующие функции:

- воспринимает вводимые пользователем информационные запросы и необходимые исходные данные;
- обрабатывает введенные и хранимые в системе данные в соответствии с известным алгоритмом и формирует требуемую выходную информацию.

С точки зрения реализации перечисленных функций ИС можно рассматривать как фабрику, производящую информацию, в которой заказом является информационный запрос, сырьем — исходные данные, продуктом — требуемая информация, а инструментом (оборудованием) — знание, с помощью которого данные преобразуются в информацию.

Коммуникативные способности ИИС характеризуют способ взаимодействия (интерфейса) конечного пользователя с системой.

Интеллектуальными считаются задачи, связанные с разработкой алгоритмов решения ранее нерешенных задач определенного типа.

Интеллект представляет собой универсальный алгоритм, способный разрабатывать алгоритмы решения конкретных задач.

Если в ходе эксплуатации ИС выяснится потребность в модификации одного из двух компонентов программы, то возникнет необходимость ее переписывания. Это объясняется тем, что полным знанием проблемной области обладает только разработчик ИС, а программа служит “недумающим исполнителем” знания разработчика. Этот недостаток устраняются в интеллектуальных информационных системах.

Недостатки ИС и их устранение в ИИС

1. Слабая адаптируемость к информационным потребностям пользователя.
2. Невозможность решать плохо формализуемые задачи.

Перечисленные недостатки устраняются в ИИС, которые имеют следующие характерные признаки:

- развитые коммуникативные способности;
- умение решать сложные, плохо формализуемые задачи (характеризуются наполовину качественным и количественным описанием, а хорошо формализуемые задачи – полностью количественным описанием);
- способность к развитию и самообучению.

Классификация ИИС

I класс: системы с интеллектуальным интерфейсом (коммуникативные способности):

1. Интеллектуальные БД;
2. Естественно-языковой интерфейс;
3. Гипертекстовые системы;
4. Контекстные системы;
5. Когнитивная графика.

II класс: экспертные системы (решение сложных задач):

1. Классифицирующие системы;
2. Доопределяющие системы;
3. Трансформирующие системы;
4. Многоагентные системы.

III класс: самообучающиеся системы (способность к самообучению):

1. Индуктивные системы;
2. Нейронные сети;
3. Системы, основанные на прецедентах;
4. Информационные хранилища.

Семантический Веб

HTML-страница описывает как представить информацию визуально в Веб-браузере и трудно поддаётся смысловому анализу компьютерами. Для неё невозможно автоматизировать даже такие тривиальные задачи, как нахождение людей, проектов, программ в Интернете.

Технология Семантический Веб (Semantic Web) позволяет компьютеру интерпретировать информацию в Вебе наравне с людьми, для чего разработана графовая модель описания ресурсов RDF (Resource Description Framework), которая является спецификацией W3C.

С помощью RDF можно создавать любые утверждения о любых ресурсах.

Графовая модель RDF

Утверждения о ресурсах в модели RDF состоят из троек.

Ресурсы и свойства представляются в виде URI, а литералы в формате Unicode. URI позволяет уникальным образом идентифицировать ресурсы в Вебе, а Unicode решает проблему мультиязычности.

RDF схема – это не XML схема

RDF схема описана в утверждениях RDF. В отличие от XML схемы определяет ресурсы (термины) предметной области, а не ограничивает структуру RDF. За ресурсами RDF схемы в спецификации W3C закреплена семантика. Пример RDF схемы, описанной с помощью RDF

Семантика данных – что это такое?

Под семантикой данных будем понимать возможность формального описания смысла передаваемых данных, делая их независимыми от приложений. Это особенно важно в контексте рассматриваемых нами перспектив развития Интернета – побеждает тот, у кого есть данные. Может быть очень много приложений, сайтов, сервисов, но сами по себе они будут очень мало чего значить. Будут выигрывать те, кто сможет предоставлять свой контент в любом, удобном пользователю контенте.

Какие данные можно использовать независимо от сервисов, в которых они используются сегодня: данные из баз данных, XML-документы, приложения в социальных сетях? Нет, потому что их семантика зашита в логике программы и/или неформально в спецификациях. Только данные снабжённые явной семантикой можно сделать действительно независимыми от приложений!

Зачем нужен RDF? Чем плох XML?

Вложенность тегов XML несет только синтаксис, но не несёт никакой семантики. XML хорош только как формат (синтаксис) для обмена данными, но не как модель описания семантики данных! Это же можно сказать и про другие популярные форматы (JSON, например).

Технология Семантический Веб успешно решает следующие задачи:

- независимость данных от приложений;
- семантическая интеграция данных;
- создание основы для повсеместного использования компьютерных агентов (сервисов);
- Data Mining;
- Экспертные системы;
- Проблемы единой авторизации. Если есть ресурс с несколькими возможными способами авторизации, и учетная запись на сайте, к которой привязываются сторонние аккаунты (VK, FB, Twi, OpenID, Oauth…), то мы можем научиться уникально идентифицировать, что это все один и тот же пользователь и связывать всю имеющуюся о нем информацию.

Задачи и проблемы Семантического Веба:

- индексация и поиск информации;
- разработка и поддержка метаданных;
- разработка и поддержка методов аннотирования;

- представление Web в виде большой, интероперабельной базы данных;
- организация машинной добычи данных;
- обнаружение (discovery) и предоставление веб-ориентированных сервисов;
- исследования в области интеллектуальных программных агентов.

Форма представления результата:
Отчет

Практическое занятие № 19

Языки, позволяющие создавать собственные серверы. Примеры.

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: Ознакомиться с возможностями создания сервера

Выполнив работу, Вы будете:

уметь:

- создавать собственный сервер;

Материальное обеспечение:

рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

Интернет играет в нашей жизни большую роль. Мы берем почту, узнаем свежую информацию, отлавливаем своих знакомых через ICQ… Но что ассоциируется со словом Интернет у большинства людей в первую очередь? Сайты. Многие при слове «Интернет» вспоминают свои любимые сайты, а некоторые (не слишком продвинутые) ставят знак равенства между WWW и Интернетом, хотя в последнем есть много другого интересного: email, IRC, p2p-сети, MUD'ы и так далее. Но World Wide Web играет доминирующую роль.

В основе WWW лежит протокол HyperText Transfer Protocol. Надо сказать, что HTTP может использоваться не только для передачи сайтов, но и для передачи всего чтобы то ни было. С помощью протокола HTTP мы можем скачать, например, недавно вышедший фильм или свежие mp3 :). В p2p-сетях HTTP-протокол применяется именно в этих целях.

В данном пособии мы рассмотрим, как написать простой веб-сервер. Я предполагаю, что вы знакомы с основами программирования winsock и умеете создавать сокеты и коннектиться к чему-нибудь :).

Основы HyperText Transfer Protocol

Идея HTTP довольно проста. Клиент шлет запрос серверу, тот рассматривает его и шлет соответствующий ответ. Ответом может быть запрошенный файл, сообщение о том, что такого файла на сервере нет или что-то еще. Примерная структура запроса следующая:

<метод> <запрашиваемый_ресурс> HTTP/1.1<\n>
<заголовочное_поле>: <значение><\n>
<заголовочное_поле>: <значение><\n>
[..заголовочных полей может быть много..]<\n>
<заголовочное_поле>: <значение><\n>
<\n>

<метод> — вид запроса. Основных два: GET и POST. Друг от друга они отличаются, главным образом, способом передачи дополнительной информации, отсылающейся вместе с запросом. В этом туториале мы рассмотрим только метод GET — функционально он похож на

POST, но несколько проще. О методе POST я расскажу во второй части данного туториала, если, конечно, таковая вообще появится на свет :).

<\n> — это два байта 0Dh, 0Ah, несомненно, хорошо знакомые всем ассемблерщикам :).

Таким образом, с методом мы определились. На данный момент запрос, который мы (в качестве клиента) должны будем послать серверу выглядит так:

GET <запрашиваемый_ресурс> HTTP/1.1<\n>
<заголовочное_поле>: <значение><\n>
<заголовочное_поле>: <значение><\n>
[...]
<заголовочное_поле>: <значение><\n>
<\n>

Теперь нам нужно задать <запрашиваемый_ресурс>. Возьмем типичную ссылка на одном из лучших сайтов по программированию в Рунете WASM.RU: http://www.wasm.ru/article.php?article=1016002

Здесь «http://» указывает на то, что используется протокол HTTP, «www.wasm.ru» говорит о том, что необходимо подсоединиться к сайту www.wasm.ru, а все, что идет после знака вопроса — это дополнительные параметры страницы. Последние используются не всегда и не на всех сайтах.

Предположим, что мы подсоединились к www.wasm.ru и хотим получить article.php с необходимыми параметрами. Очевидно, что раз мы уже подсоединились к данному серверу и знаем, что необходимо использовать протокол HTTP, то пересылать «http://www.wasm.ru» не нужно, а значит, мы пошлем только «/article.php?article=1016002». Теперь наш запрос к серверу выглядит так:

GET /article.php?article=1016002 HTTP/1.1<\n>
<заголовочное_поле>: <значение><\n>
<заголовочное_поле>: <значение><\n>
[...]
<заголовочное_поле>: <значение><\n>
<\n>

Теперь осталось разобраться с заголовочными полями. С их помощью клиент передает дополнительную информацию о себе или характере запроса. Например, довольно часто используемым заголовочным полем является ‘User-Agent’. Опера, скажем, шлет следующее:

User-Agent: Mozilla/4.0 (compatible; MSIE 5.0; Windows 98) Opera 6.02 [en]

Другим еще более важным полем является ‘Host’. В нем задается имя веб-сервера, с которого мы хотим получить документ. «Как же так», — можете сказать вы, — «Ведь мы же уже указывали имя веб-сервера, когда создавали сокет и коннектились к нему!» Да, это так. Когда мы коннектились к серверу, мы вызвали функцию gethostbyname, которой передали имя веб-сервера, а она возвратила нам адрес структуры, содержащей адрес сервера. Дело в том, что одному IP-адресу может соответствовать несколько имен сайтов, поэтому когда веб-сервер получает запрос, он должен знать к какому сайту обращается клиент (один веб-сервер может обслуживать тысячи различных сайтов, которые физически будут расположены на одной машине с одним адресом). Для этого мы пишем в ‘Host’ адрес сайта, к которому обращаемся:

Host: www.wasm.ru

Вот как теперь выглядит наш запрос:

GET /article.php?article=1016002 HTTP/1.1<\n>
User-Agent: ManualSender/1.0 :)<\n>
Host: www.wasm.ru<\n>
<\n>

Надо заметить, что хотя эти и другие заголовочные поля являются очень желательными, но, строго говоря, они не являются обязательными, то есть их может и не быть. Также необходимо обратить ваше внимание на то, что за последним заголовочным файлом следуют _два_ <\n>, а не один. Это важно.

Теперь взглянем на все вышеизложенное с точки зрения веб-сервера. Он ждет, пока к нему не поступит запрос, обрабатывает его и посылает ответ, который выглядит примерно так:

HTTP/1.1 <код_ответа> <сообщение><\n>
<заголовочное_поле>: <значение><\n>
<заголовочное_поле>: <значение><\n>
[..заголовочных полей может быть много..]<\n>
<заголовочное_поле>: <значение><\n>
<\n>
<тело_документа>

Получив запрос, сервер должен выдать клиенту код ответа (это трехзначное число), а также сопутствующее ему сообщение. Например, если запрошенный документ был найден, первая строка ответа будет примерно следующей:

HTTP/1.1 200 Ok

Если коды ответов жестко заданы стандартом (200 означает, что запрос был успешно обработан и будет возвращен соответствующий документ/информация), то <сообщение> зависит только от вашей фантазии (конечно, по смыслу оно должно совпадать с <кодом_ответа>. Например, вместо «HTTP/1.1 200 Ok» мы можем послать «HTTP/1.1 200 You want it, you'll get it» :).

В ответе сервера также могут быть заголовочные поля. Они содержат дополнительные сведения об ответе и данных, идущих вместе с ним. Далее я приведу важнейшие (для нас).

Content-Type — этот заголовок задает тип отдаваемых данных. Главнейшие типы заданы в стандарте, и от того, что вы напишите в данном поле, зависит поведение клиента при приеме данных. Например, если вы посылаете браузеру пользователя html-файл, то необходимо указать тип данных 'text/html', иначе браузер может отобразить файл неправильно, либо вообще не будет его отображать, а предложит пользователю его скачать.

Content-Length — здесь указывается длина данных (не включая сам ответ с заголовочными данными) в байтах.

Исходя из вышесказанного, ответ сервера будет выглядеть примерно так:

HTTP/1.1 200 Ok<\n>
Content-Type: text/html<\n>
Content-Length: <длина_документа><\n>
<\n>
<тело_документа>

Если мы хотим отослать простой html-файл, то можем сократить ответ до следующего:

HTTP/1.1 200 Ok<\n>
Content-Type: text/html<\n>
<\n>
<тело_документа>

В результате получаем следующее. Клиент шлет запрос на получение документа, лежащего, например, в корне сервера:

GET / HTTP/1.1<\n>
User-Agent: ManualSender/1.0 :)<\n>
Host: www.someoneserver.com<\n>
<\n>

Сервер получает этот запрос, обрабатывает и выдает корневую страницу:

HTTP/1.1 200 Ok<\n>
Content-Type: text/html<\n>
<\n>
<html>
<head><title>Добро пожаловать на HTTP-сервер!</title></head>
<body>Вы находитесь на нашем http-сервере</body>
</html>

Форма представления результата:

Выполненная работа, отчет

Тема 3.11. Создание собственных клиентов
Практическое занятие № 20

Языки, позволяющие создавать собственные клиенты. Семантика.

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: научиться создавать собственные клиенты

Выполнив работу, Вы будете:

уметь:

- создавать собственные клиенты;

Материальное обеспечение:

рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

Языки веб-программирования – это языки, которые в основном предназначены для работы с веб-технологиями. Языки веб-программирования можно условно разделить на две пересекающиеся группы: клиентские и серверные. Как следует из названия, программы на клиентских языках обрабатываются на стороне пользователя, как правило, их выполняет браузер. Это и создает главную проблему клиентских языков – результат выполнения программы (скрипта) зависит от браузера пользователя. То есть, если пользователь запретил выполнять клиентские программы, то они исполняться не будут, как бы ни желал этого программист. Кроме того, может произойти такое, что в разных браузерах или в разных версиях одного и того же браузера один и тот же скрипт будет выполняться по-разному. С другой стороны, если программист возлагает надежды на серверные программы, то он может упростить их работу и снизить нагрузку на сервер за счет программ, исполняемых на стороне клиента, поскольку они не всегда требуют перезагрузку (генерацию) страницы. Самыми распространенными клиентскими языками программирования являются:

– HTML
– CSS
– JavaScript
– VBScript
– ActionScript
– Java
– CoffeeScript

Когда пользователь дает запрос на какую-либо страницу (переходит на неё по ссылке или вводит адрес в адресной строке своего браузера), то вызванная страница сначала обрабатывается на сервере, то есть выполняются все программы, связанные со страницей, и только потом возвращается к посетителю по сети в виде файла. Этот файл может иметь расширения: HTML, PHP, ASP, ASPX, Perl, SSI, XML, DHTML, XHTML.

Работа программ уже полностью зависима от сервера, на котором расположен сайт, и от того, какая версия того или иного языка поддерживается. К серверным языкам программирования можно отнести: PHP, Perl, Python, Ruby, любой .NET язык программирования (технология ASP.NET), Java, Groovy.

Важной стороной работы серверных языков является возможность организации непосредственного взаимодействия с системой управления базами данных (или СУБД) – сервером, на котором упорядоченно хранится информация, которая может быть вызвана в любой момент. Популярными среди систем управления базами данных являются:

– Firebird
– IBM DB2
– IBM DB2 Express-C
– Microsoft SQL Server
– Microsoft SQL Server Express
– mSQL
– MySQL
– Oracle
– PostgreSQL
– SQLite
– Sybase Adaptive Server Enterprise
– ЛИНТЕР
– MongoDB

Язык HTML

За последние годы разработки для Интернета эволюционировали от статических страниц до динамических информационных систем. Некоторое время назад создание современных Web-страниц не требовало практически ничего, кроме совершенного владения языком разметки гипертекста (Hypertext Markup Language, HTML). HTML представляет собой простой язык обработки текстов; на этом языке при помощи набора тегов (tags) создается документ, который можно просматривать специальной программой просмотра Web (browser). Так, HTML-код из листинга 1.1 создает простую Webстраницу.

Листинг 1.1. Исходный код простой Web-страницы.

```
<HTML>
<HEAD><TITLE>My First Web Page</TITLE></HEAD>
<BODY BGCOLOR="WHITE">
<H2><CENTER>Добро пожаловать на мою первую Webстраничку! </CENTER></H2>
</BODY></HTML>
```

HTML – не язык программирования в том смысле, как C++ или Visual Basic; он больше напоминает средства форматирования документов с использованием управляющих последовательностей. Кодирование на HTML часто сравнивают с созданием документа в формате Microsoft Word путем набивки кодов форматирования прямо в Notepad. Очевидно, что функциональность этого крайне мала.

По многим причинам HTML – бедный язык с точки зрения программирования. Во-первых, возьмем гиперссылки (hyperlinks) – эти подчеркнутые и выделенные голубым цветом слова, которые Вы щелкаете, чтобы перейти к другой странице. Гиперссылка – это, по сути, облагороженный оператор перехода GOTO, обеспечивающий переход к жестко указанному месту приложения. Об операторе GOTO и его недостатках написана масса статей. Жестко закодированные ссылки порождают код, который очень трудно сопровождать, и если вы когда-либо писали HTML-код, то знаете, как трудно его повторно использовать и модифицировать.

Во-вторых, HTML не предоставляет никакой реальной возможности сохранять данные в процессе работы приложения. Да и вообще, в этом случае трудно даже говорить о приложении в Web. Когда каждая страница представляет собой лишенную состояния транзакцию с сервером, как вообще можно определить, где приложение начинается и где заканчивается? Сравните это с типичным клиент-серверным приложением, о начале работы которого сигнализирует двойной щелчок значка на рабочем столе, а о конце — выбор пункта Exit в меню File.

В-третьих, у HTML очень ограниченные возможности для 6 взаимодействия. Стандартный HTML довольствуется статическими Web-страницами с текстом, рисунками и ссылками на другие страницы. Подобные узлы называют Желтыми Страницами WWW (World Wide Yellow Pages), так как их формат очень напоминает страницы телефонной книги.

Разумеется, HTML обеспечивает некоторую интерактивность при помощи встроенных элементов управления (intrinsic controls) — тех самых полей ввода, которые обычно присутствуют в HTML-формах. Простые формы можно создать, например, при помощи тегов <INPUT>. Тег <INPUT>. Тег допускает применение текстовых полей (text boxes), флажков (check boxes), переключателей (radio buttons) и кнопок (buttons). Листинг 1.2 определяет HTML-форму, которая содержит текстовые поля ввода для имени, номера телефона и адреса электронной почты.

Листинг 1.2. Код для HTML-формы.

```
<HTML><HEAD><TITLE>Simple HTML
Form</TITLE></HEAD>
<BBDY BGCOLOR="WHITE">
<FORM>
<INPUT TYPE="TEXT" NAME="txtName">Имя<P>
<INPUT TYPE="TEXT" NAME="txtPhone">Телефон<P>
<INPUT TYPE="TEXT" NAME="txtEMail">Адрес
электронной почты<P>
</FORM></BODY></HTML>
```

Формы представляют собой простейшее средство взаимодействия с HTML. Пользователь заполняет ряд форм, которые затем отсылаются серверу. В процессе пересылки данные преобразуются в некий заранее определенный формат и отсылаются в текстовом виде исполняемому файлу сервера. После этого процесс на сервере может использовать полученные данные, например, для доступа к базам данных, посылки почтового сообщения или выполнения иных функций.

HTML представляет собой обычный текст, поэтому первоначально большинство разработчиков писали свои программы непосредственно в текстовых редакторах, таких как Notepad. Со временем ряд фирм предложили графические средства разработки, например, Microsoft FrontPage, дающие возможность создавать Web-страницы, не зная в явном виде HTML. Эти графические редакторы позволяют непосредственно макетировать Web-страницы без трудоемкой возни с тегами. К сожалению, мощь подобных графических редакторов оборачивается и серьезным их недостатком: у разработчиков создается впечатление, что им ни к чему изучать синтаксис и теги HTML, — а между тем трудно придумать что-нибудь более далекое от истины, чем это утверждение. Если вы даже ничего больше не вынесете из этого краткого введения в HTML, то запомните хотя бы это: чтобы быть настоящим Web разработчиком, вы должны знать HTML. Навыки редактирования страницы непосредственно в виде исходного текста позволят вам добиться желаемого эффекта независимо от того, поддерживает ли его ваш любимый графический редактор.

Форма представления результата:

Выполненная работа, отчет

Практическое занятие № 21

Языки, позволяющие создавать собственные клиенты. Примеры

Формируемые компетенции:

ПК 3.2. Проводить системотехническое обслуживание компьютерных систем и комплексов.

ПК 3.3. Принимать участие в отладке и технических испытаниях компьютерных систем и комплексов; инсталляции, конфигурировании программного обеспечения.

Цель работы: научиться создавать собственные клиенты

Выполнив работу, Вы будете:

уметь:

- создавать собственные клиенты;

Материальное обеспечение:

рабочее место, оснащенное ПК и необходимым программным обеспечением;
инструкция для работы.

Порядок выполнения работы:

Компоненты ActiveX По мере совершенствования технологии программ просмотра зависимость от платформы усилилась — она была привнесена компонентами ActiveX, технологией, основанной на COM — модели многокомпонентных объектов Microsoft (Component Object Model). Компоненты ActiveX варьируются от причудливых элементов управления, таких как движки (spinners), до невизуальных компонентов, обеспечивающих доступ к базам данных или электронной почте. Подобные компоненты делают страницы в Internet Explorer более функциональными и привлекательными, но практически бесполезными в среде, не поддерживающей ActiveX, например, в Netscape Navigator.

Компонент ActiveX добавляется в Web-страницу при помощи тега <OBJECT>, однозначно определяющего компонент для программы просмотра. Приведенный ниже код, используя тег <OBJECT>, помещает на Web-страницу элемент управления ActiveX — метку (label).

```
<OBJECT ID="Label1" WIDTH=291 HEIGHT=41
CLASSID="CLSID:978C9E23-D480-11CE-BF2D00AA003F4000"
CODEBASE="http://www.microsoft.com/activex/controls/FM20.
DLL">
<PARAM NAME="ForeColor" VALUE="65408">
<PARAM NAME="VariousPropertyBits"
VALUE="276824091">
<PARAM NAME="Caption" VALUE="Щелкни меня!">
<PARAM NAME="Size" VALUE="7691;1094">
<PARAM NAME="SpecialEffect" VALUE="1">
<PARAM NAME="FontEffects" VALUE="1073741827">
<PARAM NAME="FontHeight" VALUE="480">
<PARAM NAME="FontCharSet" VALUE="204">
<PARAM NAME="ParagraphAlign" VALUE="3">
<PARAM NAME="FontWeight" VALUE="700">
</OBJECT>
```

Значения GUID в любой операционной системе хранятся в реестре — централизованной базе данных, которая отвечает за поддержание информации о программных объектах, используемых приложениями. Когда Internet Explorer обнаруживает тег <OBJECT>, он обращается к реестру и ищет там GUID, совпадающий со значением атрибута CLASSID. Когда такой GUID найден, из реестра выбирается дополнительная информация, позволяющая отыскать файл, который соответствует данному элементу управления ActiveX.

В теге <OBJECT>можно выделить несколько ключевых составных частей, которые определяют, как именно компонент ActiveX будет размещен на странице. Атрибут ID задает имя элемента управления, посредством которого ко всем его свойствам, методам и событиям можно будет получить доступ из текста сценария.

CLASSID представляет собой буквенно-цифровой код, который однозначно идентифицирует данный компонент ActiveX среди всех остальных. Этот код, известный как глобально уникальный идентификатор (Globally Unique Identifier, GUID), не использует больше ни один компонент ActiveX. При помощи GUID Internet Explorer однозначно определяет требуемый компонент и создает его на странице. Если нужный элемент управления ActiveX на клиентской машине отсутствует, Internet Explorer обращается к атрибуту CODEBASE за информацией о том, где находится этот элемент на сервере. Следуя этой информации, файлы данного элемента управления загружаются с сервера, и элемент устанавливается на клиентской машине. Теперь Internet Explorer может свободно работать с ним.

Доступ к компонентам ActiveX посредством тега <OBJECT>не ограничивается элементами управления. Этот тег может активизировать произвольный компонент ActiveX, в том числе и те компоненты, которые можно написать на языках Visual Basic, C++ и Microsoft FoxPro. В сущности, вы легко можете расширить функциональность, доступную клиенту, написав свои собственные компоненты ActiveX и загрузив их в программу просмотра. Следует, правда, помнить, что Internet Explorer по умолчанию не загружает и не выполняет компоненты без цифровой подписи разработчика. Окончательное обеспечение компонента данными происходит через тег <PARAM>, имеющий атрибуты NAME и VALUE, при помощи которых задаются начальные значения свойств данного компонента, когда он впервые создается на Web-странице. После того, как начальные значения установлены, значения свойств легко изменить во время выполнения из текста сценария.

```
<SCRIPT LANGUAGE="VBScript"><!--
Sub Label1_DblClick(Cancel)
Label1.Font.Weight=24
Label1.Caption="Щелкни снова!"
end sub
Sub Label1_Click()
Label1.AutoSize = false
Label1.Font.Weight = 30
Label1.Caption="Еще два раза!!!!!"
Label1.SpecialEffect=1
end sub
-->
</SCRIPT>
```

Visual Basic, начиная с версии 5.0, позволяет, помимо элементов управления ActiveX, создавать документы ActiveX. Документы ActiveX представляют собой программные объекты, которые могут загружаться и работать внутри ActiveXконтейнера, такого как Internet Explorer. Документы ActiveX позволяют разработчикам на Visual Basic немедленно применить свой опыт работы на Visual Basic для создания приложений для Интернета. Что самое существенное, документы ActiveX предоставляют доступ к большей части ключевых возможностей Visual Basic в загружаемом формате.

Язык Java очень быстро приобрел популярность, и его поддерживают как Internet Explorer, так и Netscape Navigator. Апплеты, разработанные на Java при помощи таких средств, как Microsoft J++, во многом напоминают компоненты ActiveX: это самодостаточные, загружаемые фрагменты Web-страницы. Так же, как и у компонентов ActiveX, у апплетов имеется свой особый тег — <APPLET>, который дает программе просмотра указание загрузить код на Java и выполнить его. Нижеследующий код исполняет апплет на Web-странице:

```
<APPLET CODE="DBLBULB.CLASS" HEIGHT=35
WIDTH=26> </APPLET>
```

Атрибут CODE тега <APPLET>идентифицирует исходный код апплета Java практически так же, как атрибут CODEBASE определяет источник для компонента ActiveX. У апплетов могут также быть теги , задающие начальные значения. Во многих случаях апплеты представляют собой функциональные эквиваленты элементов управления ActiveX. Во всяком случае, языки описания сценариев могут обращаться к открытым функциям апплетов точно так же, как они обращаются к методам компонентов ActiveX.

Форма представления результата:

Выполненная работа, отчет