Практическое занятие № 13
Основной синтаксис PHP. Способы разделения инструкций, создания комментариев. Переменные, константы и типы данных, операторы.

Цель работы:

1. Изучение особенностей синтаксиса языка PHP.
2. Получение практических навыков в создании HTML-документов с применением языка PHP.

Оборудование и программное обеспечение:
1. ПК, локальная сеть.
2. ОС MS Windows 7 (10), офисный пакет MS Office, браузер MS Internet Explorer.

План занятия:

1. Освоение теоретической части занятия в соответствии с разделами.
2. Выполнение практической части занятия в соответствии с заданиями.
3. Повторение и закрепление основных изученных понятий и терминов.
4. Оформление отчета.

Синтаксис PHP похож на синтаксис языка C или Perl, он включает в себя операторы, разделенные между собой точкой с запятой (является обязательной), при этом ошибки в PHP по умолчанию выдаются на экран (в отличие от CGI, где все ошибки записываются в лог-файл), найти их не составляет труда, интерпретатор подскажет номер строки, в которой произошла ошибка.

Для программирования на PHP понадобится любой текстовый редактор, но для удобства он должен обеспечивать подсветку синтаксиса и нумерацию строк. Можно применять CuteHTML, входящий в комплект поставки неплохого FTP-менеджера CuteFTP последних версий. ОН удобен, без лишних ненужных функций, сам встраивается в контекстное меню, не требует инсталляции и имеет все необходимое для программирования. Также нужен комплект для работы с PHP - как правило используется Apache+PHP (не является обязательным, т.к. подходит любой сервер, например - IIS Microsoft). Но вариант Apache+PHP бесплатен и имеет большую поддержку документацией (в том числе на русском языке) и форумами.

Первый скрипт. Для того чтобы сервер знал, в каком из файлов есть код PHP, его расширение нужно сделать либо phtml, либо php3, либо php. Строго говоря, может быть назначено любое расширение, но в целях совместимости рекомендуется всегда использовать phtml. Каждая команда в PHP как правило начинается с "<?php" и заканчивается "?>" (здесь и далее без кавычек), а также несколько команд разделяются точкой с запятой. В любом месте скрипта PHP можно поставить комментарий, начинается он с "/*", а заканчивается — "*/" Если комментарий маленький и занимает только одну строчку, можно поставить "//" и таким образом легко закомментировать любую строчку до ее конца. Как обычно, пробелы, символы табуляции и перевод строки просто игнорируются и могут применяться для улучшения читабельности кода PHP.

1. ВЫВОД НА ЭКРАН И ПЕРЕМЕННЫЕ В PHP

PHP очень легко позволяет организовать вывод текста на экран. Рассмотрим пример скрипта:

```
<?php
echo "Привет, мир!";
?>
```

Этот скрипт может быть расположен в любом месте HTML-документа, и сам по себе он не несет ничего полезного, так как выводит на экран только фразу "Привет, мир!". Но таким образом мы знакомимся с одной из наиболее распространенных команд PHP — вывод информации на экран пользователя. Для того чтобы придать нашему скрипту полезные функции, давайте познакомимся с переменными. Переменная характеризуется именем, типом и значением. Имя может быть любым и включать в себя цифры, буквы английского алфавита и разрешенные символы (например, символ подчеркивания или тире). По типу переменные делятся на целые, с плавающей запятой и символьные. Значение в соответствии с типом может быть практически любым. Например, переменная a=5. Это говорит нам о том, что имя у переменной — a, тип — целочисленный, значение — 5. Вот еще примеры имен и значений:

```
<?php
$name = 6;
$h12 = 4.89;
$file_type = "path/index.phtml";
$os = "PHP для всех!";
?>
```

Все переменные в PHP должны начинаться с символа $, что позволяет интерпретатору безошибочно отличать их от команд PHP. В первой строчке нашего скрипта переменной $name присваивается значение 6, и эта переменная автоматически становится целочисленной.

Заранее описывать тип переменной не требуется, как в языках Pascal или Visual Basic, но хотя разделение на типы чисто условное, каждая переменная автоматически стремится использовать правильный тип, соответственно значению. Вторая строчка кода присваивает переменной $h12 значение 4.89, которое является значением с плавающей запятой. Третья и четвертая строчки кода присваивают своим переменным значения, являющиеся символьными строками. Все, что заключено в кавычки, будет интерпретировано как символьная строка. Если переменные не определены ранее, но используются, их значение принимается равным либо нулю, либо пустой строке в зависимости от типа.

Как и в любом языке, над переменными можно совершать любые арифметические действия, достаточно указать переменную для результата, знак равенства и перечислить в естественном порядке переменные или значения с необходимыми арифметическими знаками. Пример:

```
<?php
$a = 5;
$b = 3;
$c = 4;
$d = $a+$b-$c;
echo $d;
?>
```

Результат работы скрипта — вывод на экран цифры 4. Поддерживаются все арифметические операции и функции, многоуровневые скобки, логические операции, операции увеличения или уменьшения на единицу и многое другое. Также выполняется сравнение если — то — иначе. Для этого в PHP применяется конструкция if ( ) { } else { }. Есть различные варианты синтаксиса этого оператора, но этот — основной, и самый логичный из всех. (если) if (условие) (то) { выполняется то, что заключено в кавычки } (иначе) else { выполняется то, что заключено в кавычки }. После кавычек ставить точку с запятой, как обычно между операторами, не обязательно. Но внутри кавычек — разделение операторов между собой проводится только через точку с запятой. Можно вкладывать нескольких операторов проверки один в один. Надо внимательно считать закрывающие кавычки во избежание ошибок кода. Рассмотрим пример:

```
<?php
$a = 5;
$b = 9;
if ( $a == $b ) { echo $b-$a; } else { echo $b.$a; }
?>
```

При сравнении на истину применяется два знака равенства для того, чтобы интерпретатор мог без труда отличить сравнение от присваивания. Результат работы скрипта — 95, т.к $a не равно $b, а команда echo $b.$a; (между переменными стоит точка, а не знак арифметической операции) выводит подряд указанные переменные. Неравенство (ложь) обозначается символами !=, допустимы все остальные арифметические и логические символы и операторы (например, or, and, >, <= и т.д.).

В PHP есть средства быстрого изменения переменной на единицу в сторону увеличения или уменьшения. Для этого нужно указать имя переменной и за ним, без знака равенства, — подряд два плюса или минуса соответственно. Например, $a++; — переменная $a будет увеличена на единицу. Одновременно можно присвоить одно значение нескольким переменным — $a = $b = 4;. Обе переменных будут равны четырем. Вот еще несколько примеров нестандартных арифметических операций в PHP:

```
<?php
$b = $a = 5;       // присваиваем значения переменым $a и $b
$c = $a++;         // последующее увеличение, присваиваем $c
                   // начальное значение $a (5)
$e = $d = ++$b;    // предварительное увеличение, присваиваем $d и $e
                   // увеличенное значение$b (6) тут и $d и$e равны 6
$f = double($d++); // присвоить удвоенное значение $d до его увеличения,
                   // то есть 2*6 = 12, переменной $f
$g = double(++$e); // присвоить удвоенное значение $e после его увеличения,
                   // то есть 2*7 = 14, переменной g
$h = $g += 10;     // сначала увеличить значение $g на 10, что дает в
                   // результате 24, а затем присвоить это значение
                   // переменной $h, что также дает 24
?>
```

2. ВЛОЖЕНИЯ ФАЙЛОВ В PHP

Если создан объемный сайт, маленькое дополнение (например, в меню) в сотни файлов может потребует значительных затрат по времени. PHP решает эту проблему, позволяя вкладывать одну страницу в другую. Достигается это с помощью операторов REQUIRE и INCLUDE. После этих операторов в круглых скобках должен стоять путь к вкладываемому файлу. Например, INCLUDE ("text.phtml"). Различие между указанными операторами заключается в том, что REQUIRE подменяется содержимым указанного файла и может быть использован только один раз, а INCLUDE — вставляет и выполняет содержимое указанного файла, что позволяет применить его несколько раз, например - в цикле. В любом случае, при исполнении файла интерпретатор PHP (правильно говорить — парсер) пребывает в состоянии HTML, и для его включения код надо заключить в конструкцию <?php … ?>. Вложения файлов могут происходить только внутри серверного пространства, доступного PHP. Другими словами, Вы не можете использовать в имени файла http://

Достаточно часто встречаются сайты, ссылки которых включают в себя специальные символы — &, ?, %. Все это может быть и результатом работы PHP. Дело в том, что если в конце ссылки добавить ?имя=значение, это значение будет доступно под этим же именем в файле, куда указывает ссылка. Если необходимо добавить несколько имен, они могут быть разделены знаком &. Теперь мы можем сделать сайт, который будет доступен с помощью только одной страницы. А всю остальную информацию эта страница будет выводить на основании полученных по ссылке данных. Вид такой ссылки будет примерно таким: http://имя.ru/index.phtml?link=1. Единица в конце ссылки и есть наш параметр, который будет подставляться в файле index.phtml. Например, вот так:

```
<html>
…начало файла …
<?php
$url = "";
if ($link == 1) { $url = "name1.phtml"; }
if ($link == 2) { $url = "name2.phtml"; }
if ($link == 3) { $url = "name3.phtml"; }
if ($link == 4) { $url = "name4.phtml"; }
if ($url == "") { $url = "error.phtml"; }
INCLUDE ($url);
?>
… конец файла …
</html>
```

Написанный код учитывает ситуацию, когда посетитель по разным причинам указал неправильный параметр. В этом случае выводится заранее заготовленная страница с сообщением об ошибке. Если же параметр соответствует какому-либо из файлов сайта, он в код файла index.phtml вкладывается и исполняется. Таким образом, начало и конец остаются одинаковыми, а изменяется только середина. И какие-либо изменения уже не кажутся такими страшными, как раньше. Ведь сделать их надо только в одном файле, а отразится это на всем сайте.

Есть и другой путь. Его суть заключается в том, что у PHP есть доступ к так называемым переменным окружения сервера. Одна из этих переменных — запрашиваемый посетителем путь относительно адреса сайта. И этот путь становится нам доступен для использования. В этом случае ссылки у нас будут такого вида: http://имя.ru/index.phtml?patch/name.phtml. Вторая часть ссылки — patch/name.phtml — будет нам доступна, если мы считаем параметр $QUERY_STRING. Например, так: $add = $QUERY_STRING. Теперь изменим наш головной файл index.phtml, чтобы все работало автоматически. А если запрашиваемый параметр не будет указан (правильно говоря — будет равен пустой строке), чтобы что-то открыть, присвоим переменной $add имя файла, который должен быть открыт как главная страница. Пусть это будет файл main.phtml. Тогда код будет выглядеть следующим образом:

```
<html>
…начало файла …
<?php
$add = $QUERY_STRING;
if ($add == "") { $add = "main.phtml"; }
INCLUDE ($url);
?>
… конец файла …
</html>
```

ПРЕДОСТЕРЕЖЕНИЕ: данный метод проще первого, но открывает путь к получению информации о сервере, где расположен сайт с такой организацией структуры. Злоумышленник (или просто любопытный человек) при наличии определенных обстоятельств и знаний сможет много узнать о сервере, а это открывает прямой путь к взлому. Так что будьте осторожны и не станьте причиной больших неприятностей. Защититься от подобных проблем можно, но это уже совсем другая история.

3. ПРИМЕР СЧЕТЧИКА ПОСЕЩЕНИЙ НА PHP

Рассмотрим код скрипта, который позволит организовать на любой из страниц сайта счетчик посещений. Этот счетчик не будет полнофункциональным, так как имеет достаточно много недостатков, но как пример применения PHP вполне годится. В любом месте страницы (но только там, где это нужно) вставьте следующий код:

```
<p>Посетителей страницы —
<?php
$filename = "counter.dat";
$fp = @fopen($filename,"r");
if ($fp) { $counter=fgets($fp,10); fclose($fp); } else { $counter=0; }
$counter++;
echo $counter;
$fp = @fopen($filename,"w");
if ($fp) { $counter=fputs($fp,$counter); fclose($fp); }
?></p>
```

В том же каталоге, что и главная страницы, создайте файл **counter.dat**, закачайте его на сервер и с помощью своего FTP-менеджера измените атрибуты этого файла таким образом, чтобы он был доступен для записи. Обычно нужно установить галочки на всех атрибутах файла. Если Вы этого не сделаете, скрипт будет постоянно выдавать ошибку при попытке записи в файл. Кстати, для того чтобы этого не происходило, стоит поставить перед командой записи и открытия файла символ @, он отменит вывод сообщения о возникшей ошибке на экран посетителя. Когда атрибуты изменены, обновите Вашу страницу на сервере и обратитесь к ней по ее адресу в браузере. Вы увидите, что там, где Вы вставили код PHP, появляется строка: "Посетителей страницы — " и далее число, соответствующее количеству посещений. И никакого следа кода! Он был обработан на сервере в Интернете, а браузеру просто передан результат этого исполнения.

Алгоритм: в первой строке мы присваиваем выбранной переменной имя файла, где будет храниться число посещений; во второй — открывается соединение с этим файлом для чтения. Далее проверяется успешность соединения, и если файл существует и он доступен для чтения, из него считывается строка из 10 байт, этого достаточно для счетчика, и закрывается соединение с файлом. Показание счетчика увеличивается на единицу, его новое значение выводится на экран. На следующем этапе нужно записать новое значение счетчика, и для этого снова открывается соединение (дескриптор) с файлом, но уже на запись с очисткой содержимого файла. Если оно успешно — туда записывается новое значение счетчика и дескриптор файла закрывается.

4. ВЫПОЛНЕНИЕ ПРАКТИЧЕСКОЙ ЧАСТИ

Практическая часть работы включает в себя поэтапное выполнение разделов методического пособия в соответствии с приведенными примерами. **ВАРИАНТЫ ЗАДАНИЙ:**

Разделы	**Вариант 1**	**Вариант 2**	**Вариант 3**
1	Вывод на экран любой фразы	Вывод на экран значений	Выполнение вычислений
2	Вложение стр-ц 1-м способом	Вложение стр-ц 2-м способом	Вложение стр-ц 1-м способом
3	Размещение кода счетчика	Размещение кода счетчика	Размещение кода счетчика

5. ОФОРМЛЕНИЕ ОТЧЕТА

Отчет практической работы должен содержать: наименование и цель работы, порядок выполнения работы, номер варианта задания, исходные данные, код программы, ответы на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите основное назначение и перечислите особенности синтаксиса языка PHP.
2. Перечислите характеристики переменных. Сколько существует типов переменных в PHP?
3. Каково назначение счетчика посещений? В чем особенности кода и размещения счетчика?