PHP

- **PHP**
  - **Что такое PHP?**
  - **Некоторые преимущества PHP.**
  - **Синтаксис и грамматика.**

Что такое PHP?

- PHP — это серверный язык создания сценариев, разработанный специально для Web. В HTML-страницу можно внедрить код PHP, который будет выполняться при каждом ее посещении. Код PHP интерпретируется Web-сервером и генерирует HTML или иной вывод, наблюдаемый посетителем страницы.
- Разработка PHP была начата в 1994 г. и вначале выполнялась одним человеком, Расмусом Лердорфом (Rasmus Lerdorf).

Что такое PHP?

- PHP — это продукт с открытым исходным кодом (Open Source). У пользователя имеется доступ к исходному коду. Его можно использовать, изменять и свободно распространять другим пользователям или организациям.
- PHP означает PHP Hypertext Preprocessor (Препроцессор гипертекста PHP).
- В настоящее время основной версией PHP является четвертая. Адрес начальной страницы для PHP — http://www.php.net

Некоторые преимущества PHP

- К числу конкурентов PHP относятся Perl, ASP, Java Server Pages (JSP) и Allaire Cold Fusion.
- PHP обладает множеством преимуществ по сравнению с этими продуктами:
  - Высокая производительность
  - Наличие интерфейсов ко многим различным системам баз данных
  - Встроенные библиотеки для выполнения многих общих задач, связанных с Web
  - Низкая стоимость
  - Простота изучения и использования
  - Переместимость
  - Доступность исходного кода

Синтаксис и грамматика.

- Синтаксис PHP заимствован непосредственно из C. Java и Perl также повлияли на синтаксис данного языка.
- Есть 4 способа выхода из HTML и перехода в "режим PHP кода":

1. `<? echo("простейший способ, инструкция обработки SGML\n"); ?>`
2. `<?php echo("при работе с XML документами делайте так\n"); ?>`
3. ```
   <script language="php">
   echo ("некоторые редакторы не любят обрабатывающие инструкции");
   </script>
   ```
4. `<% echo("От PHP 3.0.4 можно факультативно применять ASP-тэги"); %>`
   ```

Разделение инструкций. Типы переменных.

- Инструкции разделяются также как в C или Perl - точкой с запятой. Закрывающий тэг (?>) тоже подразумевает конец утверждения, поэтому следующие записи эквивалентны:
  - `<php echo "Это тест"; ?>`
  - `<php echo "Это тест" ?>`
- PHP поддерживает переменные типов:
  - integer - целое
  - double - число с дробной частью
  - string - строковая переменная
  - array - массив
  - object - объектная переменная
  - pdfdoc - PDF-документ (только при наличии поддержки PDF)

Типы переменных.

- Тип переменной обычно не устанавливается программистом; вместо этого, он определяется в зависимости от контекста, в котором она используется. Чтобы указать тип переменной непосредственно, используйте инструкцию cast либо функцию settype(), но учтите, что переменная может вести себя по-разному, в зависимости от того, какой тип определен для нее в данное время.
- **Инициализация переменной.** Для инициализации переменной в PHP вы просто присваиваете ей значение. Для массивов и объектных переменных, однако, может использоваться несколько иной механизм.

Инициализация массивов.

- Массив может инициализироваться одним из двух способов: последовательным присвоением значений, или посредством конструкции array().
- При последовательном добавлении значений в массив вы просто записываете значения элементов массива, используя пустой индекс. Каждое последующее значение будет добавляться в качестве последнего элемента массива.

```
$names[] = "Jill";    // $names[0] = "Jill"
$names[] = "Jack";    // $names[1] = "Jack"
```

- Как в C и Perl, элементы массива нумеруются, начиная с 0, а не с 1.

Инициализация объектов.

- Для инициализации объектной переменной используйте новое предписание для сопоставления данного объекта объектной переменной.

```
class foo {
    function do_foo () { echo "Doing
  foo."; }
}
$bar = new foo;
$bar -> do_foo ();
```

Область переменной.

- Областью переменной является контекст, внутри которого она определена. Любая переменная, определенная внутри функции, по умолчанию ограничена локальной областью функции. Например:

```
$a = 1; /* глобальная область */
Function Test () {
    echo $a; /* ссылка на переменную
  локальной области */ }
Test ();
```

- Этот скрипт не выдаст что-либо на выходе, поскольку инструкция echo относится к локальной версии переменной $a, значение которой присваивается не внутри этой области.

Область переменной.

- Здесь имеется отличие от языка С, в том, что глобальные переменные в С автоматически действуют и внутри функций, если только не переписываются локальными определениями.
- В PHP глобальные переменные должны быть продекларированы глобально внутри функции, если предполагается их использование в данной функции. Например:

```
$a = 1; $b = 2;
Function Sum () {
    global $a, $b;
$b = $a + $b; }
Sum (); echo $b;
```

- Этот скрипт выдаст значение "3".

Глобальные переменные

- Поскольку $a и $b декларируются глобально внутри функции, ссылки на них трактуются как ссылки на их глобальные версии.
- Вторым способом доступа к глобальным переменным является использование определяемого PHP массива $GLOBALS:

```
$a = 1; $b = 2;
Function Sum () { $GLOBALS["b"] =
  $GLOBALS["a"] + $GLOBALS["b"]; }
Sum (); echo $b;
```

- Массив $GLOBALS является ассоциативным, в котором имя глобальной переменной является ключом, а значение этой переменной является значением элемента массива.

Статическая переменная

- Статическая переменная существует только в локальной области функции, но она не теряет своего значения, когда программа, при исполнении, покидает эту область:

```
Function Test () {
        static $a = 0;
        echo $a;
        $a++;
}
```

- При вызове функции Test() она будет выводить значение $a и увеличивать его.

Статическая переменная

- Статические переменные также весьма существенны, когда функции вызываются рекурсивно. Составлять рекурсивную функцию нужно внимательно, т.к. при неправильном написании можно сделать рекурсию неопределенной. Следующая простая функция рекурсивно считает до 10:

```
Function Test () {
    static $count = 0;
    $count++;
    echo $count;
    if ($count < 10) { Test (); }
    $count--;
}
```

Изменяемые переменные.

- Иногда бывает удобно давать переменным изменяемые имена. Такие имена могут изменяться динамически. Изменяемая переменная берет некое значение и обрабатывает его как имя переменной. В примере значение **hello** может быть использовано как имя переменной, посредством применения двух записанных подряд знаков доллара, т.е.:

```
$a = "hello";
$$a = "world";
```

- Две переменных определены и сохранены в символьном дереве PHP: $a с содержимым "hello" и $hello с содержимым "world".

Изменяемые переменные.

- Так, инструкция `echo "$a ${$a}";`
- осуществляет то же самое, что и инструкция:
  `echo "$a $hello";`
- а именно, обе они выводят: hello world.
- Чтобы использовать изменяемые переменные с массивами, решите проблему неоднозначности. Это означает, что если вы пишете $$a[1], то синтаксическому анализатору необходимо знать, имеете ли вы в виду использовать $a[1] как переменную, или вы предполагаете $$a как переменную, а [1] как индекс этой переменной. Синтаксис для разрешения неоднозначности такой: ${$a[1]} для первого случая и ${$a}[1] для второго.

Переменные вне PHP.

- Для передачи данных Web-серверу используется отправка формы. В форме содержатся текстовые поля, переключатели, списки и т.п. Клиент размещает введенные данные в этих полях и пересылает пакет серверу. Процессом передачи формы управляют два атрибута тега <FORM>: METHOD и ACTION. Первый атрибут - METHOD - определяет, каким именно образом данные пересылаются серверу. Атрибут может иметь два значения: POST и GET.

Переменные вне PHP.

- POST диктует программе просмотра, что данные нужно поместить внутрь формы, а GET пересылает данные как составную часть URL целевой страницы. Второй атрибут - ACTION - задает целевую страницу для обработки отправленных данных. Следующий код посылает данные формы сценарию DATA.PHP методом POST:
- **<FORM METHOD="POST" ACTION= "data.php">**
- **<P><INPUT TYPE="TEXT" NAME="Name"></P>**
- **<P><INPUT TYPE="SUBMIT"></P></FORM>**

Переменные вне PHP.

- На сервере данные полей формы можно вновь разобрать. Для этого, в зависимости от метода их отправки, в PHP используются глобальные ассоциативные массивы $HTTP_POST_VARS[] и $HTTP_GET_VARS[], ключами которых являются имена полей форм. Допустима сокращенная форма записи $_POST и $_GET соответственно, а если компилятор PHP собран с ключом **register_globals**, что можно проверить вызовом функции **phpinfo()**, то достаточно использовать переменную, имя которой совпадает с именем поля формы.

Переменные вне PHP.

- Например, следующие три строки кода вернут значение поля Name:
- $a=$HTTP_POST_VARS["Name"];
- $a=$_POST["Name"];
- $a=$Name;
- Кроме значений полей форм в гобальных ассоциативных массивах хранятся также значения теневых посылок (=$HTTP_POST_VARS[]) и сеансовых переменных (=$HTTP_SESSION_VARS[]).

Переменные вне PHP.

- Когда программой-обработчиком формы является PHP-скрипт, переменные этой формы автоматически доступны для данного скрипта PHP:

```
<form action="foo.php" method="post">
Name:<input type="text" name="name">
<br><input type="submit">
</form>
```

- При активизации формы PHP создаст переменную $name, значением которой будет то содержимое, которое было введено в поле Name данной формы.

Переменные вне PHP.

- PHP также воспринимает массивы в контексте переменных формы, но только одномерные:

```
<form action="array.php" method="post">
Name:<input type="text«
 name="personal[name]"><br>
Email:<input type="text«
 name="personal[email]"><br>
    Beer: <br>
    <select multiple name="beer[]">
    <option value="warthog">Warthog
    <option value="guinness">Guinness
    </select>
    <input type="submit">
</form>
```

Имена переменных рисунка активизации

- При активизации (запуске) формы можно использовать рисунок (изображение) вместо стандартной кнопки запуска:

```
<input type=image src="image.gif"
  name="sub">
```

- Когда пользователь нажимает кнопку мыши где-либо над таким рисунком, сопровождающая форма передается на сервер с двумя дополнительными переменными, sub_x и sub_y. Они содержат координаты места нажатия кнопки мыши пользователем внутри данного рисунка.

Переменные окружения.

- PHP автоматически создает переменные окружения, как и обычные переменные.

```
echo $HOME; /* Показывает переменную HOME, если она установлена.*/
```

- Хотя при поступлении информации механизмы GET, POST и Cookie также автоматически создают переменные PHP, иногда лучше явным образом прочитать переменную окружения, для того чтобы быть уверенным в получении ее правильной версии. Для этого может использоваться функция getenv(). Для установки значения переменной окружения пользуйтесь функцией putenv().

Изменение типа.

- В PHP тип переменной определяется по контексту, в котором она используется. Если присвоить строковое значение переменной var, var становится строкой. Если затем присвоить переменной var значение целого числа, то она станет целым.
- Примером автоматического преобразования типа в PHP может служить оператор сложения '+'. Если какой-либо из операндов является числом типа double, то все операнды оцениваются как double и результат будет иметь тип double. Иначе, эти операнды будут интерпретированы как integer и результат будет integer. Отметим, что при этом НЕ меняются типы операндов, меняется их оценка.

Изменение типа.

- Примеры:

```
$foo = "0"; // $foo - строка (ASCII 48)
$foo++; // $foo - строка "1" (ASCII 49)
$foo += 1;   // $foo является целым (2)
$foo = $foo + 1.3;//$foo - double (3.3)
$foo = 5 + "10 Little Piggies";
// $foo является целым (15)
$foo = 5 + "10 Small Pigs";
// $foo является целым (15)
```

- Если вы желаете изменить тип переменной, используйте settype().

Определение типов переменных.

- Поскольку PHP определяет типы переменных и преобразует их (в общем) по мере необходимости, не всегда очевидно какой тип данная переменная имеет в какой-то отдельный момент. Поэтому PHP включает несколько функций, которые позволяют определить текущий тип переменной. Это функции **gettype()**, **is_long()**, **is_double()**, **is_string()**, **is_array()**, и **is_object()**.

Приведение типа.

- Приведение типа работает в PHP во многом так же как в C: название требуемого типа записывается в круглых скобках перед переменной, которая должна быть приведена к данному типу.

```
$foo = 10;   // $foo is an integer
$bar = (double) $foo;// $bar - double
```

- Допускается следующее приведение типов:
  - (int), (integer) - приведение к целому
  - (real), (double), (float) - приведение к типу double
  - (string) - приведение к строке
  - (array) - приведение к массиву
  - (object) - приведение к объектной переменной.

Преобразование строк.

- Когда строковая переменная оценивается как числовая, результирующее значение и тип переменной определяются так:
- Переменная string будет оценена как double, если она содержит любой из символов '.', 'e', или 'E'. Иначе она будет оценена как integer.
- Данное значение задается начальной частью строковой переменной. Если строка начинается с допустимых цифровых данных, то это значение и будет использоваться. Иначе, будет значение 0 (ноль).

Преобразование строк.

- Примеры:

```
$foo = 1 + "10.5";//double (11.5)
$foo = 1 + "-1.3e3";// double (-1299)
$foo = 1 + "bob-1.3e3"; // integer (1)
$foo = 1 + "bob3"; // integer (1)
$foo = 1 + "10 Small Pigs";
// integer (11)
```

Манипуляции с массивом.

- PHP поддерживает как скалярные, так и ассоциативные массивы. Вы можете создать массив, используя функции list() или array(), или можно явно задать значение каждого элемента массива.

```
$a[0] = "abc";
$a[1] = "def";
$b["foo"] = 13;
```

- Можно также создать массив просто добавляя в него значения.

```
$a[] = "hello"; // $a[2] == "hello"
$a[] = "world"; // $a[3] == "world"
```

Манипуляции с массивом.

- Инициализация хеша:
- $c["one"]="abc";
- $c["two"]="def";
- $c[99]="ghi"; // аналог $c["99"]="ghi";
- Применение функции array():
- $d=array("one","two",86.22);
- $e=array("one"=>"abc", "two"=>"def");

Манипуляции с массивом.

- **Многомерные массивы:**
- $a[][0]=4; $a[1][]=5;
- $b[]["one"]="abc"; $b[3]["bb"]="abc";
- $f["first"]["second"]="123";
- $c[“aa"][5][1][“bb"]=$f;
- **Применение функции array():**
- $fruit=array("апельсин"=>array("цвет"=>"оранжевый","вкус"=>"победы"),
- "лимон"=>array("цвет"=>"желтый", "вкус"=>"кислый"));
- echo $fruit["лимон"]["цвет"]; // желтый

Манипуляции с массивом.

- Массив может сортироваться функциями **asort(), arsort(), ksort(), rsort(), sort(), uasort(), usort(),** и **uksort()** в зависимости от типа желаемой сортировки. Подсчет количества элементов массива осуществляется функцией **count().** Перемещаться по массиву позволяют функции **next()** и **prev().** Другим типовым способом перемещения по массиву является использование функции **each().**