Типы данных и инициализация переменных PHP

В PHP переменные могут содержать следующие основные типы данных:

- bool (или boolean) — логический тип данных. Может содержать значения true или false (регистр не важен);
- int (или integer) — целые числа;
- float (или double) — вещественные числа;
- string — строка;
- object — объекты;
- array — массивы;
- resource— идентификаторы ресурсов, например, файлов;
- null— означает, что переменная не содержит значения.

При инициализации переменной интерпретатор автоматически относит ее к одному из типов данных. Значение переменной присваивается с помощью оператора = так:

```
$b = true; // bool

$x = 7; // int

$y = 7.8; // float

$sl = «Строка»; // string

 $s2 = 'Строка'; // string

$arr = [0, 1]; // array

$n = null;  // NULL
```

PHP в любой момент времени изменяет тип переменной в соответствии с данными, хранящимися в ней:

```
$var = "Строка"; // тип string

$var =7; // теперь переменная имеет тип int
```

Функция gettype(<имя_переменной>) возвращает тип данных переменной (лис тинг 5.9). Для целей отладки можно также использовать функцию var_dump(<Имя_ переменной>).

```
<?php
$var = null;
echo gettype($var),
var dump ($var) ;
$var = true;
echo gettype($var) ,
var dump ($var) ;
 $var = 7;
echo gettype($var), "\n";
var dump($var) ;
$var = 7.8;
echo gettype($var), "\n";
var dump($var);
$var = 'Строка';
echo gettype($var), "\n";
var dump($var);
$var = array();
echo gettype($var) , "\n";
var dump($var);
```

Кроме того, существуют функции проверки конкретного типа данных:

- is bool (<переменная>) — возвращает true, если переменная имеет тип bool (ло гический тип данных);
- is int (<переменная>) — возвращает true, если переменная имеет тип int (целое число);
- is integer (<переменная>) — возвращает true, если переменная имеет ТИП int (целое число);
- □ is fioat (<переменная>) — возвращает true, если переменная имеет ТИП float (вещественное число);

- is doubie (<переменная>) — возвращает true, если переменная имеет тип float (вещественное число);
- is string (<переменная>) — возвращает true, если переменная имеет тип string (строка);
- is array (<Переменная>) — возвращает true, если переменная имеет ТИП array (массив);
- is object (<переменная>) — возвращает true, если переменная имеет тип object (объект);
- is resource(<переменная>)— возвращает true, если переменная имеет тип resource (ресурс);
- is null (<переменная>) — возвращает true, если переменная имеет значение null.
- Проверить тип переменной можно следующим образом:

```
$var = 7;

if (is_int ($var) ) {

echo 'Переменная $var имеет тип int';

}
```

Преобразование и приведение типов

Если функция ожидает один тип, а мы передаем другой, то интерпретатор по умол чанию автоматически пытается выполнить преобразование типов. Для явного преобразования типов данных можно воспользоваться функцией settypeo, которая преобразует тип переменной в указанный: settype (<Переменная>, <Тип>)

Пример:

```
$var = 1.5;

settype($var, "string");

var dump($var);

settype($var, "int");

var dump($var);
```

Можно также воспользоваться приведением типов. Для этого перед переменной в круглых скобках указывается тип, к которому нужно преобразовать значение переменной.

Внимание!

В отличие от функции settypeo, приведение типов не меняет тип исходной переменной.

Пример приведения типов:

```
$var = 0;

var dump($var); // int(0)

$var2 = (bool)$var;

var_dump($var2); // bool(false)
```

Значение будет преобразовано в false в следующих случаях:

- *если число равно о или о. о;*
- *если указана пустая строка или строка "0"; ▫ если указан пустой массив;*
- *если переменная содержит значение null.*

Все остальные значения будут преобразованы в true.

Для преобразования типов можно также воспользоваться функциями booivai (), intval(), floatval(), doubleval() И strval() :

```
$var = 0;

var dump($var); // int(0)

$var2 = boolval($var);

var_dump($var2); // bool(false)
```

Проверка существования переменной

С помощью оператора isset (<переменная>) можно проверить существование пере менной. Если переменная существует, то возвращается значение true. Для примера переделаем нашу первую программу так, чтобы она «здоровалась» не со всем миром, а только с нами.

```
<?php

if (isset($_GET['name']) ) {

echo 'Hello, ' . $_GET['name'];

}

else {

echo 'Введите ваше имя<Ьг>';

echo '<form action="' . $_SERVER['SCRIPT_NAME'] . '"Re echo '<input type="text" name="name">';

echo '<input type="submit" value="0K">';

echo '</form>';

}
```

При первом запуске программы появится приглашение ввести имя. Вводим свое имя (например, Николай) и нажимаем кнопку OK. В итоге отобразится приветствие Hello, Николай.

Оператор empty(<переменная>) проверяет наличие у переменной непустого, ненуле вого значения. Возвращает true, если переменная пустая, не существует или имеет нулевое значение. Например, код:

```
if (isset($str) ) echo "Существует"; else echo "Нет";

echo "<br>";

if (empty($str)) echo "Пустая”; else echo "Нет";
```

вернет следующие значения:

Нет Пустая

А если предварительно инициализировать переменную $str, например, так:

```
$str = "Строка";

if (isset($str)) echo "Существует"; else echo "Нет";

echo "<br>";

if (empty($str)) echo "Пустая"; else echo "Нет";
```

то вывод программы будет отображен Web-браузером так:

Существует Нет

Удаление переменной

Удалить переменную можно с помощью оператора unseto:

unset(<Переменная>)

Этот оператор необходим, если переменная использовалась при обработке данных большого объема и теперь не нужна. Удаление переменной позволит освободить память компьютера:

$str = "Строка";

if (isset($str)) echo "Существует"; else echo "Нет";

unset($str);

*echo "
";*

if (isset($str)) echo "Существует"; else echo "Нет";

Вывод программы:

Существует Нет

Переменные окружения

Создадим сценарий, состоящий всего из двух строк:

<?php $var = 10;

А теперь вопрос: сколько переменных доступно сценарию? Думаете, одна $var? Давайте перепишем нашу программу и добавим одну строчку:

<?php

$var = 10;

echo $_SERVER['DOCUMENT—ROOT'];

В результате работы скрипта в окне Web-браузера отобразится следующая строка:

C:/xampp/htdocs

Откуда же взялась переменная $_server [' document root '] ? Ведь мы ее не создава ли! Ответ на этот вопрос достаточно прост— эта переменная была автоматически создана интерпретатором. Такая переменная называется переменной окружения.

Рассмотренная нами ранее переменная окружения $ server [' document root '] пред ставляет собой элемент массива $_server. Это весьма примечательный массив — он доступен не только в глобальной области видимости, но в любой части скрипта, из-за чего и носит название суперглобалъного.

Приведем суперглобальные массивы:

- *$ get — массив переменных, переданных посредством метода get;*
- *$ post — массив переменных, переданных посредством метода post;*
- *$ server — массив переменных среды сервера;*
- *$ files — массив переменных, определяющих отправленные через форму файлы;*
- *$ cookie — массив cookies-переменных;*
- *$_env — массив переменных, определяющих конфигурацию среды;*
- *$_request — массив всех переменных, вводимых пользователем. Этот массив зависит ОТ значения директивы request order.*

Вывести значения всех переменных окружения позволяет код из листинга:

```
<pre>
<?php
echo "\$J3ET\n"; print—r($_GET);
echo "\n\$_POST\n"; printf r ($_POST) ;
echo "\n\$_SERVER\n"; print r($_SERVER);
echo "\n\$_FILES\n"; print_ r ($_FILES) ;
echo "\n\$_COOKIE\n"; print__r ($_COOKIE) ;
 echo "\n\$_ENV\n"; print_r($ ENV);
echo "\n\$—REQUEST\n"; print—r($_REQUEST);
?>
</pre>
```

Рассмотрим наиболее часто используемые переменные окружения:

$_server ['document_root'] — путь к корневому каталогу сервера;

$_server['remote _addr'] — IP-адрес клиента, запрашивающего ресурс;

$_server['remotE-USEr'] — имя пользователя, прошедшего аутентификацию;

$_server['Query-String'] — строка переданных серверу параметров;

$_server['лттР—USER—agent']—названиеиверсияWeb-браузераклиента;

$_server['лттР—referer']— URL-адрес, с которого пользователь перешел на наш сайт;

$_server['request_method'] — метод передачи информации (GET ИЛИ POST). Прежде чем использовать переменные окружения, необходимо проверить существование переменной с помощью оператора isset о : if (isset($_SERVER['HTTP—REFERER'])) {echo $_SERVER['’HTTP_REFERER'];

}

Предположим, что пользователь заполнил форму с одним текстовым полем, имею щим имя textl (name="textl"). При передаче данных методом get сервер сформирует следующие переменные:

$_GET['’textl']

$_REQUEST['’textl']

Если передача формы осуществлялась методом post, то сервер сформирует другие переменные:

$_POST['’textl’]

$_REQUEST['textl’]

Мы можем извлечь ее значение и присвоить обычной переменной PHP:

if (isset($_GET['textl'])) $textl = $_GET['textl']; else $textl = '';

ИЛИ

if (isset($_POST['textl’])) $textl = $_POST['textl'] ;

else $textl = '';

Аналогичную операцию проще выполнить с помощью оператора ??:

$textl=$_GET['textl’] ?? ‘’;

ИЛИ

$textl = $_POST['’textl’] ?? ‘’;

Если переменная окружения существует, то переменная $textl получит ее значе ние, в противном случае переменной $textl будет присвоена пустая строка.

Остальные переменные окружения используются реже, но по названиям их предна значение интуитивно понятно. В дальнейшем мы еще не раз будем возвращаться к переменным окружения.

Управлять порядком обработки суперглобальных массивов позволяет директива variables order в файле php.ini. В пакете XAMPP директива имеет следующее зна чение:

variables—order=«GPCS"

Буквы, указанные внутри строки, это первые буквы в именах get, post, cookie, server и env. В этом примере отсутствует буква e, т. к. буква s всегда эквивалента комбинации es. Если какая-либо буква не указана, то соответствующий массив не создается.