PHP的基本语法

网址：www.lampbrother.net

电话：400 700 1307

无兄弟不编程

PHP的基本语法

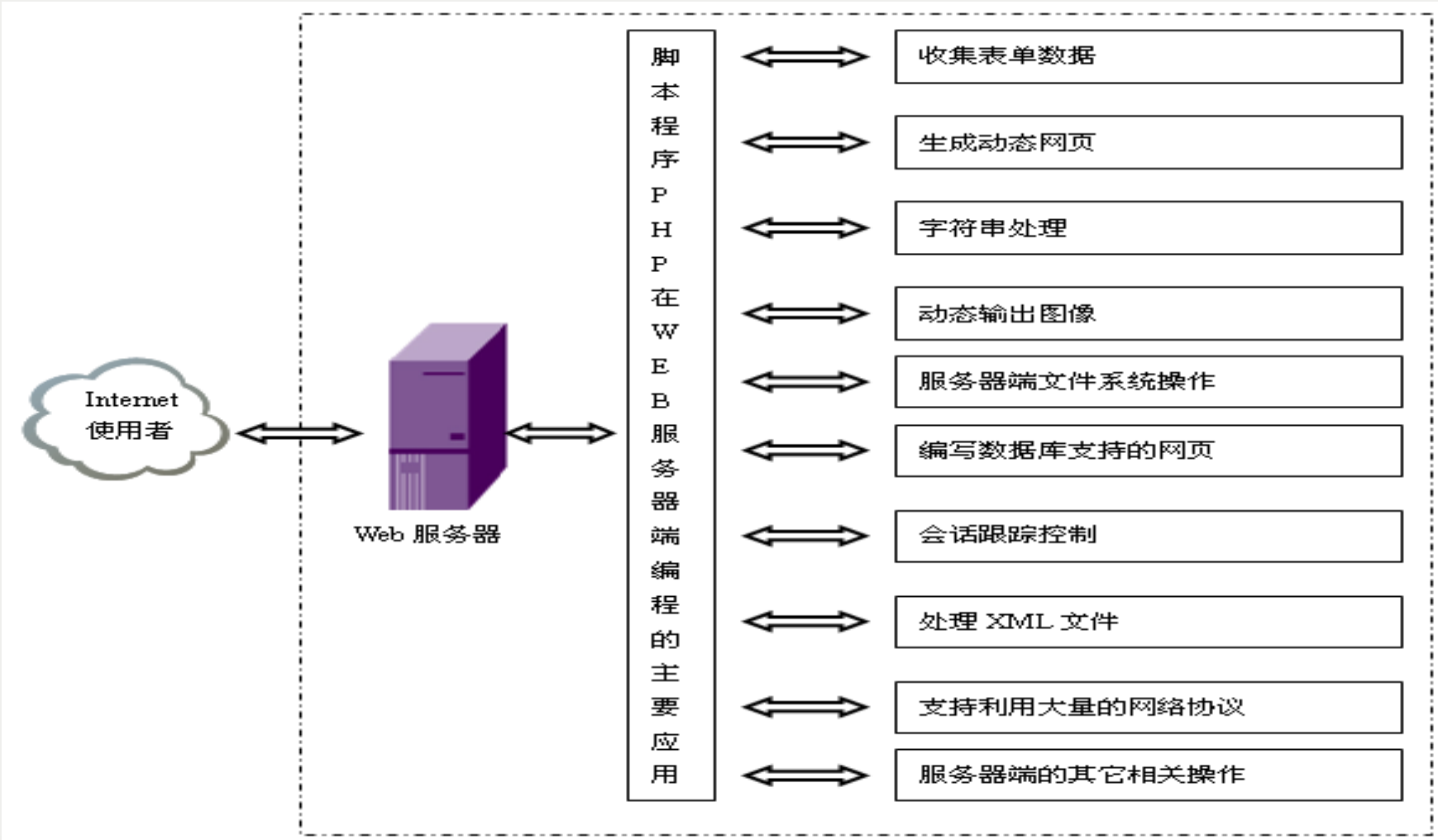
1. PHP在Web开发中的应用
2. 第一个PHP脚本语言
3. 变量
4. 变量的类型
5. 常量
6. PHP中的运算符
7. 表达式

一、 PHP在Web开发中的应用

PHP是什么？

- PHP(Hypertext Preprocessor缩写)超级文本预处理器。
- PHP是一种在服务器端执行的嵌入HTML文档的脚本语言。
- PHP是目前最流行的网站开发语言（ B/S结构）。
- PHP 独特的语法混合了 C、Java、Perl 以及 PHP 自创新的语法 。
- 支持几乎所有流行的数据库以及操作系统

❖ PHP在Web中的功能:



PHP简要历史:

- 1994年Rasmus Lerdorf第一个设计出PHP。
- 1995年6月PHP1.0的声明。
- 1996年4月发表了PHP第二版本的声明PHP/FI2.0。
- 1998年6月PHP3.0的声明。
- 2000年5月22日正式发布4.0。
- 2004 年 7 月正式发布PHP5版本
- 到2004年8月，PHP已经在全球的1700多万个网站域中安装，而且现在还在不断快速增长。
- PHP的主页: http://www.php.net

❖ 语言排名： 数据为2010年2月 来源：Tiobe网站

Position Feb 2010	Position Feb 2009	Delta in Position	Programming Language	Ratings Feb 2010	Delta Feb 2009	Status
1	1	=	Java	17.348%	-2.05%	A
2	2	=	C	16.602%	+0.76%	A
3	5	⇈	PHP	10.001%	+1.22%	A
4	3	↓	C++	9.447%	-0.19%	A
5	4	↓	(Visual) Basic	7.051%	-1.79%	A
6	6	=	C#	5.015%	-0.05%	A

❖ PHP的性能比较：

性能比较	LAMP	J2EE	ASP.NET
运行速度	较快	快	快
开发速度	快	慢	快
运行损耗	一般	较小	较大
难易程序	简单	难	简单
运行平台	Linux/Unix/windows等	绝大多数平台均可	Windows平台
扩展性	好	好	较差
安全性	好	好	较差
应用程度	较广	较广	较广
建设成本	非常低	非常高	高

PHP特点：

- 是开放源代码的，服务器端的脚本语言.
- 独立于操作系统，可以运行在几乎所有系统中.
- 支持大部分的服务器，如apache,IIS
- 支持大量的数据库
- 可以创建图象
- 还有一些其他功能在后面的高级技术详细介绍。

二、第一个PHP脚本语言

- 2.1 PHP语言标记
- 2.2 指令分割符”分号”
- 2.3 程序注释
- 2.4 在程序中使用空白的处理

❖ 示例： hello.php

```
<html>
    <head>
        <title>我的第一个PHP页面</title>
    </head>
    <body>
        <h2>
        <?php  echo “Hello, PHP!” ?>
        </h2>
    </body>
</html>
```

起始符

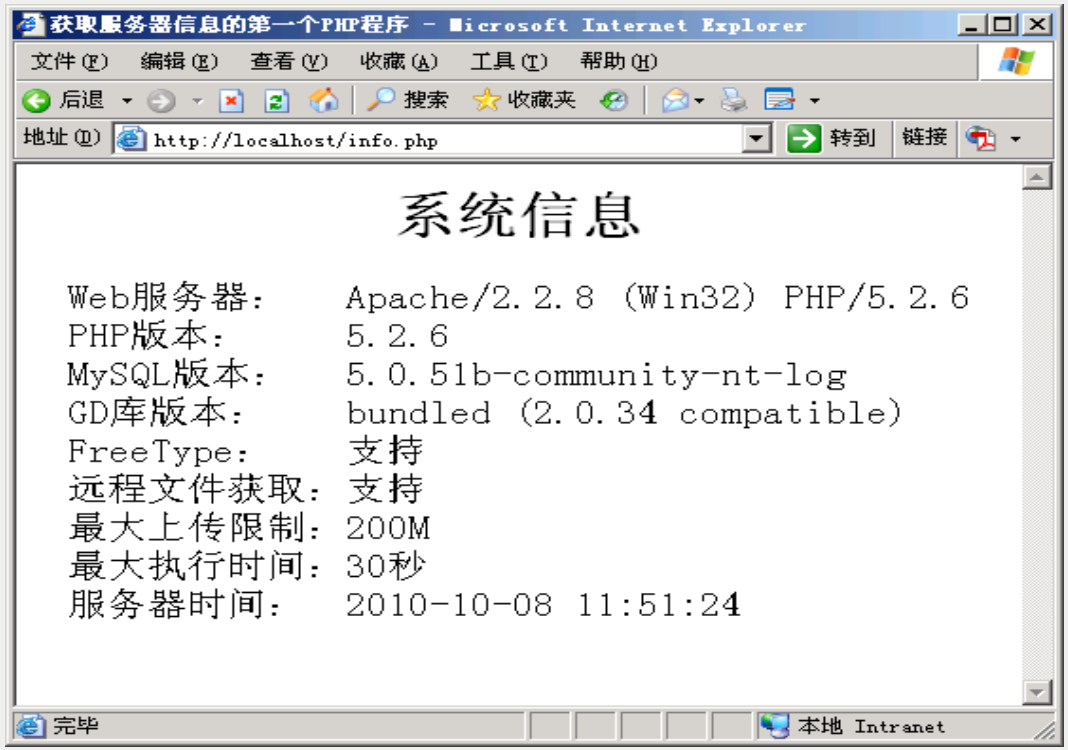
结束符

这就是PHP脚本

文件后缀名为.php结尾，上传到Web服务器的文档根目录下，通过浏览器访问Web服务器管理下的PHP文件，就可以运行PHP文件。

- PHP的开发步骤：
  - 使用编辑器创建一个包含源代码的磁盘文件
  - 将文件上传到web服务器上
  - 通过浏览器访问Web服务器运行程序
- 示例： info.php
- 代码详见：P139

  里面具体语法含义在后面的课程中讲到。



2.1 PHP语言标记

- 我们用<?php来表示PHP标识符的起始，然后放入PHP语句并通过加上一个终止标识符?>来退出PHP模式。可以根据自己的需要在HTML文件中像这样开启或关闭PHP模式。大多数的嵌入式脚本语言都是这样嵌入到HTML中并和HTML一起使用，例如CSS、JavaScript、PHP、ASP以及JSP等。

```
<html>
    <head>
        <style> body{ background:#ccc;} </style>
    </head>
    <body>
        <script> alert("客户端时间"+(new Date())); </script>
        <?php   echo "服务器端的时间".date("Y-m-d H:i:s"); ?>
    </body>
</html>
```

CSS

JavaScript

PHP

PHP语言嵌入HTML中的位置

```
<html>
    <head>
        <title> <?php   echo “PHP 语言标记的使用” ?> </title>
    </head>
    <body <?php echo ’bgcolor=“#cccccc”’ ?> >
        <?php  if($expression){  ?>
          <p align=“ <?php echo “center” ?> ”>This is true</p>
        <?php  }else{ ?>
           <p>This is false</p>
        <?php  } ?>
    </body>
</html>
```

嵌入到页面的标题处

嵌入到html标签属性中

在HTML中更高级的分离技术

使用不同的四对标记

- 以<?php开始和以?>结束标记是标准风格，这是PHP推荐使用的标记风格。
- 以<script language="php">开始和<script>结束是长风格标记，这种标记最长，总是可用的，但我们并不常用。
- 以<?开始和以?>结束标记是简短风格的标记，是最简单的，但是系统管理员偶尔会禁用掉它，因为它会干扰XML文档的声明。只用通过php.ini配置文件中的指令short_open_tag打开后就可以使用。
- 以<%开始和以%>结束标记是ASP风格的标记，可以在php.ini配置文件设定中启用了asp_tags选项就可以使用它（默认是禁用的），习惯了ASP风格的可以使用它。

2.2 指令分割符”分号”

- PHP同C或Perl以及Java一样，语句分为两种:
- 一种是在程序中使用结构定义语句例如流程控制、函数与类的定义等，是用大括号来标记代码块，在大括号后面不要用分号。
- 另一种是在程序中使用功能执行语句，如变量的声明、内容的输出、函数的调用等，是用来在程序中执行某些特定功能的语句，这种语句也可称为指令，PHP需要在每个指令后用分号结束。
- 和其他语言不一样的是，在PHP中右括号(?>)前的分号不是必选的。

2.3 程序注释

对于阅读代码的人来说，注释其实就相当于代码的解释和说明。注释可以用来解释脚本的用途、脚本编写人、为什么要按如此的方法编写代码、上一次修改的时间等等。

- PHP支持C、C++和Shell脚本风格的注释，如下：
  - //... ... 单行注释
  - /* ... ... */多行注释 （注意：不能嵌套）
  - # ... ... 脚本注释
- 程序员在编程时使用注释是一种良好的习惯，优点：
  - 写过不合适的代码注释
  - 写帮助文档
  - 调试程序
- 注意： 注释要写在代码的上面或是右边

2.4 在程序中使用空白的处理

- 一般来说，空白符（空格、Tab制表符、换行）在PHP中无关紧要。可以将一个语句展开成任意行，或者将语句紧缩在一行。
- 可以利用这个灵活的格式来使代码更具有可读性（通过排列分配、缩进等）。一些懒惰的程序员利用这种自由的格式创建根本无法阅读的代码，这是不提倡的。

- 使用两个空行
  - 一个源文件的两个代码段
  - 两个类的声明
- 在以下情况使用一个空行
  - 两个函数声明之间
  - 函数内的局部变量和函数的第一条语句之间
  - 注释或者单行注释之前
  - 一个函数的两个逻辑代码段

三、变量

- 3.1 变量的声明
- 3.2 变量的命名
- 3.3 可变变量
- 3.4 变量的引用赋值

3.1 变量的声明

- 变量是用于临时存储值的容器。这些值可以是数字、文本、或者复杂得多的排列组合。是用于跟踪几乎所有类型信息的简单工具。
- PHP是一种非常弱的类型语言。在大多数编程语言中，变量只能保持一种类型的数据，而且这个类型必须在使用变量前声明，例如C语言中。而在PHP中，变量的类型通常不是由程序员设定的，确切地说，是根据该变量使用的上下文在运行时（即变量的值）决定的。PHP不要求在使用变量之前声明变量，当第一次给一个变量赋值时，你才创建了这个变量。

PHP变量的声明:

- PHP的变量声明是以$符开始的，后面跟英文的大写，小写，下划线。但不能以数字开头。

```
<?php
        $a=100;                 //声明一个变量d，赋予整数100
        $b="string";            //声明一个变量d，赋予字串string
        $c=true;                //声明一个变量d，赋予布尔值true
        $d=99.99;               //声明一个变量d，赋予浮点数99.99
        $key=$a;                //声明一个key变量，并将a变量的值赋予
        $a=$b=$c=$d="value";    //同时声明多个变量，并赋予相同的值
?>
```

- 可以使用函数unset()释放指定的变量，isset()函数检测变量是否设置，empty()函数检查一个变量是否为空。

3.2 变量的命名

- 变量名与 PHP 中其它的标签一样遵循相同的规则。一个有效的变量名由字母或者下划线开头，后面跟上任意数量的字母，数字，或者下划线。
- 变量的名称是对大小写敏感的。
- 但内置结构和关键字以及用户自定义的类名和函数名都是不区分大小写的。如：echo、while、function名称等。

```
<?php
        echo “this is a test”;
        Echo “this is a test”;
        $name=“tarzan”;
        $Name=“skygao”;
        echo $name.$Name;        //输出：tarzanskygao
?>
```

这两行输出是一样的

这是两个不同的变量

3.3 可变变量

- 有时候使用可变变量名是很方便的。就是说，一个可变变量获取了一个普通变量的值作为这个可变变量的变量名。 例如:

```
<?php
        $a = 'hello';                   //普通变量
        $$a = ‘world’;                  //可变变量
        echo "$a ${$a}";                //输出：  hello world
        echo "$a $hello";               //输出：  hello world
?>
```

3.4 变量的引用赋值

- 引用操作符&可以在关联赋值中使用，就像一个别名，使得变量都指向了内存的相同地址。

```
<?php
    $a=5;
    $b=&$a;
    echo $b;        //输出5
    $a=7;
    echo $b;        //输出7
?>
```

- 通过unset($a)重置变量与内存的关联

四、变量的类型

- 4.1 类型介绍
- 4.2 布尔型(boolean)
- 4.3 整型(integer)
- 4.4 浮点型(float或double)
- 4.5 字符串(String)
- 4.6 数组(Array)
- 4.7 对象(Object)
- 4.8 资源类型(Resource)
- 4.9 NULL类型
- 4.10 伪类型介绍
- 4.11 数据类型之间相互转换

4.1 类型介绍

数据类型:

PHP 支持八种原始类型。

- 四种标量类型:
  - 布尔型（boolean）
  - 整型（integer）
  - 浮点型（float）（浮点数，也作double）
  - 字符串（String）
- 两种复合类型:
  - 数组（Array）
  - 对象（Object）
- 最后是两种特殊类型:
  - 资源（Resource）
  - NULL

- 在PHP中，变量的类型通常不是由程序员设定的，确切地说，是根据该变量使用的上下文在运行时（即变量的值）决定的。
- 使用函数var_dump( )查看表达式的值和类型。

```php
<?php
        $bool=TRUE;             //赋一个布尔值
        $str="foo";             //赋一个字符串
        $int=12;                //赋一个整型值

        var_dump($bool);        //输出：bool(true)
        var_dump($str);         //输出：string(3) "foo"
        var_dump($int);         //输出：int(12)
?>
```

4.2 布尔型(boolean)

- 这是最简单的类型。boolean 表达了真值，可以为 TRUE 或 FALSE，即“真”或“假”。
- 当其他类型转换为 boolean类型 时，以下值被认为是 FALSE:
  - 布尔值 FALSE
  - 整型值 0（零）
  - 浮点型值 0.0（零）
  - 空白字符串和字符串"0"
  - 没有成员变量的数组
  - 没有单元的对象（仅适用于 PHP 4）
  - 特殊类型 NULL（包括尚未设定的变量）

所有其它值都被认为是 TRUE（包括任何资源）。

```php
<?php
  var_dump((bool)"");          //bool(false)
  var_dump((bool)"false");     //bool(true)
  var_dump((bool)-1);          //bool(true)
  var_dump((bool)0);           //bool(false)
?>
```

4.3 整型(integer)

- 整型值可以用十进制，十六进制或八进制符号指定，前面可以加上可选的符号（- 或者 +）代表数值的正负。

```php
<?php
   $a = 1234;       // 十进制数
   $a = -123;       // 一个负数
   $a = 0123;       // 八进制数（等于十进制的 83）
   $a = 0x1A;       // 十六进制数（等于十进制的 26）
?>
```

- 整数值有最大的使用范围，这与平台有关，对于32位系统而言范围：-2147483648～2147483647,PHP不支持无符号整数。如果超出了则变成了float型。

```php
$large_number=2147483648;
var_dump($large_number);          //输出：float(2147483648)
```

4.4 浮点型(float或double)

- 浮点数（也叫双精度数或实数）是包含小数部分的数。通常用来表示整数无法表示的数据，如金钱值、距离值、速度值等。浮点数的字长和平台相关，尽管通常最大值是 1.8e308 并具有 14 位十进制数字的精度。
- 可以用以下任何语法定义:

```
<?php
   $a = 1.234;
   $a = 1.2e3;        //相当于1.2*10^3即1200
   $a = 7E-10;        //相当于7*10^-10即0.0000000007
?>
```

- 注意事项: 例: floor((0.1+0.7)*10) 通常会返回 7 而不是预期中的 8，因为该结果内部的表示其实是7.9。就是不可能精确的用有限位数表达某些十进制分数。所以永远不要相信浮点数结果精确到了最后一位，也永远不要比较两个浮点数是否相等。如果确实需要更高的精度，应该使用任意精度数学函数或者 gmp 函数。

4.5 字符串(String)

- 字符串的定义

  string是一系列字符。在 PHP 中，字符和字节一样，也就是说，一共有 256 种不同字符的可能性。这也暗示 PHP 对 Unicode 没有本地支持。

  注: 一个字符串变得非常巨大也没有问题，PHP 没有给字符串的大小强加实现范围，所以完全没有理由担心长字符串。

- 语法:
  - 字符串可以用三种字面上的方法定义:
    - 单引号 ''
    - 双引号 " "
    - 定界符 <<<

- 单引号: 指定一个简单字符串的最简单的方法是用单引号（字符 ′）括起来。
- 要表示一个单引号，需要用反斜线（ \）转义，和很多其它语言一样。如果在单引号之前或字符串结尾需要出现一个反斜线，需要用两个反斜线表示。注意如果试图转义任何其它字符，反斜线本身也会被显示出来！所以通常不需要转义反斜线本身。
- 注：单引号字符串中出现的变量不会被变量的值替代。

```
<?php
   echo ‘this is a simple string’;              //输出：this is a simple string
   echo ‘this is a \’simple\‘ string’;          //输出：this is a 'simple' string
   echo ‘this \n is \r a \t simple string\\’;   //输出：this \n is \r a \t simple string\
   $str=100;
   echo ‘this is a simple $str string’;         //输出：this is a simple $str string
?>
```

- 双引号：如果用双引号（"）括起字符串，PHP 懂得更多特殊字符的转义序列：
- 注：双引号字符串最重要一点是其中的变量名会被变量值替代。此外，如果试图转义任何其它字符，反斜线本身也会被显示出来！转义字符如下表所示：

序列	含义
\n	换行（LF 或 ASCII 字符 0x0A（10））
\r	回车（CR 或 ASCII 字符 0x0D（13））
\t	水平制表符（HT 或 ASCII 字符 0x09（9））
\\	反斜线
\$	美元符号
\”	双引号
\[0-7]{1,3}	此正则表达式序列匹配一个用八进制符号表示的字符
\x[0-9A-Fa-f]{1,2}	此正则表达式序列匹配一个用十六进制符号表示的字符

❖ 示例：

```php
<?php
    $beer='Heineken';
    echo "$beer's taste is great";        //输出：Heineken's taste is great

    echo "He drank some $beers";          //输出：He drank some

    echo "He drank some ${beer}s";        //输出：He drank some Heinekens

    echo "He drank some {$beer}s";        //输出：He drank some Heinekens
?>
```

没有$beers这个变量

使用{ }包含起来，就可以将变量分离出来。最后这两种都可以

- 定界符：另一种给字符串定界的方法使用定界符语法（“<<<”）。应该在 <<< 之后提供一个标识符，然后是字符串，然后是同样的标识符结束字符串。
- 定界符中标识符的命名规则与变量的命名规则一样。只能包含字母数字下划线，而且必须以下划线或非数字字符开始。
- 注：结束标识符所在的行不能包含任何其它字符，可能除了一个分号（;）之外。这尤其意味着该结束标识符不能被缩进，而且在分号之前和之后都不能有任何空格或制表符。如果破坏了这条规则使得结束标识符不“干净”，则它不会被视为结束标识符，PHP 将继续寻找下去。如果在这种情况下找不到合适的结束标识符，将会导致一个在脚本最后一行出现的语法错误。
- 不能用定界符语法初始化类成员。用其它字符串语法替代。
- 定界符文本的表现和双引号字符串一样，只是没有双引号

4.6 数组(Array)

- PHP 中的数组实际上是一个有序图。图是一种把 *values* 映射到 *keys* 的类型。此类型在很多方面做了优化，因此可以把它当成真正的数组来使用，或列表（矢量），散列表（是图的一种实现），字典，集合，栈，队列以及更多可能性。因为可以用另一个 PHP 数组作为值，也可以很容易地模拟树。本书将用一章介绍数组的声明与使用，这里仅作简要说明。
- PHP中可以使用多种方式构建一个数组，在这里我们只用array()语言结构来新建一个array。它接受一定数量用逗号分隔的 *key => value* 参数对。

语法结构：

```
array( [key =>] value , ... )
// key 可以是integer或者string类型
// value 可以是任何值
```

```
<?php
    $arr = array(
        "foo" => "bar", 12 => true);
?>
```

4.7 对象(Object)

- 在PHP中，对象和数组一样都是一种复合数据类型。但对象是一种更高级的数据类型。一个对象类型的变量，是由一组属性值和一组方法构成，其中属性表明对象的一种状态，方法通常用来表明对象的功能。本书将用一章的内容来介绍对象的使用，这里仅作简要的说明。要初始化一个对象，用 *new* 语句将对象实例到一个变量中。

```php
<?php
    class foo{                      //类的定义
        function do_foo(){          //类中方法的定义
            echo "Doing foo.";
        }
    }
    $bar = new foo;                 //初始化类foo创建一个对象bar
    $bar->do_foo();                 //通过对象bar调用方法do_foo输出： Doing foo.
?>
```

4.8 资源类型(Resource)

- 资源是一种特殊变量，保存了到外部资源的一个引用。资源是通过专门的函数来建立和使用的。
- 由于资源类型变量保存有为打开文件、数据库连接、图形画布区域等的特殊句柄，因此无法将其它类型的值转换为资源。
- PHP4Zend引擎引进了资源计数系统，可以自动检测到一个资源不再被引用了（和 Java 一样）。这种情况下此资源使用的所有外部资源都会被垃圾回收系统释放。由此原因，很少需要用某些 free-result 函数来手工释放内存。

```php
<?php
    $file_handle=fopen("info.txt","w");
    var_dump($file_handle);              //resource(3) of type (stream)
    $link_mysql=mysql_connect("localhost","root","root");
    var_dump($link_mysql);               //resource(4) of type (mysql link)
?>
```

4.9 NULL类型

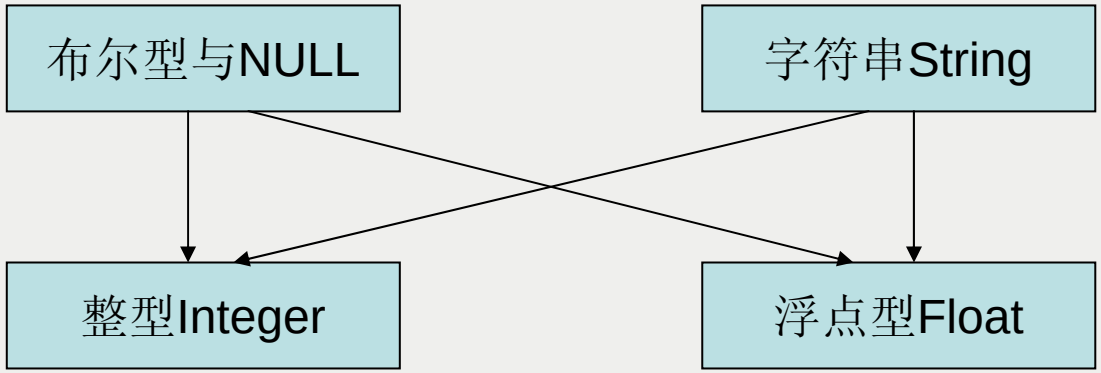
- 特殊的 NULL 值表示一个变量没有值。NULL类型唯一可能的值就是NULL，表示一个变量的值为空，NULL不区分大小写。
- 在下列情况下一个变量被认为是 NULL:
  - 被赋值为 NULL值的变量。
  - 尚未被赋值的变量。
  - 被unset()函数销毁的变量。

4.10 伪类型介绍

- 伪类型并不是PHP语言中的基本数据类型，只是因为PHP是弱类型语言，所以在一些函数中，一个参数可以接收多种类型的数据，还可以接收别的函数作为回调函数使用。为了确保代码的易读性在本书中介绍一些伪类型的使用。
  - *mixed:* 说明一个参数可以接受多种不同的（但并不必须是所有的）类型。
  - *number:* 说明一个参数可以是 integer 或者 float。
  - callback: 有些诸如 call_user_function() 或 usort() 的函数接受用户自定义的函数作为一个参数。Callback 函数不仅可以是一个简单的函数，它还可以是一个对象的方法，包括静态类的方法。
- 一个 PHP 函数用函数名字符串来传递。可以传递任何内置的或者用户自定义的函数，除了 array()，echo()，empty()，eval()，exit()，isset()，list()，print() 和 unset()。

4.11 数据类型之间相互转换

- PHP 在变量定义中不需要（或不支持）明示的类型定义;变量类型是根据使用该变量的上下文所决定的。
- 类型转换是指将变量或值从一种数据类型转换成其他数据类型。转换的方法有两种:
  - 自动转换
  - 强制转换



数据类型自动转换的关系

类型强制转换

- PHP 中的类型强制转换和 C 中的非常像：在要转换的变量之前加上用括号括起来的目标类型。
- 允许的强制转换有：
  - (int)，(integer) - 转换成整型
  - (bool)，(boolean) - 转换成布尔型
  - (float)，(double)，(real) - 转换成浮点型
  - (string) - 转换成字符串
  - (array) - 转换成数组
  - (object) - 转换成对象
- 注意在括号内允许有空格和制表符，为了将一个变量还原为字符串，还可以将变量放置在双引号中。

- 变量类型的测试函数:
  - is_bool():判断是否是布尔型
  - is_int()、is_integer()和is_long():判断是否为整型。
  - is_float()、is_double()和is_real(): 判断是否为浮点型
  - is_string(): 判断是否为字符串
  - is_array(): 判断是否为数组
  - is_object(): 判断是否为对象
  - is_resource(): 判断是否为资源类型
  - is_null(): 判断是否为null
  - is_scalar(): 判断是否为标量
  - is_numberic(): 判断是否是任何类型的数字和数字字符串
  - is_callable(): 判断是否是有效的函数名
- 函数: bool settype (mixed var, string type) 是将变量 *var* 的类型设置成 *type*。

五、常量

- 5.1 常量的定义与使用
- 5.2 常量与变量
- 5.3 预定义常量

- 常量是一个简单值的标识符（名字）。如同其名称所暗示的，在脚本执行期间一个常量一旦被定义，就不能再改变或取消定义。常量默认为大小写敏感。按照惯例常量标识符总是大写的。
- 常量名和其它任何 PHP 标签遵循同样的命名规则。合法的常量名以字母或下划线开始，后面跟着任何字母，数字或下划线。
- 常量的范围是全局的。不用管作用域就可以在脚本的任何地方访问常量。
- 我们可以用 define() 函数来定义常量。

5.1 常量的定义与使用

- 使用define()函数来定义常量。一个常量一旦被定义，就不能再改变或者取消定义。

语法：**bool define (string name, mixed value [, bool case_insensitive])**

- 其中name表示常量名，value表示常量值或表达式，但常量只能包含标量数据（boolean，integer，float和string）。第三个为可选参数case_insensitive设置为true时则表示常量名不区分大小写。

```
<?php
        define("CON_INT",100);
        echo CON_INT;                                   //输出：100
        define("GREETING","Hello you",true);
        echo GREETING;                                  //输出：Hello you
        echo constant ("Greeting");                     //输出：Hello you
?>
```

5.2 常量与变量

- 常量和变量不同:
  - 常量前面没有美元符号（$）;
  - 常量只能用 define() 函数定义，而不能通过赋值语句;
  - 常量可以不用理会变量范围的规则而在任何地方定义和访问;
  - 常量一旦定义就不能被重新定义或者取消定义;
  - 常量的值只能是标量。
- 可以用函数 constant() 来读取常量的值。
- 用 get_defined_constants() 可以获得所有已定义的常量列表

5.3 预定义常量

- 详见P161表5-3

六、PHP中的运算符

- 5.1 算数运算符
- 5.2 字符串运算符
- 5.3 赋值运算符
- 5.4 比较运算符
- 5.5 逻辑运算符
- 5.6 位运算符
- 5.7 其他运算符
- 5.8 运算符的优先级

- 运算符是可以通过给出的一或多个值（用编程行话来说，表达式）来产生另一个值（因而整个结构成为一个表达式）的东西。所以可以认为函数或任何会返回一个值（例如print）的结构是运算符，而那些没有返回值的（例如echo）是别的东西。
- 有三种类型的运算符:
  - 一元运算符，只运算一个值，例如 !（取反运算符）或 ++（加一运算符）。
  - 二元运算符，有两个操作数，PHP支持的大多数运算符都是这种。
  - 三元运算符：？：。它应该被用来根据一个表达式在另两个表达式中选择一个，而不是用来在两个语句或者程序路线中选择。把整个三元表达式放在扩号里是个很好的主意。

5.1 算数运算符

运算符	意义	示例	结果
+	加法运算	$a+$b	$a和$b的和
-	减法/取负运算	$a-$b	$a和$b的差
*	乘法运算	$a*$b	$a和$b的积
/	除法运算	$a/$b	$a和$b的商
%	求余运算符（取模运算）	$a%$b	$a和$b的余数
++	累加1	$a++或++$a	$a的值加1
--	递减1	$a--或--$a	$a的值减1

5.2 字符串运算符

- 有两个字符串运算符:
  - 第一个是连接运算符（“.”），它返回其左右参数连接后的字符串。
  - 第二个是连接赋值运算符（“.=”），它将右边参数附加到左边的参数后。

```
<?php
        $a = "Hello ";
        $b = $a . “World!”;          // 现在$b的值： Hello World!

        $a = "Hello ";
        $a .= “World!”;              // 现在$a的值： Hello World!
?>
```

5.3 赋值运算符

- 基本的赋值运算符是“=”。一开始可能会以为它是“等于”，其实不是的。它实际上意味着把右边表达式的值赋给左边的运算数。

运算符	意义	示例
=	将一个值或表达式的结果赋给变量	$x=3
+=	将变量与所赋的值相加后的结果赋给该变量	$x+=3等价于$x=$x+3
-=	将变量与所赋的值相减后的结果赋给该变量	$x-=3等价于$x=$x-3
*=	将变量与所赋的值相乘后的结果赋给该变量	$x*=3等价于$x=$x*3
/=	将变量与所赋的值相除后的结果赋给该变量	$x/=3等价于$x=$x/3
%=	将变量与所赋的值求模后的结果赋给该变量	$x%=3等价于$x=$x%3
.=	将变量与所赋的值相连后的结果赋给该变量	$x.="H"等价于$x=$x."H"

5.4 比较运算符

运算符	描述	说明	示例
>	大于	当左边大于右边时返回true，否则返回false	$a>$b
<	小于	当左边小于右边时返回true，否则返回false	$a<$b
>=	大于等于	当左边大于等于右边时返回true，否则false	$a>=$b
<=	小于等于	当左边小于等于右边时返回true，否则false	$a<=$b
==	等于	两边操作数的值相等时返回true，否则false	$a==$b
===	全等于	两边值相等并且类型相等返回true，否则false	$a===$b
<>或!=	不等于	两边值不等时返回true，否则返回false	$a<>$b $a!=$b
!==	非全等于	两边值或类型有不等时返回true，否则false	$a!==$b

5.5 逻辑运算符

运算符	描述	说明	示例
and或&&	逻辑与	当两边操作数都为true时，返回true，否则返回false	$a and $b $a && $b
or或||	逻辑或	当两边操作数都为false时，返回false，否则返回true	$a or $b $a || $b
not或!	逻辑非	当操作数为true时返回false，否则返回true	not $b !$b
xor	逻辑异或	当两边操作数只有一个为true时，返回true，否则返回false	$a xor $b

5.6 位运算符

运算符	描述	说明	示例
&	按位与	只有参与运算的两位都为1时，运算结果才为1，否则为0.	$a & $b
|	按位或	只有参与运算的两位都为0时，运算结果才为0，否则为1.	$a | $b
^	按位异或	只有参与运算的两位不同，运算结果才为1，否则为0.	^$b
~	按位非	将用二进制表示的操作数中的1变成0，0变成1.	~ $a
<<	左移	将左边的操作数在内存中的二进制数据右移右边操作数指定的位数，右边移空的部分补上0	$a<<$b
>>	右移	将左边的操作数在内存中的二进制数据左移右边操作数指定的位数，左边移空的部分补上0	$a>>$b

5.7 其他运算符

运算符	描述	示例
? :	三元运算符，可以提供简单的逻辑判断。	$a<$b?$c=1:$c=0
``	反引号(` `)是执行运算符，PHP将尝试将反引号中的内容作外壳命令来执行，并将其输入信息返回	$a=`ls -al`
@	错误控制运算符，当将其放置在一个PHP表达式之前，该表达式可能产生的任何错误信息都被忽略掉。	@表达式
=>	数组下标指定符号，通过此符号指定数组的键与值。	键=>值
->	对象成员访问符号，访问对象中的成员属性或成员方法。	对象->成员
instanceof	类型运算符，用来测定一个给定的对象是否来自指定的对象类。	对象 instanceof 类名

5.8 运算符的优先级

- 详见P173的表5-11

七、表达式

- 表达式是 PHP 最重要的基石。在 PHP 中，几乎所写的任何东西都是一个表达式。简单但却最精确的定义一个表达式的方式就是“任何有值的东西”。
- 最基本的表达式形式是常量和变量。当键入“$a = 5”。
- 稍微复杂的表达式例子就是函数。
- 其他还有运算符与操作数构成的也成表达式。如：比较表达式$a>5、$a==5。

谢谢！