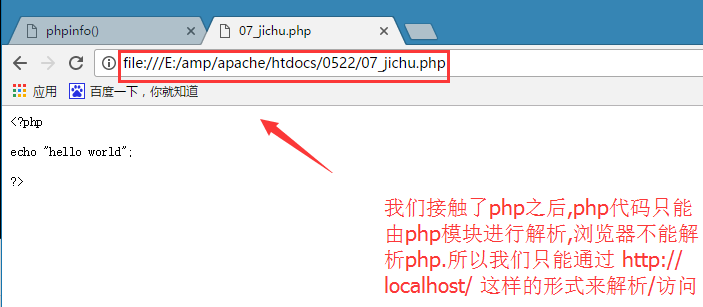
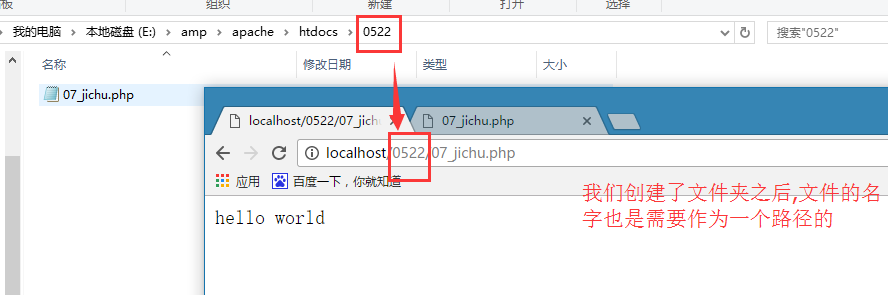
昨日回顾



# 第7讲 基本PHP语法





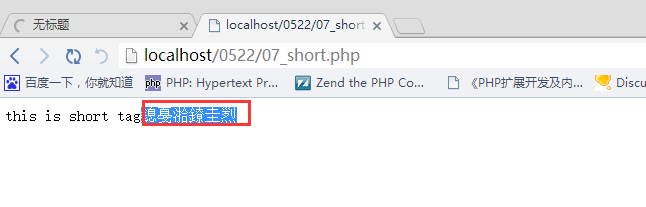
## PHP标签

标记分区PHP代码从指定位置开始解析到指定位置结束解析.

为什么要有php标签?

* 当html和php代码进行混编的时候,后缀名一定是php
* 混编的时候,php引擎也不是所有的代码都执行,他只执行属于php代码的那一部分,我们如何区分哪些是php代码呢?就必须依靠php的标签来进行区分!

中文乱码问题:



有两种解决方式:

第一种方式: 改变浏览器的编码格式

第二种方式: 在php的代码中添加一个header ;

header(“content-type:text/html;charset=utf-8”);

### <?php ?>

这个标签是php的标准标签,也是使用最广泛的标签,也是推荐大家使用的标签

标签是以 <?php 开始

以 ?> 结束



等大家以后学项目的时候,再使用: 如果 脚本最后是以php代码结尾的, 结尾的 ?> 可以省略

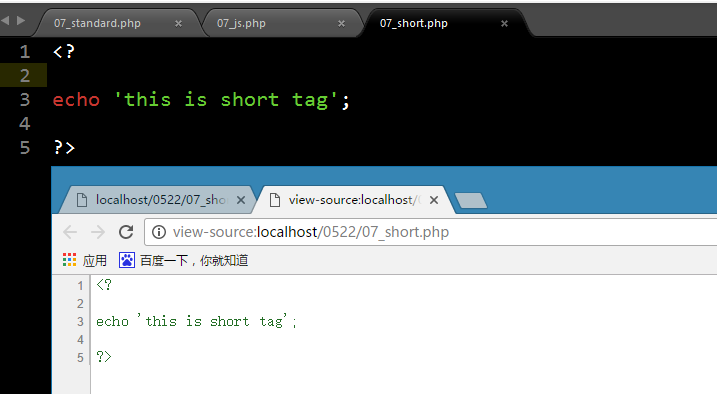
### <script language=”php”></script>



这种标签也是标准标签,但是第一:容易和js混淆 第二:麻烦,懒得写!

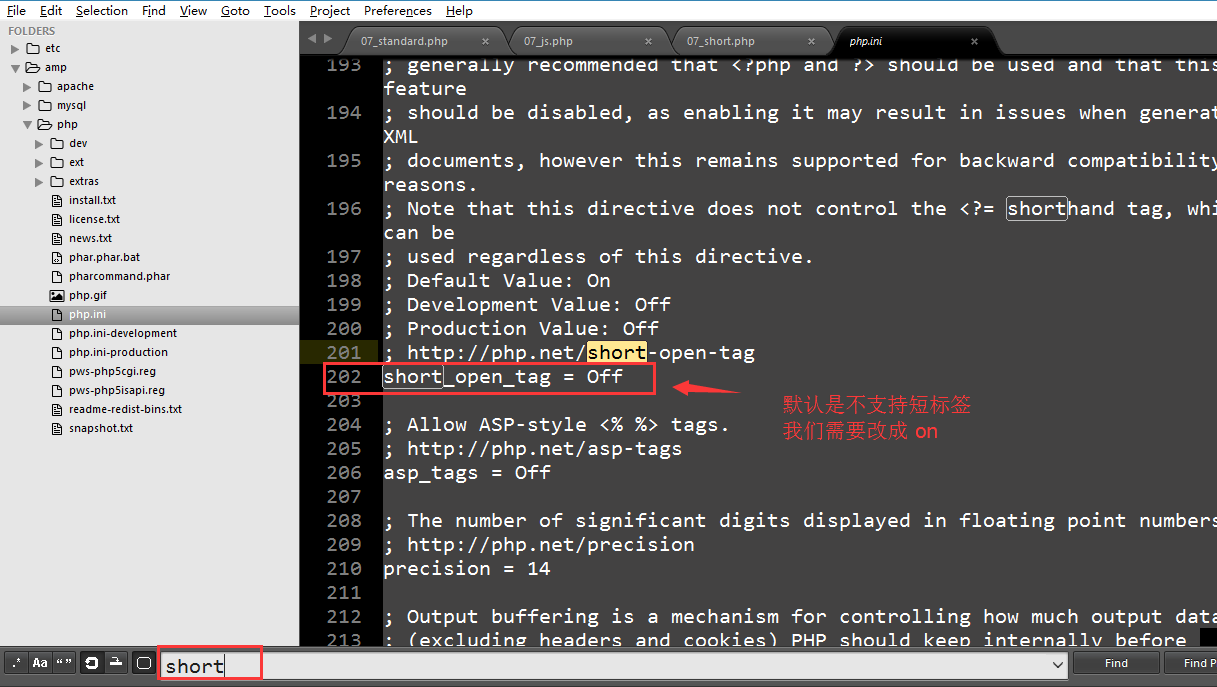
这种标签大家认识就可以了.

### <? ?>



默认情况下php不支持短标签,我们需要修改php.ini配置文件来让php模块解析短标签

修改php.ini文件,我们需要找到php.ini然后搜索 short



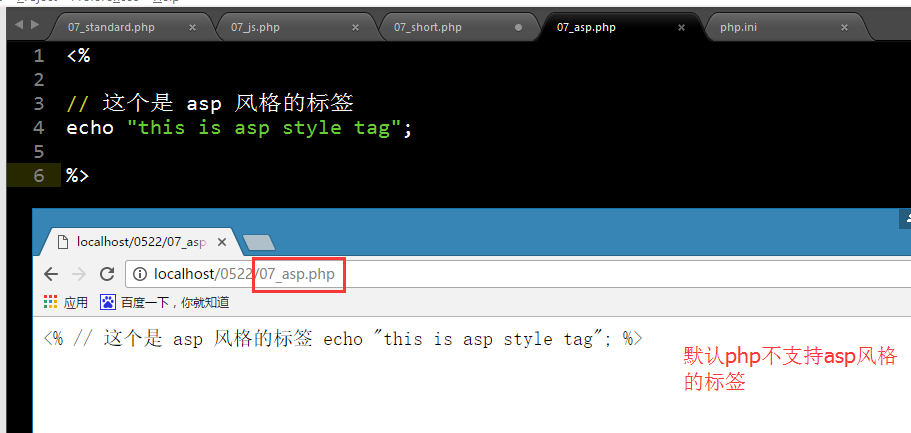
保存之后,重启apache



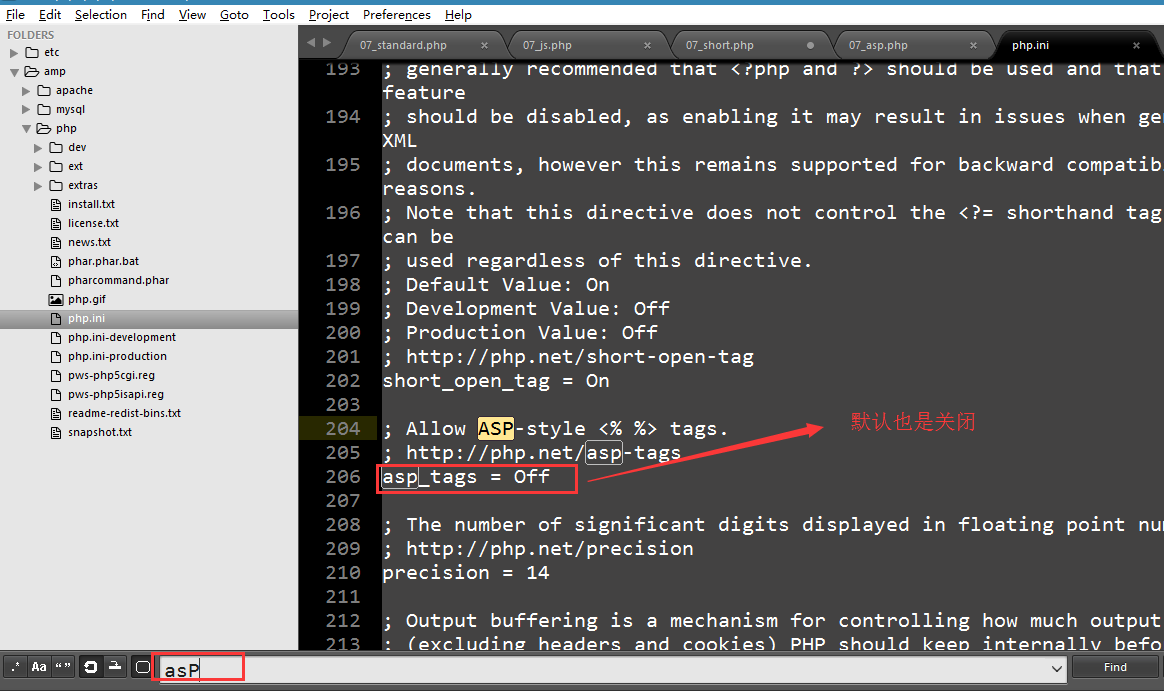
不推荐大家使用,认识就可以了!

### <% %>

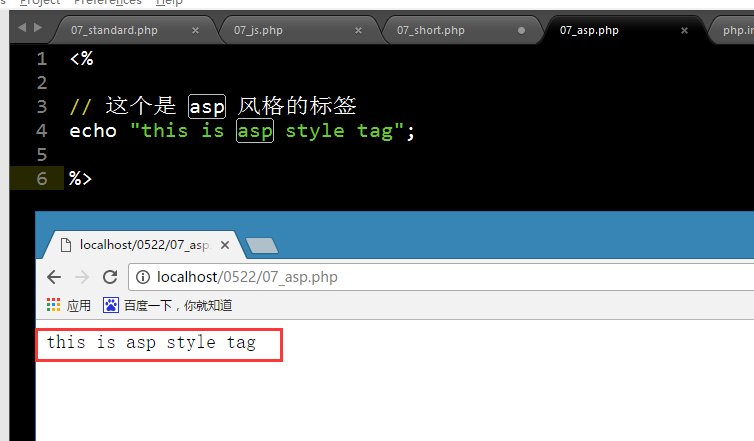
asp风格的标签



我们在php.ini中搜索 asp 或者是 在 short\_open\_tag 的下边



修改之后重启apache

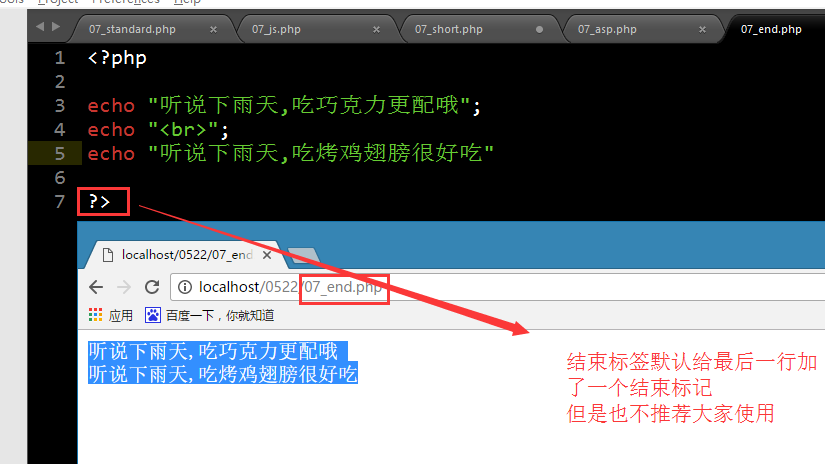


## 语句结束符

要求是每条语句都要以分号(;)结束



有一个特例,最后一个语句可以没有分号.但是也不推荐



## 注释

### 行注释

// 行注释

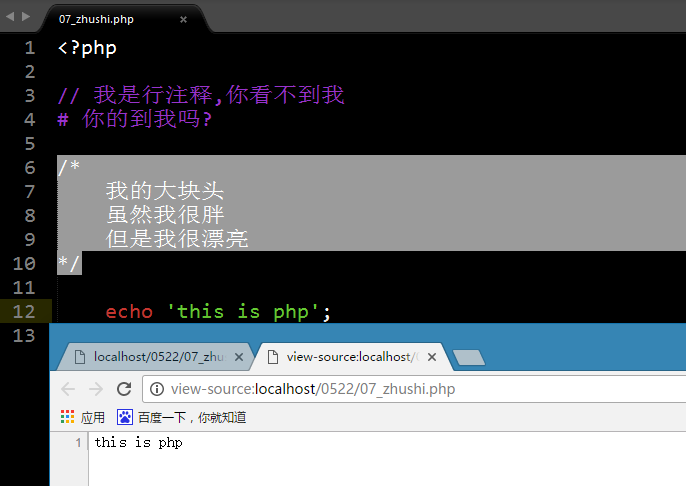
# 这个也是注释

### 块注释

/\*

块注释

\*/



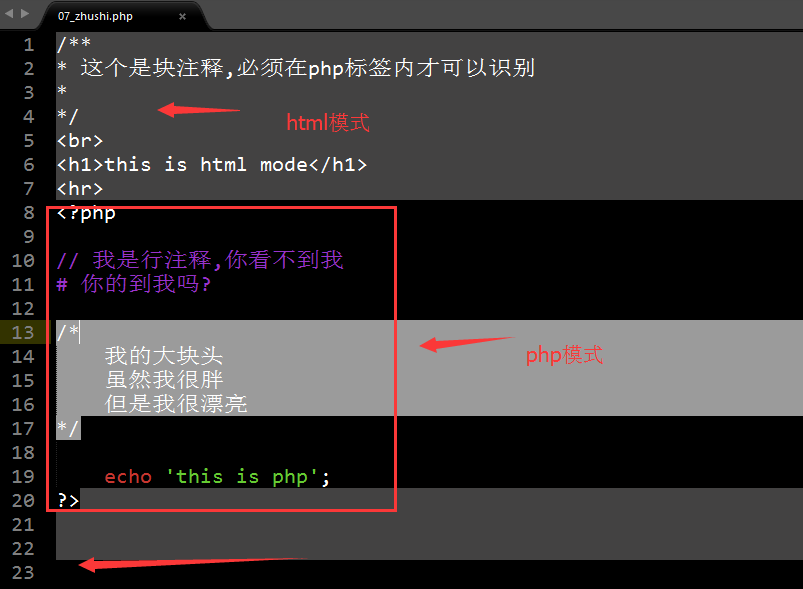
在php脚本中只存在两种模式:

php模式: php标签内就是 php模式

html模式: php标签外就是html模式

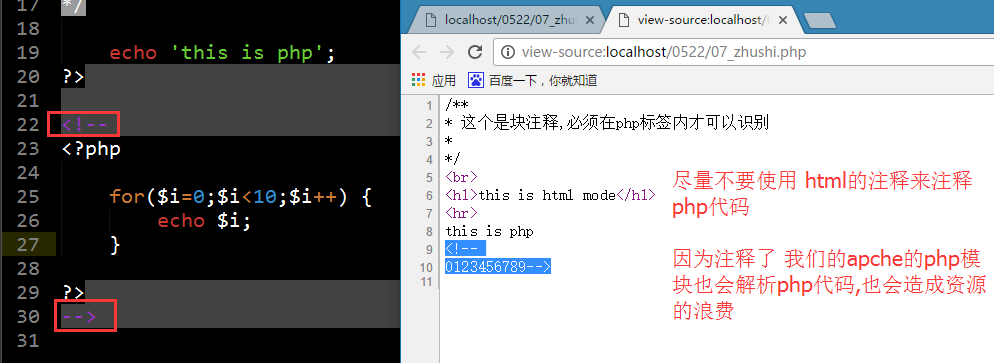
php模式里的代码会被php模块解析

html模块的代码不会被解析,会被apache原样返回,但是会被浏览器解析



### 注意事项

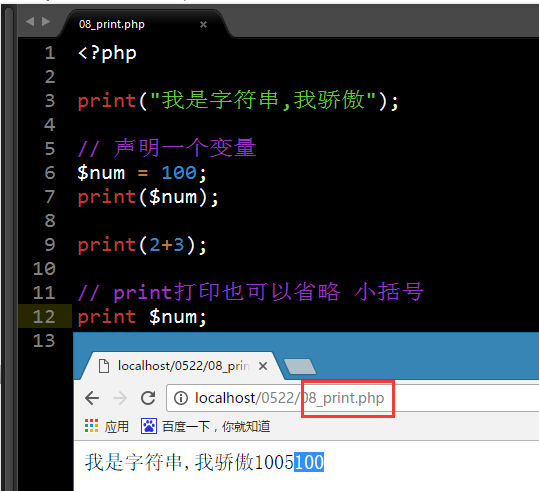
* php注释只有在php代码(php模式)中才能生效
* 千万不要使用html的注释去注释php语句



# 第8讲 常见的输出语句

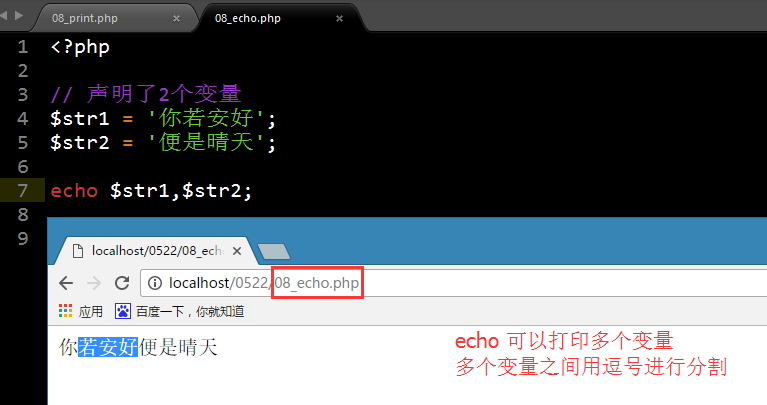
## print语句

print语句可以打印 变量,表达式,还有字符串.

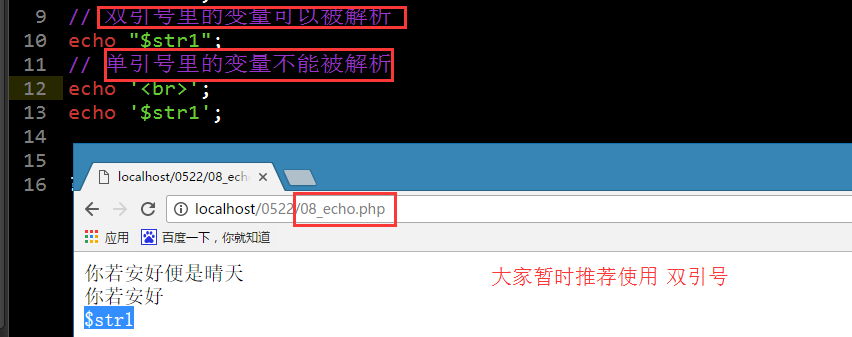


print不可以打印2个变量.

## echo语句



简要说一下 单引号字符串和双引号字符串的一个区别:



echo不能打印数组

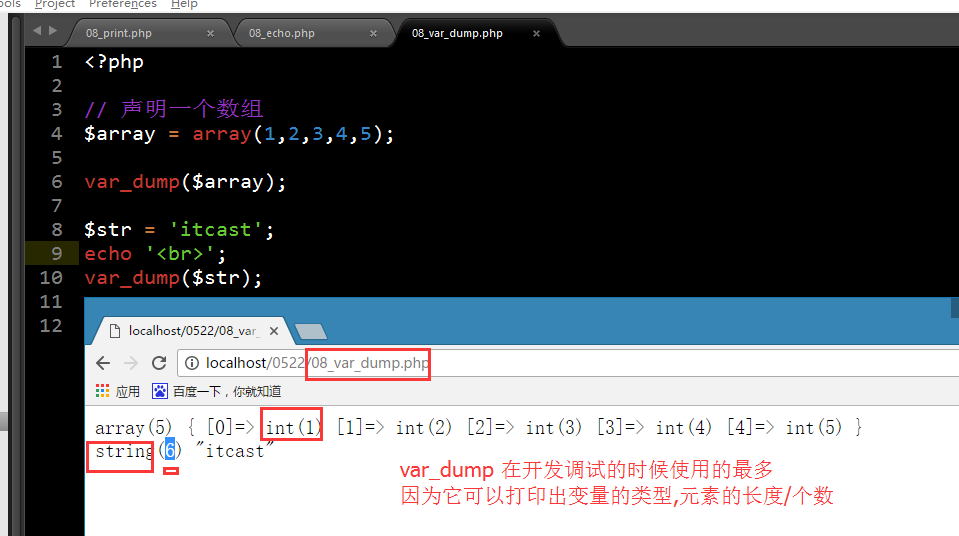


## var\_dump语句

var\_dump 可以打印数组



var\_dump 可以打印出比较详细的信息



## printf语句

比较擅长输出由静态文本和其他变量所组成的混合产物

基本语法规则:

printf语句分为2部分:

前半部分是用双引号括起来的字符串,如果字符串中需要出现变量的时候,就用一个变量占位符来表示,占位符是以%开头,而且不同的占位符表示不同的输出类型

后半部分是变量列表,而且变量列表和前面的占位符是一一对应的

常见的变量占位符:

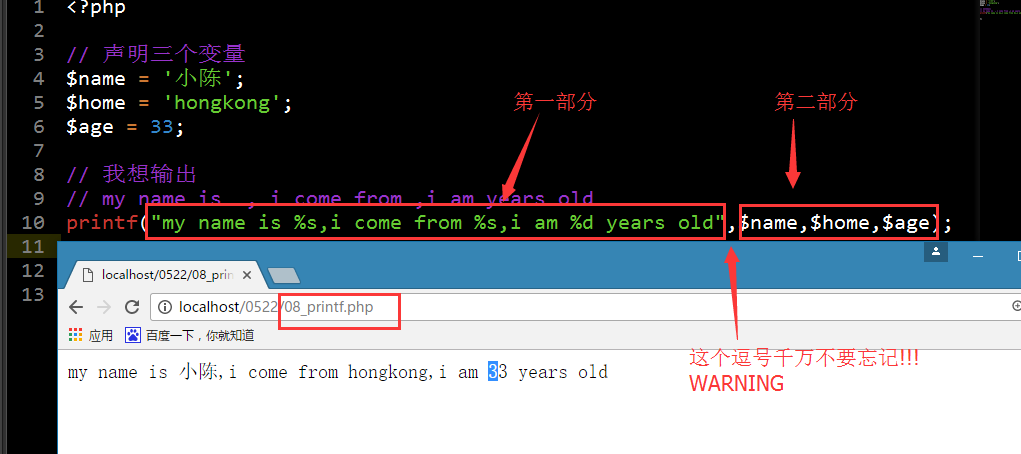
%d：将后面的变量认为是一个整数，并显示为有符号十进制数

%o：将后面的变量认为是一个整数，并显示为无符号八进制数

%x：将后面的变量认为是一个整数，并显示为无符号十六进制数

%f：将后面的变量认为是一个浮点数，并显示为浮点数

%s：将后面的变量认为是一个字符串，并显示为字符串



## print\_r语句



# 第9讲 变量

## 概念

### 什么是量

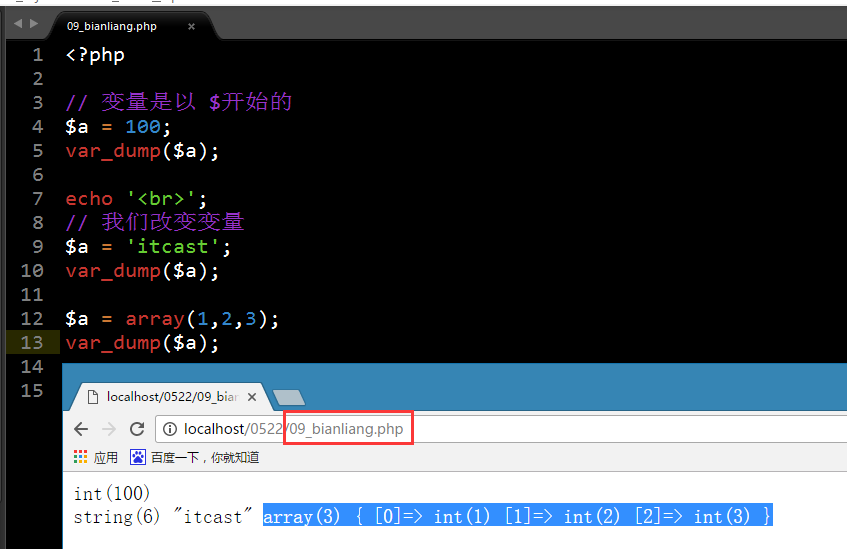
量其实就是 数据.

### 什么是变量

变量就是 在一个脚本周期内,其值可以改变的量.

### 什么叫做一个脚本周期

一个脚本周期就是 当apache将请求交给php模块解析开始,到 php模块解析完成为止,这个就是一个 脚本周期



## 变量的组成

### 变量组成

$a = 100;

分2部分 $a 和 100

$a 是变量名

100 是变量的值

变量名 和变量值 之间是啥关系呢? 引用和被引用的关系

在内存中是 变量名 引用 变量值 ; 变量值 被 变量名所引用;

### 内存组成

堆区：空间大，但是速度慢，一般存放数组、对象等复杂的数据类型！

栈区：空间小，但是速度快，一般存所占空间不变的数据，比如变量名，常量名、函数名、数组名！

代码区：就是存函数或对象的方法等代码

全局数据区：主要存放变量、常量、静态变量！



### 深度剖析一下$a = 100;



① 在全局区的变量区 分配一块内存空间用户保存100这个值,同时这个内存空间有一个地址,这个地址举例为1000 1000

② 在栈区开辟一块内存空间,$a

③ $a用于保存 变量区的地址(1000 1000),同时变量名对变量区的数据有一个引用关系

### 深度剖析一下echo $a

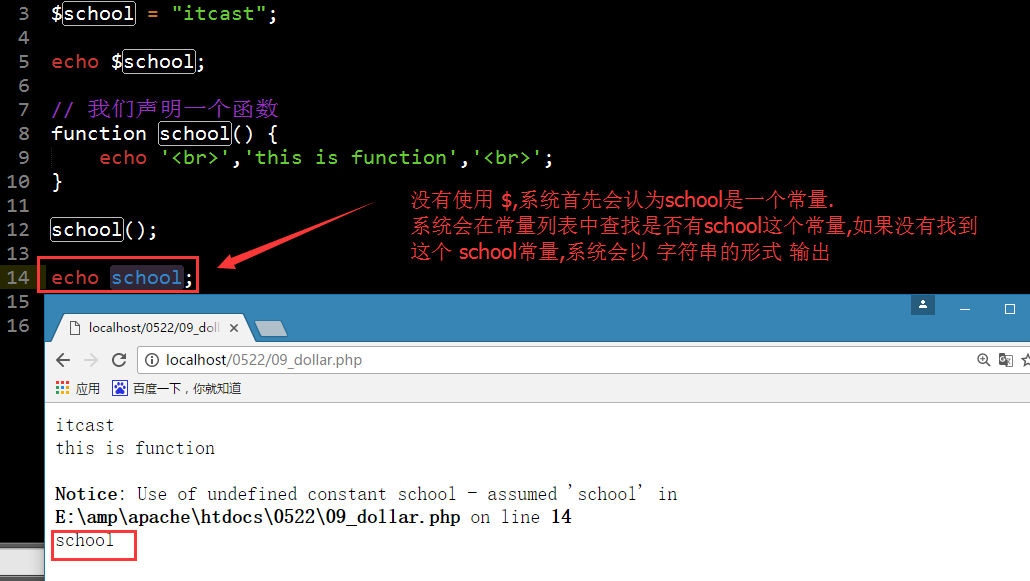
① 在栈区找到变量$a

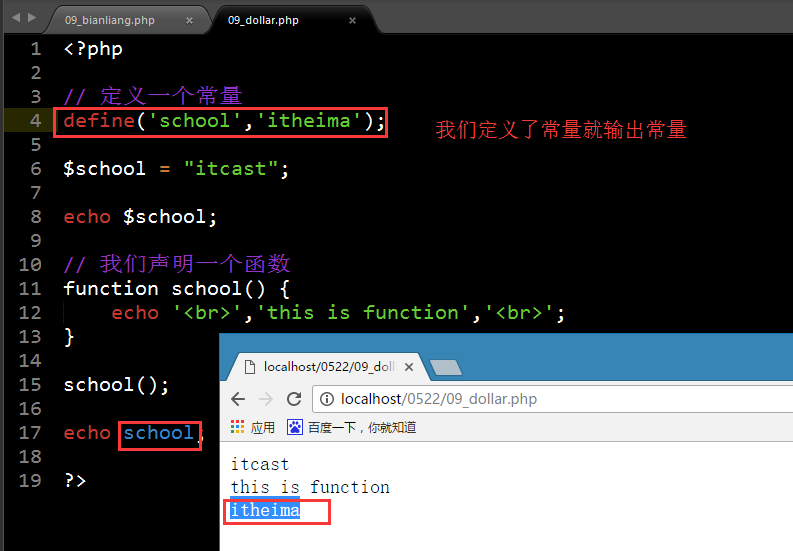
② 根据$a存在的地址,找到变量区

③ 找到变量区的地址之后,就找到了对应的数值,然后输出出来

## 变量的基本语法:$

$是用于声明后边为一个变量. $其实并不属于变量名的一部分



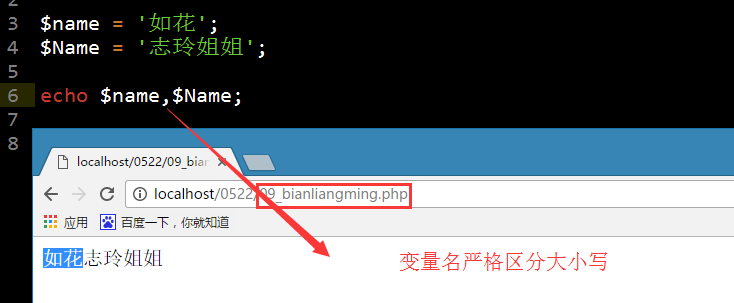


## 变量的命名规则

变量命名遵循标识符的命名规则。标识符用来标识信息的名称。如变量名，常量名，函数名等

标识符的命名规则：

* 由字母数字下划线等多个字符组成，以字母或者下划线开头。
* 变量命名严格区分大小写。



* 不使用系统关键字
* 建议命名使用英文单词，见名知意。

多个单词组成的变量：StudentName

驼峰命名法

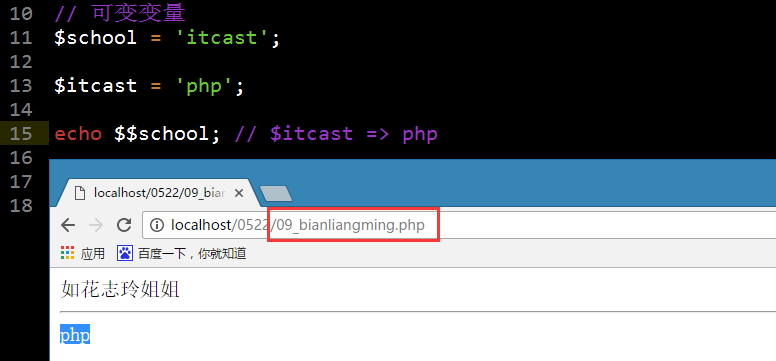
* 第一个单词首字母小写，其余单词的首字母大写（小驼峰）举例: studentName
* 每个单词的首字母都大写（大驼峰）,举例: StudentName
* 使用下划线进行分割: 举例 student\_name

类名一般使用 大驼峰. 函数一般使用 小驼峰 数据库一般使用 下划线

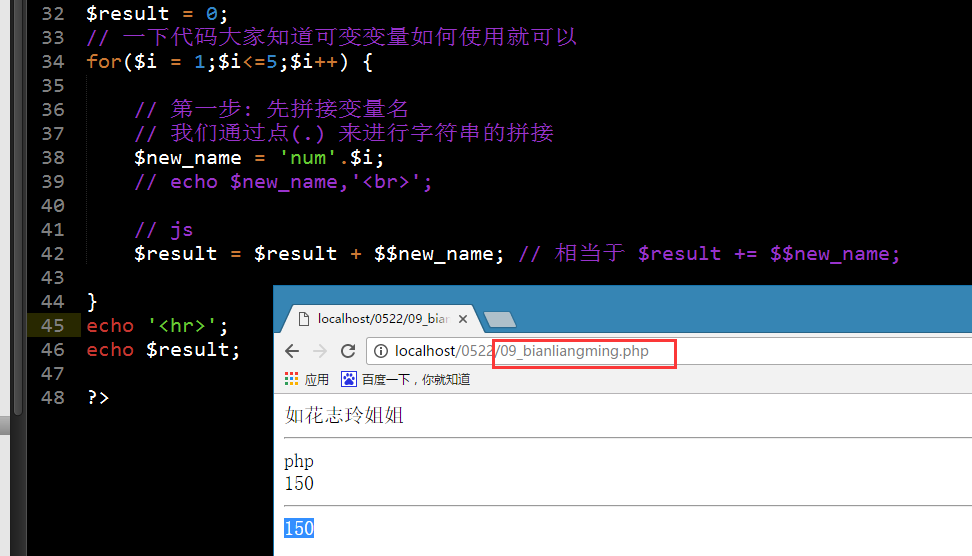
## 可变变量

PHP支持动态改变变量名。即可变变量的名称由变量的值组成

我们的变量名也是可以作为一个变量的,就叫做 可变变量!!!



以后还会接触 可变函数, 可变类等可变类型

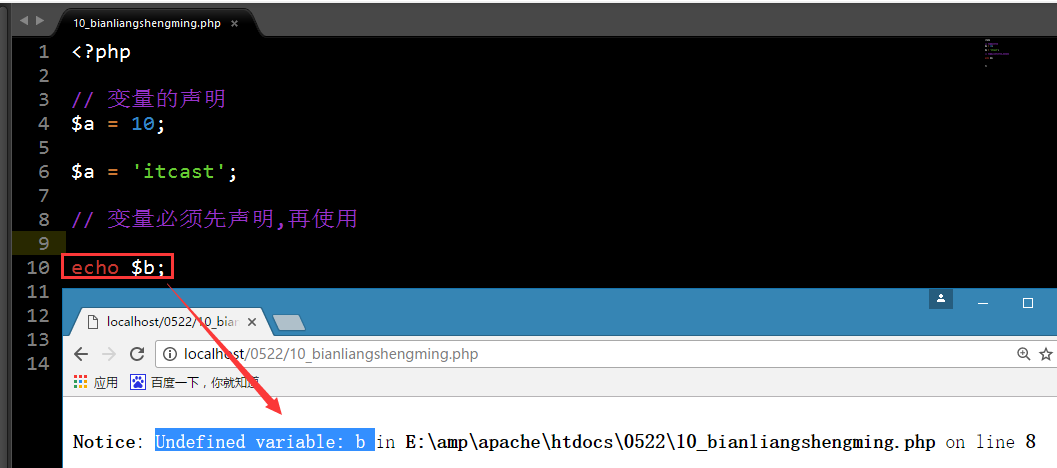


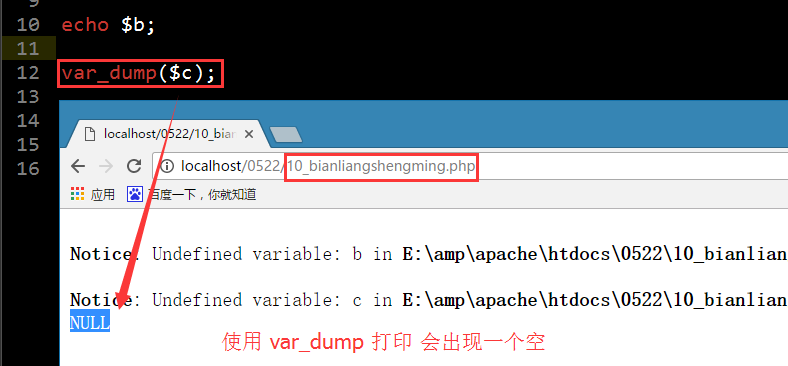
# 第10讲 变量的基本操作

量就是数据,数据的操作无外乎就是 增删改查

## 变量声明(增)

* php不需要显示的声明变量（比如c语言中的int a；），变量的声明一般和赋值同时进行，也叫作变量的初始化！
* 变量可以是任意的类型，而不需要指定具体的类型，或者说可以放置任意数据类型的值！
* 虽然php变量不需要显示的声明，但并不代表可以直接使用一个完成没有初始化的变量！

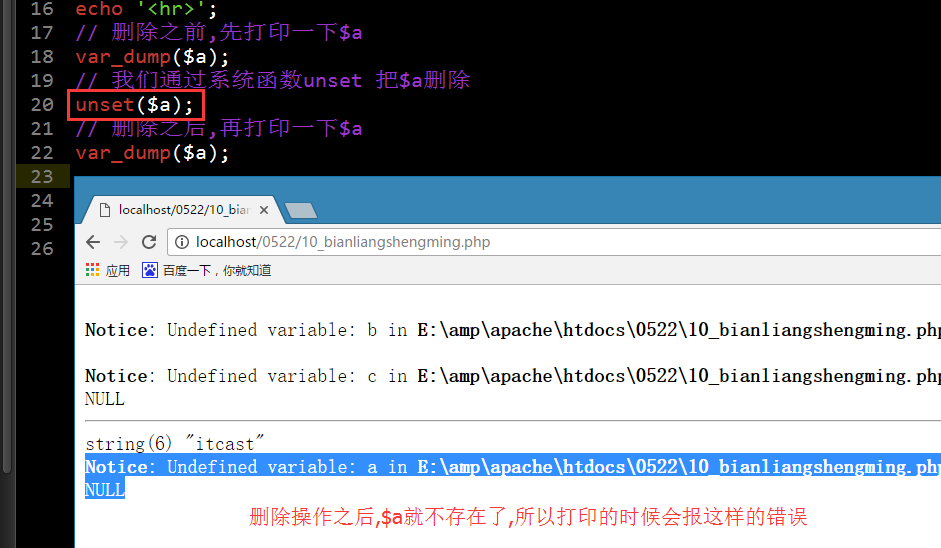




* 虽然php变量不需要指定类型，但不并代表该变量没有数据类型，变量的数据类型就是其保存的值的类型！我们是通过变量的值来确定变量的类型！

## 删除变量(删)

删除操作是采用系统的函数 unset($变量名);



深度剖析一下 unset($a) 在内存中做了那些工作???



① 在栈区把$a这个变量移除

② 把栈区$a对 变量区的引用移除

全局区的变量区数据并不是由unset进行回收的,而是由php的核心解释器负责管理和回收的.

## 变量的修改(改)

直接赋新值就可以!!!

## 获取变量的值(查)

echo ,var\_dump 来获取变量的值

# 第11讲 变量间的传值

将一个变量传递给另外一个变量

$a = 100;

$b = $a;

## 值传递

$a = 100;

$b = $a; 这里 $b = $a是值传递;

值传递可以理解成为 复制 传递;



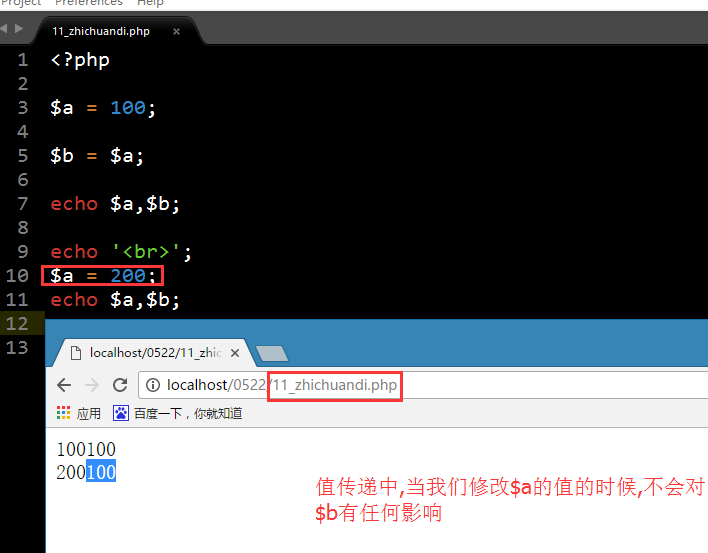
① 在全局区的变量区 重新开辟一块内存空间,它的地址是 1000 1001.但是内存的值还是原来变量的值 100.

② 在栈区重新开辟一块内存空间$b

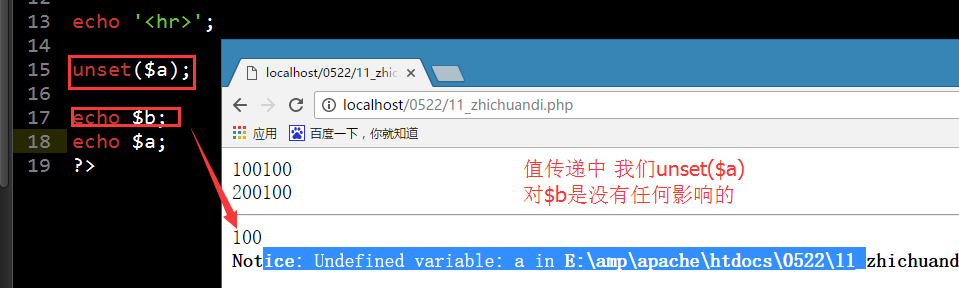
③ 栈区的$b所存储的是 新开辟的变量区的空间地址1000 1001,同时$b对变量区有一个引用.

看图说话?

问题一: 当我修改 $a的值的时候,$b有没有影响?



问题二: 当我unset($a)的时候,$b还有没有值?



## 引用传递

引用传递，就是将一个变量的内存空间地址赋值给另外一个变量！

问题: 我们如何获取到一个变量的内存空间地址.

我们使用 & 取地址符.

$a = 100;

$b = & $a;



引用传递

($b = &$a)

将一个变量的内存空间地址赋值给另外一个变量

我们通过求地址运算符(&)来得到变量的地址空间

剖析$b = &$a

获得a变量所在的内存空间的地址

在栈区里面开辟一个内存空间,存放变量名b

将a变量所在的内存空间的地址复制一份给b

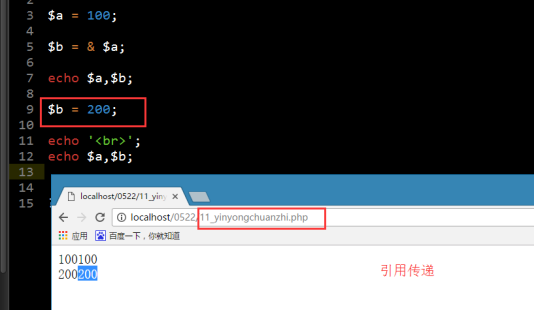
①获得a变量所在的内存空间的地址

② 在栈区重新开辟一块内存空间$b

③ 在$b的栈区保存 找到的那个地址 1000 1000,同时$b对变量有一个引用关系

看图说话:

问题一: 当$b的值改变之后,$a,有没有影响???



问题二: 当 unset($a)的时候, 对$b有没有影响???

unset所做的操作就是 2步,将栈区的变量移除,第二个:将引用移除.

# 第12讲 预定义变量

## 概念

预定义变量就是系统已经为我们定义好的变量,我们拿来就用就可以.

我们有9个预定义变量.

今天我们先接触1个, 3天后会接触另外1个

$\_SERVER:获取服务器端的一些信息

$\_GET,$\_POST,$\_REQUEST: 表单提交

$\_COOKIE,$\_SESSION:会话

$\_FILES:文件上传相关.

$\_ENV: 很少使用,和php的命令行相关

$GLOBALS:变量的作用域有关

预定义变量都是 数组.

### $\_SERVER

是一个数组，保存执行环境及服务器的信息。

### $\_GET

接收表单GET方式传递的数据

get 传递数据的时候,一般是在浏览器中有显示

get不安全

get有一般传输 4K

### $\_POST

接收表单POST方式传递的数据

post一般用于上传图片/文件等二进制的数据

post比较安全

post传输的数据量比较大

### $\_REQUEST

接收表单传递的数据（GET方式及POST方式）

request = get + post

# 第13讲 常量

## 概念

常量 的值不会发生改变.

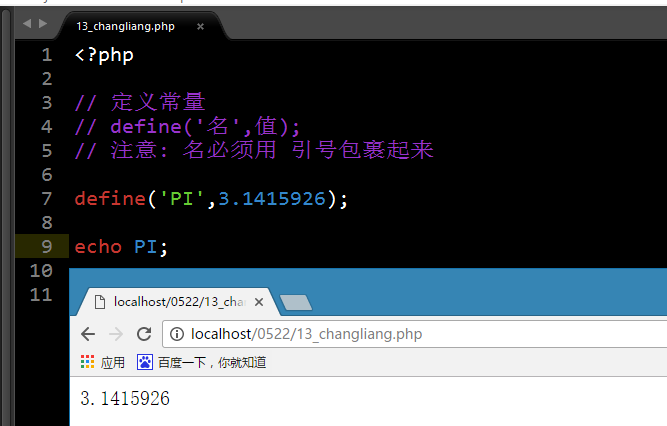
特点: 就是值不会发生改变

## 常量的定义

### 语法

使用系统函数define(名,值);

名一定要用引号包裹起来



### 深度剖析一下define(‘PI’,3.14)



① 在全局区的常量区分配一块内存空间,用于存放3.1415926这个值,同时有一个地址为 1000 1111

② 在栈区开辟一块内存空间 PI

③ 栈区用于存放 常量区的地址 1000 1111,同时 对常量区有一个引用关系

常量区在内存中 只允许写入一次!!!

以后我们学习面向对象的时候, 面向对象有一个关键字 const 也可以定义常量

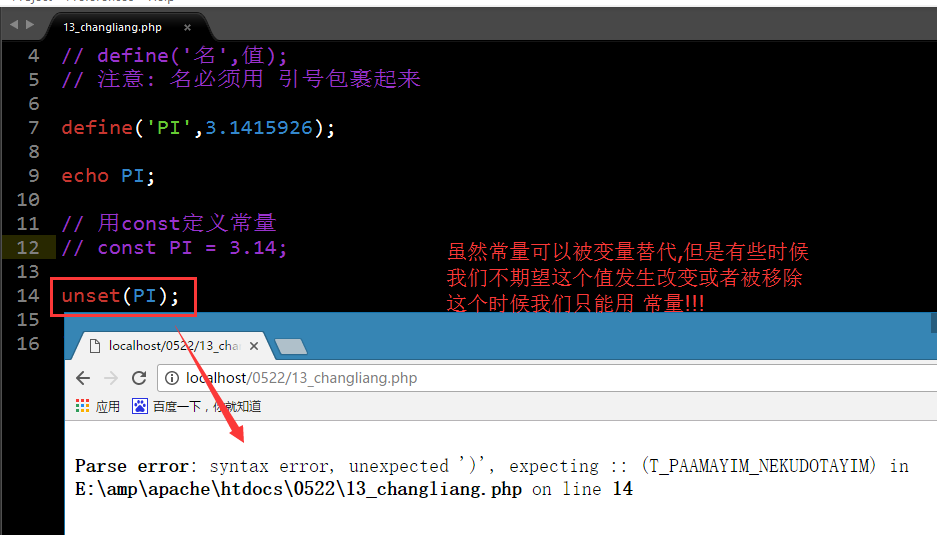
const 常量名 = 值;

注意: 常量名千万不要有$



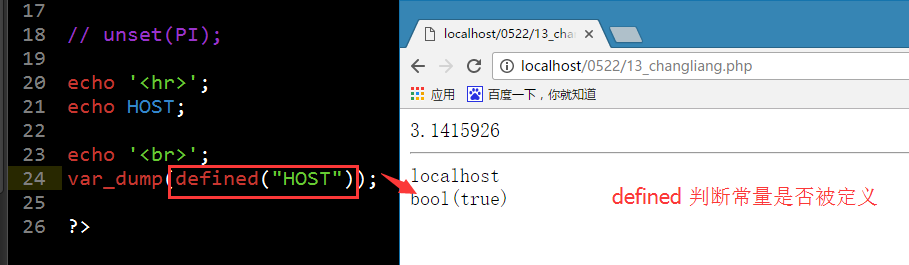
## 常量的语法意义

凡是使用常量的地方都可以用变量来代替,但是常量的语法意义主要是用于某个值不需要变化的时候，通常用来规范数据，保证数据不被修改！



## 判断常量是否存在

defined(名),返回一个bool值



## 常量的命名规则

① 常量千万不要使用 $

② 常量一般都使用 大写,为的就是和 变量进行区分

③ 命名规则和变量类似

# 第14讲 预定义常量

## 概念

预定义常量:系统预先定义好的常量

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 |

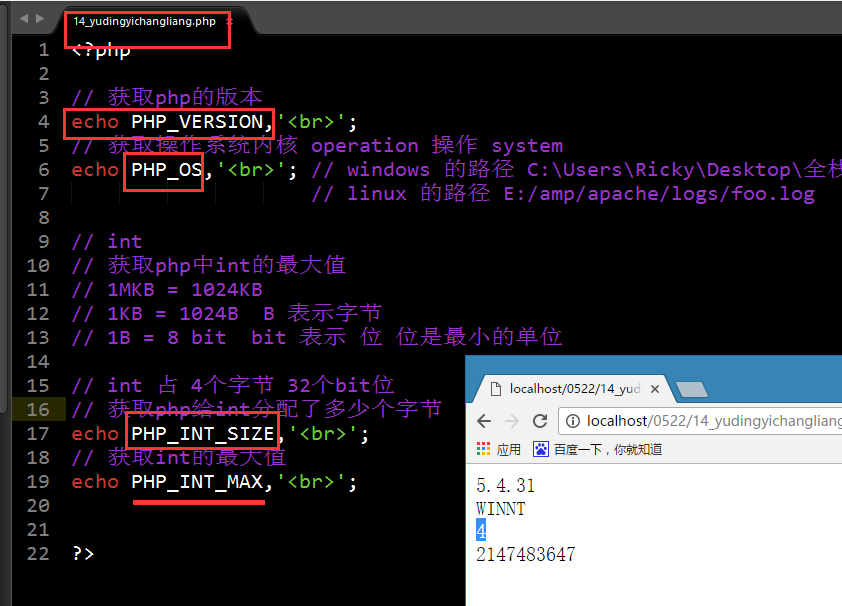
8个 bit 为一个 字节 1B

## 常用预定义常量

PHP\_OS:输出操作系统的内核

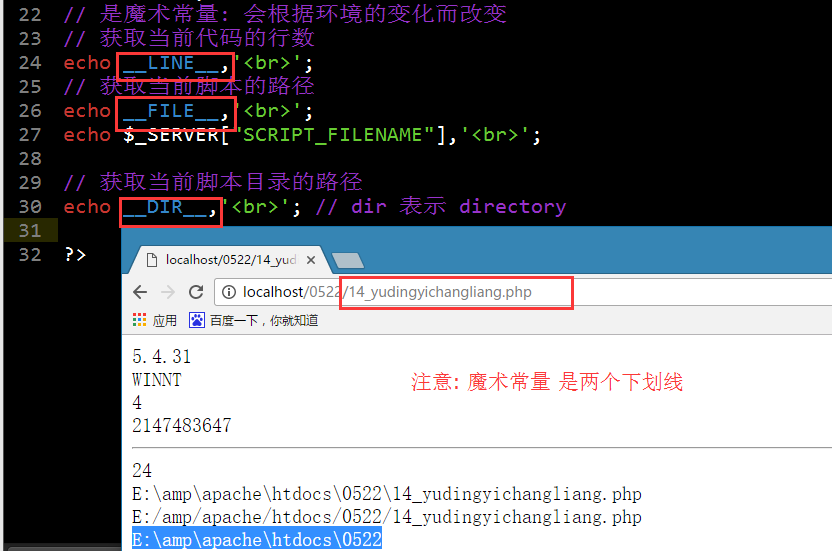
PHP\_VERSION:输出php的版本号

PHP\_INT\_MAX:输出int的最大数  
PHP\_INT\_SIZE:输出int类型所占的字节数



\_\_FILE\_\_:输出当前文件所在的路径  
\_\_DIR\_\_:输出当前文件所在的目录

\_\_LINE\_\_:输出当前的代码所在的行数



以后面向对象的时候还会学习其他的预定义常量



魔术常量. 魔术常量会根据环境的变化而改变.那它真的就是常量吗???

计算机 能够识别的 二进制 010101

针对于 010101 汇编语言,汇编语言是最接近于机器语言 010101 Add

汇编也是晦涩难懂,也很难写.

高级语言. c java.都是高级语言, 高级语言需要进行一个编译的过程,编译的过程就是将高级语言转换成 计算机能够识别的二进制语言.

编译型语言: c,java

解释型语言: html.

php是解释型语言.

php为了提高执行效率,会先进行一次预编译,把脚本文件预编译成一个opcode文件,这个opcode文件能够被php的核心解释器识别.

# 第15讲 数据类型简介

## 前言

编程的目的就是为了更好的去改造现实世界，首先要模拟现实世界！

著名的计算机科学家 沃思曾经提出了一个公式：

**程序 = 数据结构 + 算法**

**数据结构**：描述数据的类型及其内在的组织形式（在内存中的形式）

**算法**：解决问题的方法、步骤（往往决定事情的成败）

其实，现实生活中，到处都是程序！

厨师 = 原材料（油盐酱醋鸡鸭鱼蛋等）+ 操作步骤（菜谱）

音乐家 = 原材料（音符）+ 操作步骤（乐谱）

工人 = 原材料（沙子水泥钢筋涂料等）+ 操作步骤（图纸）

php程序 = php数据结构 + 算法

php数据结构总共有8种.

整形 – int

浮点型 – float

字符串 –string

布尔型 – bool

数组 – array

对象 – object

空 – null

资源型 – resource

## 标量类型(简单类型)

整形 – int

浮点型 – float

字符串 –string

布尔型 – bool

## 复合类型

容器

数组 – array

对象 – object

## 特殊类型

空 – null

资源型 – resource

空更多的是一种描述

资源型就是获取外界的数据