web服務器軟件: apache, tomcat(java), iis 等…

NodeJS不需要web服務器軟件也可能做服務端。

IP internet protocol 網絡之間的互聯協議。ip是唯一的。

域名Domain Name eg. [www.google.com](http://www.google.com)

特殊ip : 127.0.0.1

特殊域名 localhost

DNS 域名系統domain name system. 經由domain name去取得主機的ip

eg: [www.goole.com](http://www.goole.com) 的域名,取得的ip是 8.8.8.8。

端口port

虛擬端口 默認80不可用。

物理端口: 接口, 接線用,rj45接口。

動態網址php 比 靜態網址html 多了一層 數據庫



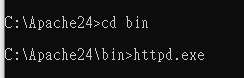
我們利用模板引擎和數據庫,nodeJS 也可以使靜態網址html變成動態。

啟動apache server

Open a command prompt window and cd to the \Apace24\bin folder on the drive you unzipped the zip file to.

To Start Apache in the command prompt type:

*httpd.exe*



<http://127.0.0.1:80>

localhost 都可以去到apache page 就說明服務器啟動成功!

c:/Apache24



httpd.exe

1. 服務器進程: 運行之後才能夠工作
2. 用來查看apache具有哪些功能以及配置文件是否有錯。
   1. httpd –M查看使用模塊(static 靜態加載 apache啟動就加載好了,全局的,可以直接使用 shared 動態加載,使用時才會加載)
   2. httpd –t 驗查配置文件是否有效

httpd.conf: 配置文件下

1.DocumentRoot 根目錄

DocumentRoot "${SRVROOT}/htdocs"

2.ServerName

ServerName localhost:80

3.listen 端口號port

#Listen 12.34.56.78:80

Listen 80

如果修改了Config文件,要重啟apache服務器才有效。

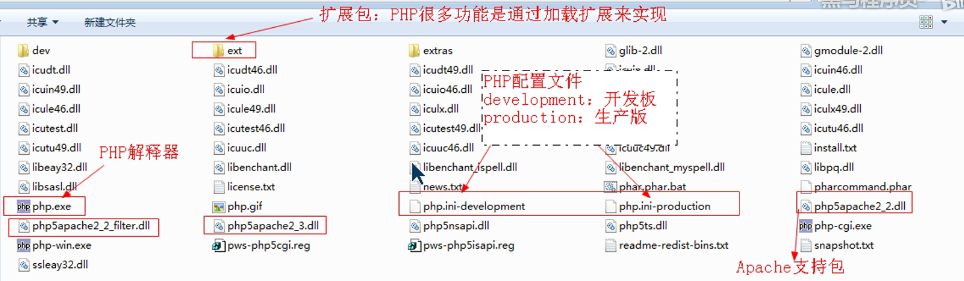
4.本機DNS域名

C:\Windows\System32\drivers\etc 內的hosts 文件

修改可以修本機的DNS域名

php 安裝

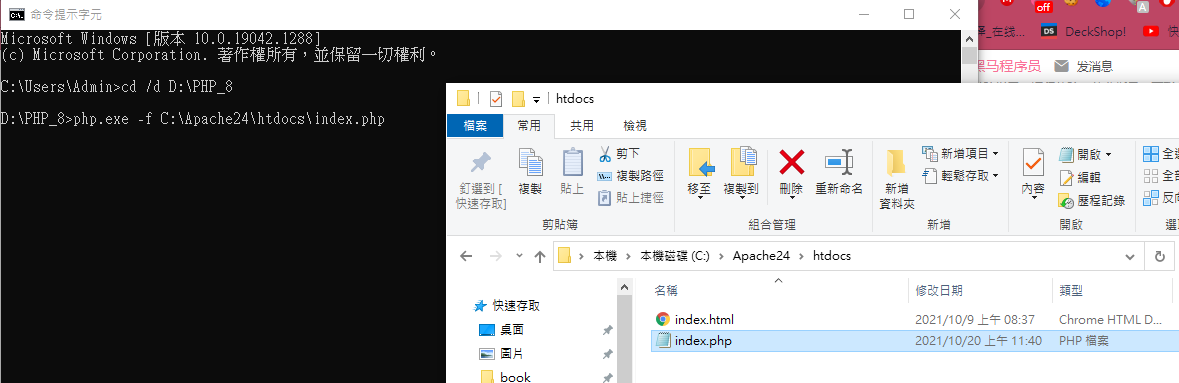
php底層是C語言 VC是編譯環境 我們php8是vc16。

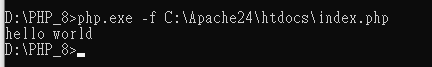


php.exe可以解析php代碼,轉變成HTML代碼, 讓瀏覽器解析。

1. 通過CMD控制器進入到php.exe的目錄。
2. 通過php.exe命令來指定要解析的php文件腳本,

php.exe -f PHP(文件路徑)





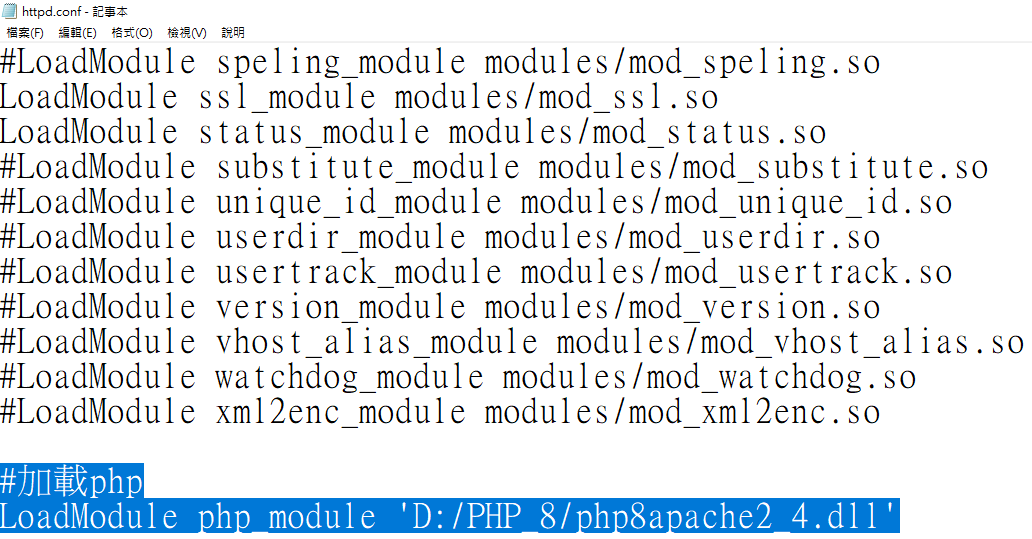
配置apache加載php模塊, 使瀏覽器可以讀取apche加載後的php文件。

1. 在apache 配置文件中加載對應的php模塊。

LoadModule php\_module (php 所提供的模塊鏈接所在路徑)

(php8apache2\_4.dll)

所提供的模塊鏈接所在路徑: D:\PHP\_8\php8apache2\_4.dll



鈄槓方向要小心。

之後用httpd –t 測試一下加載php是否成功。

之後用httpd –M 測試一下加載php是否成功。

會看到php\_module (shared) shared是需要時會加載使用。

version問題:

在對應的file的路徑下:

php –version 查看php版本

httpd –vserion 查看apache版本

如果一個32位元,一個64位元, 就不能加載了。

x86等於32位元

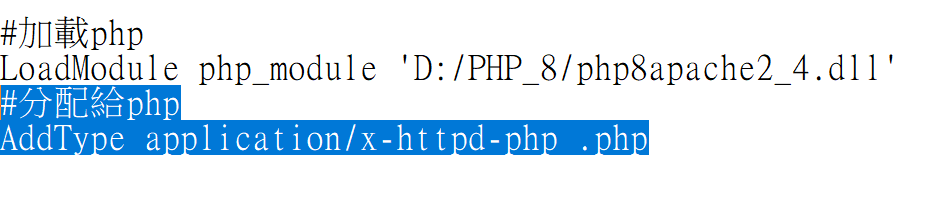
由於我們是32位元的apache,所以我們去下載一個32位元的php才能加載。

1. apache分配工作給php模塊 的設置

文件後綴是.php就分配給php模塊。

寫到httpd.conf文件內

AddType application/x-httpd-php .php

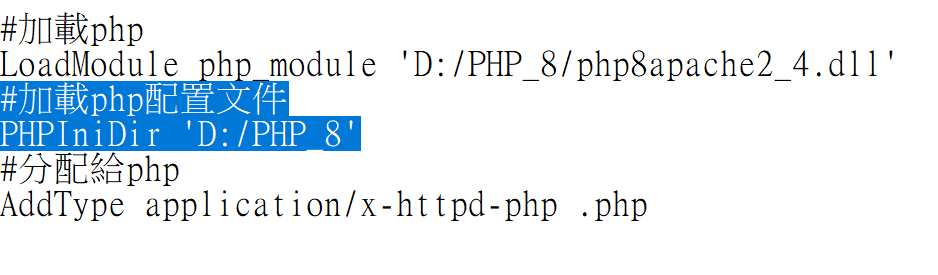


1. 將php的配置文件加載到apache配置文件中(共同生效)

3.1在apache中指定php配置文件所在路徑

PHPIniDir php.ini 所在路徑

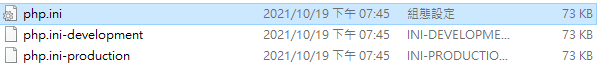




3.2由於php.ini是有development和production , php.ini本身是不存在的,

所以我們需要格式化:

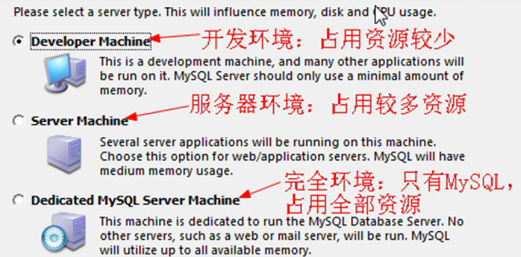
3.2.1複製php.ini-development, 改名為php.ini。

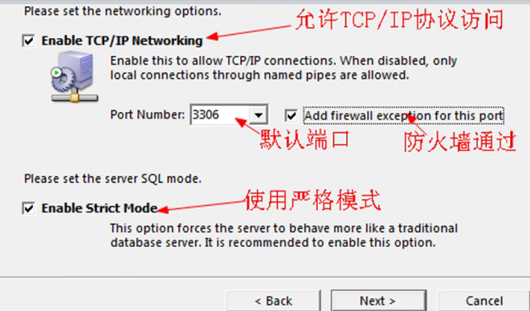


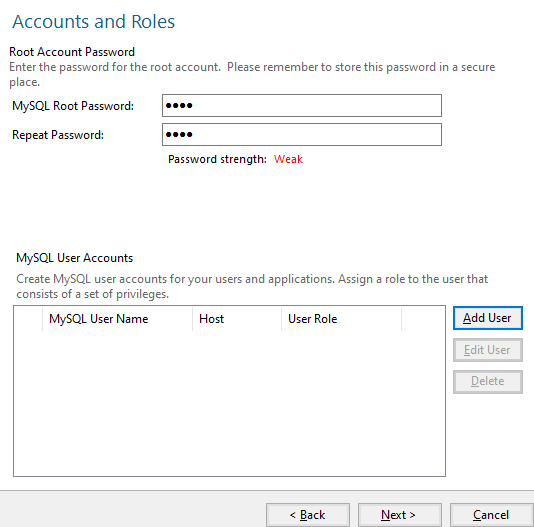
就完成了。

安裝與配置 MySQL

mySQL配置

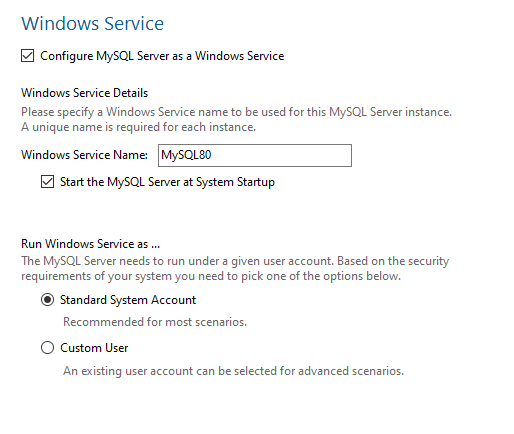


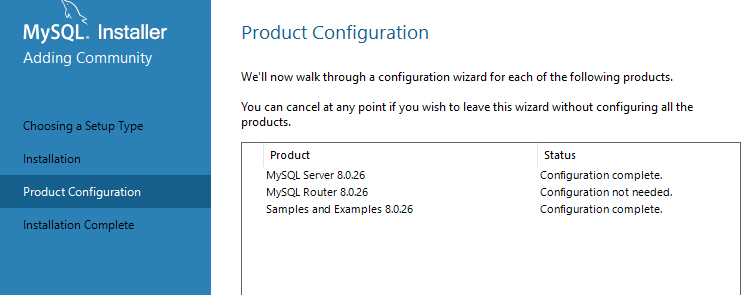




我們用root 做為password.

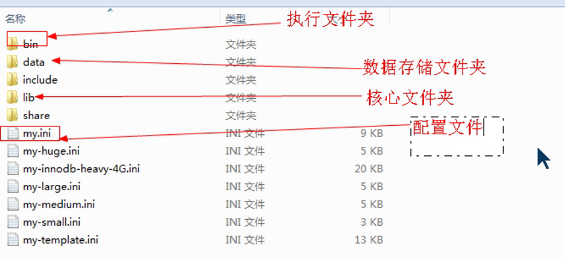
window service





mysql路徑:

C:\Program Files\MySQL\ MySQL Server 8.0



由於我們是默認default安裝路徑,所以data和 myini在ProgramData

C:\ProgramData\MySQL\MySQL Server 8.0

\*\*\*\*\*

mysql路徑:

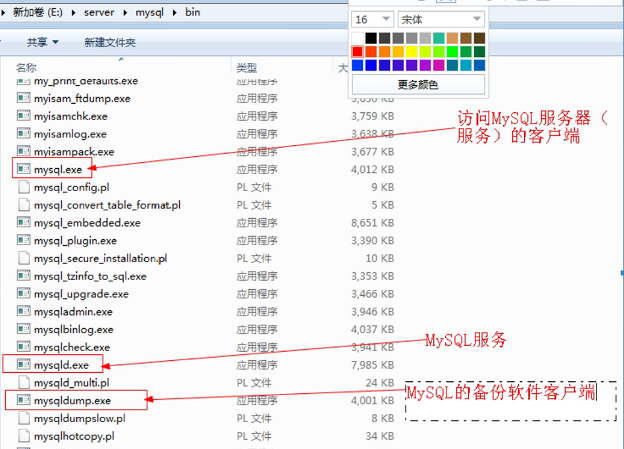
C:\Program Files\MySQL\ MySQL Server 8.0

mysql data與my.ini路徑:

C:\ProgramData\MySQL\MySQL Server 8.0

MySQL server內bin常用命令:

C:\Program Files\MySQL\MySQL Server 8.0\bin



軟件設計結構 C/S B/S

C/S : Client客戶端/Server服務端,

C/S:用戶需要安裝客戶端產品才能訪問服務器,而且只能訪問一種軟件(當前自己)

B/S : Browser瀏覽器/Server服務端

B/S: 用戶只需要安裝瀏覽器,就可以訪問所有的服務器(B/S架構服務)

\*\*\*\*\*

MySQL訪問流程:

mysql是一種C/S架構的軟件,需要通過客戶端來訪問服務端(mysql提示也提供了其他模式的訪問:通過一些插件擴展來充當客戶端)

mysql.exe是服務器的客戶端。

mysqld.exe是服務器本身



1. 啟用mysql客戶端: mysql.exe, 這軟件本身可以通過cmd開啟。

配置好環境變量,可以直接在cmd默認路徑下跑mysql.exe。



1. MySQL客戶端訪問服務端需要進行尋找匹配: 連接與認證

連接: IP和端口確認, 如果是本地都可以省略

-h 主機地址 🡪 -hlocalhost(可以是ip)

-P 端口 🡪-P3306 (3306是mysql默認的port) 大寫P

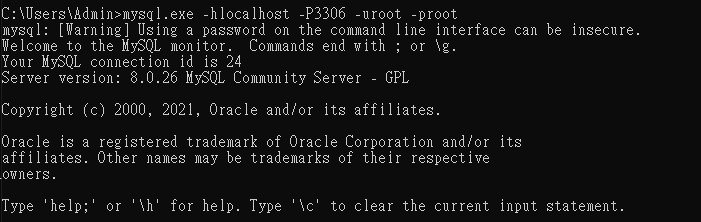
認證: 通過用戶名和密碼進入服務器

-u 用戶名🡪-uroot 不可以省略(只有匿名用戶可以省略)

-p密碼🡪-proot 小寫P

mysql.exe –hlocalhost –P3306 –uroot -proot





1. 退出命令 \q

可以-p 輸入回車,系統會再叫你輸入密碼,提高安全性



php連接mysql數據庫

php本身不具備操作mysql的能力,需要借助php操作mysql的擴展來實現。

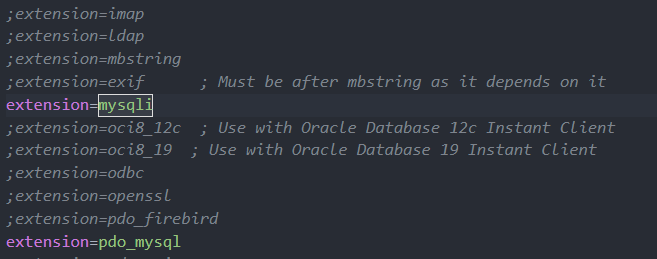
1. php加載mysql擴展:

在php.ini文件中

找到dpo\_mysql 開放它出來加載。 (去除分號)

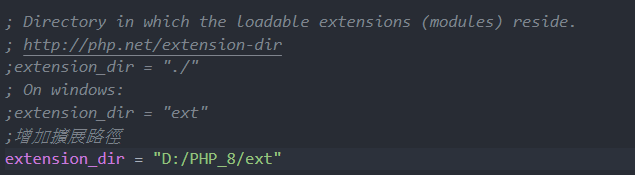
找到mysqli 開放它出來加載。 (去除分號)

\*\*\*\*\*php7之後就是開放以上2種來加載mysql擴展



1. 在php.ini文件中 增加擴展路徑

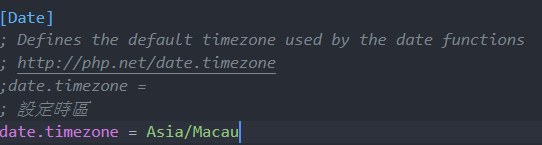
(因為我們php加載文件被apache加載走了,所以我們要手動設回來)



3.php timezone時區

自PHP 5開始 時區(date.timezone) 一開始是未設置所以會以標準時區 也就是GMT+0

所以在使用date相關的函式的時候 都會少了八個小時，所以必須去修改PHP.ini的設定。



WampServer是一款由法國人開發的Apache Web服務器、PHP解釋器以及MySQL數據庫的整合軟件包。免去了開發人員將時間花費在繁瑣的配置環境過程，從而騰出更多精力去做開發。W是window

LAMP是 Linux Apache Mysql Php。

WAMP是Window Apache Mysql Php。

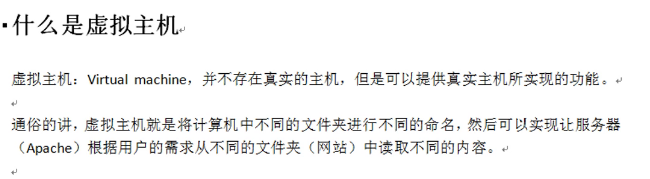
LNPM是 Linux Nginx MySQL PHP。

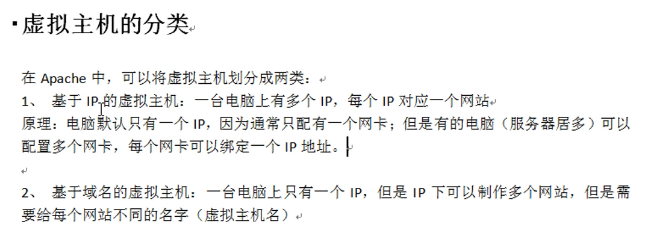
Nginx是非同步框架的網頁伺服器，也可以用作反向代理、負載平衡器和HTTP快取。

Apache HTTP Server是Apache軟體基金會的一個開放原始碼的網頁伺服器軟體，可以在大多數電腦作業系統中運行。由於其跨平台和安全性，被廣泛使用，是最流行的Web伺服器軟體之一。

配置虛擬主機

一台服務器十分貴,如果只能部署一個網站,就十分浪費,所以需要通過其他渠道來實現一台主機上部署多個網站。





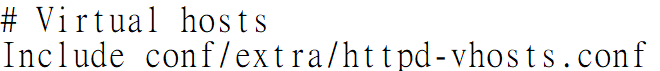
我們用的是 基於域名的虛擬主機。

配置基於域名的虛擬主機

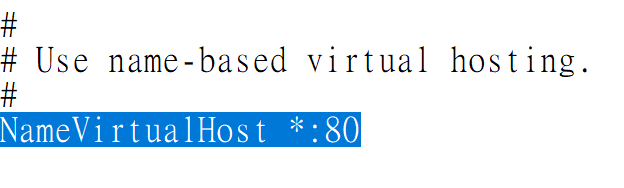
1.在apache的httpd.conf內:

找到virtual hosts

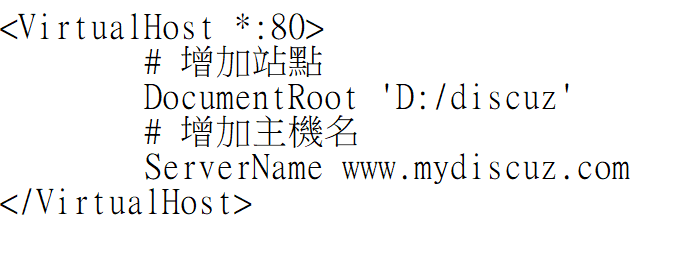
默認是 #注釋 沒有開啟的。 我們去除#



2.之後到conf/extra/httpd-vhosts.conf下, 寫上NameVirtualHost \*:80



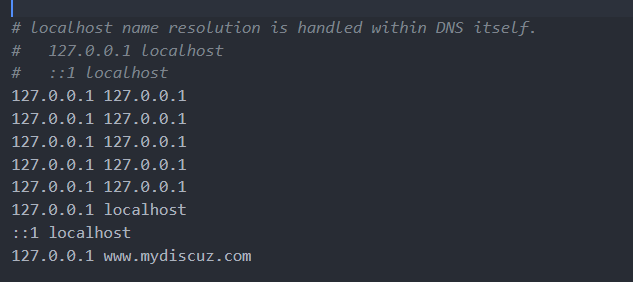
在下面增加站點DocumentRoot ,增加主機名ServerName www.mydiscuz.com



1. 增加域名解析DNS: hosts

C:\Windows\System32\drivers\etc 內的hosts 文件

增加127.0.0.1 www.mydiscuz.com



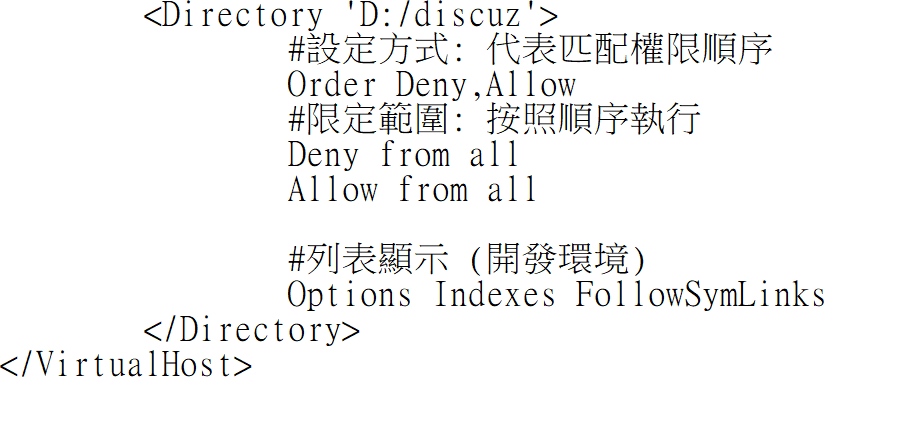
1. 配置站點DocumentRoot權限 <Directory ‘站點路徑’>





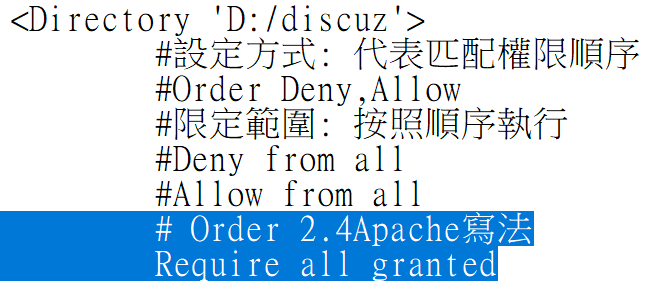
1. 配置訪問列表: 當訪問是一個目錄的時候(沒有指定具體要訪問的文件)

Options Indexes [FollowSymLinks], 如果沒有訪問目錄, 那麼列表訪問顯示所有文件。



Order在2.4改了寫法



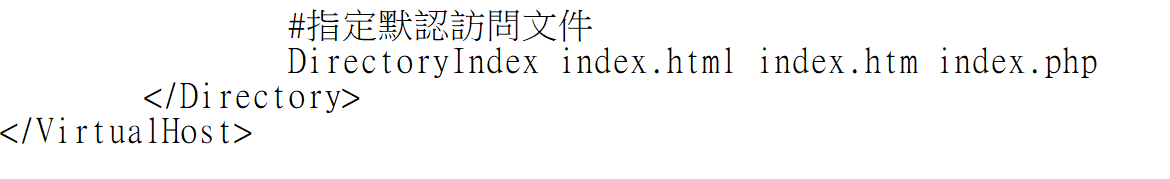


1. DirectoryIndex 默認的文件列表, 使用空格分離



DirectoryIndex index.html index.htm index.php

默認index.html, 沒有就找index.htm 再沒有就找index.php



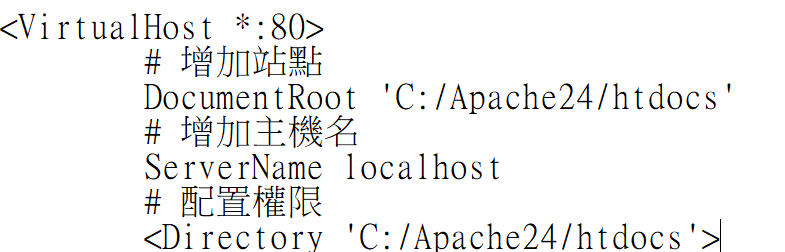
1. \*\*\*\*\*

一旦開啟虛擬主機配置文件,默認的主機地址localhost就不再生效, 如果想要生效.

為localhost增加單獨的虛擬主機。

比如: 我們想經由localhost這個域名去Apache的htdocs站點。

* 1. Apache的vhost.conf下DocumentRoot 改做Apache的htdocs
  2. Directorys權限的路徑也改做Apache的htdocs
  3. ServerName 改做localhost



現在,我們在同一個ip與port 80下, 我們現在利用不同的域名就可以去不同的網站。



C:\Windows\System32\drivers\etc\hosts

hosts文件的用途:

當查詢一個主機名稱(域名 eg: www.abc.com)時，  
如果這個檔案沒定義的話，  
就會去查DNS。

比如設定了 1.2.3.4 [www.hello.com](http://www.hello.com)

當我們輸入hello.com時, 會直接跑去1.2.3.4的ip地址。

php語法



腳本標記: <script language=”php”> \*php代碼\* </script>

標準標記(常用)<?php \*php代碼\* ?>

php語句分隔符 ;

在php中是以行為單位,結束要加; 分號。

php變量:

$var1 = 0;

訪問變量: echo $var1;

*// 刪除變量 unset(xx);*

    unset($var2);

變量命名規則:

不可以數字開頭

$var1; //可以

$\_var1; //可以

$1var; //不可以

預定義變量: 提前定義變量, 預定義變量都是數組。

$\_GET: 獲取所有表單以GET提交的數據

$\_POST: 獲取所有表單以POST提交的數據

$\_REQUEST: 獲取GET和POST提交的數據

$GLOBALS:PHP中所有全局變量

$\_SERVER: 服務器信息

$\_SESSION: session會話數據

$\_COOKIE: cookie會話數據

$\_ENV:環境信息

$\_FILES: 用戶上傳的文件信息

以上紅色是常用。

可變變量: 如果一個變量保存的值剛好是另一個變量的名字, 那麼可以通過訪問一個變量得到另外一個變量的值(不是指針!!!): 在變量前面多加一個$符號。

EG:

$a = ‘b’;

$b = ‘bb’;

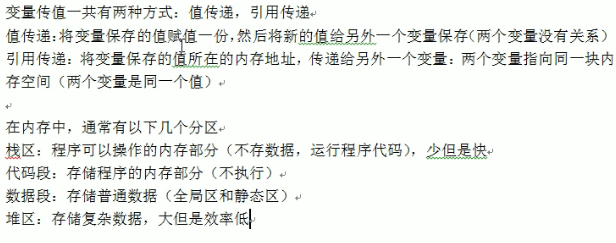
由於$a 是’b’

$$a 就可以是 $’b’ = ‘bb’;

所以$$a = ‘bb’;

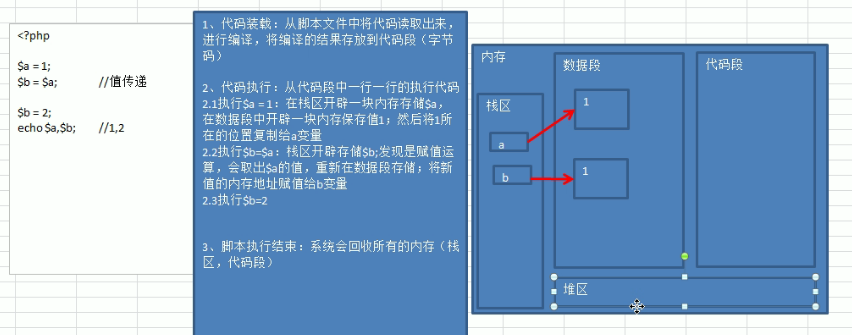
變量傳值:

變量傳值有2種方式: 值傳遞, 引用傳遞

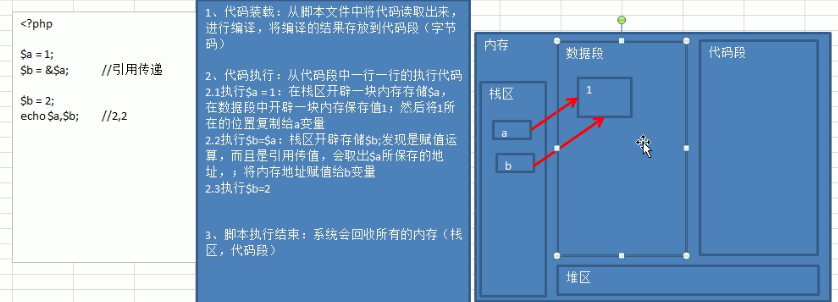
.

.

值傳遞: $a = $b;

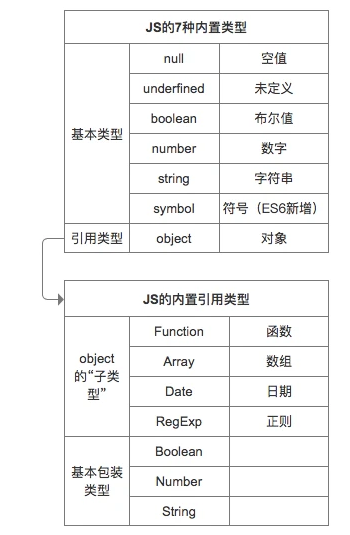


引用傳遞: $a = &$b;



\*\*\*\*\*

在JS中, 基本類型是值傳遞, 引用類型是引用傳遞



在PHP和C++中, 值傳遞和引用傳遞是要手動的。

$a = $b; 值傳

$a = &$b; 引用傳

常量:

2種常量定義方式:

define(‘常量名’ , 常量值);

const 常量名 = 值;

define可以定義 特殊字符名的常量

define(‘=\_=’ , ’smile’);

define的第三個參數是 大小寫敏感default = false;

系統常量:

PHP\_VERSION PHP版本號

PHP\_INT\_SIZE 整形int 大小 (4)

PHP\_INT\_MAX 整形能表現的最大值 (2147483647)

PHP\_INT\_SIZE 返回4 是 4個字節的意思。

php 魔術常量: \_\_常量名\_\_ 這種就是魔術常量。

魔術常量會跟環境而變化 , 用戶改變不了。

\_\_DIR\_\_ : 當前被執行的腳本所在電腦的絕對路徑

\_\_FILE\_\_: 當前被執行的腳本所在電腦的絕對路徑(帶文件名)

\_\_LINE\_\_:當前所屬行數

\_\_NAMESPACE\_\_:當前所屬的命名空間

\_\_CLASS\_\_: 當前所屬的類

\_\_METHOD\_\_: 當前所屬的方法

php數據類型:

php本身是弱語言,變量本身沒有數據類型。

**php的8種數據類型**:

php將數據分為 3大類8小類:

第一大類:簡單(基本)數據類型(4小類):

整形int/integer 系統分配4個字節

浮點型float / double: 系統分配8個字節, 表示小數或整數存不下的整數

字符串型 String, 系統根據字符串長度分配

布爾型: bool / Boolean , true/false。

第二大類:複合數據類型(2小類):

對象類型object: 存放對象

數組類型array: 一次性存多個數據

第三大類:特殊數據類型(2小類):

資源類型resource: 存放資源數據(php外部數據, 如數據庫,文件)

空類型NULL: 只有一個值,就是NULL(不能運算)。

**數據類型轉換**:

在很多的條件下,需要指定的數據類型, 需要外部數據(當前php取得的數據), 轉換成目標數據類型。

在php中有2種數據轉換方式:

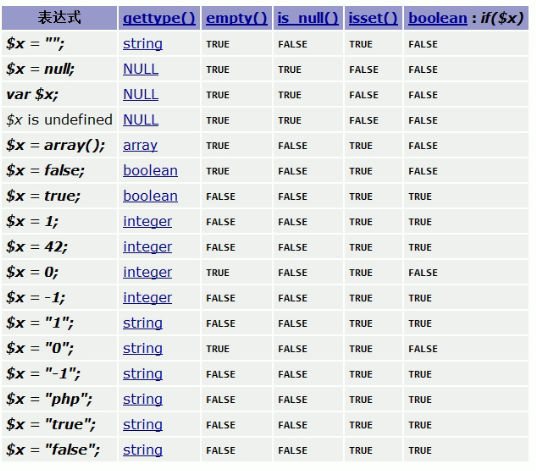
1. 自動轉換: 系統根據需求自動轉換(用得比較多,但效率低)。
2. 強制(手動)轉換: 根據我們自己需要手動轉換。

在轉換過程中,用得比較多的是轉boolean和轉數值類型(算術運算)。

強制轉換規則: 在變量之前增加一個括號(), 然後在里面寫上對應類型: int/integer…

如果轉換NULL類型 要用到unset()

轉換表:



其他類型轉數值的說明:

1. Boolean trun為1, false為0
2. 字符串轉數值有自己的規則
   1. 以字母開頭的,永遠是0
   2. 以數字開頭的,取到有其他非數字字符串為止。(不會同時包含2個小點數)

php7開始沒有自動轉換了。\*\*\*\*\*

只有手動轉換。

08-數據類型判斷函數

is\_類型名(變量) 返回bool值,是否類型名與變量的類型相同。

返回: true返回1, flase什麼也不返回。

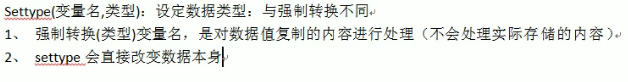
因為bool類型不能用echo查看, 可使用var\_dump結構查看。

var\_dump(var1,var2…);

比如: var\_dump(is\_string($a));

gettype(變量, ‘想設置成的類型’): 返回 該變量的類型名的字符串。

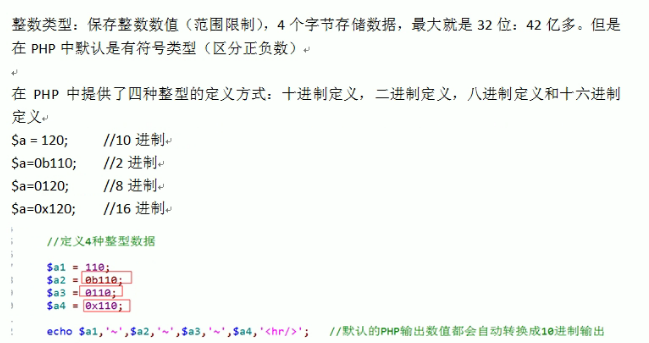
settype 改變類型



settype(變量名,類型) 會直接改變數據本身的類型

而(float)$a; 是不會改變$a本身。

php 整數類型進制



php進制轉換函數:

decbin():10進制轉2進制

decoct();10進制轉8進制

dechex(); 10進制轉16進制

bindec(): 2進制轉10進制

浮點數



在php中最好不要做float計算, 不精確。

**var\_dump()** 函數用於輸出變量的相關信息。

**var\_dump()** 函數顯示關於一個或多個表達式的結構信息，包括表達式的類型與值。數組將遞歸展開值，通過縮進顯示其結構。

empty(): 判斷數據值是否為空, 不是NULL。 如果空 返回true。不為空返回false;

isset():判斷數據存儲的變量本身是否存在, 存在變量返回true, 不存在返回false;

php運算符

賦值運算符 = php可以連貫賦值 $a = $b = 10;

算術運算符

+

-

\*

/ 除法和余法不能除0

%

比較運算符

>

<

>=

<=

== 數值相等就可以, 數據類型不相同也可以。

!=

=== 全等 大小和數據類型都要相等

!== 不全等

邏輯運算符

&& 邏輯與

|| 邏輯或

! 邏輯非

\*\*\*\*\*

連接運算符

是php中將多個字符串拼接的一種符號

. (dot): 將兩個字符串連接到一起 $a . $b

.= :複合運算, 將左邊的內容 與 右邊的內容 連接起來, 然後重新賦值給左邊。(相當於+=)

$a .= $b; 相當於 $a = $a . $b;

錯誤抑制符

在php中有一些錯誤可以提前預知,但是這些錯誤可能無法避免, 但是又不希望報錯給用戶。 就可以用錯誤抑制符。

@:在可能出錯的表達式前面使用@

Warning

Prior to PHP 8.0.0, the value of the severity passed to the custom error handler was always 0 if the diagnostic was suppressed. This is no longer the case as of PHP 8.0.0.

在 PHP 8.0.0 之前，如果診斷被抑制，則傳遞給自定義錯誤處理程序的嚴重性值始終為 0。 從 PHP 8.0.0 開始，情況不再如此。

三目運算符(同js)

A(condition) ? b : c;

複合三目運算

A(condition) ? ( (Bcondtion) ? c : d ) : e

自操作運算符

++

--

$a++; 相當於 $a = $a+1;

前置與後置

++$a;

$a++;

前置是先自增再自我賦值。

後置是先自我賦值(計算表達式)再自增。

\*=

+=

-=

/=

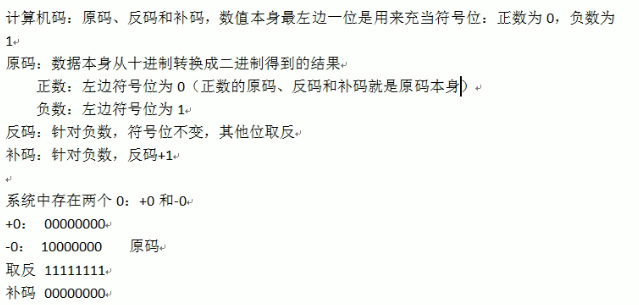
%=

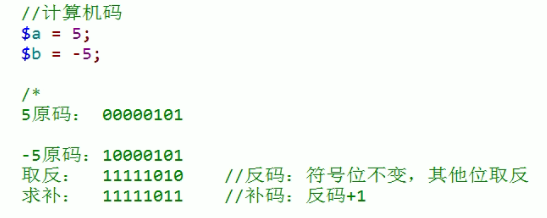
都是自操作運算符

計算機碼

原碼,反碼,補碼







位運算符

位運算: 取出計算機中最小的單位,(位bit)進行運算

&

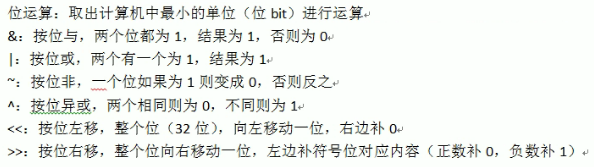
|

~

^

<<

>>



if else 語句

如同js與c++。

if (c) {xx} else if (c){xx} else{xxx};

switch語句

如同js。

switch(var){

case 1: //如果var === 1

執行here

break;

case2:

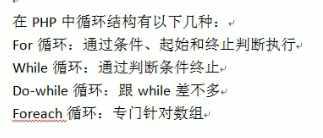
break;

default:

break;

}

for循環



如同js, java。

循環控制

continue; 馬上重新開始循環

break; 循環直接結束

php 替代語法

//php中endfor

<table border=1>

    <?php for($i=1;$i<10;$i++) :?>

    <tr>

    <?php for($j=1;$j<=$i;$j++) :?>

        <td>

            <?php echo $i . '\*' . $j . '=' . $i\*$j ?>

        </td>

        <?php endfor; ?>

    </tr>

    <?php endfor; ?>

</table>

for (x;x;x) : endfor;

for(x;x;x;) { };

php 替代語法:

左大括號{ 變成 :

右大括號} 變成 end+對應標記;

if(){} 🡪 if() : endif;

switch(){} 🡪switch(): endswitch;

for

while

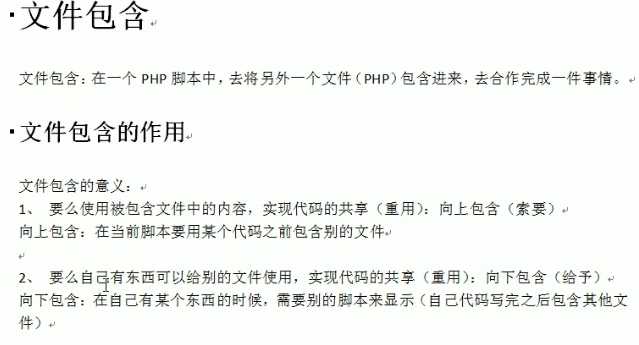
foreach

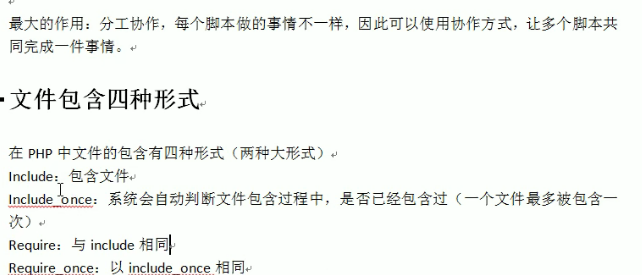
\*\*\*\*\*

15-php文件包 (import)

向上包含(import加載/引入) 索要

向下包含(export 導出) 給予





向上包含 include require

include , include\_once

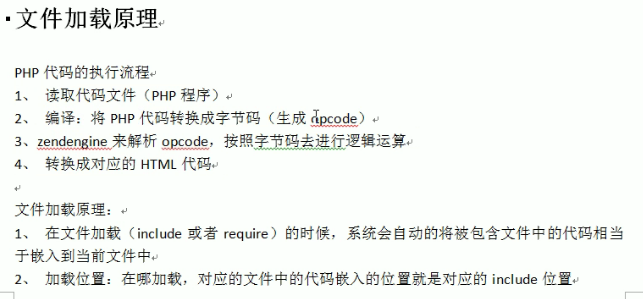
require , require\_once (include與require相同)

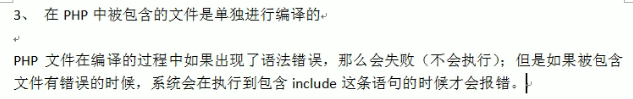
寫法:

include ‘文件路徑’; include(‘文件路徑’);

文件加載原理:

php文件加載會直接在加載的對應位置嵌入被加載文件的代碼。





include與require分別

include與include\_once ,

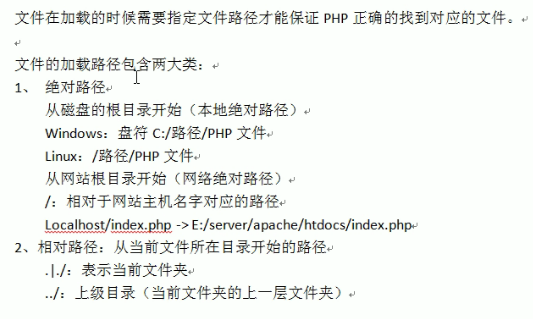
include\_once只會跑一次, 如果前面有include同一個文件, include\_once也不會執行。



include錯誤級別較低: 不會阻止代碼執行 warning

require錯誤級別較高: 如果出錯,下面代碼不再執行 fatal error

文件加載路徑



嵌套include



相對路徑是相對的

\*\*\*\*\* ./ 指的是執行文件當前目錄。

php 函數function

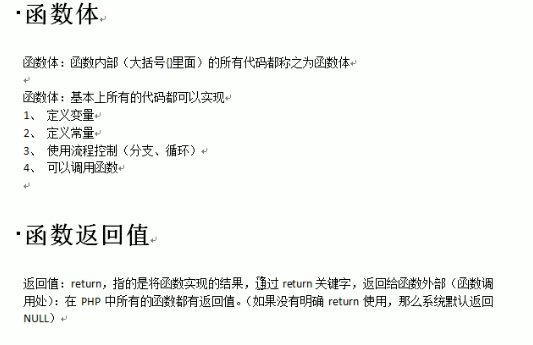
基本與js相同。

引用傳參數 與 值傳參數

js是 引用數據類型就引用傳參, 基本類型是傳值調用。

php是與C++相同, 要用&號 引用傳參。

函數體

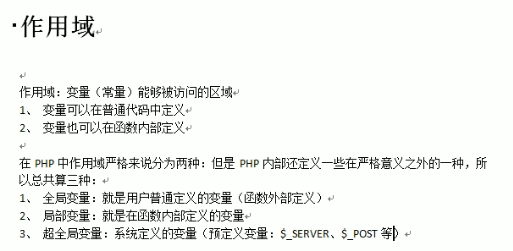


php作用域

全局變量

局部變量

超全局變量: 系統定義的 $\_SERVER, $\_POST等



js下 局部作用域可以訪問全局變量。

\*\*\*\*\*

但php不可以,要傳參才能在函數(局部作用域)訪問全局變量。

php有一個超全局變量$GLOBALS 是一個數組,會放一些全局變量,

我們可以用超全局變量$GLOBALS獲取全局變量。

. $GLOBALS獲取全局變量時, 是用變量名做為索引的。

$g2 = 'hello';

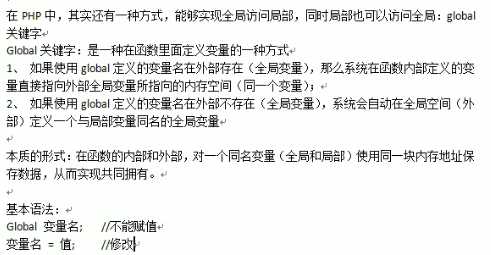
    function fun($g){

        echo $g , '</br>';

        echo $GLOBALS['g2'] , '</br>'; *// $g2*

    }

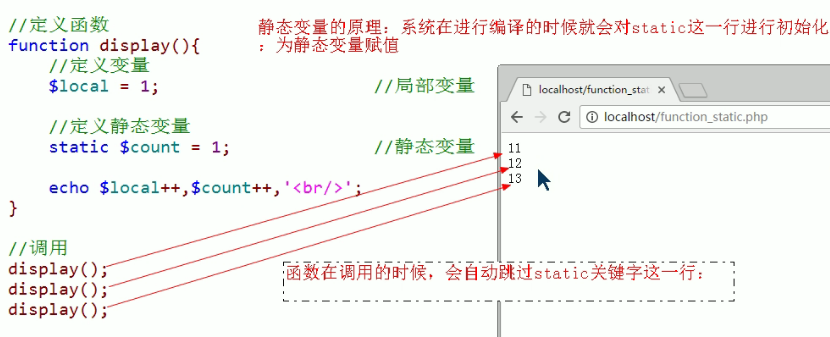
global關鍵字



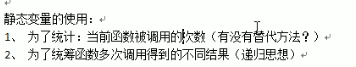
global的定義和賦值要分開。

靜態變量 static

php靜態變量是為了實現跨函數共享數據(同一個函數多次調用)



static靜態變量的使用場景



可變函數

當一個變量與函數名相同時, 就可以使用變量+()來充當函數名使用。

$變量();

可變函數的使用



也就是callback 回調函數

php 匿名函數

寫法:

$變量名 = function(){};

用變量去接收匿名函數

調用時$變量();

\*\*\*

接收了匿名函數的變量, 也是 可變函數。

closure 閉包



php閉包: \*\*\*\*\*

js的閉包是直接可以用的, php要加關鍵字use(變量名)。

function fn1(){

        $name = \_\_FUNCTION\_\_;

        $fn2 = function() use($name){

            echo $name;

        };

        $fn2();

    };

    fn1();

php的作用域局分十分精確。

php偽類型

php偽類在3大類8小類之外的類, 主要有2種

Mixed混合的: 可以是多種php的數據類型

Number數值的: 數值的, 可以是任意數值類型(int,float,double)

\*\*\*\*\*

php常用系統函數

1. 有關輸出的函數

print() 類似echo,本質是一種結構(不是函數),如同include, 可以不需要使用()括號。

print()會返回1 所以如果echo print(); 最後會多輸出一個1。

print\_r(): 類似於var\_dump, 但是比var\_dump簡單,不會輸出數據的類型,只會輸出值(數組打印使用比較多)

1. 有關時間的函數

date()**date** ( string $format , int $timestamp = ? ) : string

timestamp 是可选的，默认值为 [time()](mk:@MSITStore:D:\PHP_8\manual\php_enhanced_zh.chm::/res/function.time.html)。

time()time — 返回当前的 Unix 时间戳

microtime()microtime — 返回当前 Unix 时间戳和微秒数

strtotime()

1. 有關數學的函數

max() 返回參數中最大的一個

min() 如果參數只有一個數組,返回最小值, 如果只是數值,返回參數中最小的一個。

rand() 返回區間的RANDOM整數, 如果只有一個參數,返回0到參數之間的。

mt\_rand() 與rand相同,底層結構不同,效率更高。

round() 四舍五入

cell() 向上取整

floor() 向下取整

pow() 次方 pow(2,8) == 2的8次方。

abs() 絕對值

sqrt() 平方根

1. 有關函數的函數

function\_exists() 判斷指定函數的名字是否在內存中存在。(幫助用戶不去使用不存在的函數,讓代碼安全性提高)

func\_get\_arg() 在自定義函數中去獲取指定數值對應的參數

func\_get\_args()在自定義函數中去獲取所有參數(返回數組)

func\_num\_args()獲取當前自定義函數的參數數量

php錯誤處理

錯誤分類:

1. 語法錯誤: 用戶寫的代碼不符合語法規範,在編譯過程中不通過,所以代碼不會執行(Paser error)
2. 運行時錯誤: 代碼編譯通過,但是代碼在執行過程中會出現一些條件不滿足導致的錯誤。(runtime error)
3. 邏輯錯誤: 程序員在寫代碼時不夠規範, 出現了邏輯性錯誤, 代碼正常執行,但是得不到想要的結果。 (常常出現在程序員寫錯, 但不報錯, 取難的!!)

錯誤代號:

php的錯誤代碼都是系統常量(我們可以直接調用)

1. 系統錯誤:

E\_PARSE: 編譯錯誤,代碼不會執行

E\_ERROR: fatal error, 致命錯誤,代碼不能正確執行( 出錯的位置斷掉)

E\_WARNING: warning 警告錯誤, 不影響代碼執行, 但是可能得不到想要的結果

E\_NOTICE: notice通知錯誤, 不影響代碼執行, 只影響美觀。

1. 用戶錯誤:

E\_USER\_ERROR: fatal error ,代碼不能正確執行( 出錯的位置斷掉)

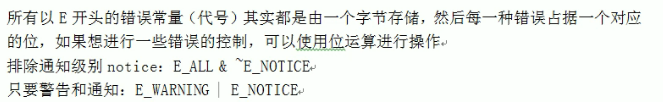
E\_USER\_WARNING: 不影響代碼執行, 但是可能得不到想要的結果

E\_USER\_NOTICE: notice通知錯誤, 不影響代碼執行, 只影響美觀。

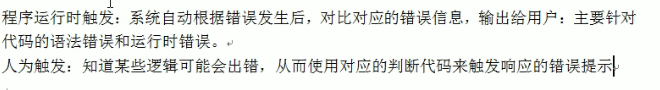
用戶在使用自定義錯誤觸發的時候, 會使用到的錯誤代號(系統不會用到)

1. 其他:

E\_ALL 代表所有的錯誤(通常在進行錯誤的控制時使用比較多), 建議在開發過程(開發環境)中使用。



錯誤觸發



trigger\_error(觸發錯誤提示):

**trigger\_error** ( string $error\_msg , int $error\_type = E\_USER\_NOTICE ) : bool

第一個參數:

error\_msg

该 error 的特定错误信息，长度限制在了 1024 个字节。超过 1024 字节的字符都会被截断。

第二個參數:

error\_type

该 error 所特定的错误类型。仅 E\_USER 系列常量对其有效，默认是 **E\_USER\_NOTICE**。

比如改為 E\_USER\_ERROR 變成fatal error

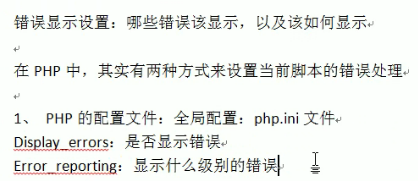
trigger\_error('<b>除數不能為0!</b>',E\_USER\_ERROR);

php處理腳本讓瀏覽器按照指定 字符編碼 解析:

header(‘Content-Type:text/html;charset=utf-8’);

錯誤設置

php.ini文件可設置腳本的錯誤處理





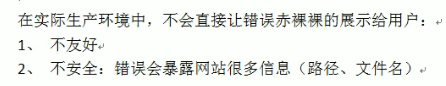
error\_reporting() 沒有參數代表獲取當前系統錯誤處理對應級別

ini\_set(配置文件中的配置項,配置值)

ini\_set(‘error\_reportying’,’E\_ALL’);

ini\_set(‘display\_errors’,1);

錯誤日志設置



1 開啟日志功能 log\_errors=on;

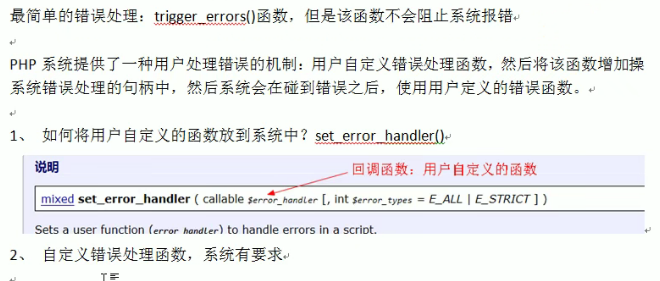


2指定路徑 error\_log = ‘路徑’;

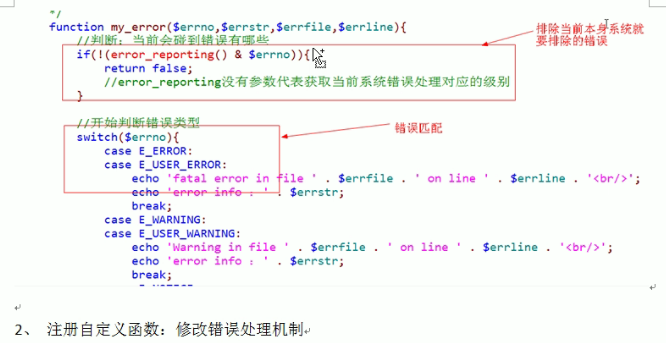


自定義錯誤處理

**set\_error\_handler** ( [callable](mk:@MSITStore:D:\PHP_8\manual\php_enhanced_zh.chm::/res/language.types.callable.html) $error\_handler , int $error\_types = E\_ALL | E\_STRICT ) : [mixed](mk:@MSITStore:D:\PHP_8\manual\php_enhanced_zh.chm::/res/language.types.declarations.html#language.types.declarations.mixed)



error\_reporting() 沒有參數代表獲取當前系統錯誤處理對應級別



28-自定義錯誤提示-set\_error\_handler.php

字符串類型php

4種定義方式

1. 單引號
2. 雙引號
3. nowdoc字符串

$str = <<< ‘EOD’

hello

EOD;

1. heredoc字符串

$str = <<< EOD

hello

EOD;

EOD是邊界符



字符串轉義

在php中常用的轉義符號:

\’ 在單引號字符串中顯示單引號

\” 在雙引號字符串中顯示雙引號

\r 代表回車 理論上是回到當前行的首位置

\n 代表新一行

\t 類似tab, 輸出4個空格

\$ 在php中使用$符號作為變量符號, 因此需要特定識別

\*\*\*\*\*

php中 單引號只能夠識別 \’ , 雙引號只不能夠識別 \’。

所以 在php中 字符串我們用雙引號。

而且雙引號能夠在字符串中識別 變量。

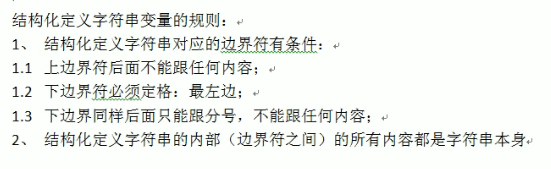
為了在雙引號內好區分 變量, 我們用專業標識符區分 {};

$a = 100;

$str3 = “abc{$a}def”;

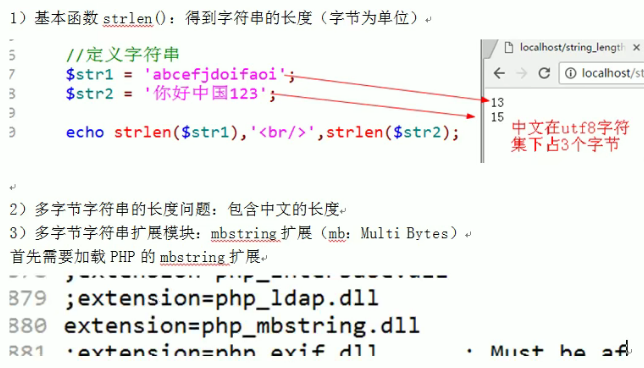
\*\*

結構化字符串php7開始 EOD邊界符不需要頂格了。



字符串長度問題

strlen() 得到字符串的長度(字節為單位) 中文在utf8中是3個字節



mb\_strlen要開啟mbstring的extension (在php.ini配置)

字符串相關函數\*\*\*\*\*

1.轉換函數

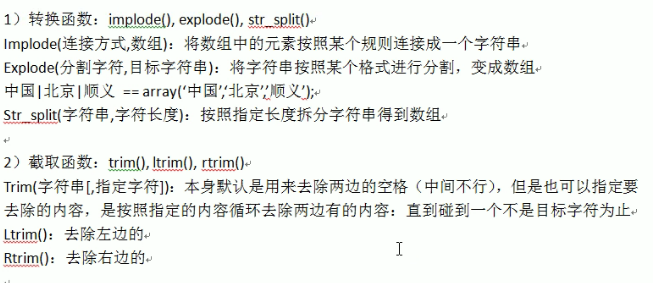
implode(連接方式, 目標數組) js中的join() 數組轉換為字符串

explode(分開字符串, 目標字符串) js中的split() 字符串轉換數組

str\_split(字符串,多少字符串放到一個索引default=1) 按照指定長度拆分字符串 得到 數組 。

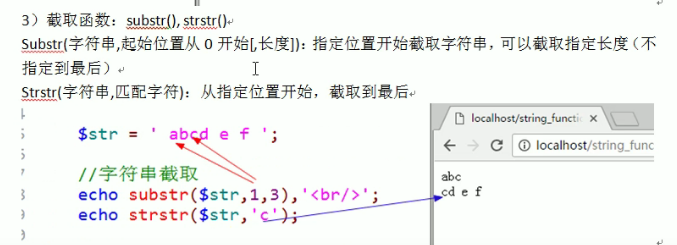
mb\_str\_split() 如果中文要用到mb擴展

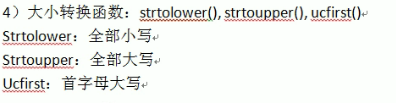
2.截取函數



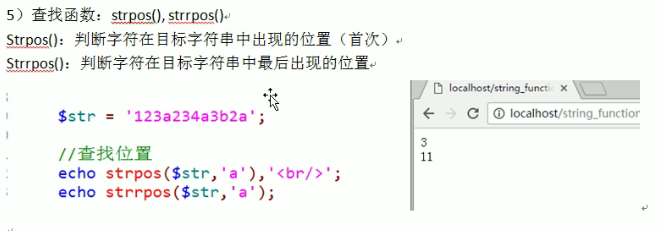
substr($str1 ,1 ,3); 返回由索引1到3的字符串

strstr(目標字符串,指定字符串), 由指定字符串開始, 取到最後 ()





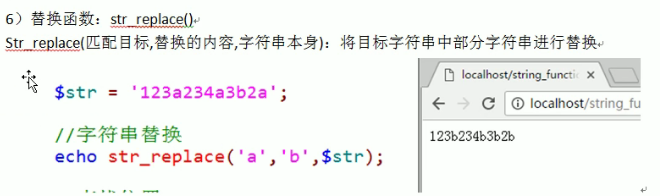
strtolower() strtoupper ucfirst



strpos()

strrpos()

str\_replace() 替換字符串



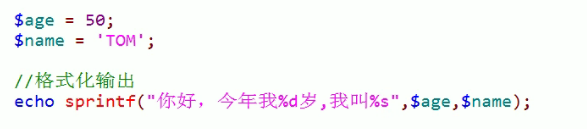
7.格式化輸出數據 printf sprint

printf(輸出字符串有占位符,順序占位內容)

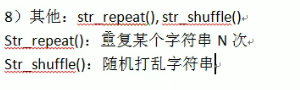
sprint(輸出字符串有占位符,順序占位內容)

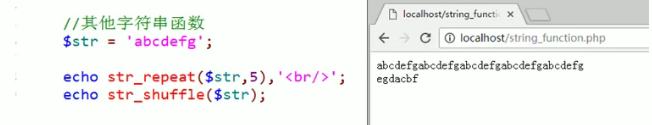
%d 是 十進制整數

%s 是字符串



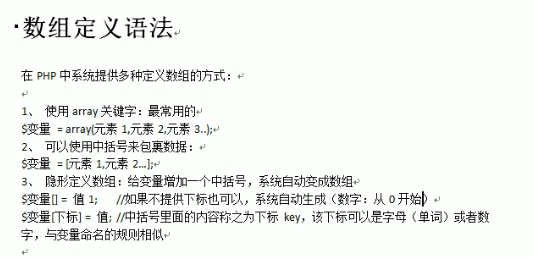
1. 其他 str\_repeat() str\_shuffle()

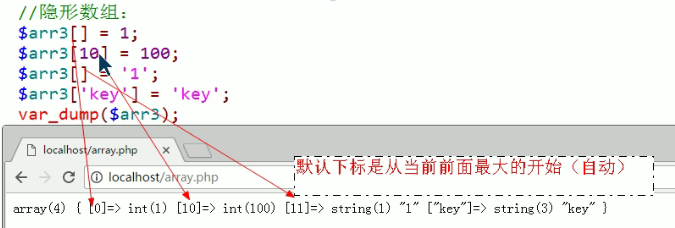




php數組

php數組定義方式

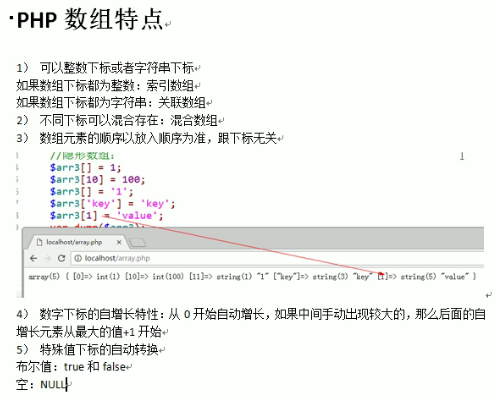




\*\*\*

php數組的索引可以是字符串, 也可以打亂。

php數組特點

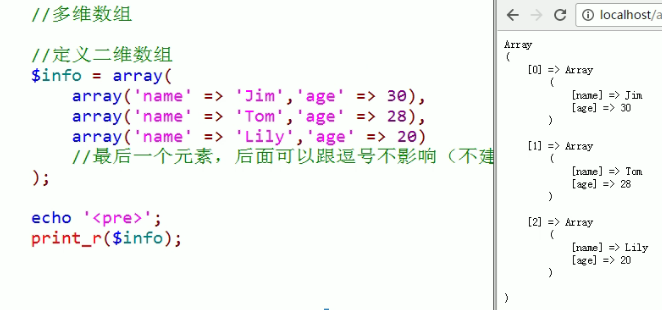


6.php數組元素沒有類型限制, 可以混合。

7.與java不同,php和js數組不限制長度,可自由加減。

php數組是很大的數據, 存放在堆區, 為當前數組分配一塊連續的內存。

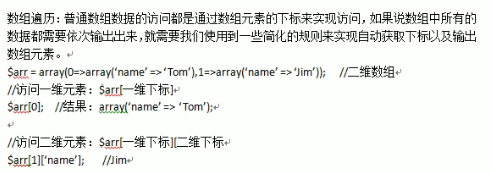
二維數組



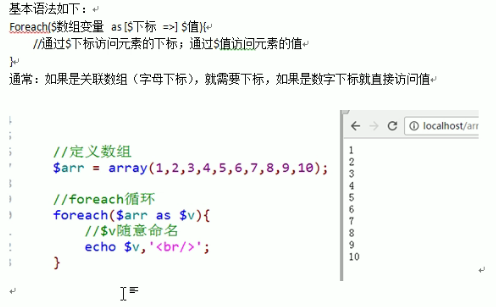
多維數組與異形數組

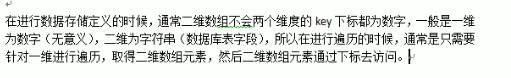


編歷數組



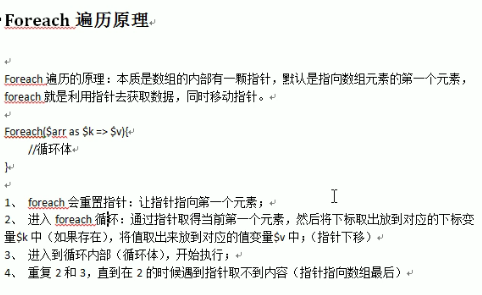
foreach() 語法







foreach原理



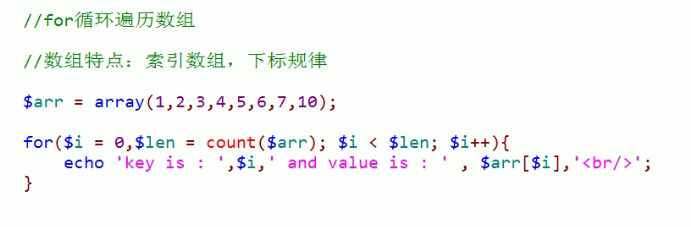
\*\*\*\*\*

count() — 計算數組中的單元個數，或對象中的屬性個數

php中用count來獲取數組長度 js是array.length

php for循環編歷數組





while配合each和list編歷數組

each(): each能夠由一個數組中獲取當前數組指針所指向的元素的下標和值,拿到之後將數組指針下移, 同時將拿到的元素下標和值以一個四個元素的數組返回。

0下標🡪取得元素的下標

1下標🡪取得的元素的值

key下標🡪取得元素的下標值

value下標🡪取得元素的值

\*\*\*

each函数已自 PHP 7.2.0 起废弃。强烈建议不要使用本函数。

list()

list — 把数组中的值赋给一组变量

**list** ( [mixed](mk:@MSITStore:D:\PHP_8\manual\php_enhanced_zh.chm::/res/language.types.declarations.html#language.types.declarations.mixed) $var , [mixed](mk:@MSITStore:D:\PHP_8\manual\php_enhanced_zh.chm::/res/language.types.declarations.html#language.types.declarations.mixed) ...$vars = ? ) : array

\*\* PHP 5 里，**list()** 从最右边的参数开始赋值； PHP 7 里，**list()** 从最左边的参数开始赋值。

list()不是真正的函数，而是语言结构。 **list()** 可以在单次操作内就为一组变量赋值。

\*\*\*\*\*

php數組相關函數 數組函數 數組方法

排序函數 (\*\*\*\*\*排序是引用傳參,直接影響原數組)

sort() 升序排序(按照索引排序, 按照值排序) 返回bool

rsort() 逆序排序

asort() 升序排序(按照值排序,保留值對應的索引)

arsort() 逆序排序

ksort() 升序排序:(按照索引排序, 原索引對應的值不變)

krsort() 逆序排序

shuffle(): (按照值打亂排序)

指針函數

reset() 重置指針 將數組指針回到首位

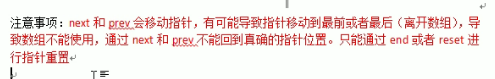
end() 重置指針 將數組指針指到最後一位元素

next() 指針下移,取得下一個元素的值

prev() 指針上移,取得上一個元素的值

current() 獲取指針當前指向的元素值

key() 獲取當前指針對應的索引



其他函數

count() 返回數組元素個數 相當於array.length

後面:

array\_push() 數組是引用傳參, 在數組尾增加元素

array\_pop() 數組是引用傳參, 在數組尾抽走元素

前面:

array\_shift () 數組是引用傳參,數組前面取出一個元素

array\_unshift() 數組是引用傳參,數組前面增加一個元素

array\_reverse() 數組元素反過來(值傳參)

in\_array() 判斷一個元素在數組中是否存在

**in\_array** ( [mixed](mk:@MSITStore:D:\PHP_8\manual\php_enhanced_zh.chm::/res/language.types.declarations.html#language.types.declarations.mixed) $needle , array $haystack , bool $strict = **false** ) : bool

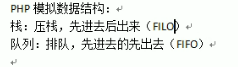
返回bool

array\_keys() 返回一個數組,數組內存有目標數組所有索引

array\_values() 返回一個數組,數組內存有目標數組所有值



php模擬數據結構 FILO FIFO 先進後出,先進先出



\*\*\*

php 數組 方括號 []

$arr = array();

$arr[] = 1;

[] 索引不給鍵key時, 會自動取最大索引+1, 相當於push一個新的值。

ltrim($string , [needle]) 刪除字符串needle(含needle)開始左面所有字符串。

rtrim($string , [needle])

strstr($string , needle , [true/false])

true:取字符串頭與needle之間 /

false:取needle與字符串尾之間(包含needle)。

strrchr($string , needle) 取needle與字符串尾之間(包含needle)。

舊php的strrchr是不包含needle的

is\_array()

is\_dir()

in\_array()