PART10

- 在被载入文件中return语句的作用

1，一个载入语句，如果载入成功，其实是有返回值的，为1，如果载入失败，则返回的是false。
（虽然我们通常不去使用该返回值）。

```php

主文件开始位置:
<?php
header('Content-type:text/html; charset=utf-8');



    echo "<br />主文件中的位置A";

    include "./liucheng.php";   //要载入的文件

    echo "<br />主文件中的位置B";



?>

<?php
include "./liucheng.php";//成功就有返回值1，但这里不用他
$v1 = include "./liucheng.php";//
echo "<br/>"; var_dump($v1);

?>

```

```
主文件开始位置:
主文件中的位置Aabcd132
---
A
当前时间2017-01-19 11:39:10
wait
当前时间2017-01-19 11:39:10
25A25
主文件中的位置B abcd132
---
A
当前时间2017-01-19 11:39:10
wait
当前时间2017-01-19 11:39:10
25A25abcd132
---
A
当前时间2017-01-19 11:39:10
wait
当前时间2017-01-19 11:39:10
25A25
int(1)
```

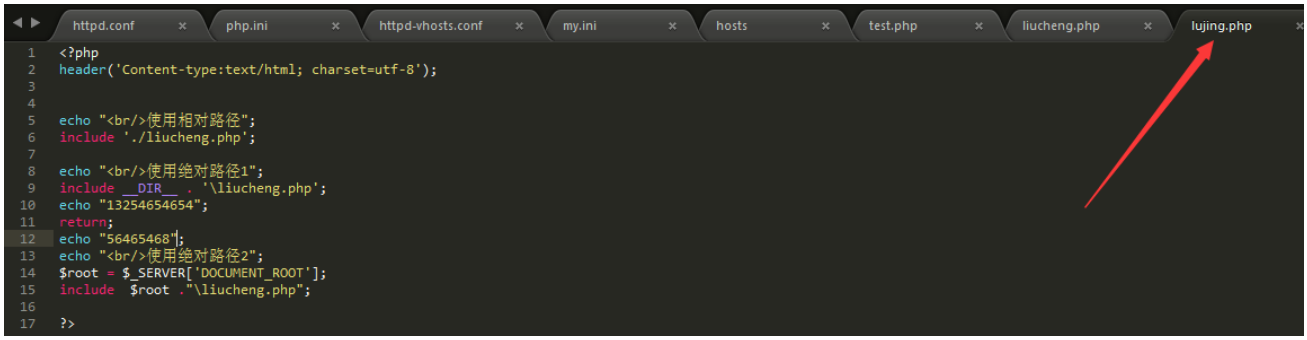
但，如果被载入文件中有return语句，此时就有另外的机制和作用：
2，return语句此时的作用是终止载入过程——该return语句的后续被载入文件的代码不再载入。

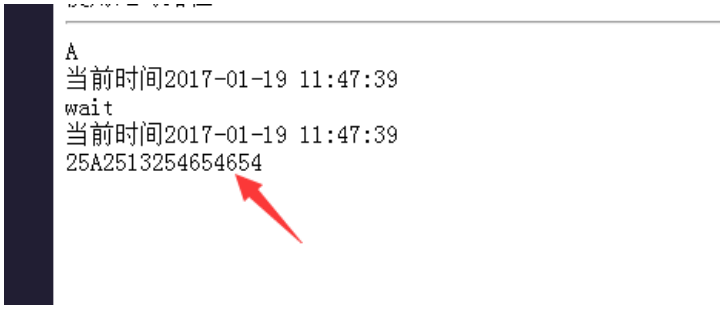
```php


?>

<?php
include "./liucheng.php";//成功就有返回值1，但这里不用他
$v1 = include "./liucheng.php";//
echo "<br/>"; var_dump($v1);
include "./lujing.php";
?>

```





3，return语句也可以用于该被载入文件载入时返回一个数据，形式为：return XX数据；

- 错误处理

错误的分类
通常分3种：
语法错误：程序运行之前，都要先检查语法。如果语法有错误，就会立即报错，并且不会去执行程序。
运行时错误：就是在程序语法检查通过后，，开始运行程序并在此过程中遇到的错误。常见的有3中：
提示性错误、警告性错误、致命错误。

逻辑错误：
指的是，程序本身可以正常执行，没有报错——但“计算结果”却错了。

错误的分级
php语言中，将各种错误进行了不同级别的分类归纳，并形成大约有10几个级别的错误，这就是技术层面的错误分级。
每一级别的错误，都有一个“代号”，这个代号其实也就是一个系统内部的“常量而已”。比如：

- 系统常见错误：

E_ERROR： 致命错误
E_WARNING: 警告性错误
E_NOTICE: 提示性错误
用户可自定义的错误：
E_USER_ERROR： 自定义致命错误
E_USER_WARNING: 自定义警告性错误
E_USER_NOTICE: 自定义提示性错误
其他：
E_STRICT: 严谨性语法检查错误
E_ALL ： 代表“所有错误”。
详细参考相应手册！

错误代号的实际值：
示例：

```php
<?php
header('Content-type:text/html; charset=utf-8');
?>
<?php
function GetBinStr($e){
    $s = decbin($e);//这是一个二进制数字字符串
    //str_pad($str1,长度n,$str2,位置w)函数的作用是：
    //将字符串$str1，用字符串$str2填充到指定长度n，
    //而且可以指定填充的位置W：左边填充还是右边填充；
    $s1 = str_pad($s,16,"0", STR_PAD_LEFT);
    return $s1;
}
echo "<pre>";
echo "<br />E_ERROR=" . E_ERROR . ", \t\t其对应二进制值为：" . GetBinStr(E_ERROR);
echo "<br />E_WARNING=" . E_WARNING. ", \t\t其对应二进制值为：" . GetBinStr(E_WARNING);
echo "<br />E_NOTICE=" . E_NOTICE. ", \t\t其对应二进制值为：" . GetBinStr(E_NOTICE);
echo "<br />E_USER_NOTICE=" . E_USER_NOTICE. ", \t其对应二进制值为：" . GetBinStr(E_USER_NOTICE);
echo "<br />E_ALL=" . E_ALL. ", \t\t其对应二进制值为：" . GetBinStr(E_ALL);
echo "</pre>";

?>
```

```
E_ERROR=1,              其对应二进制值为：0000000000000001
E_WARNING=2,            其对应二进制值为：0000000000000010
E_NOTICE=8,             其对应二进制值为：0000000000001000
E_USER_NOTICE=1024,     其对应二进制值为：0000010000000000
E_ALL=30719,            其对应二进制值为：0111011111111111
```

- 错误的触发

错误的触发，就是让错误“发生”。
有两种方式会触发错误：
系统触发：程序运行到某行代码，确实出现了某种错误，此时系统就会报错——这就是触发了系统错误。
系统触发的典型错误有这3种：
E_NOTICE:　　提示性错误：会输出错误提示，并继续执行后续代码；
E_WARNING:　警告性错误：会输出错误提示，并继续执行后续代码（也可能看具体情况，比如require)
E_ERROR：　　致命错误：导致程序无法执行后续语句；

- 自定义触发：

当我们处理某些数据的时候，本来数据本身是没有错误的，但根据具体应用（业务）的需要，会要求数据满足某种条件，而该数据并不满足的时候，我们就可以在程序中“主动”去触发（创建）一个错误，以表明该数据的“非法性”。
语法形式：
trigger_error(“错误提示信息内容”，3中用户错误代号之一);
其中触发了用户的致命错误(E_USER_ERROR)，也会终止程序的后续执行。

示例：

```php
<?php
header('Content-type:text/html; charset=utf-8');
?>
<?php
echo "<br/>cc12";

$age = 900;
if($age >127 || $age <0){

    trigger_error("age error",E_USER_ERROR);//异常数据处理
}
else {
    echo "你的年龄是：$age";//正常数据处理
}
echo "ccs35";

?>
```

```
cc12
Fatal error: age error in E:\wamp\zhandian\error1.php on line 10
```

- 错误报告的显示问题

所谓错误报告，就是显示在网页上的错误提示内容！
有关错误报告，有2个问题需要处理：
是否显示错误报告（display_errors）：
有2种做法可以来设定是否显示：
做法1：在php.ini文件中，设定display_erros的值，为on（显示），或为off（不显示）
注意：前提条件都是我们apache已经装载了php.ini文件——这一点，需要在apache的配置文件httpd.config中加入如下一行：
PHPIniDir “php.ini文件的位置（路径）”

```
544  ; leave this setting off on production servers.
545  ; Default Value: Off
546  ; Development Value: On
547  ; Production Value: Off
548  ; http://php.net/display-startup-errors
549  display_startup_errors = On
550
551  ; Besides displaying errors, PHP can also log errors to locations such as
552  ; server-specific log, STDERR, or a location specified by the error_log
553  ; directive found below. While errors should not be displayed on productio
554  ; servers they should still be monitored and logging is a great way to do
555  ; Default Value: Off
```

方法2：
直接在php的脚本文件中设使用函数ini_set()来对其进行设置：

```
1  <?php
2  header('Content-type:text/html; charset=utf-8');
3  ?>
4  <?php
5  ini_set("display_errors",0);
6  echo "<br/>cc12";
7
8  $age = 900;
9  if($age >127 || $age <0){
10
11     trigger_error("age error",E_USER_ERROR);//异常数据处理
12 }
13 else {
14     echo "你的年龄是：$age";//正常数据处理
15 }
16 echo "ccs35";
17
18 ?>
```

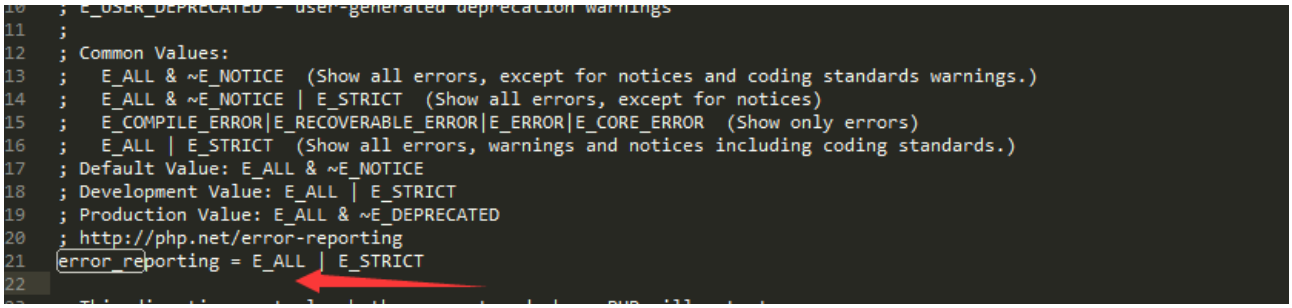
当然，如果设置为1，就是显示！

```
cc12
```

注意：
1，不管哪种形式，该单词是一样的：display.errors
2，使用php.ini配置，影响的是全局（即所有php网页）；
3，在某个脚本代码中使用ini_set()设置，就只影响该脚本代码本身——这是常用的方式。
4，脚本中的设置优先于php.ini中的设置。

- 显示哪些级别的错误报告（error_reporting）：

显然，前提是“display_errors”设置为On(或1），表示可以显示。
显示哪些级别的错误报告，也有2个做法：
方法1：



这个值目前代表“所有错误”，都显示。
修改为：error_reporting = E_NOTICE　此时就只显示E_NOTICE级别的错误
error_reporting = E_NOTICE | E_WARING | E_ERROR　　//显示该3种；
error_reporting = E_ERROR | E_USER_ERROR　　//显示该2种严重错误
要想代表真正的“所有错误”，应该写为：E_ALL | E_STRICT
方法2：在当前的脚本代码中：
跟php.ini中设置其实是一样，举一些例子如下：
ini_set(“error_reporting”, E_NOTICE);　//就显示该一个级别的错误
ini_set(“error_reporting”, E_NOTICE | E_WARNING), //显示2个级别
ini_set(“error_reporting”, E_NOTICE | E_WARNING | E_ERROR), 　//显示3个级别
ini_set(“error_reporting”, E_ALL | E_STRICT), 　//这才代表显示所有错误！

错误日志的记录问题
错误日志其实就是错误报告，只是它会“写入文件中”，此时就称为错误日志！
也有2个问题，每个问题也有2种做法：
是否记录log_errors：

php.ini中：log_errors = On　或 Off
脚本中：ini_set(“log_erros”, 1); 或 0

补充一句：
1：ini_set(“php配置项”，值)；　//用于脚本中设置php.ini中是某项的值。
2,：$v1 = ini_get(“php配置项”)；//用于获取php.ini中是某项的值

记录到哪里error_log：
一般就只有2个写法：
写法1：直接使用一个文件名，此时系统会自动在每个文件夹下都建立该文件名，并用其记录该文件夹下的所有网页文件发生的错误信息。
在PHP脚本语句中
ini_set(“error_log”,”my_error.txt”);//记录到该文件里

写法2：使用一个特殊的名字“syslog”，则此时所有错误信息都会记录到系统的“日志文件”中。
系统日志文件在这里：控制面板》管理工具》事件查看器》window日志》应用程序：
ini_set(“error_log”,”syslog”);//记录到系统文件里

- 定义错误处理器

什么叫错误处理器？
就是一旦发生错误，用来处理该错误的一种“机器”——其实就是一个函数。
自定义错误处理，就是指：让系统不要去处理错误了，而完全由我们（开发者）来对错误进行处理：显示和记录。

做法2步：
第一步：
设定要用于处理错误的函数名！
set_error_handler(“f1”);
第二步：

去定义该函数！
function f1(){
//这里可以任意写代码：自然正常是去显示错误报告，和记录错误日志。
}
示例：

```php
echo "<br />E_ALL=" . E_ALL. ", \t\t其对应二进制值为：" . GetBinStr(E_ALL);
echo "</pre>";

//自定义错误处理器



//我们准备要自己来定义错误“处理器”了：
//第1步：设定要作为错误处理的函数名：
set_error_handler("my_error");
//第2步：定义该函数：
//该函数需要定义4个形参，分别代表：
//$errCode：代表错误代号（级别）
//$errMsg：  代表错误信息内容
//$errFile：代表发生错误的文件名
//$errLine：代表发生错误的行号
//注意，该函数我们不要在程序中调用，而是，一发生错误就会被自动调用
//而且会传入该4个实参数据
function my_error($errCode,$errMsg,$errFile,$errLine){
        echo "<h1/>cuowu ";
    $str = "";
    $str .= "<p><font color='red'>大事不好，发生错误：</font>";
    $str .= "<br />错误代号为：" . $errCode;
    $str .= "<br />错误内容为：" . $errMsg;
    $str .= "<br />错误文件为：" . $errFile;
    $str .= "<br />错误行号为：" . $errLine;
    $str .= "<br />发生时间为：" . date("Y-d-m H:i:s");
    $str .= "</p>";
    echo $str;  //输出该“构建”的错误完整处理结果
    //也可以将该内容“写入”到某个文件中去，也就是所谓记录错误日志！
    //但，今天我们就做不到了—这涉及到文件操作！
}



echo "$mm";
?>
```