流程控制

循环结构

循环的中断

循环中，有两种中断语句可以使用：

break：
　　用于完全终止某个循环，让执行流程进入到循环语句后面的语句；

continue：
　　用于停止当前正在进行的当次循环，而进入到循环的“下一次”过程中去（通常就是循环的开始位置）；

在 php 中，该两个循环有更强的能力：中断“更多层”的循环，语法如下：

```
break    正整数 n;              //比如 1,2，3；
continue    正整数 n;           //比如 1,2，3；
```

循环的“层”，指的是从当前中断语句（break 或 continue）算起，往代码的“外部”数循环的个数，就是层数。比如：

```
for(...){          //循环 1
    for(...){          //循环 2
        for(...){          //循环 3
            break    2;   //此时会中断循环 2：其实指中断“2 层”
                          //对此 break 语句，循环 3 是其“第一层”，循环 2 是其第 2 层，循环 1 是其第 3 层
        }
        contine    2;    //此时会中断循环 12：其实指中断“2 层”
                          //对此 contine 语句，循环 2 是其“第 1 层”，循环 1 是其第 2 层，
    }
}
```

do　while 循环：

for 循环语句形式：

```
for（【循环变量初始化】;　【循环变量的条件判断】;　【循环变量的改变】){
//循环体语句。。。。
}
```

while 循环语句形式：
【循环变量初始化】
while(【循环变量的条件判断】){
//循环体语句。。。
【循环变量的改变】
}
do while 循环语句形式：
【循环变量初始化】
do {
//循环体语句。。。
【循环变量的改变】
}while(【循环变量的条件判断】);

说明：
1，do while 会先进入循环体执行一次（不判断条件）；
2，然后，判断循环条件是否满足，如果满足，又会回到 do 的开始位置（进入循环体）执行——这就是循环的正常情况。
3，如果不满足，就结束循环。

流程控制的替代语法

- if (...) : 。。。。 endif;
- if (...) : 。。。else: 。。。 endif;
- if (...): 。。。elseif(...) : 。。。 elseif(...): 。。。 else: 。。。 endif;
- switch(...) : case ... case ... endSwitch;
- while(...): 。。。endwhile;
- for(...; ...; ...): 。。。 endfor;

goto 语句：

学此语句的目的是：不要用它！
语法形式：

//程序从这里开始：

标识符 2：
　　语句群 1.。。。。。

goto 标识符 1;　　//含义是：立即跳转到标识符 1 所在位置的下一行继续执行
　　语句群 2.。。。。。

标识符 1:
　　语句群 3.。。。。。

goto　标识符 2:;

控制脚本执行顺序

die(字符串)/exit(字符串)：

输出该字符串后，立即停止 php 的执行！即后续程序不再执行，包括后续的其他所有 php 和 html 代码部分。

exit 是 die 的同义词。他们也可以不加字符串，而是直接停止。

sleep（$n)

让程序停止运行指定的秒数。然后等待过了那个时间后，就继续运行！

注意，其单位是“秒”；

文件加载

综述和基本语法：

1，有 4 个文件加载语句：include, require, include_once, require_once
2，他们的使用形式完全一样，比如：　include　“要加载的文件路径”; 或：　include (“要加载的文件路径”);
3，他们的含义也几乎完全一样：只是在加载失败时或是否重复加载这种情况，有所不同。
4，他们可以载入 php 或 html 文件；

文件加载的路径问题：

前提说明：以下的说明举例，以 include 为例，也适用于其他 3 个加载语句；

有 3 中路径形式可以使用：

相对路径：

是相对于当前网页文件所在的位置来定位某个被加载的文件位置，主要依赖以下 2 个特殊的路径符号：

./ ：表示当前位置，即当前网页文件所在的位置（目录）；

../ ：表示上一级位置，即当前网页文件所在的位置的上一级位置（目录）；

我们需要用这 2 个符号来表达位置信息，比如：

include ‘./page1.php’; //表示当前网页文件所在位置的 page1.php 文件；

include ‘../page2.php’;

include ‘../ab/page3.html’;

绝对路径：

绝对路径又分 2 种：

本地绝对路径：

比如：

include “c:/d1/d2/p1.php”;

include “f:/f1/abc/p2.html”;

特别注意：我们其实几乎都不应该在代码中直接写这种本地绝对路径！

但，其实我们这种本地绝对路径的写法是很常用的！

网络绝对路径：

比如：

include “http://www.abc.com/p1.php”;

include “http://www.baidu.com/index.php”;

“无路径”（不推荐）：

形式就是没有给出路径信息，而只给出文件名，我们不推荐。

比如：include ‘page1.php’; //此时通常其实 php 语言引擎会在当前网页目录下找该文件。

文件载入和执行过程详解

- 第 1 步：从 include 语句处退出 php 脚本模式（进入 html 代码模式）
- 第 2 步：载入 include 语句所设定的文件中的代码，并执行之（如同在当前文件中一样）
- 第 3 步：退出 html 模式重新进入 php 脚本模式，继续执行之后的代码

4 个载入语句的区别

include 和 require 的区别：

include 载入文件失败时（即没有找到该文件），报一个“提示错误”，然后继续执行后续代码；
require 载入文件失败时，报错并立即终止执行。
通常，require 用于在程序中，后续的代码依赖于载入的文件的时候。

include_once 和 require_once 的区别：

同 include 和 require 的区别：

include 和 include_once 的区别：

include 载入的文件不判断是否重复，只要有 include 语句，就会载入一次——即此时可能导致重复载入。

include_once 载入的文件会有内部判断机制是否“前面代码”已经载入过，如果载入过，就不再载入。

require 和 require_once 的区别：

同 include 和 include_once 的区别。