# HTTP协议

## HTTP协议初步认识

### HTTP协议概念

HTTP协议，即超文本传输协议(Hypertext transfer protocol)。是一种详细规定了浏览器和万维网(WWW = World Wide Web)服务器之间互相通信的规则，通过因特网传送万维网文档的数据传送协议。

HTTP协议是用于从WWW服务器传输超文本到本地浏览器的传送协议。它可以使浏览器更加高效，使网络传输减少。它不仅保证计算机正确快速地传输超文本文档，还确定传输文档中的哪一部分，以及哪部分内容首先显示(如文本先于图形)等。

### HTTP协议特点

1. 客户/服务器模式：客户端（浏览器）/服务端
2. 简单快速：客户向服务器请求服务时，只需传送请求方法和路径。由于HTTP协议简单，使得HTTP服务器的程序规模小，因而通信速度很快。
3. 灵活：HTTP允许传输任意类型的数据对象（MIME类型）

4）无连接：无连接的含义是限制每次连接只处理一个请求。服务器处理完客户的请求，并收到客户的应答后，即断开连接。采用这种方式可以节省传输时间。

5）无状态：HTTP协议是无状态协议。无状态是指协议对于事务处理没有记忆能力。缺少状态意味着如果后续处理需要前面的信息，则它必须重传，这样可能导致每次连接传送的数据量增大。另一方面，在服务器不需要先前信息时它的应答就较快。

### HTTP协议分类

1）http请求协议：浏览器向服务器发起请求的时候需要遵循的协议

2）http响应协议：服务器向浏览器发起响应的时候需要遵循的协议

## HTTP请求

### 请求行

1. 形式：请求方式 资源路径 协议版本号
2. GET /index.php HTTP/1.1

最早的时候HTTP协议有过1.0，请求行独占一行（第一行）

### 请求头

请求头就是各项协议内容：具体的协议内容不会每次都使用全部

1）Host：请求的主机地址（必须）

2）Accept：当前请求能够接收服务器返回的类型（MIME类型）

3）Accept-Language：接收的语言

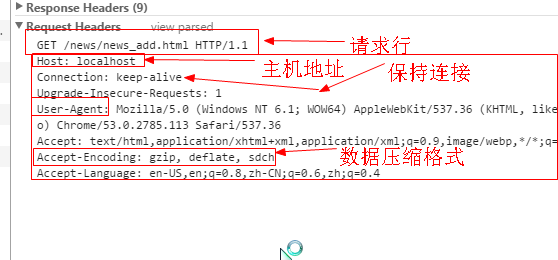
4）User-Agent：客户浏览器所在点的一些信息

请求头不固定数量，每个请求协议也是独占一行，最后会有一行空行（用来区分请求头和请求体）

### 请求体

请求数据：POST请求会有请求体。GET请求所有的数据都是跟在URL之后，会在请求行中的资源路径上体现。

基本格式：资源名字=资源值&资源名字=资源值…



## HTTP响应

### 响应行

1）形式：协议版本号 状态码 状态消息（独占一行）

HTTP/1.1 200 ok

2）200 ok： 成功

3）403 Forbidden： 没权限访问

4）404 Not Found： 未找到页面

5）500 Server Internal Error： 服务器内部错误

### 响应头

具体协议内容

1）时间：Wed, 16 Sep 2017 11:43:33 GMT

2）服务器：Server: Apache/2.2.22 (Win32) PHP/5.3.13

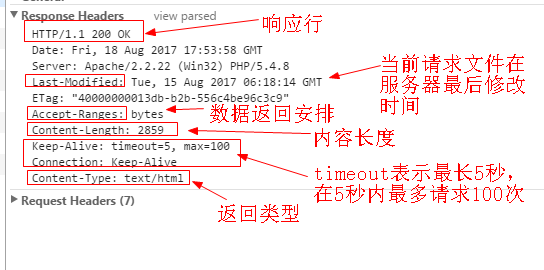
3）内容长度：Content-Length: 1571，数据具体的字节数（响应体）

4）内容类型：Content-Type: text/html：告诉浏览器对应的数据格式

列举了几个常见的响应头，并不是全部：响应头一个占一行，最后一行空行（区分响应头和响应体）

### 响应体

实际服务器响应给浏览器的内容



### 常用HTTP状态码

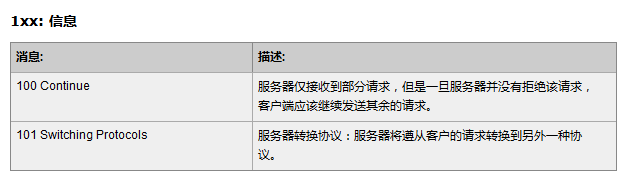
状态码200：成功

状态码403：forbidden，拒绝访问（没有权限）

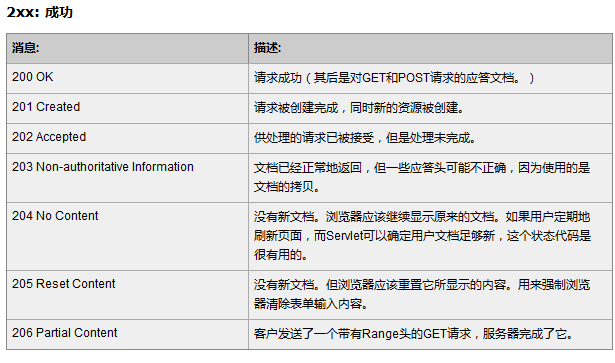
状态码404：NOT FOUND，找不到

状态码500：服务器问题

服务器正在处理过程中



服务器正常且正确处理



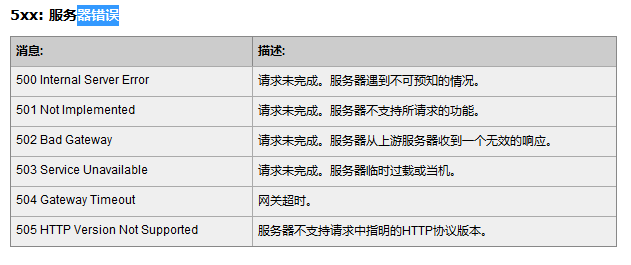
请求的目标已经转移或者需要更新



客户端出错了



服务器出错



## HTTP响应

### 常见HTTP响应设置及使用

PHP中针对HTTP协议（响应）进行了底层设计，可以通过函数header来实现修改HTTP响应（响应头）

注意事项：

1、 Header可以设计HTTP响应，因为HTTP协议特点是：响应行，响应头（空行结尾），响应体。认为通过header设计响应头的时候，不应该有任何内容输出，所以一旦产生内容输出（哪怕一个空格），系统都会认为响应头已经结束而响应体开始了，所以如果先输出内容后设置响应头（header使用），理论设置无效；

2、 在PHP5以后，增加程序缓存内容：允许服务器脚本在输出内容的时候，不直接返回浏览器而是先在服务器端使用程序缓存保留（php.ini中使用output\_buffering），有了该内容之后，在程序缓存内会自动调整响应头和响应体（允许响应头在已经输出的内容之后再设置），但是此时会报错（警告）。

总结：header设置响应体之前不要有任何输出

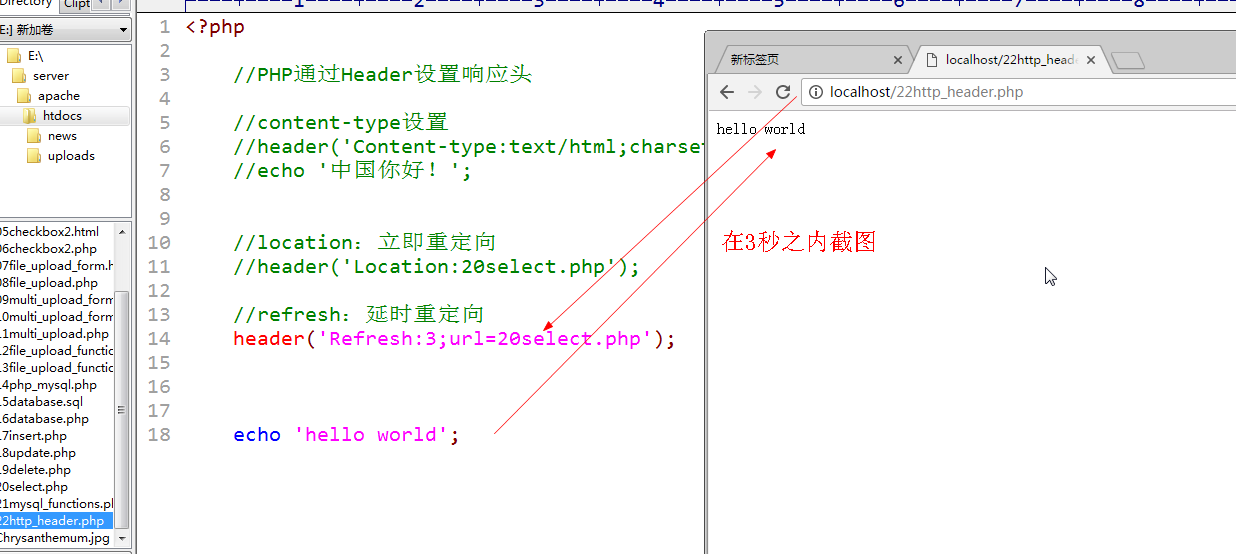
Location：重定向，立即跳转（响应体不用解析）

浏览器在解析服务器响应的时候：先判定响应行，继续响应头，最后响应体：location是在响应头中，所以浏览器一旦见到该协议项，不再向下解析。



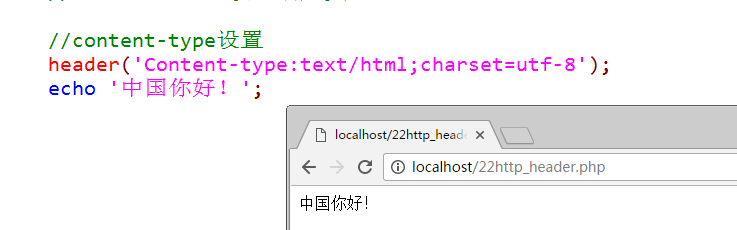
Refresh：重定向，定时跳转（响应体会解析）

延时重定向：浏览器会根据具体时间延迟后在访问指定跳转链接：浏览器在准备跳转访问之前，会继续解析HTTP协议（响应头和响应体）



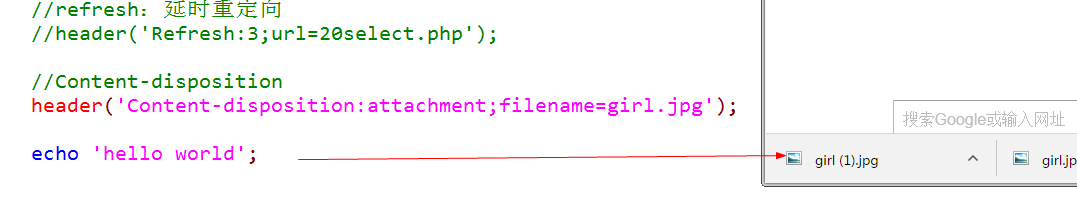
Content-type：内容类型，MIME类型

通过内容告知（MIME类型），浏览器正确解析内容



Content-disposition：内容类型，MIME类型扩展，激活浏览器文件下载对话框

浏览器在解析内容的时候，默认是直接解析：那么有时候需要浏览器不解析，当做内容下载成文件



## PHP模拟HTTP请求

### 原理

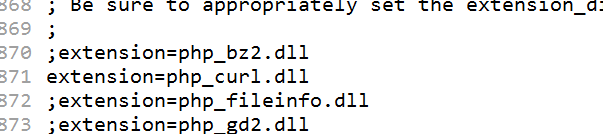
PHP可以通过模拟HTTP协议发起HTTP请求

CURL是一个非常强大的开源库，支持很多协议，包括HTTP、FTP、TELNET等，我们使用它来发送HTTP请求。它给我 们带来的好处是可以通过灵活的选项设置不同的HTTP协议参数，并且支持HTTPS。CURL可以根据URL前缀是“HTTP” 还是“HTTPS”自动选择是否加密发送内容。

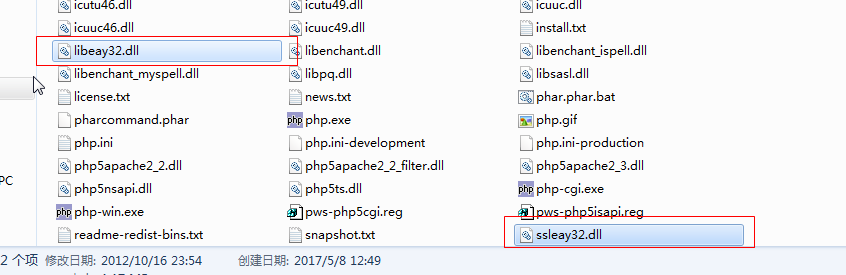
前提条件：HTTP协议的客户端/服务端模式，HTTP协议不局限于一定要浏览器访问

### Curl扩展库使用

1、 开启CURL扩展

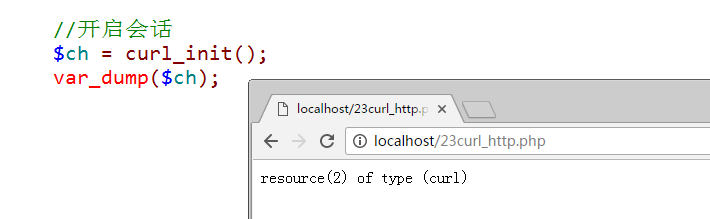


2、 有的时候PHP版本会出现即使开启了扩展也无法使用扩展的可能：因为CURL找不到对应的dll文件：需要将相关的DLL文件放到C:windows



3、 重启Apache应用

1）建立连接：curl\_init()：激活一个CURL连接功能



2）设置请求选项：curl\_setOpt()：设定选项

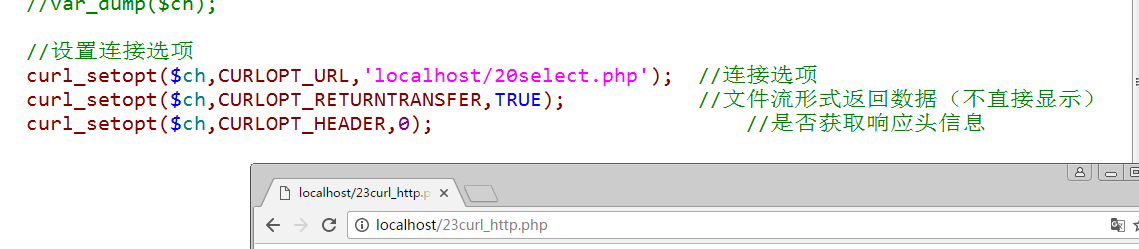
CURLOPT\_URL：连接对象

CURLOPT\_RETURNTRANSFER:将服务器执行的结果（响应）以文件流的形式返回给请求界面（PHP脚本）

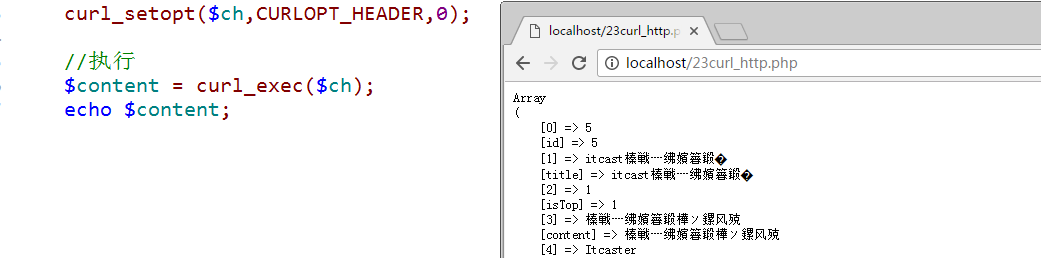
CURLOPT\_POST：是否才有POST方式发起请求（默认请求是GET）

CURLOPT\_POSTFIELDS：用来传递POST提交的数据，分为两种方式：字符串（name=abc&password=123）以及数组形式（array(‘name’=>’abc’,…)）

CURLOPT\_HEADER：是否得到响应的header信息（响应头），默认不获取



3）执行请求：curl\_exec()：执行选项（与服务器发起请求），得到服务器返回的内容



乱码原因：20select.php中的内容有告知浏览器，但是当前是被服务器脚本23curl\_http.php访问的，没有做解析；输出给浏览器之后，需要当前23curl\_http.php告知浏览器对应的字符集

4）关闭连接：curl\_close()：关闭资源



# 文件编程

## 文件编程初步认识

### 文件编程的必要性

文件编程指利用PHP代码针对文件（文件夹）进行增删改查操作。

在实际开发项目中，会有很多内容（文件上传、配置文件等）具有很多不确定性，不能在一开始就手动的创建，需要根据实际需求和数据本身来进行管理，这个时候就可以使用PHP文件编程来实现代码批量控制和其他操作。

### 文件操作的分类

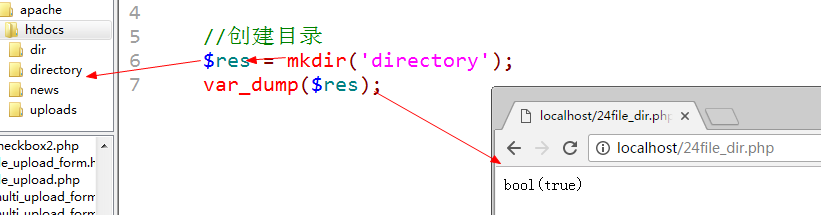
1） 目录操作：文件夹，用来存放文件的特殊文件

2） 文件操作：用来存放内容

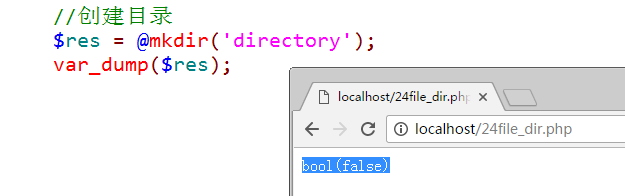
## 目录操作

### 文件操作创建目录结构

1）mkDir(路径名字)：创建成功返回true，创建失败返回false

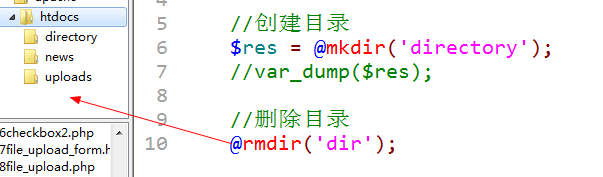


有些操作为的就是得到一个想要的结果，如果结果本身就存在，那么可以忽略得到过程的错误：抑制错误



### 删除目录

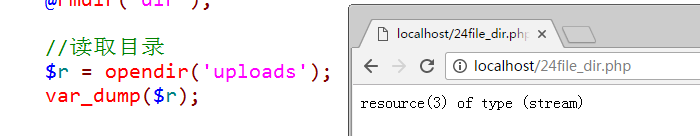
1）rmDir(指定文件夹路径)：移出文件夹



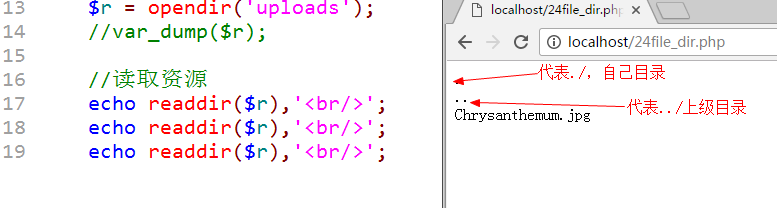
### 读取目录

读取方式：将文件夹（路径）按照资源方式打开

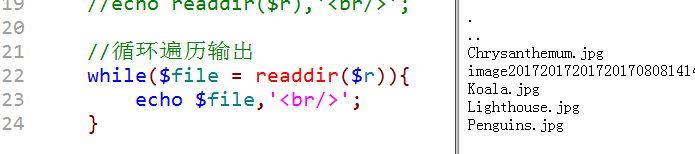
1）openDir()：打开资源，返回一个路径资源，包含指定目录下的所有文件（文件夹）



2）readDir()：从资源中读取指针所在位置的文件名字，然后指针下移，直到指针移出资源

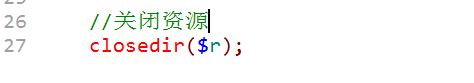


读取所有内容：遍历操作



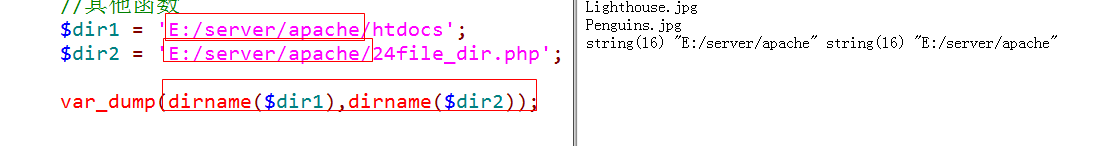
### 关闭目录

1）closeDir()：关闭资源

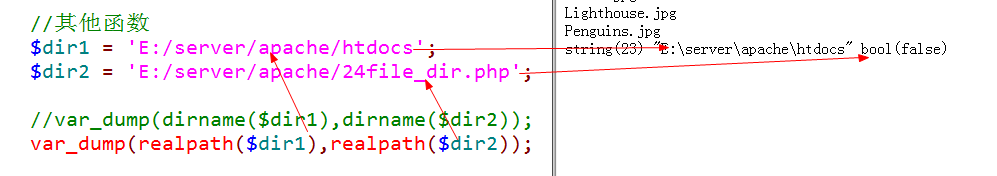


### 其他目录操作

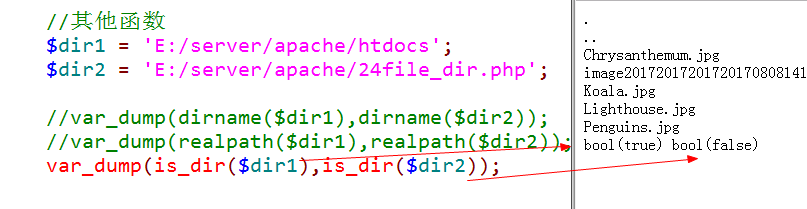
1）dirName(一个路径)：得到的是路径的上一层路径



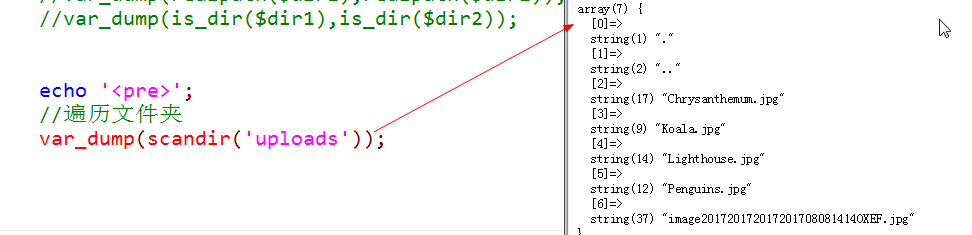
2）realPath(一个路径)：得到真实路径（目录路径），如果是文件那么得到的结果是false



3）is\_dir()：判断指定路径是否是一个目录



4）scandir()：封装版的opendir\readdir\closedir，获取一个指定路径下的所有文件信息，以数组形式返回



## 目录操作

### 递归遍历目录

递归遍历目录：指定一个目录的情况下，将其下的所有文件和目录，及其目录内部的所有内容都输出出来。

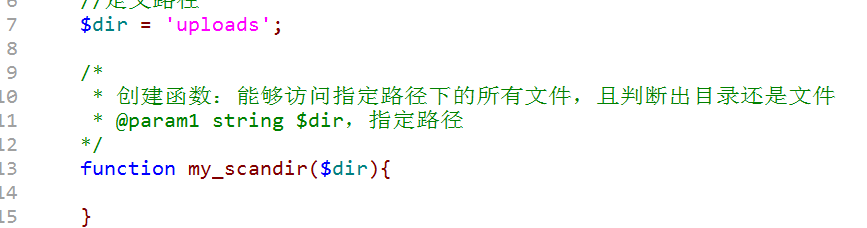
递归算法：将大问题切成相似的小问题（最小单位），然后可以调用解决大问题的方法来解决小问题。

递归函数：函数如果自己内部调用自己，该函数称之为递归函数。

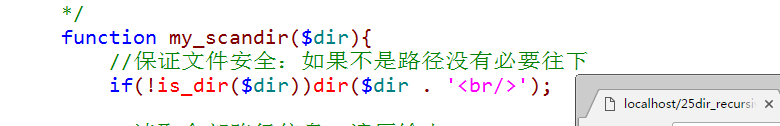
递归遍历目录的思维逻辑

1、 设计一个能够遍历一层文件的函数

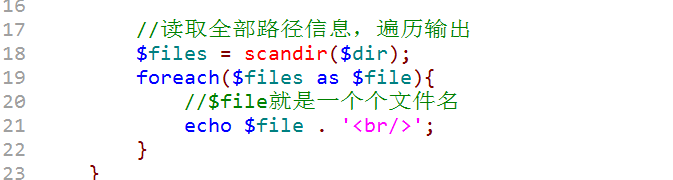
a. 创建函数



b. 安全判定：是路径才访问

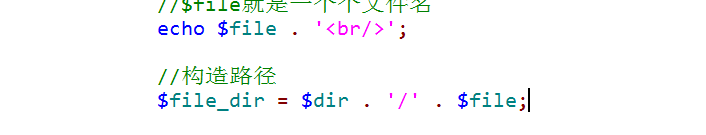


c. 读取全部内容，遍历输出

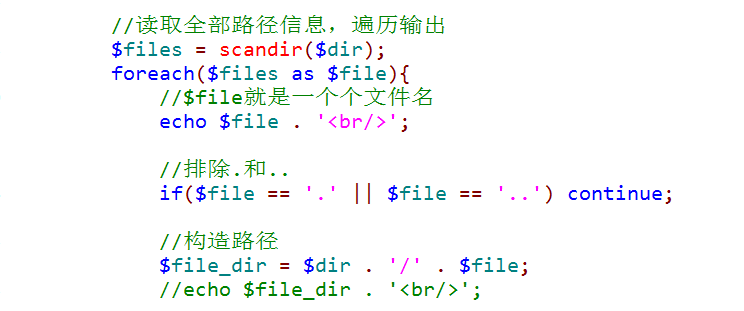


2、 找到递归点：遍历得到的文件是目录，应该调用当前函数（调用自己）：

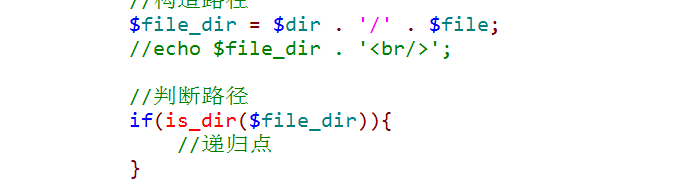
a. 需要构造路径（遍历得到的结果只是文件的名字）



b. 需要注意排除.和..



c. 判断是路径还是文件



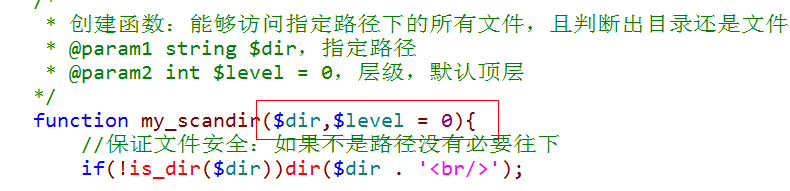
d. 递归调用函数



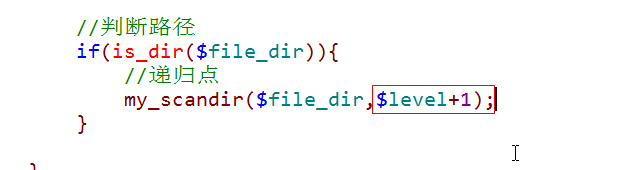
3、 找到递归出口：遍历完这个文件夹之后，发现没有任何子文件夹（函数不再调用自己）：自带递归出口

4、 如何显示层级关系？函数第一次运行遍历的结果是最外层目录，内部调用一次说明进入一个子目录，子目录再调用一次函数进行孙子目录…如果能够在第一次调用的时候给个标记，然后在进入的时候，通过标记的变化来识别层级关系，就可以达到目的：该标记还能代表层次关系：缩进

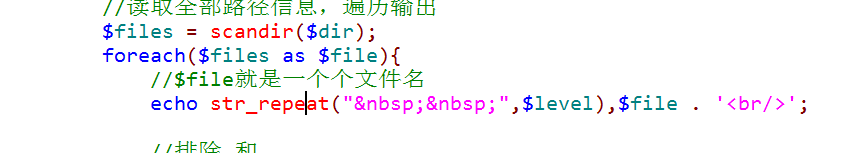
a. 在函数参数中增加一个标记：默认值为0



b. 递归调用函数的时候也需要使用该参数：但是是属于当前层级的子层，所以+1



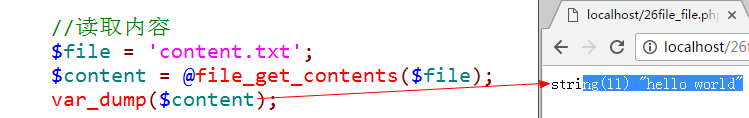
c. 根据层级来实现缩进：str\_repeat()



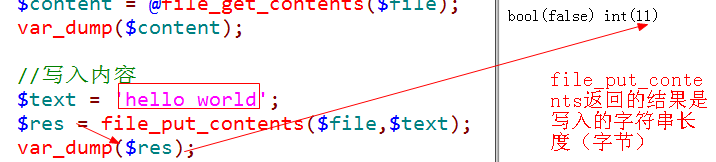
## 文件操作

### PHP5常见文件操作函数

1）file\_get\_contents(文件路径)：获取指定文件的所有内容，如果路径不存在最好做安全处理



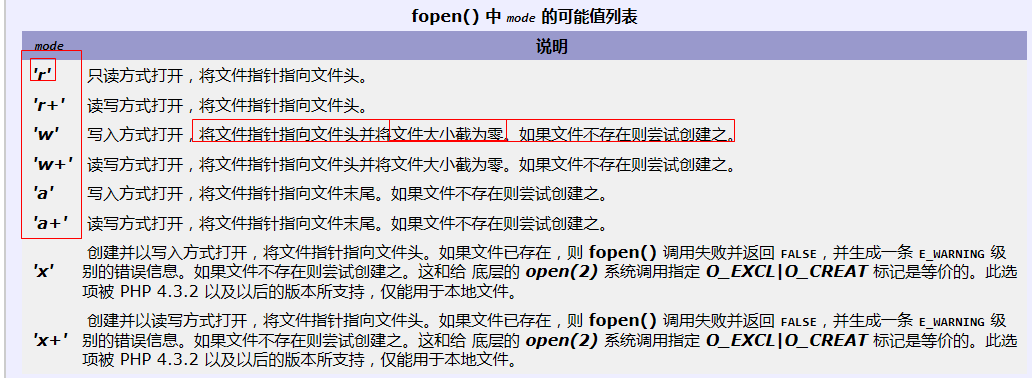
2）file\_put\_contents(文件路径，内容)：将指定内容写入到指定文件内：如果当前路径下不存在指定的文件，函数会自动创建（如果路径不存在，不会创建路径）

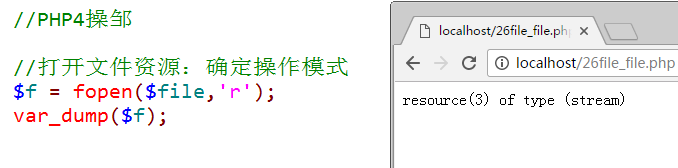


### PHP4常见文件操作函数

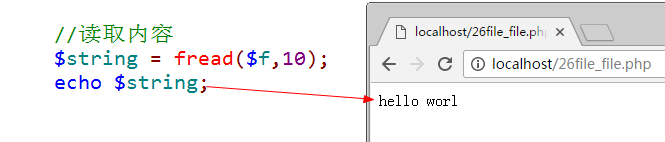
PHP4中是将文件操作用资源形式处理：不论是读还是写都依赖资源指针：文件内容中指针所在位置。

1）fopen(文件路径，打开模式)：打开一个文件资源，限定打开模式

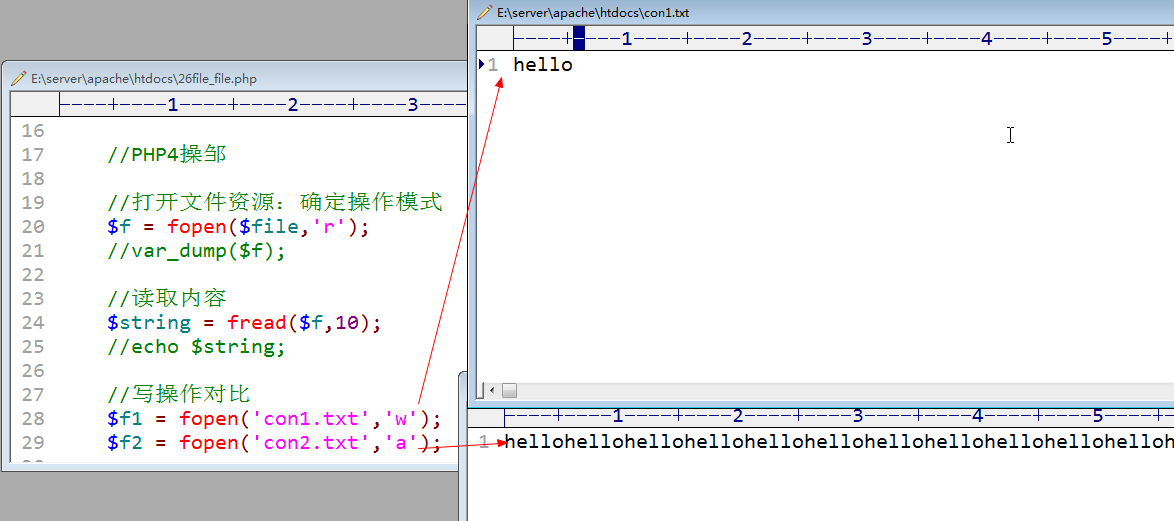




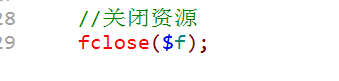
2）fread(资源，长度)：从打开的资源中读取指定长度的内容（字节）



3）fwrite(资源，内容)：向打开的资源中写入指定的内容



4）fclose(资源)：关闭资源



### 其他文件操作函数

1）is\_file()：判断文件是否正确（不识别路径）

2）filesize()：获取文件大小

3）file\_exists()：判断文件是否存在（识别路径）

4）unLink()：取消文件名字与磁盘地址的连接（删除文件）

5）filemtime()：获取文件最后一次修改的时间

6）fseek()：设定fopen打开的文件的指针位置

7）fgetc()：一次获取一个字符

8）fgets()：一次获取一个字符串（默认行）

9）file()：读取整个文件，类似file\_get\_contents，区别是按行读取，返回一个数组

## 文件操作

### 文件下载

文件下载：从服务器将文件通过HTTP协议传输到浏览器，浏览器不解析保存成相应的文件。

提供下载方式可以使用HTML中的a标签：<a href=”互联网绝对文件路径”>点击下载</a>

1、 缺点1：a标签能够让浏览器自动下载的内容有限：浏览器是发现如果解析不了才会启用下载

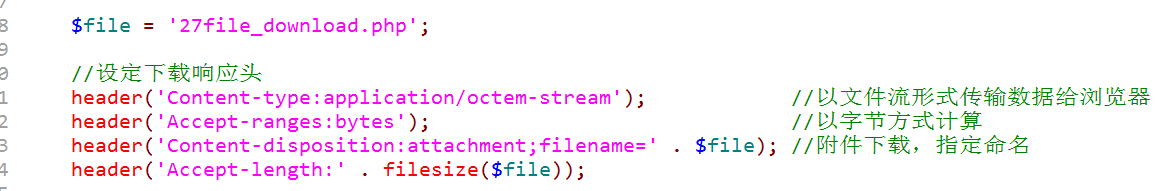
2、 缺点2：a标签下载的文件存储路径会需要通过href属性写出来，这样会暴露服务器存储数据的位置（不安全）

PHP下载：读取文件内容，以文件流的形式传递给浏览器：在响应头中告知浏览器不要解析，激活下载框实现下载。

1. 指定浏览器解析字符集

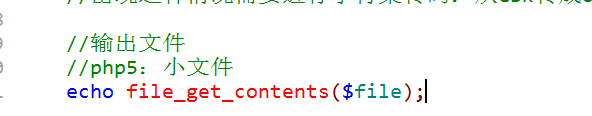


1. 设定响应头
   1. 设定文件返回类型：image/jpg||application/octem-stream
   2. 设定返回文件计算方式：Accept-ranges：bytes
   3. 设定下载提示：Content-disposition:attachment;filename=’文件名字’
   4. 设定文件大小：Accept-length：文件大小（字节）



1. 读取文件
2. 输出文件

方案1：如果文件较小，可以使用PHP5的文件函数操作：file\_get\_contents



方案2：文件比较大（网络不好），可以使用PHP4 的文件操作方式：一次读一点

