表单传值

概念

表单传值即浏览器通过表单元素将用户的选择或者输入的数据提交给后台服务器语言。

为什么使用表单传值

动态网站（Web2.0）的特点就是后台根据用户的需求定制数据，所谓的“需求”就是用户通过当前的选择或者输入的数据信息，表单就是这些数据的承载者。

表单传值方式

GET 传值

1）form 表单
<form method=”GET”>表单元素</form>
2）a 标签
<a href=”www.itcast.cn/index.php?学科=PHP”>
3）location 对象的 href 属性
<script>location.href=”www.itcast.cn/index.php?data=PHP”</script>
4）location 对象的 assign()方法
<script>location.assign(“www.itcast.cn/index.php?data=PHP”)</script>

POST 传值

1）post 表单方式的基本设定
<form method=”POST”>表单元素</form>
2）post 方式跟 get 方式的区别
1、 Get 传输的数据主要用来获取数据，不改变服务器上资源：get 只是用来获取内容
2、 Post 传输的数据主要用来增加数据，改变服务器上资源：POST 会改变服务器上数据内容
3、 传输方式上 post 必须使用 form 表单，而 get 可以使用 form 表单和 URL
4、 get 传输数据可以在 URL 中对外可见，而 post 不可见：GET 传值最终会在浏览器的地址栏中全部显示：?数据名=数据值&数据名 2=数据值 2…
5、 get 和 post 能传输的数据大小不同，get 为 2K，post 理论无限制（事实上，GET 和 POST 本身没有数据长度限制，但是浏览器厂家做了一些限制）

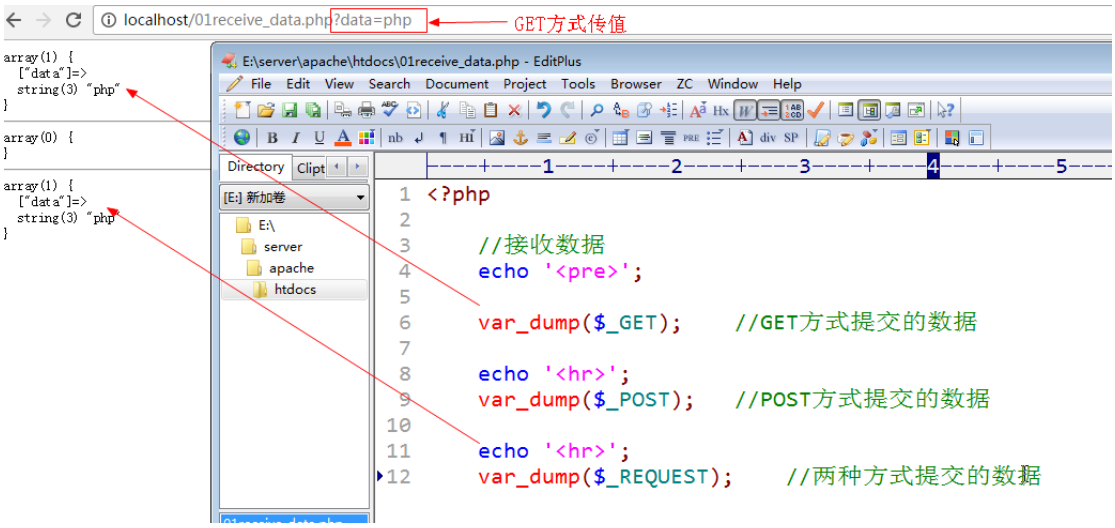
6、 get 和 post 能够传输的数据格式有区别：get 传输简单数据（数值/字符串），post 可以提交复杂数据（二进制等）

PHP 接收数据的三种方式

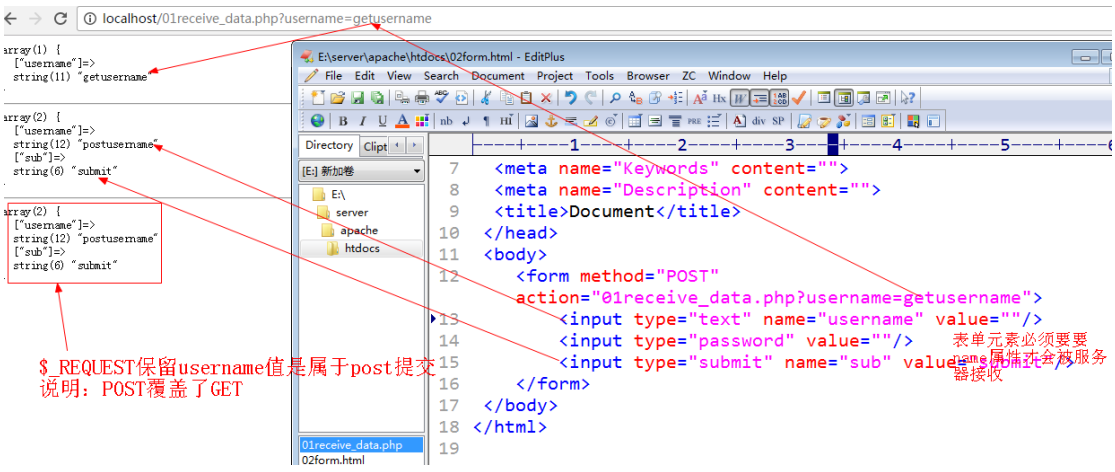
不管是$_GET/$_POST/$_REQUEST，三个都是 PHP 超全局（没有范围限制）预定义数组，表单元素的“name”属性的值作为数组的下标，而 value 属性对应的值就是数组的元素值

$_GET 方式：接收 GET 方式提交的数据
$_POST 方式：接收 POST 方式提交的数据
$_REQUEST 方式：接收 POST 或者 GET 提交的所有数据
1）$_REQUEST 所存储数据的内容：将$_POST 和$_GET 合并存储到一个数组
2）$_REQUEST 和$_POST 与$_GET 的联系：如果 GET 和 POST 中有同名数组元素（下标），POST 会覆盖 GET（PHP 中数组元素下标具有唯一性），这个可以在 php.ini 中进行配置

GET/POST/REQUEST 关系



证明在 REQUEST 中 POST 会覆盖 GET



PHP 处理复选框数据

复选框表单项的命名方式

复选框：通常是将一类内容以同样（同名）的形式传递给后台，数据库存储通常是一个字段存储。复选框的特点：选中才会提交

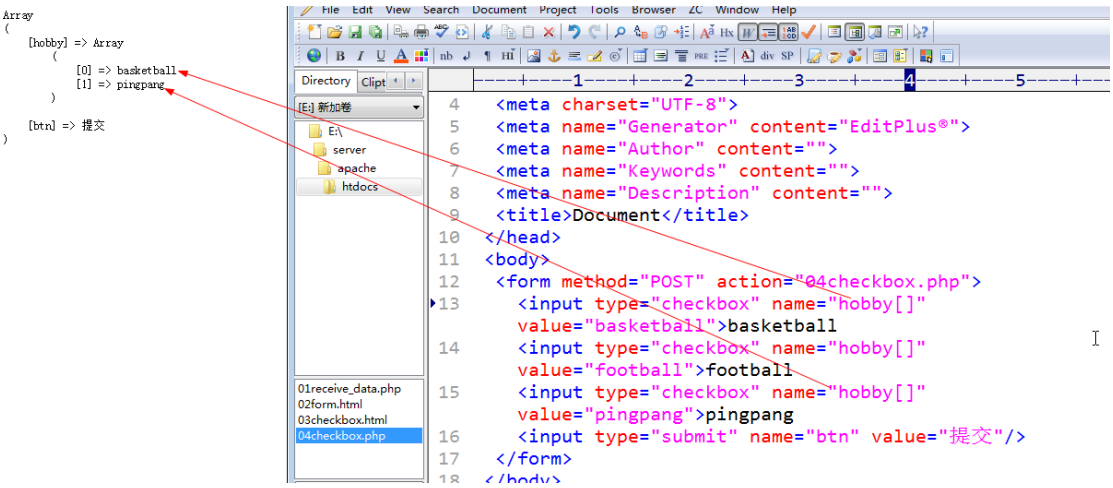
1、 在浏览器端，checkbox 的 name 属性的值不论什么都会被浏览器毫无保留的提交
2、 在 PHP 中$_POST/$_GET 都会对同名 name 属性进行覆盖

解决方案：浏览器不识别[]（浏览器不认为有特殊性），但是 PHP 认为[]有特殊性：系统自动认为该符号是数组的形式，所以 PHP 就会自动的将同名的但是带有[]的元素组合到一起形成一个数组

```
</head>
<body>
 <form method="POST" action="04checkbox.php">
   <input type="checkbox" name="hobby[]" value="basketball">basketball
   <input type="checkbox" name="hobby[]" value="football">football
   <input type="checkbox" name="hobby[]" value="pingpang">pingpang
   <input type="submit" name="btn" value="提交"/>
 </form>
</body>
```

复选框数据的接收形式

PHP 会自动组合同名元素的为数组



PHP 处理复选框数据

复选框数据的常见处理

1）单选按钮的数据处理

Radio button：可以出现多个选择项，但是只能选择其中一个

1、 表单中使用的 name 属性，使用同名即可：只能选中一个

2、 后台接收数据也不需要额外处理

3、 数据库存储的话只需要一个字段存储普通数据即可（数字或者字符串）

```
6
7       <input type="radio" name="gender" value="1"
        checked="checked"/>男
8       <input type="radio" name="gender" value="2"/>女
9
```

4、 PHP 拿到数据之后，组织 SQL 直接存储到数据表即可

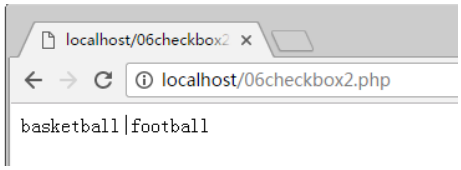
2）多选按钮的数据处理

1、 表单中 name 属性使用数组格式：名字[]（一类复选框数据使用一个）

2、 后台接收到数据之后，是一个数组（数组不能存储到数据库）

3、PHP 需要将数组转换成指定格式的字符串：使用分隔符分隔每一个元素并且形成字符串：implode(’ 分隔符’ ,数组)

```
14
15      //数组转换成有格式的字符串
16      $hobby_string = implode($hobby,'|');
17      echo $hobby_string;
18
```

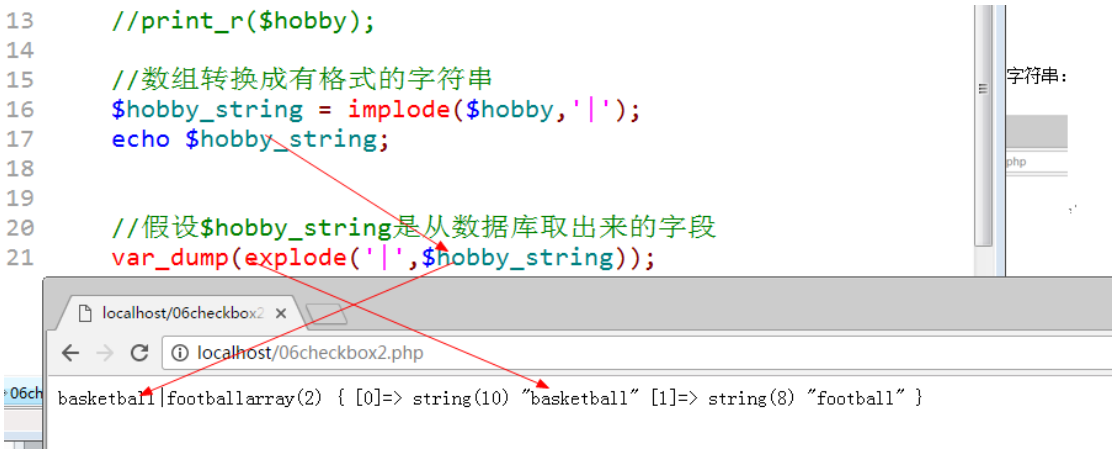


4、 PHP 组织 SQL 直接存储到数据库

取出来复选框数据显示

1、 如果是反过来操作，那么取出数据之后使用 explode 把字符串变成数组

```
13      //print_r($hobby);
14
15      //数组转换成有格式的字符串
16      $hobby_string = implode($hobby,'|');
17      echo $hobby_string;
18
19
20      //假设$hobby_string是从数据库取出来的字段
21      var_dump(explode('|',$hobby_string));
```



2、 在 HTML 显示当中，通过判断复选框元素是否在数组中存在，来确定复选框 checkbox 是否有 checked=“checked”属性：in_array()

3）其他常规同名表单项的数据处理
除开 radio button 单选框和 checkbox 复选框，很少会出现同名的表单项。如果非要使用同名的来进行管理，那么可以采用 checkbox 方式进行操作

1、 表单中同名增加[]
2、 PHP 接收时数组处理
3、 PHP 转换成有格式的字符串
4、 数据库字符串存储

复选框细节

如果复选框没有选中，那么浏览器就不会提交。因此在 PHP 接收使用复选框（单选框）数据的时候，应该先判断是否存在该数据

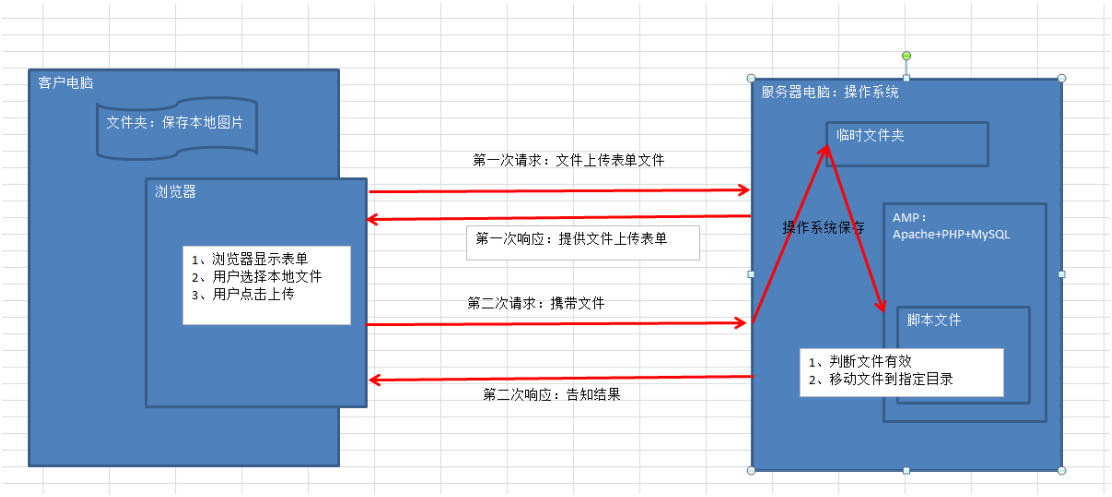
```
0
1       header('Content-type:text/html;charset=utf-8');
2       $hobby = isset($_POST['hobby']) ? $_POST['hobby'] :
        array();
3       //print_r($hobby);
4
5       //数组转换成有格式的字符串
6       $hobby_string = implode($hobby,'|');
7       echo $hobby_string;
```

文件上传

原理

文件上传：文件从用户本地电脑通过传输方式（Web 表单）保存到服务器所在电脑指定的目录下。

1、 增加文件上传的表单：浏览器请求一个服务器的 HTML 脚本（包含文件上传表单）
2、 用户从本地选择一个文件（点击上传框（按钮））
3、 用户点击上传：文件会通过物联网传输到服务器上
4、 服务器操作系统会将文件保存到临时目录：是以临时文件格式保存（windows 下 tmp）
5、 服务器脚本开始工作：判断文件有效
6、 服务器脚本将有效文件从临时目录移动到指定的目录下（完成）



表单写法

1）method 属性：表单提交方式必须为 POST

2）enctype 属性：form 表单属性，主要是规范表单数据的编码方式

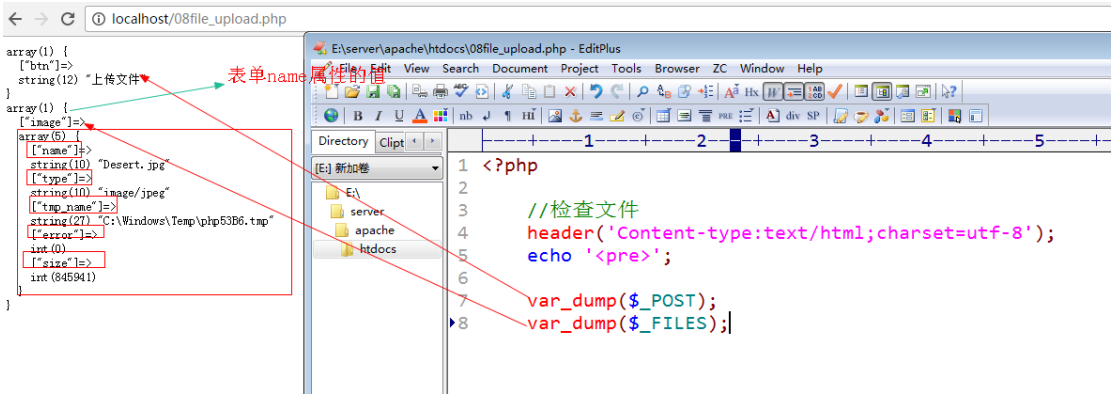
属性值

值	描述
application/x-www-form-urlencoded	在发送前编码所有字符（默认）
multipart/form-data	不对字符编码。 在使用包含文件上传控件的表单时，必须使用该值。
text/plain	空格转换为 "+" 加号，但不对特殊字符编码。

3）上传表单：file 表单

```
</head>
<body>
 <form method="POST" enctype="multipart/form-data" action="08file_upload.php">
   <input type="file" name="image" />
    <input type="submit" name="btn" value="上传文件"/>
 </form>
</body>
</html>
```

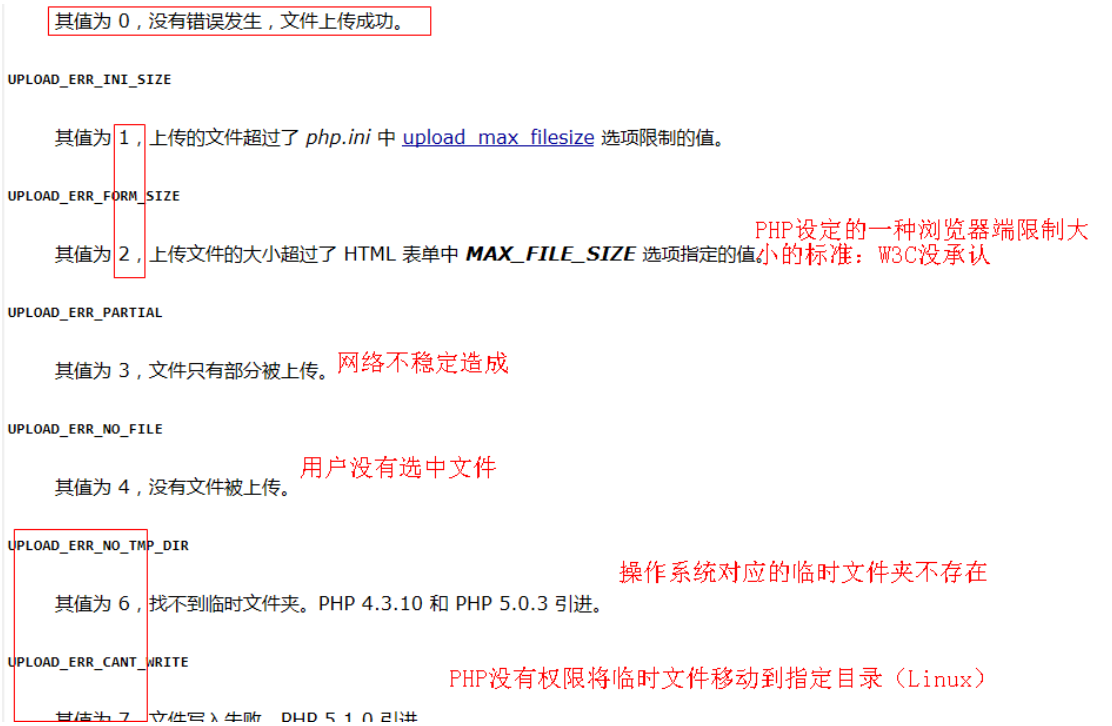
在 PHP 中，有一个预定义变量$_FILES 是专门用来存储用户上传的文件的。



测试：如果没有 enctype 属性，那么上传文件会是什么样子？

$_FILES 变量详解

1）name：文件在用户（浏览器端）电脑上实际存在的名字（实际用来保留后缀）
2）tmp_name：文件上传到服务器后操作系统保存的临时路径（实际用来给 PHP 后期使用）
3）type：MIME（多功能互联网邮件扩展）类型，用来在计算机中客户端识别文件类型（确定软件）
4）error：文件上传的代号，用来告知应用软件（PHP）文件接收过程中出现了什么问题（PHP 后期根据代码进行文件判断）



5）size：文件大小（PHP 根据实际需求来确定是否该保留）

移动临时文件到目标位置

文件上传之后会保存到$_FILES 中，那么访问文件信息的形式就是$_FILES['表单 name 属性值']['元素信息']

1）判断是否为上传的文件：is_uploaded_file()

```php
//1、取得文件信息
$file = $_FILES['image'];

//2、判断是否是上传文件：临时文件
if(is_uploaded_file($file['tmp_name'])){
    //是上传文件
}else{
    //不是上传文件
    echo '文件上传失败！';
}
```

2）移动文件：move_uploaded_file()

```php
//2、判断是否是上传文件：临时文件
if(is_uploaded_file($file['tmp_name'])){
    //是上传文件
    if(move_uploaded_file($file['tmp_name'],'uploads/' .
    $file['name'])){
        echo '文件保存成功！';
    }else{
        echo '文件保存失败！';
    }
}else{
```

多文件上传

当商品需要上传多个图片进行展示的时候：那么需要使用多文件上传

针对一个内容但是不同文件说明：同名表单

```html
</head>
<body>
 <form method="POST" enctype="multipart/form-data" action="10multi_upload.php">
   <input type="file" name="image[]" />
   <input type="file" name="image[]" />
   <input type="file" name="image[]" />
   <input type="submit" name="btn" value="批量上传"/>
 </form>
</body>
```

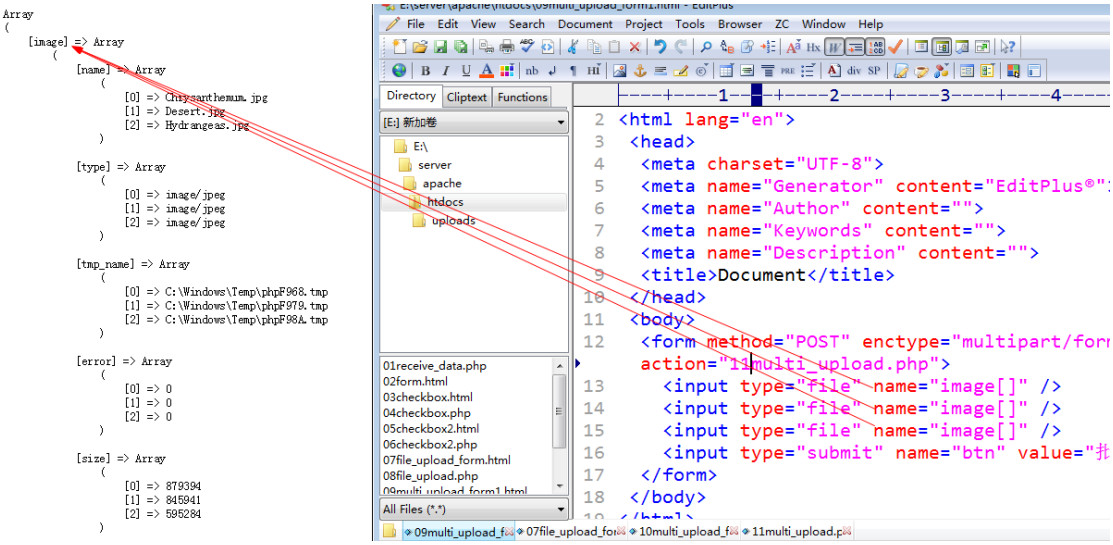
当商品需要进行多个维度图片说明的时候：需要使用多文件上传

针对是不同内容所以表单名字不一样：批量解决问题

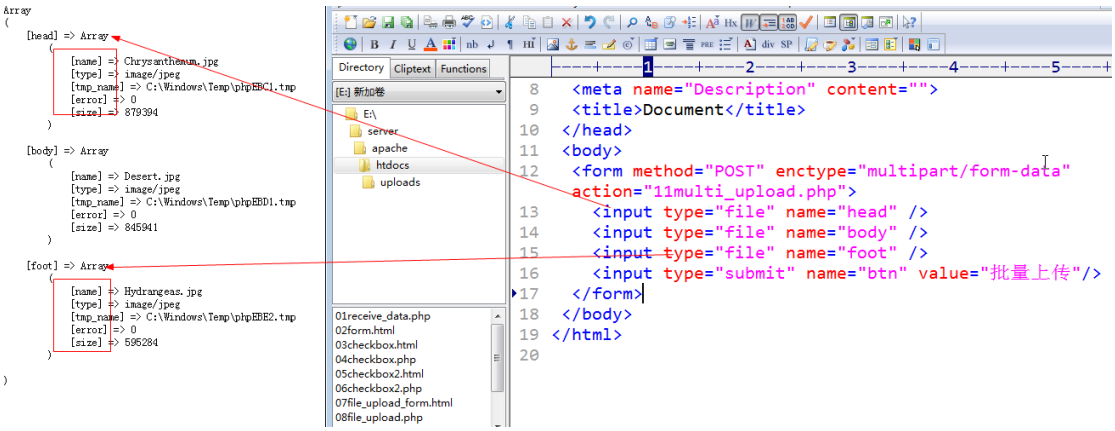
```
<body>
 <form method="POST" enctype="multipart/form-data" action="11multi_upload.php">
   <input type="file" name="head" />
   <input type="file" name="body" />
   <input type="file" name="foot" />
   <input type="submit" name="btn" value="批量上传"/>
 </form>
```

多文件上传的$_FILES 变量的数据结构形式

批量上传：同名表单：将表单名字形成一个数组，而且同时将文件对应的五个要素：name Tmp_name、size、type、error 都形成对应数量的数组，每个文件上传对应数组元素的下标都是一样的：name[0] 和 type[0]是属于同一个文件



批量上传：不同名表单：每个文件都会形成一个属于自己独立的 5 个元素的数组



当商品需要上传多个图片进行展示的时候：那么需要使用多文件上传

针对一个内容但是不同文件说明：同名表单

```
<body>
 <form method="POST" enctype="multipart/form-data" action="10multi_upload.php">
   <input type="file" name="image[]" />
   <input type="file" name="image[]" />
   <input type="file" name="image[]" />
   <input type="submit" name="btn" value="批量上传"/>
 </form>
</body>
```

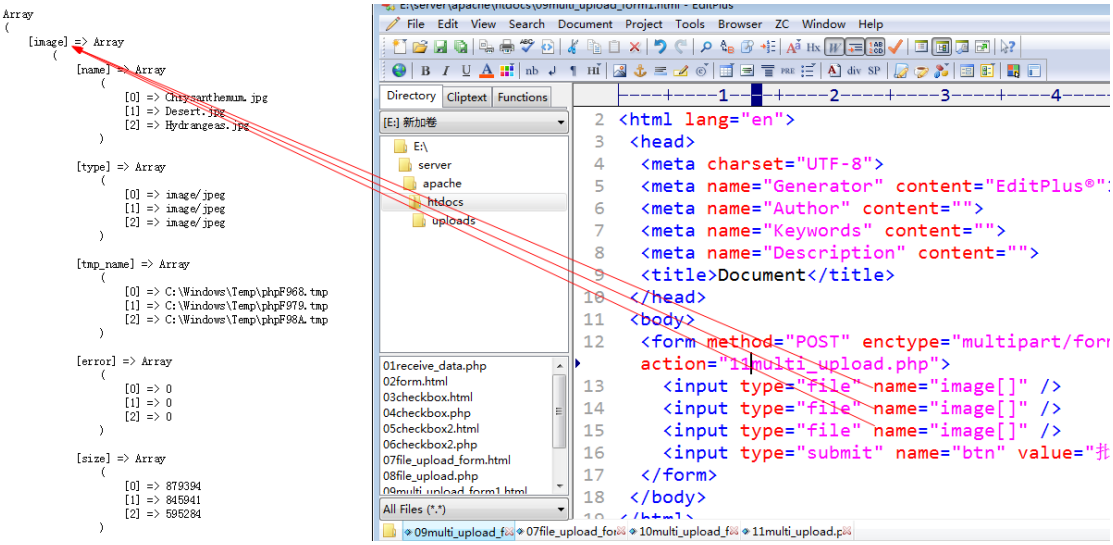
当商品需要进行多个维度图片说明的时候：需要使用多文件上传

针对是不同内容所以表单名字不一样：批量解决问题

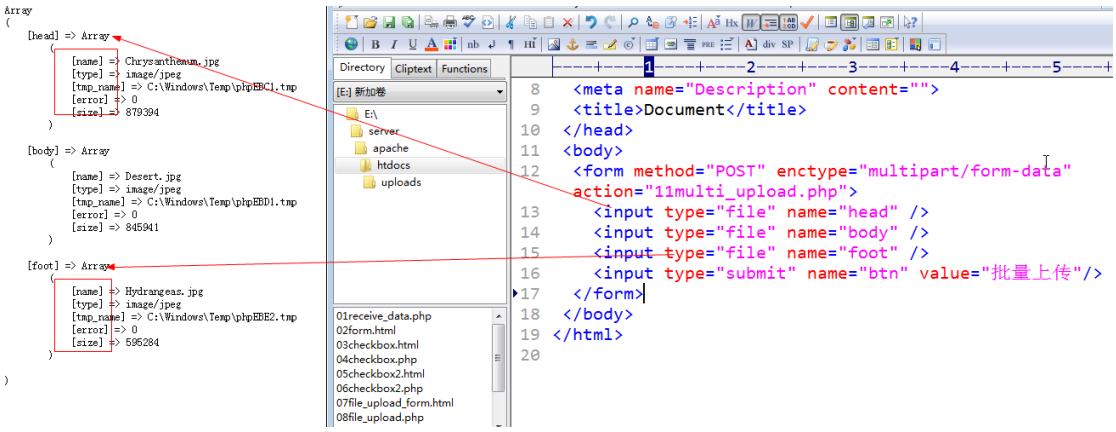
```
<body>
 <form method="POST" enctype="multipart/form-data" action="11multi_upload.php">
   <input type="file" name="head" />
   <input type="file" name="body" />
   <input type="file" name="foot" />
   <input type="submit" name="btn" value="批量上传"/>
 </form>
```

多文件上传的$_FILES 变量的数据结构形式

批量上传：同名表单：将表单名字形成一个数组，而且同时将文件对应的五个要素：name Tmp_name、size、type、error 都形成对应数量的数组，每个文件上传对应数组元素的下标都是一样的：name[0] 和 type[0]是属于同一个文件



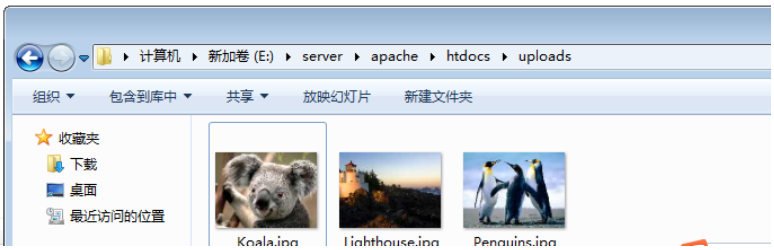
批量上传：不同名表单：每个文件都会形成一个属于自己独立的 5 个元素的数组



对多文件信息的遍历读取和处理

1、 不同名多文件上传处理方式：按照表单名字从$_FILES 中取出来就可以直接使用（明确知道表单中有多少个文件上传）；如果不确定表单中有多少个文件上传，不适合挨个去取（效率不高），可以通过遍历$_FILES 数组，挨个取出来实现文件上传

```php
//不同名文件遍历处理方式
foreach($_FILES as $file){
    //$file就是一个完整的上传文件信息：找到临时路径，指定存放路径
    if(is_uploaded_file($file['tmp_name'])){
        //存储
        move_uploaded_file($file['tmp_name'],'uploads/' . $file['name']);
    }
}
```



2、 同名多文件上传：想办法得到一个文件对应的五个元素数组。从$_FILES 中把对应的 name\tmp_name\size\error\type 挨个取出来，然后存放到不同的数组中。

```php
//同名文件遍历处理方式
//此时遍历$_FILES没有意义，只有一个数组元素；应该变量$_FILES['userfile']['任意一个要素: name|tmp_name|size|error|type']
//判断元素存在而且是数组：name有代表是文件，name元素有多个（数组）代表是同名批量上传
if(isset($_FILES['image']['name']) && is_array($_FILES['image']['name'])){
    //遍历构造数组元素
    $images = array();      //存储所有的文件信息，一个元素代表一个文件（数组）
    foreach($_FILES['image']['name'] as $k => $file){
        $images[] = array(
            'name' => $file,
            'tmp_name' => $_FILES['image']['type'][$k],
            'type' => $_FILES['image']['tmp_name'][$k],
            'error' => $_FILES['image']['error'][$k],
            'size' => $_FILES['image']['size'][$k]
        );
    }

}

print_r($images);
```

文件上传后续问题

实现上传功能代码的重复利用：封装文件上传函数
功能：上传文件
条件：条件判断
需要上传的文件信息：对应的 5 个元素的数组
1、 文件类型是否合适？外部指定 MIME 类型
2、 文件存储到什么位置？外部指定
3、 文件格式限制（文件后缀）？外部限定

4、 文件大小限制？外部指定

结果：实现文件上传

1、 成功：结果能够在以后看到：需要将文件的路径和文件名字返回（存储到数据库）

2、 失败：返回 false，指定错误原因（引用参数）

1）封装出一个上传函数

```php
/*
 * 实现文件上传（单）
 * @param1 array $file，需要上传的文件信息：一维5元素数组（name\tmp_name\error\size\type)
 * @param2 array $allow_type，允许上传的MIME类型
 * @param3 string $path，存储的路径
 * @param4 string &$error，如果出现错误的原因
 * @param5 array $allow_format = array()，允许上传的文件格式
 * @param6 int $max_size = 2000000，允许上传的最大值
*/
function upload_single($file,$allow_type,$path,&$error,$allow_format = array(),$max_size =
2000000){


}
```

2）判断文件是否有效

```php
 * @param6 int $max_size = 2000000，允许上传的最大值
*/
function upload_single($file,$allow_type,$path,&$error,$allow_format = array(),$max_size =
2000000){
    //判断文件是否有效
    if(!is_array($file) || !isset($file['error'])){
        //文件无效
        $error = '不是一个有效的上传文件！';
        return false;
    }
```

3）判断保存路径是否有效

```php
        $error = '不是一个有效的上传文件！';
        return false;
    }

    //判断文件存储路径是否有效
    if(!is_dir($path)){
        //路径不存在
        $error = '文件存储路径不存在！';
        return false;
    }
```

4）判断文件本身上传的过程中是否有错误：error

```php
//判断文件上传过程是否出错
switch($file['error']){
    case 1:
    case 2:
        $error = '文件超出服务器允许大小！';
        return false;
    case 3:
        $error = '文件上传过程中出现问题，只上传一部分！';
        return false;
    case 4:
        $error = '用户没有选中要上传的文件！';
        return false;
    case 6:
    case 7:
        $error = '文件保存失败！';
        return false;
}
```

2）文件类型的处理
3）文件格式的处理
4）文件大小的处理
5）命名冲突的处理

文件上传后续问题

实现上传功能代码的重复利用：封装文件上传函数
功能：上传文件
条件：条件判断
需要上传的文件信息：对应的 5 个元素的数组
1、 文件类型是否合适？外部指定 MIME 类型
2、 文件存储到什么位置？外部指定
3、 文件格式限制（文件后缀）？外部限定
4、 文件大小限制？外部指定
结果：实现文件上传
1、 成功：结果能够在以后看到：需要将文件的路径和文件名字返回（存储到数据库）
2、 失败：返回 false，指定错误原因（引用参数）

1）封装出一个上传函数

```php
/*
 * 实现文件上传（单）
 * @param1 array $file，需要上传的文件信息：一维5元素数组（name\tmp_name\error\size\type)
 * @param2 array $allow_type，允许上传的MIME类型
 * @param3 string $path，存储的路径
 * @param4 string &$error，如果出现错误的原因
 * @param5 array $allow_format = array()，允许上传的文件格式
 * @param6 int $max_size = 2000000，允许上传的最大值
*/
function upload_single($file,$allow_type,$path,&$error,$allow_format = array(),$max_size =
2000000){


}
```

2）判断文件是否有效

```
 * @param6 int $max_size = 2000000, 允许上传的最大值
*/
function upload_single($file,$allow_type,$path,&$error,$allow_format = array(),$max_size =
2000000){
    //判断文件是否有效
    if(!is_array($file) || !isset($file['error'])){
        //文件无效
        $error = '不是一个有效的上传文件！';
        return false;
    }
```

3）判断保存路径是否有效

```
        $error = '不是一个有效的上传文件！';
        return false;
    }

    //判断文件存储路径是否有效
    if(!is_dir($path)){
        //路径不存在
        $error = '文件存储路径不存在！';
        return false;
    }
```

4）判断文件本身上传的过程中是否有错误：error

```
    //判断文件上传过程是否出错
    switch($file['error']){
        case 1:
        case 2:
            $error = '文件超出服务器允许大小！';
            return false;
        case 3:
            $error = '文件上传过程中出现问题，只上传一部分！';
            return false;
        case 4:
            $error = '用户没有选中要上传的文件！';
            return false;
        case 6:
        case 7:
            $error = '文件保存失败！';
            return false;
    }
```

5）文件类型的处理：通过 MIME 匹配即可

```
16
17      //判断MIME类型
18      if(!in_array($file['type'],$allow_type)){
19          //该文件类型不允许上传
20          $error = '当前文件类型不允许上传！';
21          return false;
22      }
```

6）文件格式的处理：后缀名的问题

```php

        //判断后缀是否允许
        //取出后缀
        $ext = ltrim(strrchr($file['name'],'.'),'.');
        if(!empty($allow_format) && !in_array($ext,$allow_format)){
            //不允许上传
            $error = '当前文件的格式不允许上传！';
            return false;
        }
```

7）文件大小的处理

```php

        //判断当前文件大小是否满足当前需求
        if($file['size'] > $max_size){
            //文件过大
            $error = '当前上传的文件超出大小，最大允许' . $max_size . '字节!';
            return false;
        }

```

8）移动到指定目录

```php
         //构造文件名字
         //移动到指定目录
         if(!is_uploaded_file($file['tmp_name'])){
             //文件不是上传的
             $error = '错误：不是上传文件！';
             return false;
         }

         if(move_uploaded_file($file['tmp_name'],$path . '/' . $file['name'])){
             //成功
             return $file['name'];
         }else{
             //移动失败
             $error = '文件上传失败！';
             return false;
         }

```

8）命名冲突的处理：上传同名文件？中文名字文件怎么办？

```php
         //构造文件名字：类型_年月日+随机字符串.$ext
         $fullname = strstr($file['type'],'/',TRUE) . date('YYYYmmdd');
         //产生随机字符串
         for($i = 0;$i < 4;$i++){
             $fullname .= chr(mt_rand(65,90));
         }
         //拼凑后缀
         $fullname .= '.' . $ext;

         //移动到指定目录
         if(!is_uploaded_file($file['tmp_name'])){
             //文件不是上传的
             $error = '错误：不是上传文件！';
             return false;
         }

         if(move_uploaded_file($file['tmp_name'],$path . '/' . $fullname)){
             //成功
             return $fullname;
         }else{
```