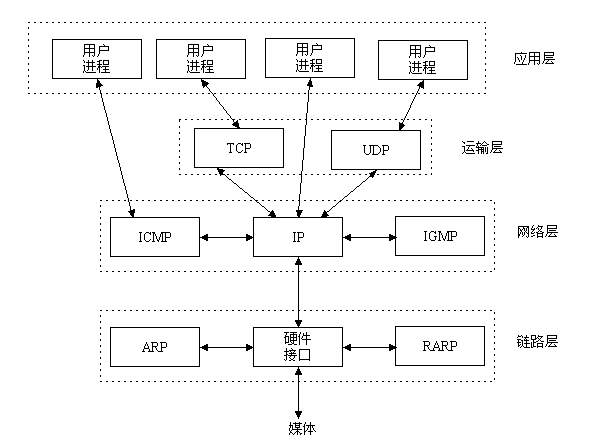
# PHP的Socket网络编程入门指引

一般人认为php的用途就是简单的网站建设，其实php语言也可以进行高深层次的网站功能性开发，这篇文章通过案例演示详细介绍了php的socket通信，对每一段代码进行了分析，每一步骤都配有详细说明，感兴趣的的朋友可以参考下

对 TCP/IP 、 UDP 、 Socket 编程这些词你不会很陌生吧？随着网络技术的发展，这些词充斥着我们的耳朵。

**那什么是TCP/IP、UDP？**  
TCP/IP（Transmission Control Protocol/Internet Protocol）即传输控制协议/网间协议，是一个工业标准的协议集，它是为广域网（WANs）设计的。  
UDP（User Data Protocol，用户数据报协议）是与TCP相对应的协议。它是属于TCP/IP协议族中的一种。  
这里有一张图，表明了这些协议的关系。



TCP/IP协议族包括运输层、网络层、链路层。现在你知道TCP/IP与UDP的关系了吧。  
**Socket在哪里呢？**  
在图1中，我们没有看到Socket的影子，那么它到底在哪里呢？还是用图来说话，一目了然。



原来Socket在这里。  
**Socket是什么呢？**　　Socket是应用层与TCP/IP协议族通信的中间软件抽象层，它是一组接口。在设计模式中，Socket其实就是一个门面模式，它把复杂的TCP /IP协议族隐藏在Socket接口后面，对用户来说，一组简单的接口就是全部，让Socket去组织数据，以符合指定的协议。  
你会使用它们吗？  
　　前人已经给我们做了好多的事了，网络间的通信也就简单了许多，但毕竟还是有挺多工作要做的。以前听到Socket编程，觉得它是比较高深的编程知识，但是只要弄清Socket编程的工作原理，神秘的面纱也就揭开了。  
　　一个生活中的场景。你要打电话给一个朋友，先拨号，朋友听到电话铃声后提起电话，这时你和你的朋友就建立起了连接，就可以讲话了。等交流结束，挂断电 话结束此次交谈。 生活中的场景就解释了这工作原理，也许TCP/IP协议族就是诞生于生活中，这也不一定。



　　先从服务器端说起。服务器端先初始化Socket，然后与端口绑定(bind)，对端口进行监听(listen)，调用accept阻塞，等待 客户端连接。在这时如果有个客户端初始化一个Socket，然后连接服务器(connect)，如果连接成功，这时客户端与服务器端的连接就建立了。客户 端发送数据请求，服务器端接收请求并处理请求，然后把回应数据发送给客户端，客户端读取数据，最后关闭连接，一次交互结束。

**socket相关函数：**----------------------------------------------------------------------------------------------  
socket\_accept() 接受一个Socket连接  
socket\_bind() 把socket绑定在一个IP地址和端口上  
socket\_clear\_error() 清除socket的错误或者最后的错误代码  
socket\_close() 关闭一个socket资源  
socket\_connect() 开始一个socket连接  
socket\_create\_listen() 在指定端口打开一个socket监听  
socket\_create\_pair() 产生一对没有区别的socket到一个数组里  
socket\_create() 产生一个socket，相当于产生一个socket的数据结构  
socket\_get\_option() 获取socket选项  
socket\_getpeername() 获取远程类似主机的ip地址  
socket\_getsockname() 获取本地socket的ip地址  
socket\_iovec\_add() 添加一个新的向量到一个分散/聚合的数组  
socket\_iovec\_alloc() 这个函数创建一个能够发送接收读写的iovec数据结构  
socket\_iovec\_delete() 删除一个已经分配的iovec  
socket\_iovec\_fetch() 返回指定的iovec资源的数据  
socket\_iovec\_free() 释放一个iovec资源  
socket\_iovec\_set() 设置iovec的数据新值  
socket\_last\_error() 获取当前socket的最后错误代码  
socket\_listen() 监听由指定socket的所有连接  
socket\_read() 读取指定长度的数据  
socket\_readv() 读取从分散/聚合数组过来的数据  
socket\_recv() 从socket里结束数据到缓存  
socket\_recvfrom() 接受数据从指定的socket，如果没有指定则默认当前socket  
socket\_recvmsg() 从iovec里接受消息  
socket\_select() 多路选择  
socket\_send() 这个函数发送数据到已连接的socket  
socket\_sendmsg() 发送消息到socket  
socket\_sendto() 发送消息到指定地址的socket  
socket\_set\_block() 在socket里设置为块模式  
socket\_set\_nonblock() socket里设置为非块模式  
socket\_set\_option() 设置socket选项  
socket\_shutdown() 这个函数允许你关闭读、写、或者指定的socket  
socket\_strerror() 返回指定错误号的详细错误  
socket\_write() 写数据到socket缓存  
socket\_writev() 写数据到分散/聚合数组

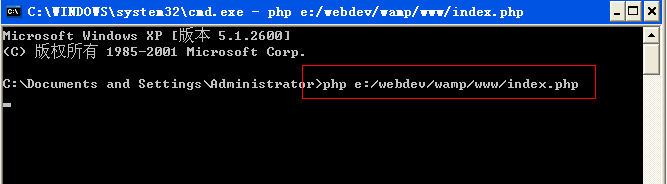
**案例一：socket通信演示**

**服务器端：**

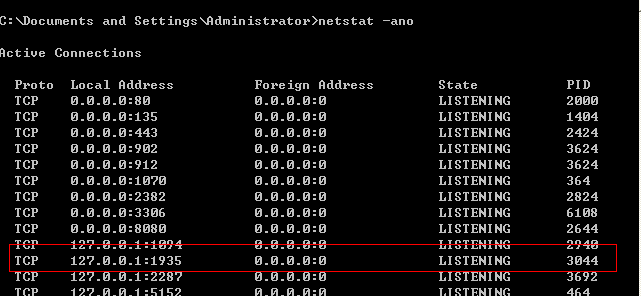
[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1 | 1 <?php 2 //确保在连接客户端时不会超时 3 set\_time\_limit(0); 4 5 $ip = '127.0.0.1'; 6 $port = 1935; 7 8 /\* 9 +------------------------------- 10 \* @socket通信整个过程 11 +------------------------------- 12 \* @socket\_create 13 \* @socket\_bind 14 \* @socket\_listen 15 \* @socket\_accept 16 \* @socket\_read 17 \* @socket\_write 18 \* @socket\_close 19 +-------------------------------- 20 \*/ 21 22 /\*---------------- 以下操作都是手册上的 -------------------\*/ 23 if(($sock = socket\_create(AF\_INET,SOCK\_STREAM,SOL\_TCP)) < 0) { 24 echo "socket\_create() 失败的原因是:".socket\_strerror($sock)."\n"; 25 } 26 27 if(($ret = socket\_bind($sock,$ip,$port)) < 0) { 28 echo "socket\_bind() 失败的原因是:".socket\_strerror($ret)."\n"; 29 } 30 31 if(($ret = socket\_listen($sock,4)) < 0) { 32 echo "socket\_listen() 失败的原因是:".socket\_strerror($ret)."\n"; 33 } 34 35 $count = 0; 36 37 do { 38 if (($msgsock = socket\_accept($sock)) < 0) { 39 echo "socket\_accept() failed: reason: " . socket\_strerror($msgsock) . "\n"; 40 break; 41 } else { 42 43 //发到客户端 44 $msg ="测试成功！\n"; 45 socket\_write($msgsock, $msg, strlen($msg)); 46 47 echo "测试成功了啊\n"; 48 $buf = socket\_read($msgsock,8192); 49 50 51 $talkback = "收到的信息:$buf\n"; 52 echo $talkback; 53 54 if(++$count >= 5){ 55 break; 56 }; 57 58 59 } 60 //echo $buf; 61 socket\_close($msgsock); 62 63 } while (true); 64 65 socket\_close($sock); 66 ?> |

这是socket的服务端代码。然后运行cmd，注意是自己的程序存放路径啊。

[](http://files.jb51.net/file_images/article/201508/201581190033001.png?20157119045)

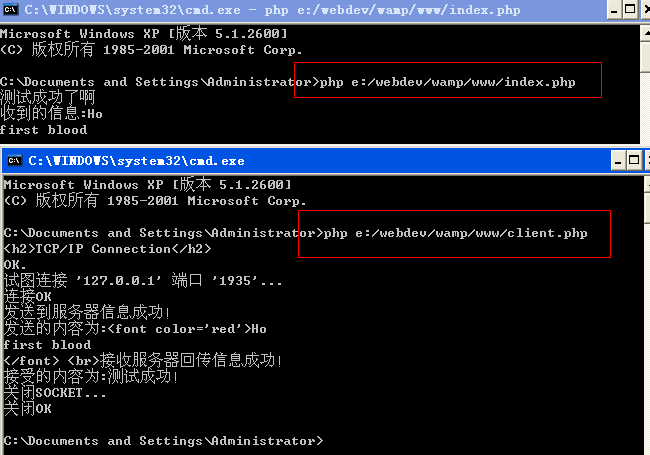
没有反映，对现在服务端的程序已经开始运行，端口已经开始监听了。运行netstat -ano可以查看端口情况，我的是1935端口

[](http://files.jb51.net/file_images/article/201508/201581190106545.png?20157119116)

看，端口已经处于LISTENING状态了。接下来我们只要运行客户端程序即可连接上。上代码

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1 | 1 <?php 2 error\_reporting(E\_ALL); 3 set\_time\_limit(0); 4 echo "<h2>TCP/IP Connection</h2>\n"; 5 6 $port = 1935; 7 $ip = "127.0.0.1"; 8 9 /\* 10 +------------------------------- 11 \* @socket连接整个过程 12 +------------------------------- 13 \* @socket\_create 14 \* @socket\_connect 15 \* @socket\_write 16 \* @socket\_read 17 \* @socket\_close 18 +-------------------------------- 19 \*/ 20 21 $socket = socket\_create(AF\_INET, SOCK\_STREAM, SOL\_TCP); 22 if ($socket < 0) { 23 echo "socket\_create() failed: reason: " . socket\_strerror($socket) . "\n"; 24 }else { 25 echo "OK.\n"; 26 } 27 28 echo "试图连接 '$ip' 端口 '$port'...\n"; 29 $result = socket\_connect($socket, $ip, $port); 30 if ($result < 0) { 31 echo "socket\_connect() failed.\nReason: ($result) " . socket\_strerror($result) . "\n"; 32 }else { 33 echo "连接OK\n"; 34 } 35 36 $in = "Ho\r\n"; 37 $in .= "first blood\r\n"; 38 $out = ''; 39 40 if(!socket\_write($socket, $in, strlen($in))) { 41 echo "socket\_write() failed: reason: " . socket\_strerror($socket) . "\n"; 42 }else { 43 echo "发送到服务器信息成功！\n"; 44 echo "发送的内容为:<font color='red'>$in</font> <br>"; 45 } 46 47 while($out = socket\_read($socket, 8192)) { 48 echo "接收服务器回传信息成功！\n"; 49 echo "接受的内容为:",$out; 50 } 51 52 53 echo "关闭SOCKET...\n"; 54 socket\_close($socket); 55 echo "关闭OK\n"; 56 ?> |

[](http://files.jb51.net/file_images/article/201508/201581190235281.png?20157119244)

至此客户端已经连接上服务端了。

**案例二：代码详解**

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | // 设置一些基本的变量  $host = "192.168.1.99";  $port = 1234;  // 设置超时时间  set\_time\_limit(0);  // 创建一个Socket  $socket = socket\_create(AF\_INET, SOCK\_STREAM, 0) or die("Www.wannuoda.com not createsocket\n");  //绑定Socket到端口  $result = socket\_bind($socket, $host, $port) or die("Www.wannuoda.com not bind tosocket\n");  // 开始监听链接  $result = socket\_listen($socket, 3) or die("Www.wannuoda.com not set up socketlistener\n");  // accept incoming connections  // 另一个Socket来处理通信  $spawn = socket\_accept($socket) or die("Www.wannuoda.com not accept incomingconnection\n");  // 获得客户端的输入  $input = socket\_read($spawn, 1024) or die("Www.wannuoda.com not read input\n");  // 清空输入字符串  $input = trim($input);  //处理客户端输入并返回结果  $output = strrev($input) . "\n";  socket\_write($spawn, $output, strlen ($output)) or die("Www.wannuoda.com not write  output\n");  // 关闭sockets  socket\_close($spawn);  socket\_close($socket); |

下面是其每一步骤的详细说明:

1.第一步是建立两个变量来保存Socket运行的服务器的IP地址和端口.你可以设置为你自己的服务器和端口(这个端口可以是1到65535之间的数字),前提是这个端口未被使用.

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 设置两个变量  $host = "192.168.1.99" ;  $port = 1234 ; |

2.在服务器端可以使用set\_time\_out()函数来确保PHP在等待客户端连接时不会超时.

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2 | // 超时时间  set\_time\_limit ( 0 ); |

3.在前面的基础上,现在该使用socket\_creat()函数创建一个Socket了—这个函数返回一个Socket句柄,这个句柄将用在以后所有的函数中.

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 创建Socket  $socket = socket\_create ( AF\_INET , SOCK\_STREAM , 0 ) or die( "Www.wannuoda.com not create  socket\n" ); |

第一个参数”AF\_INET”用来指定域名;  
第二个参数”SOCK\_STREM”告诉函数将创建一个什么类型的Socket(在这个例子中是TCP类型)

因此,如果你想创建一个UDP Socket的话,你可以使用如下的代码:

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 创建 socket  $socket = socket\_create ( AF\_INET , SOCK\_DGRAM , 0 ) or die( "Www.wannuoda.com not create  socket\n" ); |

4.一旦创建了一个Socket句柄,下一步就是指定或者绑定它到指定的地址和端口.这可以通过socket\_bind()函数来完成.

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 绑定 socket to 指定地址和端口  $result = socket\_bind ( $socket , $host , $port ) or die( "Www.wannuoda.com not bind to  socket\n" ); |

5.当Socket被创建好并绑定到一个端口后,就可以开始监听外部的连接了.PHP允许你由socket\_listen()函数来开始一个监听,同时你可以指定一个数字(在这个例子中就是第二个参数:3)

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 开始监听连接  $result = socket\_listen ( $socket , 3 ) or die( "Www.wannuoda.com not set up socket  listener\n" ); |

6.到现在,你的服务器除了等待来自客户端的连接请求外基本上什么也没有做.一旦一个客户端的连接被收到,socket\_accept()函数便开始起作用了,它接收连接请求并调用另一个子Socket来处理客户端–服务器间的信息.

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3  4 | //接受请求链接  // 调用子socket 处理信息  $spawn = socket\_accept ( $socket ) or die( "Www.wannuoda.com not accept incoming  connection\n" ); |

这个子socket现在就可以被随后的客户端–服务器通信所用了.

7.当一个连接被建立后,服务器就会等待客户端发送一些输入信息,这写信息可以由socket\_read()函数来获得,并把它赋值给PHP的$input变量.

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 读取客户端输入  $input = socket\_read ( $spawn , 1024 ) or die( "Www.wannuoda.com not read input\n" );  ?& gt ; |

socker\_read的第而个参数用以指定读入的字节数,你可以通过它来限制从客户端获取数据的大小.

注意:socket\_read函数会一直读取壳户端数据,直到遇见\n,\t或者\0字符.PHP脚本把这写字符看做是输入的结束符.

8.现在服务器必须处理这些由客户端发来是数据(在这个例子中的处理仅仅包含数据的输入和回传到客户端).这部分可以由socket\_write()函数来完成(使得由通信socket发回一个数据流到客户端成为可能)

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3  4 | // 处理客户端输入并返回数据  $output = strrev ( $input ) . "\n" ;  socket\_write ( $spawn , $output , strlen ( $output )) or die( "Www.wannuoda.com not write  output\n" ); |

9.一旦输出被返回到客户端,父/子socket都应通过socket\_close()函数来终止

[Copy to clipboard]  
PHP CODE:

[?](http://www.jb51.net/article/70900.htm)

|  |  |
| --- | --- |
| 1  2  3 | // 关闭 sockets  socket\_close ( $spawn );  socket\_close ( $socket ); |

以上就是php的socket通信的案例详解，希望对大家进一步学习socket通信知识有所帮助。