



中国数据 岁末经典巨献

租用服务器 送 笔记本电脑

此广告已经过期,暂时停止,如要续费请联系我们!

当前位置: [新云网络](#) → [网络学院](#) → [PHP编程](#) → [其它相关](#) → [PHP和Socket](#)

日期：2005-08-27 14:17:31 来源：CSDN

本站精彩推荐

PHP和Socket

[-] 减小字体 [+] 增大字体

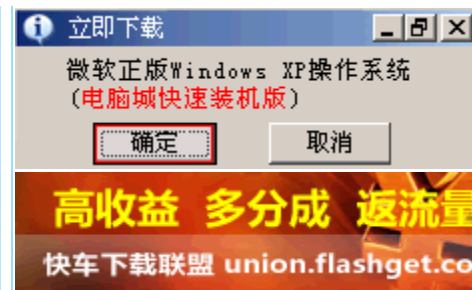
翻译: heiyeluren <heiyeluren_gmail_com>

◆ Socket基础

◇ 产生一个客户端

在这一章里你将了解到迷人而又让人容易糊涂的套接字（Sockets）。Sockets在PHP中是没有充分利用的功能。今天你将看到产生一个能使用客户端连接的服务器，并在客户端使用socket进行连接，服务器端将详细的处理信息发送给客户端。

◆ Socket 基础



本类热门阅览

- 1 正则表达式使用详解
- 2 在IIS下PHP环境配置的目录权限
- 3 PHP配置全攻略之Windows篇
- 4 header() 函数使用说明
- 5 IIS环境下安装PHP5手记

表一：协议

名字/常量 描述

AF_INET 这是大多数用来产生socket的协议，使用TCP或UDP来传输，用在IPv4的地址

AF_INET6 与上面类似，不过是来用在IPv6的地址

AF_UNIX 本地协议，使用在Unix和Linux系统上，它很少使用，一般都是当客户端和服务器的同一台及其上的时候使用

表二：Socket类型

名字/常量 描述

SOCK_STREAM 这个协议是按照顺序的、可靠的、数据完整的基于字节流的连接。这是一个使用最多的socket类型，这个socket是使用TCP来进行传输。

SOCK_DGRAM 这个协议是无连接的、固定长度的传输调用。该协议是不可靠的，使用UDP来进行它的连接。

SOCK_SEQPACKET 这个协议是双线路的、可靠的连接，发送固定长度的数据包进行传输。必须把这个包完整的接受才能进行读取。

SOCK_RAW 这个socket类型提供单一的网络访问，这个socket类型使用ICMP公共协议。（ping、traceroute使用该协议）

SOCK_RDM 这个类型是很少使用的，在大部分的操作系统上没有实现，它是提供给数据链路层使用，不保证数据包的顺序

表三：公共协议

名字/常量 描述

ICMP 互联网控制消息协议，主要使用在网关和主机上，用来检查网络状况和报告错误信息

UDP 用户数据报文协议，它是一个无连接，不可靠的传输协议

TCP 传输控制协议，这是一个使用最多的可靠的公共协议，它能保证数据包能够到达接受者那儿，如果在传输过程中发生错误，那么它将重新发送出错数据包。

现在你知道了产生一个socket的三个元素，那么我们就在php中使用socket_create()函数来产生一个socket。这个socket_create()函数需要三个参数：一个协议、一个socket类型、一个公共协议。socket_create()函数运行成功返回一个包含socket的资源类型，如果没有成功则返回false。

Resource socket_create(int protocol, int socketType, int commonProtocol);

现在你产生一个socket，然后呢？php提供了几个操纵socket的函数。你能够绑定socket到一个IP，监听一个socket的通信，接受一个socket；现在我们来查看一个例子，了解函数是如何产生、接受和监听一个socket。

```
<?php
$commonProtocol = getprotobyname("tcp");
$socket = socket_create(AF_INET, SOCK_STREAM, $commonProtocol);
socket_bind($socket, 'localhost', 1337);
socket_listen($socket);
// More socket functionality to come
?>
```

上面这个例子产生一个你自己的服务器端。例子第一行，

```
$commonProtocol = getprotobyname("tcp");
```

使用公共协议名字来获取一个协议类型。在这里使用的是TCP公共协议，如果你想使用UDP或者ICMP协议，那么你应该把getprotobyname()函数的参数改为“udp”或“icmp”。还有一个可选的办法是不使用getprotobyname()函数而是指定SOL_TCP或SOL_UDP在socket_create()函数中。

```
$socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
```

例子的第二行是产生一个socket并且返回一个socket资源的实例。在你有了一个socket资源的实例以后，你就必须把socket绑定到一个IP地址和某一个端口上。

```
socket_bind($socket, 'localhost', 1337);
```

在这里你绑定socket到本地计算机（127.0.0.1）和绑定socket到你的1337端口。然后你就需要监听所有进来的socket连接。

```
socket_listen($socket);
```

在第四行以后，你就需要了解所有的socket函数和他们的使用。

表四：Socket函数

函数名 描述

6 [在线人数统计源代码](#)

7 [PHP和Socket](#)

8 [如何使用PHP中的字符串函数](#)

9 [关于下载doc,xls文件的办法\(J](#)

10 [PHP安全配置](#)

11 [在Windows2000ADV下配置Apach](#)

12 [简体中文转换为繁体中文的PHP](#)

相关文章

[stevens那本socket的第一部分...](#)

[unix下编写socket程序的一般步...](#)

[用Socket发送电子邮件](#)

[用Socket发送电子邮件\(PHP\)](#)

[为了表示歉意,再贴一个 Socke...](#)

[记得原来有人问过用socket实现...](#)

[使用Sockets](#)

[Windows Socket API 使用经验...](#)

[功能强大的网络软件，揭开Soc...](#)

[初用SOCKET做聊天室程序后所想...](#)

[超cool !!!!! 截获WINSOCKET...](#)

[用Socket和MSHTML对象模型创建...](#)

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
(注: 函数介绍删减了部分原文内容, 函数详细使用建议参考英文原文, 或者参考PHP手册)

以上所有的函数都是PHP中关于socket的, 使用这些函数, 你必须把你的socket打开, 如果你没有打开, 请编辑你的php.ini文件, 去掉下面这行前面的注释:

```
extension=php_sockets.dll
```

如果你无法去掉注释, 那么请使用下面的代码来加载扩展库:

```
<?php
if(!extension_loaded('sockets'))
{
if(strtoupper(substr(PHP_OS, 3)) == "WIN")
{
dl('php_sockets.dll');
```

```
}  
else  
{  
dl('sockets.so');  
}  
}  
?>
```

如果你不知道你的socket是否打开，那么你可以使用phpinfo()函数来确定socket是否打开。你通过查看phpinfo信息了解socket是否打开。如下图：

查看phpinfo()关于socket的信息

◆ 产生一个服务器

现在我们把第一个例子进行完善。你需要监听一个指定的socket并且处理用户的连接。

```
<?php  
$commonProtocol = getprotobyname("tcp");  
$socket = socket_create(AF_INET, SOCK_STREAM, $commonProtocol);  
socket_bind($socket, 'localhost', 1337);  
socket_listen($socket);  
// Accept any incoming connections to the server  
$connection = socket_accept($socket);  
if($connection)  
{  
    socket_write($connection, "You have connected to the socket...\n\r");  
}  
?>
```

你应该使用你的命令提示符来运行这个例子。理由是因为这里将产生一个服务器，而不是一个Web页面。如果你尝试使用Web浏览器来运行这个脚本，那么很有可能它会超过30秒的限时。你可以使用下面的代码来设置一个无限的运行时间，但是还是建议使用命令提示符来运行。

```
set_time_limit(0);
```

在你的命令提示符中对这个脚本进行简单测试：

Php.exe example01_server.php

如果你没有在系统的环境变量中设置php解释器的路径，那么你将需要给php.exe指定详细的路径。当你运行这个服务器端的时候，你能够通过远程登陆（telnet）的方式连接到端口1337来测试这个服务器。如下图：

上面的服务器端有三个问题：1. 它不能接受多个连接。2. 它只完成唯一的一个命令。3. 你不能通过Web浏览器连接这个服务器。

这个第一个问题比较容易解决，你可以使用一个应用程序去每次都连接到服务器。但是后面的问题是你需要使用一个Web页面去连接这个服务器，这个比较困难。你可以让你的服务器接受连接，然后些数据到客户端（如果它一定要写的话），关闭连接并且等待下一个连接。

在上一个代码的基础上再改进，产生下面的代码来做你的新服务器端：

```
<?php  
// Set up our socket  
$commonProtocol = getprotobyname("tcp");  
$socket = socket_create(AF_INET, SOCK_STREAM, $commonProtocol);  
socket_bind($socket, 'localhost', 1337);  
socket_listen($socket);  
// Initialize the buffer
```

```

$buffer = "NO DATA";
while(true)
{
    // Accept any connections coming in on this socket
    $connection = socket_accept($socket);
    printf("Socket connected\r\n");
    // Check to see if there is anything in the buffer
    if($buffer != "")
    {
        printf("Something is in the buffer...sending data...\r\n");
        socket_write($connection, $buffer . "\r\n");
        printf("Wrote to socket\r\n");
    }
    else
    {
        printf("No Data in the buffer\r\n");
    }
    // Get the input
    while($data = socket_read($connection, 1024, PHP_NORMAL_READ))
    {
        $buffer = $data;
        socket_write($connection, "Information Received\r\n");
        printf("Buffer: " . $buffer . "\r\n");
    }
    socket_close($connection);
    printf("Closed the socket\r\n\r\n");
}
?>

```

这个服务器端要做什么呢？它初始化一个**socket**并且打开一个缓存收发数据。它等待连接，一旦产生一个连接，它将打印“**Socket connected**”在服务器端的屏幕上。这个服务器检查缓冲区，如果缓冲区里有数据，它将把数据发送到连接过来的计算机。然后它发送这个数据的接受信息，一旦它接受了信息，就把信息保存到数据里，并且让连接的计算机知道这些信息，最后关闭连接。当连接关闭后，服务器又开始处理下一次连接。（翻译的烂，附上原文）

This is what the server does. It initializes the socket and the buffer that you use to receive and send data. Then it waits for a connection. Once a connection is created it prints “Socket connected” to the screen the server is running on. The server then checks to see if there is anything in the buffer; if there is, it sends the data to the connected computer. After it sends the data it waits to receive information. Once it receives information it stores it in the data, lets the connected computer know that it has received the information, and then closes the connection. After the connection is closed, the server starts the whole process again.

◆ 产生一个客户端

处理第二个问题很容易的。你需要产生一个**php**页连接一个**socket**，发送一些数据进它的缓存并处理它。然后你又个处理后的数据在还顿，你能够发送你的数据到服务器。在另外一台客户端连接，它将处理那些数据。

To solve the second problem is very easy. You need to create a PHP page that connects to a socket, receive any data that is in the buffer, and process it. After you have processed the data in the buffer you can send your data to the server. When another client connects, it will process the data you sent and the client will send more data back to the server.

下面的例子示范了使用socket:

```
<?php
// Create the socket and connect
$socket = socket_create(AF_INET, SOCK_STREAM, SOL_TCP);
$connection = socket_connect($socket,'localhost', 1337);
while($buffer = socket_read($socket, 1024, PHP_NORMAL_READ))
{
    if($buffer == "NO DATA")
    {
        echo("<p>NO DATA</p>");
        break;
    }
    else
    {
        // Do something with the data in the buffer
        echo("<p>Buffer Data: " . $buffer . "</p>");
    }
}
echo("<p>Writing to Socket</p>");
// Write some test data to our socket
if(!socket_write($socket, "SOME DATA\r\n"))
{
    echo("<p>Write failed</p>");
}
// Read any response from the socket
while($buffer = socket_read($socket, 1024, PHP_NORMAL_READ))
{
    echo("<p>Data sent was: SOME DATA<br> Response was:" . $buffer . "</p>");
}
echo("<p>Done Reading from Socket</p>");
?>
```

这个例子的代码演示了客户端连接到服务器。客户端读取数据。如果这是第一时间到达这个循环的首次连接，这个服务器将发送“NO DATA”返回给客户端。如果情况发生了，这个客户端在连接之上。客户端发送它的数据到服务器，数据发送给服务器，客户端等待响应。一旦接受到响应，那么它将把响应写到屏幕上。

Tags :

作者: Matt Rutledget

[本日: 8 本周: 8 本月: 243 总数: 2339] [[返回上一页](#)] [[打 印](#)]

全球最好的动态免费空间站

06LA.com

1000M

支持ASP、PHP、.NET、CGI、SQL...真正的全能免费空间!

>我要注册

华夏名网新平台试运行


送大礼

com 域名 特惠 49 注册

“在线”照片编辑平台，
开启照片DIY之旅！



立刻免费编辑

 特价服务器租用
限量发售 月付仅需 **666元**

1G全能空间=195元/年 送数据库(代理四折)
电信、网通、双线 总有一款适合您 www.hb666.net

	好的评价 如果您觉得此文章好，就请您 89%(51)		支持		差的评价 如果您觉得此文章差，就请您 11%(6)		反对
---	-------------------------------	---	----	---	------------------------------	---	----

- 上一篇文章: [php的几个配置文件函数](#)
- 下一篇文章: [PHP的Socket函数参考](#)

文章评论 评论内容只代表网友观点，与本站立场无关！

评论摘要(共 0 条，得分 0 分，平均 0 分) [查看完整评论](#)
