aideehorn

aideehorn

浏览: 40269 次

性别:

来自: 北京

我现在离线

详细资料 留言簿

搜索本博客

搜索

最近访客 >>更多访客

rybby zhouky4665

jimylion Lyishuai

博客分类

PHP socket 网络编程实例[转] | HOWTO setup Freenx on Fedora

2009-08-26

PHP Socket 编程

PHP Socket 编程

PHP , Socket

socket

Socket协议的形象描述

1.一个是发动机(Socket),提供了网络通信的能力

一个是轿车(Http)，提供了具体的方式

2.socket的英文原义是“孔”或“插座”。在这里作为4BDS UNIX的进程通信机制，取后一种意义。socket非常类似于电话插座。以一个国家级电话网为例。电话的通话双方相当于相互通信的2个进程，区号是它的 网络地址；区内一个单位的交换机相当于一台主机，主机分配给每个用户的局内号码相当于socket号。任何用户在通话之前，首先要占有一部电话机，相当于 申请一个socket；同时要知道对方的号码，相当于对方有一个固定的socket。然后向对方拨号呼叫，相当于发出连接请求(假如对方不在同一区内，还 要拨对方区号，相当于给出网络地址)。对方假如在场并空闲(相当于通信的另一主机开机且可以接受连接请求)，拿起电话话筒，双方就可以正式通话，相当于连 接成功。双方通话的过程，是一方向电话机发出信号和对方从电话机接收信号的过程，相当于向socket发送数据和从socket接收数据。通话结束后，一 方挂起电话机相当于关闭socket，撤消连接。

在电话系统中，一般用户只能感受到本地电话机和对方电话号码的存在，建立通话的过程，话音传输的过程以及整个电话系统的技术细节对他都是透明的，这也与 socket机制非常相似。socket利用网间网通信设施实现进程通信，但它对通信设施的细节毫不关心，只要通信设施能提供足够的通信能力，它就满足 了。

至此，我们对socket进行了直观的描述。抽象出来，socket实质上提供了进程通信的端点。进程通信之前，双方首先必须各自创建一个端点，否则是没 有办法建立联系并相互通信的。正如打电话之前，双方必须各自拥有一台电话机一样。在网间网内部，每一个socket用一个半相关描述：

(协议，本地地址，本地端口)

一个完整的socket有一个本地唯一的socket号，由操作系统分配。

最重要的是，socket 是面向客户 / 服 务器模型而设计的，针对客户和服务器程序提供不同的socket 系统调用。客户随机申请一个socket (相当于一个想打电话的人可以在任何一台入网电话上拨号呼叫)，系统为之分配一个socket号；服务器拥有全局公认的 socket，任何客户都可以向它发出连接请求和信息请求(相当于一个被呼叫的电话拥有一个呼叫方知道的电话号码)。

socket利用客户 / 服 务器模式巧妙地解决了进程之间建立通信连接的问题。服务器socket 半相关为全局所公认非常重要。读者不妨考虑一下，两个完全随机的用户进程之间如何建立通信？假如通信双方没有任何一方的socket 固定，就好比打电话的双方彼此不知道对方的电话号码，要通话是不可能的。

- 全部博客 (98)
- ruby (9)
- rails (6)
- 技术宏览 (8)
- 数据库 (1)
- mac(Leopard) (3)
- 临时记录(待完善) (9)
- Iphone (0)
- Cocoa/Object-c/Iphone 开发 (1)
- javascript (15)
- php (12)
- 服务器 (32)
- html, div, css (2)

其他分类

- 我的收藏 (35)
- 我的论坛主题贴 (100)
- 我的所有论坛贴 (1)
- 我的精华良好贴 (0)

最近加入圈子

存档

- 2009-09 (3)
- 2009-08 (25)
- 2009-07 (15)
- 更多存档...

最新评论

- Web开发人员的利器：Ruby ...

  刚想学习Ruby 一看了这文章

Socket 接口是访问 Internet 使用得最广泛的方法。 如果你有一台刚配好TCP / IP协议的主机，其IP地址是202 . 120.127 . 201 ， 此时在另一台主机或同一台主机上执行ftp 202.120 . 127.201 ，显然无法建立连接。因 " 202.120.127.201 " 这台主机没有运行FTP服务软件。同样， 在另一台或同一台主机上运行浏览软件 如Netscape，输入 " http://202.120.127.201 " ， 也无法建立连接。现在，如果在这台主机上运行一个FTP服务软件（该软件将打开一个Socket， 并将其绑定到21端口），再在这台主机上运行一个Web 服务软件（该软件将打开另一个Socket，并将其绑定到80端口）。这样，在另一台主机或同一台主机上执行ftp 202.120 . 127.201 ， FTP客户软件将通过21端口来呼叫主机上由FTP 服务软件提供的Socket，与其建立连接并对话。而在netscape中输入 " http://202.120.127.201 " 时，将通过80端口来呼叫主机上由Web服务软件提供的Socket，与其建 立连接并对话。

在 Internet上有很多这样的主机，这些主机一般运行了多个服务软件，同时提供几种服务。每种服务都打开一个Socket，并绑定到一个端口上，不同的 端口对应于不同的服务。Socket正如其英文原意那样，象一个多孔插座。一台主机犹如布满各种插座的房间，每个插座有一个编号，有的插座提供220伏交 流电， 有的提供110伏交流电，有的则提供有线电视节目。 客户软件将插头插到不同编号的插座，就可以得到不同的服务。

---

1 . 什么是socket 所谓socket通常也称作 " 套接字 " ，用于描述IP地址和端口，是一个通信链的句柄。应用程序通常通过 " 套接字 " 向网络发出请求或者应答网络请求。 以J2SDK - 1 . 3为例，Socket和ServerSocket类库位于java . net 包中。ServerSocket用于服务器端，Socket是建立网络连接时使用的。在连接成功时，应用程序两端都会产生一个Socket实例，操作这个 实例，完成所需的会话。对于一个网络连接来说，套接字是平等的，并没有差别，不因为在服务器端或在客户端而产生不同级别。不管是Socket还是 ServerSocket它们的工作都是通过SocketImpl类及其子类完成的。

重要的Socket API：java . net . Socket继承于java . lang . Object ，有八个构造器，其方法并不多，下面介绍使用最频繁的三个方法，其它方法大家可以见JDK - 1 . 3文档。

Accept方法用于产生 " 阻塞 " ，直到接受到一个连接，并且返回一个客户端的Socket对象实例。 " 阻塞 " 是一个术语，它使程序运行暂时 " 停留 " 在这个地方，直到一个会话产生，然后程序继续；通常 " 阻塞 " 是由循环产生的。

getInputStream方法获得网络连接输入，同时返回一个IuputStream对象实例。

getOutputStream方法连接的另一端将得到输入，同时返回一个OutputStream对象实例。 注意：其中getInputStream和getOutputStream方法均会产生一个IOException，它必须被捕获，因为它们返回的流对 象，通常都会被另一个流对象使用。

2 . 如何开发一个Server - Client模型的程序 开发原理：

服务器，使用ServerSocket监听指定的端口，端口可以随意指定（由于1024以下的端口通常属于保留端口，在一些操作系统中不可以随意使用，所以建议使用大于1024的端口），等待客户连接请求，客户连接后，会话产生；在完成会话后，关闭连接。

客户端，使用Socket对网络上某一个服务器的某一个端口发出连接请求，一旦连接成功，打开会话；会话完成后，关闭Socket。客户端不需要指定打开的端口，通常临时的、动态的分配一个1024以上的端口。

Socket接口是TCP / IP网络的API，Socket接口定义了许多函数或例程，程序员可以用它们来开发TCP / IP网络上的应用程序。要学Internet上的TCP / IP网络编程，必须理解Socket接口。 Socket接口设计者最先是将接口放在Unix操作系统里面的。如果了解Unix系统的输入和输出的话，就很容易了解Socket了。网络的Socket数据传输是一种特殊的I / O，Socket也是一种文件描述符。Socket也具有一个类似于打开文件的函数调用Socket()，该函数返回一个整型的Socket描述符，随后的连接建立、数据传输等操作都是通过该Socket实现的。

常用的Socket类型有两种：流式Socket（SOCK_STREAM）和数据报式Socket（SOCK_DGRAM）。流式是一种面向连接的Socket，针对于面向连接的TCP服务应用；数据报式Socket是一种无连接的Socket，对应于无连接的UDP服务应用。 Socket建立为了建立Socket，程序可以调用Socket函数，该函数返回一个类似于文件描述符的句柄。socket函数原型为：int socket(int domain , int

把Ruby在Web方面的开发 都说神了 不过感 ...

-- by hedahai119

- ruby 经典电子书免费下载 ...

  很好，可惜排版影响阅读

  -- by supercode

- 在Windows环境中使用版本 ...

  斜体，眼睛难受

  -- by Xsen

- Prototype弹出框-颜色可 ...

  为什么值不能够通过form来提交了

  -- by elvishehai

- rails plugin list

  不错，有个不能用啦

  -- by 夜鸣猪

评论排行榜

- ruby 经典电子书免费下载--英文原版
- 只出现一次的JavaScript Alert提示窗口
- 弹出的窗口在规定时间内自动关闭，而且不作 ...
- javascript 渐隐遮罩效果
- div 隐藏和显示



[什么是RSS?]

type，int protocol);domain指明所使用的协议族，通常为PF_INET，表示互联网协议族（TCP / IP协议族）；type参数指定socket的类型：SOCK_STREAM 或SOCK_DGRAM，Socket接口还定义了原始Socket（SOCK_RAW），允许程序使用低层协议；protocol通常赋值 " 0 " 。Socket()调用返回一个整型socket描述符，你可以在后面的调用使用它。 Socket描述符是一个指向内部数据结构的指针，它指向描述符表入口。调用Socket函数时，socket执行体将建立一个Socket，实际上 " 建立一个Socket " 意 味着为一个Socket数据结构分配存储空间。 Socket执行体为你管理描述符表。两个网络程序之间的一个网络连接包括五种信息：通信协议、本地协议地址、本地主机端口、远端主机地址和远端协议端 口。Socket数据结构中包含这五种信息。 socket在测量软件中的使用也很广泛

让我们以一个简单的例子开始 --- 一个接收输入字符串，处理并返回这个字符串到客户端的TCP服务．下面是相应的代码：

PHP 代码：

```
1. <?
2. // 设置一些基本的变量
3. $host = "192.168.1.99";
4. $port = 1234;
5. // 设置超时时间
6. set_time_limit(0);
7. // 创建一个Socket
8. $socket = socket_create(AF_INET, SOCK_STREAM, 0) or die("Could not create
9. socket\n");
10. //绑定Socket到端口
11. $result = socket_bind($socket, $host, $port) or die("Could not bind to
12. socket\n");
13. // 开始监听链接
14. $result = socket_listen($socket, 3) or die("Could not set up socket
15. listener\n");
16. // accept incoming connections
17. // 另一个Socket来处理通信
18. $spawn = socket_accept($socket) or die("Could not accept incoming
19. connection\n");
20. // 获得客户端的输入
21. $input = socket_read($spawn, 1024) or die("Could not read input\n");
22. // 清空输入字符串
23. $input = trim($input);
24. //处理客户端输入并返回结果
25. $output = strrev($input) . "\n";
26. socket_write($spawn, $output, strlen ($output)) or die("Could not write
27. output\n");
28. // 关闭sockets
29. socket_close($spawn);
30. socket_close($socket);
```

```
31. ?>
```

复制代码

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 .

PHP 代码 :

```
1. <?
2. // 设置两个变量
3. $host = "192.168.1.99";
4. $port = 1234;
5. ?>
```

复制代码

---

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 .

PHP 代码 :

```
1. <?
2. // 超时时间
3. set_time_limit(0);
4. ?>
```

复制代码

---

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 .

PHP 代码 :

```
1. <?
2. // 创建Socket
3. $socket = socket_create(AF_INET, SOCK_STREAM, 0) or die("Could not create
4. socket\n");
5. ?>
```

复制代码

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 :

PHP 代码 :

```
1. <?
2. // 创建 socket
3. $socket = socket_create(AF_INET, SOCK_DGRAM, 0) or die("Could not create
4. socket\n");
5. ?>
```

复制代码

---

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 .

PHP 代码 :

```
1. <?
2. // 绑定 socket to 指定地址和端口
3. $result = socket_bind($socket, $host, $port) or die("Could not bind to
4. socket\n");
5. ?>
```

复制代码

---

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)

PHP 代码 :

```
1. <?
2. // 开始监听连接
3. $result = socket_listen($socket, 3) or die("Could not set up socket
4. listener\n");
5. ?>
```

复制代码

---

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_assept()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 .

PHP 代码 :

```
1. <?
2. //接受请求链接
3. // 调用子socket 处理信息
4. $spawn = socket_accept($socket) or die("Could not accept incoming
5. connection\n");
6. ?>
```

复制代码

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 .

PHP 代码 :

```
1. <?
2. // 读取客户端输入
3. $input = socket_read($spawn, 1024) or die("Could not read input\n");
4. ?>
```

复制代码

---

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)

PHP 代码 :

```
1. <?
2. // 处理客户端输入并返回数据
3. $output = strrev($input) . "\n";
4. socket_write($spawn, $output, strlen ($output)) or die("Could not write
5. output\n");
6. ?>
```

复制代码

---

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

PHP 代码 :

```
1. <?
2. // 关闭 sockets
3. socket_close($spawn);
4. socket_close($socket);
5. ?>
```


PHP socket 网络编程实例[转] | HOWTO setup Freenx on Fedora

01:07 | 浏览 (520) | 评论 (0) | 分类: php | 相关推荐

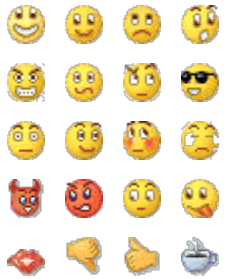
评论

发表评论

表情图标

B I U Quote Code List Img URL Flash Table

字体颜色: 标准 字体大小: 标准 对齐: 标准



提示：选择您需要装饰的文字, 按上列按钮即可添加上相应的标签

您还没有登录，请登录后发表评论(快捷键 Alt+S / Ctrl+Enter) 提交