Perl 发送邮件 Perl 面向对象

Perl Socket 编程

Socket又称"套接字"，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，使主机间或者一台计算机上的进程间可以通讯。

本章节我们为大家接收 Perl 语言中如何使用 Socket 服务。

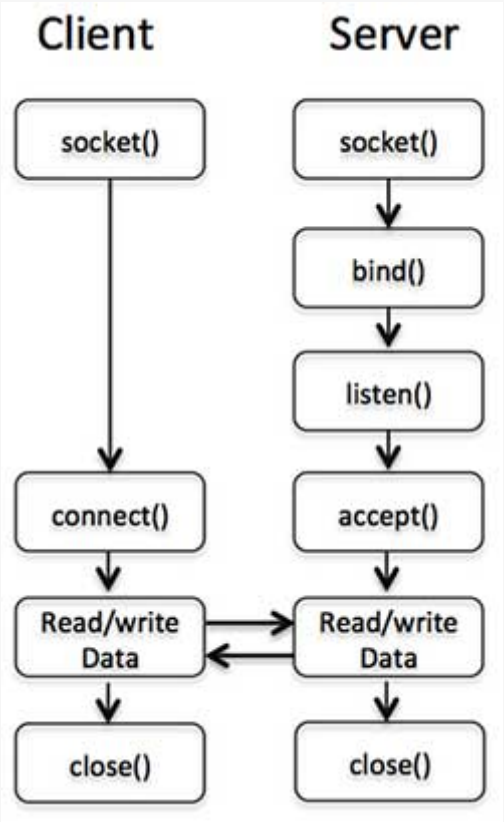
创建服务端

- 使用 **socket** 函数来创建 socket服务。
- 使用 **bind** 函数绑定端口。
- 使用 **listen** 函数监听端口。
- 使用 **accept** 函数接收客户端请求。

创建客户端

- 使用 **socket** 函数来创建 socket 服务。
- 使用 **connect** 函数连接到 socket 服务端。

以下图表演示了客户端与服务端之间的通信流程：



服务端 socket 函数

socket 函数

Perl 中，我们用 socket（ ）函数来创建套接字，语法格式如下：

```
socket( SOCKET, DOMAIN, TYPE, PROTOCOL );
```

参数解析：

- **DOMAIN** 创建的套接字指定协议集。 例如：
  - `AF_INET` 表示IPv4网络协议
  - `AF_INET6` 表示IPv6
  - `AF_UNIX` 表示本地套接字（使用一个文件）
- **TYPE** 套接字类型可以根据是面向连接的还是非连接分为SOCK_STREAM或SOCK_DGRAM
- **PROTOCOL** 应该是 **(getprotobyname('tcp'))[2]**。指定实际使用的传输协议。

所以 socket 函数调用方式如下：

```
use Socket     # 定义了 PF_INET 和 SOCK_STREAM

socket(SOCKET,PF_INET,SOCK_STREAM,(getprotobyname('tcp'))[2]);
```

bind() 函数

使用 bind() 为套接字分配一个地址：

```
bind( SOCKET, ADDRESS );
```

SOCKET 一个socket的描述符。 ADDRESS 是 socket 地址（TCP/IP）包含了三个元素:

- 地址簇 (TCP/IP, 是 AF_INET, 在你系统上可能是 2)
- 端口号 (例如 21)
- 网络地址 (例如 10.12.12.168)

使用socket()创建套接字后，只赋予其所使用的协议，并未分配地址。在接受其它主机的连接前，必须先调用bind()为套接字分配一个地址。
简单实例如下：

```
use Socket        # 定义了 PF_INET 和 SOCK_STREAM

$port = 12345;    # 监听的端口
$server_ip_address = "10.12.12.168";
bind( SOCKET, pack_sockaddr_in($port, inet_aton($server_ip_address)))
   or die "无法绑定端口! \n";
```

or die 在绑定地址失败后执行。

通过设置 setsockopt() 可选项 SO_REUSEADDR 设置端口可立即重复使用。

pack_sockaddr_in() 函数将地址转换为二进制格式。

listen() 函数

当socket和一个地址绑定之后，listen()函数会开始监听可能的连接请求。然而，这只能在有可靠数据流保证的时候使用：

```
listen( SOCKET, QUEUESIZE );
```

SOCKET : 一个socket的描述符。

QUEUESIZE : 是 一个决定监听队列大小的整数，当有一个连接请求到来，就会进入此监听队列；当一个连接请求被accept()接受，则从监听队列中移出；当队列满后，新的连接请求会返回错误。

一旦连接被接受，返回0表示成功，错误返回-1。

accept() 函数

accept() 函数接受请求的socket连接。如果成功则返回压缩形式的网络地址，否则返回FALSE：

```
accept( NEW_SOCKET, SOCKET );
```

NEW_SOCKET : 一个socket的描述符。

SOCKET : 一个socket的描述符。

accept() 通常应用在无限循环当中：

```
while(1) {
   accept( NEW_SOCKET, SOCKT );
   .......
}
```

以上实例可以实时监听客户端的请求。

客户端函数

connect() 函数

connect()系统调用为一个套接字设置连接，参数有文件描述符和主机地址。

```
connect( SOCKET, ADDRESS );
```

以下创建一个连接到服务端 socket 的实例：

```
$port = 21;    #  ftp 端口
$server_ip_address = "10.12.12.168";
connect( SOCKET, pack_sockaddr_in($port, inet_aton($server_ip_address)))
    or die "无法绑定端口! \n";
```

完整实例

接下来我们通过一个完整实例来了解下所有 socket 函数的应用：

服务端 server.pl 代码：

实例

```
#!/usr/bin/perl -w
# Filename : server.pl
use strict;
use Socket;
# 使用端口 7890 作为默认值
my $port = shift || 7890;
my $proto = getprotobyname('tcp');
my $server = "localhost"; # 设置本地地址
# 创建 socket, 端口可重复使用，创建多个连接
socket(SOCKET, PF_INET, SOCK_STREAM, $proto)
or die "无法打开 socket $!\n";
setsockopt(SOCKET, SOL_SOCKET, SO_REUSEADDR, 1)
or die "无法设置 SO_REUSEADDR $!\n";
# 绑定端口并监听
bind( SOCKET, pack_sockaddr_in($port, inet_aton($server)))
or die "无法绑定端口 $port! \n";
listen(SOCKET, 5) or die "listen: $!";
print "访问启动: $port\n";
# 接收请求
my $client_addr;
while ($client_addr = accept(NEW_SOCKET, SOCKET)) {
# send them a message, close connection
my $name = gethostbyaddr($client_addr, AF_INET );
print NEW_SOCKET "我是来自服务端的信息";
print "Connection recieved from $name\n";
close NEW_SOCKET;
}
```

打开一个终端，执行以下代码：

```
$ perl sever.pl
访问启动: 7890
```

客户端 client.pl 代码：

实例

```
#!/usr/bin/perl -w
# Filename : client.pl
use strict;
use Socket;
# 初始化地址与端口
my $host = shift || 'localhost';
my $port = shift || 7890;
my $server = "localhost"; # 主机地址
# 创建 socket 并连接
socket(SOCKET,PF_INET,SOCK_STREAM,(getprotobyname('tcp'))[2])
or die "无法创建 socket $!\n";
connect( SOCKET, pack_sockaddr_in($port, inet_aton($server)))
or die "无法连接: port $port! \n";
my $line;
while ($line = <SOCKET>) {
print "$line\n";
}
close SOCKET or die "close: $!";
```

打开另外一个终端，执行以下代码：

```
$ perl client.pl
我是来自服务端的信息
```

Perl 发送邮件 Perl 面向对象