← Ruby 发送邮件 – SMTP Ruby XML, XSLT 和 XPath 教程 →

Ruby Socket 编程

Ruby提供了两个级别访问网络的服务，在底层你可以访问操作系统，它可以让你实现客户端和服务器为面向连接和无连接协议的基本套接字支持。

Ruby 统一支持应用程序的网络协议，如FTP、HTTP等。

不管是高层的还是底层的。ruby提供了一些基本类，让你可以使用TCP,UDP,SOCKS等很多协议交互，而不必拘泥在网络层。这些类也提供了辅助类，让你可以轻松的对服务器进行读写。

接下来就让我们来学习如何进行 Ruby Socket 编程

什么是 Sockets

应用层通过传输层进行数据通信时，TCP和UDP会遇到同时为多个应用程序进程提供并发服务的问题。多个TCP连接或多个应用程序进程可能需要 通过同一个TCP协议端口传输数据。为了区别不同的应用程序进程和连接，许多计算机操作系统为应用程序与TCP / IP协议交互提供了称为套接字 (Socket)的接口，区分不同应用程序进程间的网络通信和连接。

生成套接字，主要有3个参数：通信的目的IP地址、使用的传输 层协议(TCP或UDP)和使用的端口号。Socket原意是"插座"。通过将这3个参数结合起来，与一个"插座"Socket绑定，应用层就可以和传输 层通过套接字接口，区分来自不同应用程序进程或网络连接的通信，实现数据传输的并发服务。

Sockets 词汇解析：

选项	描述
domain	指明所使用的协议族，通常为 PF_INET, PF_UNIX, PF_X25, 等等。
type	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。
hostname	网络接口的标识符： • 字符串, 可以是主机名或IP地址 • 字符串 "<broadcast>", 指定 INADDR_BROADCAST 地址。 • 0 长度的字符串, 指定INADDR_ANY • 一个整数，解释为主机字节顺序的二进制地址。
port	port是端口的编号，每个服务器都会监听客户端连接的一个或多个端口号，一个端口号可以是 Fixnum 的端口号，包含了服务器名和端口。

简单的客户端

以下我们通过给定的主机和端口编写了一个简单的客户端实例，Ruby TCPSocket 类提供了 open 方法来打开一个 socket。

TCPSocket.open(hosname, port) 打开一个 TCP 连接。

一旦你打开一个 Socket 连接，你可以像 IO 对象一样读取它，完成后，你需要像关闭文件一样关闭该连接。

以下实例演示了如何连接到一个指定的主机，并从 socket 中读取数据，最后关闭socket：

实例

```
require 'socket' # Sockets 是标准库
hostname = 'localhost'
port = 2000
s = TCPSocket.open(hostname, port)
while line = s.gets # 从 socket 中读取每行数据
puts line.chop # 打印到终端
end
s.close # 关闭 socket
```

简单的服务

Ruby 中可以使用 TCPServer 类来写个简单的服务。TCPServer 对象是 TCPSocket 的工厂对象。

现在我们使用 TCPServer.open(hostname, port) 来创建一个 TCPServer 对象。

接下来调用 TCPServer 的 accept 方法，该方法会等到一个客户端连接到指定的端口，然后返回一个的TCPSocket对象，表示连接到该客户端。

实例

```
require 'socket' # 获取socket标准库
server = TCPServer.open(2000) # Socket 监听端口为 2000
loop { # 永久运行服务
client = server.accept # 等待客户端连接
client.puts(Time.now.ctime) # 发送时间到客户端
client.puts "Closing the connection. Bye!"
client.close # 关闭客户端连接
}
```

现在，在服务器上运行以上代码，查看效果。

多客户端TCP服务

互联网上，大多服务都有大量的客户端连接。

Ruby的Thread类可以很容易地创建多线程服务，一个线程执行客户端的连接，而主线程在等待更多的连接。

实例

```
require 'socket' # 获取socket标准库
server = TCPServer.open(2000) # Socket 监听端口为 2000
loop { # 永久运行服务
Thread.start(server.accept) do |client|
client.puts(Time.now.ctime) # 发送时间到客户端
client.puts "Closing the connection. Bye!"
client.close # 关闭客户端连接
```

```
end
}
```

在这个例子中，socket永久运行，而当server.accept接收到客户端的连接时，一个新的线程被创建并立即开始处理请求。而主程序立即循环回，并等待新的连接。

微小的Web浏览器

我们可以使用socket库来实现任何的 Internet 协议。以下代码展示了如何获取网页的内容：

实例

```
require 'socket'
host = 'www.w3cschool.cc' # web服务器
port = 80 # 默认 HTTP 端口
path = "/index.htm" # 想要获取的文件地址
# 这是个 HTTP 请求
request = "GET #{path} HTTP/1.0\r\n\r\n"
socket = TCPSocket.open(host,port) # 连接服务器
socket.print(request) # 发送请求
response = socket.read # 读取完整的响应
# Split response at first blank line into headers and body
headers,body = response.split("\r\n\r\n", 2)
print body # 输出结果
```

要实现一个类似 web 的客户端，你可以使用为 HTTP 预先构建的库如Net::HTTP。

以下代码与先前代码是等效的：

实例

```
require 'net/http' # 我们需要的库
host = 'www.w3cschool.cc' # web 服务器
path = '/index.htm' # 我们想要的文件
http = Net::HTTP.new(host) # 创建连接
headers, body = http.get(path) # 请求文件
if headers.code == "200" # 检测状态码
print body
else
puts "#{headers.code} #{headers.message}"
end
```

以上我们只是简单的为大家介绍 Ruby 中socket的应用，更多文档请查看：Ruby Socket 库和类方法

← Ruby 发送邮件 – SMTP　　Ruby XML, XSLT 和 XPath 教程 →

点我分享笔记