Subject: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Tue, 16 Jan 2007 06:22:03 GMT
View Forum Message <> Reply to Message

Topic title:  Project:  SocketArchive.dll
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

The Socket Archive is a dynamic link library that makes coding Tcp and Udp connections extremely easy for even the beginner programmer.  The dll consists of 2 main classes which are the Tcp and Udp classes.

Usually it takes anywhere over 100 lines of code to program a working connection whether Tcp or Udp.  What I have done is condensed this code for the developers to around 2-4 lines of code.  The dll itself contains over 500 lines of code, but all the programmer has to do is reference it in their project, code 2 lines of code and everything else is handled for them.  This dll is fast in performance and there is hardly any delay in connections.  So far the furthest connection was tested from Pueblo, Colorado to the UK and back with less than 2-3 seconds in delay.

As of now, the project is composed of only the Tcp portion in which the only coded part is the TcpListener which listens for incoming connections and handles the TcpClients.  The server can accept as many clients as your computer or server can handle and puts each of them on their own thread.  This dll is thread safe, so there isn't anything to worry about having cross-thread errors or performance loss issues.

The dll is coded in C++/CLI using .NET 2.0 technology.  I have utilized the code to optimize this for optimal performance.  The code also maintains C++ ISO Standards.

As of now, the following is what has been done:

Tcp Class
This class is the parent class that houses all Tcp content inside the dll.

Listener Class
The Listener class houses all the methods and handling of the object that handles the incoming client connects and incoming client data.  Any connections that are incoming from a remote client are accepted and placed on their own threads to prevent single-thread complications and delays when sending and receiving data.  All data is transferred via a network stream object and is converted between bytes and other data as needed.

The listener class only has one constructor in which you do not need to pass any arguements.  You simply create and initialize a new instance of this class and the listener handles everything for you.  The listener operates by automatically selecting your active network adapters IP Address and listens on Tcp port 80 so there is no port forwarding required.

The custom function that starts the listener has been implemented as StartListener().  (Default: Start() and then Listen().

Other custom implementations will include:

ListenIncomingText();
ListenIncomingBytes();
ListenIncomingFile();
ListenIncomingFiles();

As well as:

ReplyOutgoingText();
ReplyOutgoingBytes();
ReplyOutgoingFile();
ListenIncomingFiles();

Client Class
The Tcp Client has not yet been implemented yet, but will include all functions required to send and receive data as well as connections.

Other advantages of this dll are that all information required to be displayed to the user or application operator is passed to a single console window which will handle all application messages.  The choice of a console was to give further optimal performance over that of a graphical user interface design.

The Udp class has yet to be implemented or thought over, but will be very similar to that of the Tcp class.

Please leave any comments/suggestions/flaming/ect. in this thread.

Thank you.

---

## Subject: Re: A project that "could" be beneficial to the Renegade community.
## Posted by jnz on Tue, 16 Jan 2007 08:56:27 GMT

View Forum Message <> Reply to Message

It is good, in fact very

and yes, it took me alot of code in the Seye to create a connection. with proper error handling. if had this dll sooner the client could of been finished sooner

---

## Subject: Re: A project that "could" be beneficial to the Renegade community.
## Posted by dead6re on Tue, 16 Jan 2007 09:21:19 GMT

View Forum Message <> Reply to Message

I feel .NET code bloats the class/application.

Does it really need to include .NET? I have a very simple class that I use for IRC connections that also allows me to hook specific messages into my own functions, for example

IRC.Hook("MODE", IRC_ModeHandler);

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by jnz on Tue, 16 Jan 2007 09:32:11 GMT
View Forum Message <> Reply to Message

dead6re wrote on Tue, 16 January 2007 09:21I feel .NET code bloats the class/application.

Does it really need to include .NET? I have a very simple class that I use for IRC connections that also allows me to hook specific messages into my own functions, for example

IRC.Hook("MODE", IRC_ModeHandler);

portibility of .NET isn't great . It is possible with this class to hook functions as well although you gotter reference it and code the bit that hooks manualy.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Tue, 16 Jan 2007 12:46:40 GMT
View Forum Message <> Reply to Message

dead6re wrote on Tue, 16 January 2007 02:21I feel .NET code bloats the class/application.

Does it really need to include .NET? I have a very simple class that I use for IRC connections that also allows me to hook specific messages into my own functions, for example

IRC.Hook("MODE", IRC_ModeHandler);

You just need to make sure the user has .NET on their computer (which they most likely have the latest version already). Other than that, you don't need .NET for anything, everything else is handled for you. Aside from that, this dll runs fast and efficiently with no performance loss. The feedback is provided via consoles so there is no extra memory requirements either such as a GUI would need.

As for .NET "bloating" an application, it can only if you code inefficiently.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by =HT=T-Bird on Tue, 16 Jan 2007 22:23:50 GMT
View Forum Message <> Reply to Message

Dave Anderson wrote on Tue, 16 January 2007 06:46dead6re wrote on Tue, 16 January 2007 02:21I feel .NET code bloats the class/application.

Does it really need to include .NET? I have a very simple class that I use for IRC connections that also allows me to hook specific messages into my own functions, for example

IRC.Hook("MODE", IRC_ModeHandler);

You just need to make sure the user has .NET on their computer (which they most likely have the latest version already).  Other than that, you don't need .NET for anything, everything else is handled for you.  Aside from that, this dll runs fast and efficiently with no performance loss.  The feedback is provided via consoles so there is no extra memory requirements either such as a GUI would need.

As for .NET "bloating" an application, it can only if you code inefficiently.

Problem 1: not portable platform-wise (a well-written native sockets class should not have too many problems dealing with the differences between NT (WinSock2) and BSD sockets)
Problem 2: doesn't fit into everybody's apps (some people have severe issues with C++ native<->CLR interop)
Problem 3: CLR is a lot of overhead for some people (like when you are trying not to drag down people's SFPS)

---

## Subject: Re: A project that "could" be beneficial to the Renegade community.
## Posted by Dave Anderson on Tue, 16 Jan 2007 22:30:53 GMT

View Forum Message <> Reply to Message

Quote:Problem 1: not portable platform-wise (a well-written native sockets class should not have too many problems dealing with the differences between NT (WinSock2) and BSD sockets)
Problem 2: doesn't fit into everybody's apps (some people have severe issues with C++ native<->CLR interop)

Quote:Problem 3: CLR is a lot of overhead for some people

Then this wouldn't be the kind of thing that type of coder would use.  But for the people who do know C++/CLI and don't have trouble, its ideal  (And I know a few people on here who do use C++/CLI, this isn't just for one group of coders here.)

Quote:(like when you are trying not to drag down people's SFPS)

You're narrowing this down to just server related stuff in that part.  You have to keep in mind that this isn't something that is meant for one purpose only;  like connecting to servers.  You have a lot of flexibility with a dll like this and it can be used for many purposes, server related or not.  I'm not saying that my dll has any performance gain over native sockets, but there is nothing in this dll that would cause a performance loss in a server from a logical standpoint.

---

## Subject: Re: A project that "could" be beneficial to the Renegade community.

Posted by jnz on Tue, 16 Jan 2007 23:28:21 GMT
View Forum Message <> Reply to Message

not only that, NR uses CLR .NET. that doesn't cause alot of overhead at all. in fact 20Mb of ram is nothing compaired to what some server have.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Tue, 16 Jan 2007 23:44:42 GMT
View Forum Message <> Reply to Message

EDIT:  //...

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Nightma12 on Tue, 16 Jan 2007 23:49:37 GMT
View Forum Message <> Reply to Message

gamemodding wrote on Tue, 16 January 2007 17:28not only that, NR uses CLR .NET. that doesn't cause alot of overhead at all. in fact 20Mb of ram is nothing compaired to what some server have.

NR uses COM...

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by jnz on Tue, 16 Jan 2007 23:55:20 GMT
View Forum Message <> Reply to Message

bah! i assumed because you use microsoft visual studio

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Tue, 16 Jan 2007 23:57:19 GMT
View Forum Message <> Reply to Message

I don't believe COM is a form of C++, its the Component Object Model that defines how objects interact within a single application or between applications.  Dan is talking about CLI, a form of C++ developed by Microsoft to maintain C++ ISO standards.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Sir Kane on Thu, 18 Jan 2007 18:01:49 GMT
View Forum Message <> Reply to Message

You should've written that dll in C or something and use structures or something to pass parameters around. Pretty easy to make it work on different platforms that way.

But nice idea anyways.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Thu, 18 Jan 2007 20:24:58 GMT
View Forum Message <> Reply to Message

Thanks.  I'm not familiar with the protocols Renegade uses, but what other protocols besides TCP and UDP does it use, if any?

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by jnz on Thu, 18 Jan 2007 21:58:28 GMT
View Forum Message <> Reply to Message

correct me if im wrong, but its just UDP?

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Thu, 18 Jan 2007 22:13:06 GMT
View Forum Message <> Reply to Message

I wouldn't know (which is why I am asking), but I thought I heard somewhere of it using TCP.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by icedog90 on Thu, 18 Jan 2007 23:21:45 GMT
View Forum Message <> Reply to Message

I think it uses TCP.

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Thu, 18 Jan 2007 23:23:01 GMT
View Forum Message <> Reply to Message

Silent Kane wrote on Thu, 18 January 2007 11:01You should've written that dll in C or something and use structures or something to pass parameters around. Pretty easy to make it work on different platforms that way.

But nice idea anyways.

Oh, and yeah, the class has multiple constructors so you can choose to not pass a parameter (its handled by two custom properties:  IP and Port), or you can specify parameters as you see fit. Right now the one that doesn't pass parameters doesn't make connections correctly, but I will get it working soon.

---

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Sir Kane on Fri, 19 Jan 2007 15:19:01 GMT
View Forum Message <> Reply to Message

Renegade uses UDP for everything except WOL stuff.

---

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Fri, 19 Jan 2007 20:06:26 GMT
View Forum Message <> Reply to Message

Thank very much you.

---

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by icedog90 on Sat, 20 Jan 2007 19:31:44 GMT
View Forum Message <> Reply to Message

Yeah, that's what I was thinking of (WOL).

---

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Mon, 22 Jan 2007 12:46:45 GMT
View Forum Message <> Reply to Message

http://dcomproductions.net/Products/SocketArchive/Default.aspx

Above is the link to the official project page on my website.  It has the same information as in my original post except you can find 4 images of what the test program look like (2 images) and of what the server console looks like (2 images).

---

Subject: Re: A project that "could" be beneficial to the Renegade community.
Posted by Dave Anderson on Mon, 22 Jan 2007 20:57:10 GMT
View Forum Message <> Reply to Message

Update:  I have finished coding much functionality in this dll.  Changes are as follows:

*StartCommandLine();

The StartCommandLine() function asserts the Tcp Server into a command line interface mode where the server operator can use server commands such as !help, !clients, etc. This is used to manually operate the server.

*StartListener();
The StartListener() function puts the Tcp Server into a listening state where the server will continually listen for a pending connection. When the pending state returns true, the StartListener() function accepts the client and returns. This particular function is used to manually operate and handle incoming connections. To set the server in automatic mode, please see StartListenerEx();

*StartListenerEx();
NOT IMPLEMENTED. However, this function resembles the above function except for it repeatedly loops and catches all client connnections.

*Command Line Interface Commands
!Clients | Displays a list of the currently connected clients.
!CloseClient | Disconnects a remote client from the server.
!EndCLI | Exits the command line interface mode.
!Help | Displays a list of valid commands.
!Start | Sets the server in a manual listening state.
!StartEx | Sets the server in a automatic listening state.
.
.
.
. many more to come.

*Note that this dll is explicitly for consoles in this build. Future builds will contain GUI support, but possibly not up until version 2.0.0.0.

Other Features
The dll supports up to 100 clients asynchronysly and only uses ~18MB of RAM.