

Forum:[Hack Forums](#)

Thread: [IonNetwork - A powerful, extensible networking library](#)

phrootoo - December 12, 2014, 7:17 am

Although I should probably announce this earlier to the release, i would like to see what peoples opinions are on the project.

IonNetwork is essentially a server thats extremely customisable. The server itself comes as a executable and allows users to create plugins that interact with the server.

The server works with 'Requests'. These are json serialized classes that consist of the following data:

method - The name of the method the client is to access

parameters - An array of objects defined by a string 'key'.

The requests can be sent from both the client AND the server and are entirely processed by the code. All you have to do is handle them and send them.

So, what does it look like? Well, let me run through an example. Here is a basic 'skeleton' of a script:

```
Code class Plugin : IonNetwork.Plugins.IPlugin
{
    public string GetPluginName()
    {
        return "Sample plugin";
    }

    public string GetPluginDescription()
    {
        return "This plugin handles the basics of a standard plugin and shows you what you can ";
    }

    public void OnRequest(IonNetwork.TcpNetworking.NetworkClient Client,
IonNetwork.Networking.Request Request)
    {

    }

    public void OnConnect(IonNetwork.TcpNetworking.NetworkClient Client)
    {

    }
}
```

Within this, you can see he standard 'plugin name' and 'plugin description'. These are descriptors that will be called by the server to identify plugins.

OnRequest and OnConnect are the main methods that will be called when a network client interacts

with the server. OnConnect is called when a new client connects. Do remember that EVERY plugin will have a OnConnect. Within OnConnect, you can use it to send information such as version numbers etc. This would be achieved by the following:

```
Code public void OnConnect(IonNetwork.TcpNetworking.NetworkClient Client)
{
    IonNetwork.Networking.Request request = new IonNetwork.Networking.Request("serverinfo"); //
    Takes a method name as a constructor
    request.AddParameter("clientid", Client.uniqueId);
    request.AddParameter("localtime", DateTime.Now);
}
This code would send a "serverinfo" packet to the client with their clientid and localtime of the
server. You can then implement the handler on the client side to process this easily.
```

OnRequest is raised when a new request is sent by a client to the server. The request is sent to ALL plugins, again, so you need to bare this in mind.

Here is an example of OnRequest:

```
Code public void OnRequest(IonNetwork.TcpNetworking.NetworkClient Client,
IonNetwork.Networking.Request Request)
{
    if(Request.method == "authenticate")
    {
        if(Request.GetParameter("token") == "icanhazentry")
        {
            IonNetwork.Networking.Request request = new Request("authenticateresponse");
            request.AddParameter("response", "ok");
        }
    }
}
Now you're probably thinking "Yeah, that's cool, but it must be difficult to implement it on the client
side". Well, infact, here's all you need to implement it on the client:
```

```
Code Networking.NetworkConnection connection = new
Networking.NetworkConnection("localhost", 8888);

    connection.RequestHandlerEvent += new
Networking.NetworkConnection.RequestHandler(Handler);
    connection.NetworkErrorHandler += new
Networking.NetworkConnection.NetworkError(ErrorHandler);

    connection.Connect();
```

Every time a new request is called, the RequestHandler is called and that handles the request. Pretty neat, if you ask me!

The question is, would anybody be interested in playing around with this? I'll get it completed fairly shortly, but wanted to know if anybody fancied making some plugins or possibly creating a demo for it? In return you'll get a free license when it's released![/code]

Shedletsky - December 12, 2014, 8:32 am

Woah, look's like it would be great.

phrootoo - December 12, 2014, 8:35 am

Also, something else i forgot was the ability to have 'settings' for the connection. These can be used to store objects of any type. Updated code here:

```
Code public void OnRequest(IonNetwork.TcpNetworking.NetworkClient Client,
IonNetwork.Networking.Request Request)
{
    if(Request.method == "authenticate")
    {
        if(Request.GetParameter("token") == "icanhazentry")
        {
            IonNetwork.Networking.Request request = new Request("authenticateresponse");
            request.AddParameter("response", "ok");
            // Set setting for client
            Client.SetProperty("authenticated", true);
        }
    }
}
```

Cornflower Blue ???-?— - December 12, 2014, 9:02 am

A networking lib totally needs a plugin system, especially one that has a plugin name and plugin description.

The networking is the back-end, what is the purpose of having the name and description?

Other than that, nice release, I am sure many people will find a use for it. Oh wait, all the skids who make rats use VB. Scratch that, nobody will use it.

DragonHunter - December 12, 2014, 10:40 am

Looking good so far I guess, but where is the actual library / source code :P

Increment - December 12, 2014, 1:16 pm

(Dec 12, 2014 11:40 AM)DragonHunter Wrote: [\[CITE\]](#) Looking good so far I guess, but where is the actual library / source code :P

Was going to ask the same thing ..

Anyway, it looks well done OP. Looking forward for the source.

phrootoo - December 12, 2014, 1:51 pm

I'll get around to releasing the source when i get back in to the office tomorrow. It's still in dev, so it's a little rusty. fucking cool for a days work though, if i do say so myself.

you dont really need the source though, if im honest. just a compiled server and the DLL's for scripting it :D

(Dec 12, 2014 10:02 AM)Cornflower Blue ???-?— Wrote: [\[CITE\]](#) A networking lib totally needs a plugin system, especially one that has a plugin name and plugin description.

The networking is the back-end, what is the purpose of having the name and description?

Other than that, nice release, I am sure many people will find a use for it. Oh wait, all the skids who make rats use VB. Scratch that, nobody will use it.

Name a descriptor is practically just so you know what shits on there. I guess i'll make a panel in the future, but right now it's just a console. Yknow, enable/disable plugins section with the name of

the plugin and its description. possibly get the community to work on plugins then who knows where it goes from there :D

Cornflower Blue ???-? — - December 12, 2014, 4:05 pm

(Dec 12, 2014 02:51 PM)phrootoo Wrote: [\[CITE\]](#) Name a descriptor is practically just so you know what shits on there. I guess i'll make a panel in the future, but right now it's just a console. Yknow, enable/disable plugins section with the name of the plugin and its description. possibly get the community to work on plugins then who knows where it goes from there :D

Why would you need to enable and disable parts of a networking framework?

The networking will be the backend of the program, either you are intending this to be something that I am not getting, or you are making something silly.

phrootoo - December 12, 2014, 4:33 pm

(Dec 12, 2014 05:05 PM)Cornflower Blue ???-?— Wrote: [\[CITE\]](#) Why would you need to enable and disable parts of a networking framework?

The networking will be the backend of the program, either you are intending this to be something that I am not getting, or you are making something silly.

Plugins are created externally. Users have the ability to create plugins, and possibly share plugins. The interface will allow the user to enable and disable plugins.

FYI this is not stupid at all, infact, you'd be stupid to say its stupid.

phrootoo - January 7, 2015, 8:37 am

Update on this: I'm looking for people to test it and play with it. Free license keys for people who test it :-)

I've implemented a few new features to it, so now the base is a lot better. I've also included a sample plugin, which allows you to user the system as a login/registration system in about 150 lines. Fully networked.