Subject: Is it possible to use Core/Rpc in non blocking mode?
Posted by steffen on Tue, 23 Apr 2013 09:08:10 GMT
View Forum Message <> Reply to Message

Hi,

My application is talking to a json-rpc server which normally uses two connections at a time. One for foreground request/response actions and another for long polling background requests.

The response from the long polling requests is actually reverse json-rpc requests.

Now I'm wondering if it is possible to use SocketWaitEvent, like in the GuiWebCrawler example, to listen for data on multiple JsonRpcRequest at once?
Or am I forced to use a thread for each simultaneous JsonRpcRequest?

Also is it possible to use the http upgrade feature on a JsonRpcRequest, to make it into a Html5 websocket connection?
I think my server will be able to handle websocket connections pretty soon. Then I would only need one bidirectional connection.

If none of these features exists, I would be happy to make a patch/extension for them, so a few hints in the right direction is welcome.

Regards,
Steffen

Subject: Re: Is it possible to use Core/Rpc in non blocking mode?
Posted by nlneilson on Tue, 23 Apr 2013 15:23:25 GMT
View Forum Message <> Reply to Message

steffen wrote on Tue, 23 April 2013 02:08 json-rpc server
two connections at a time.

Or am I forced to use a thread for each simultaneous JsonRpcRequest?

Hi Steffen
I didn't know so had to look up what json-rpc was so will not be much help on that.

method - A String with the name of the method to be invoked.
params - An Array of objects to be passed as parameters to the defined method.
id - A value of any type, which is used to match the response with the request that it is replying to.

Servers usually can accept multiple connections in the same thread.
I don't see where another thread would be necessary.

I just send and receive a bunch of characters up to 3000 so far.
Then I have some code for sorting and acting on what was received and when necessary reply.

What are you trying to do that is more complicated than that?

Subject: Re: Is it possible to use Core/Rpc in non blocking mode?
Posted by steffen on Tue, 23 Apr 2013 16:57:16 GMT
View Forum Message <> Reply to Message

Hi nlneilson,

It is not the protocol or data transfers that is causing me problems. It is general socket handling.

My application communicates over GSM a modem, which are automaticly behind NAT routers from all ISP's I have used.

To let the server send notifications to the client, the client either has to poll the server continuously, raising the traffic cost. Or it can use a technique called long polling, where the server delays the response until there is something to send or a timeout occurs.
In most cases we have found that a 4 minute timeout keeps the modems online, at a minimal traffic cost. The response is simply a 204 No data available.

Now with the current JsonRpcRequest the json requests work fine, but it blocks the calling thread for up to 4 minutes.

Simultaneously I have another JsonRpcRequest handling the communication going from the client to the server. This request gets a response immediately, but in case of errors there could also be some sort of timeout here. So it also gets a thread of its own.

So instead of the Execute method used today I would rather if I could send the requests to the SocketWaitEvent and loop through a list to handle the sockets, just like the GuiWebCrawler example does it.

Another thing is that the json-rpc specifies array requests, where multiple requests can be put into a json array and send in the same transmission. This adds more payload to the HTTP/TCP/IP packets, so it is not always sending 90% headers.
This would require an outgoing queue for the json requests.

I have looked into the RpcRequest::Execute method and I think I could make a derived class with a couple of methods to use it in non blocking mode.

At last I think if first I have the non blocking mode running, it will not be so big a step to let it run over a Html5 websocket connection. Without knowing the details, I think it is just a bidirectional socket connection.

Regards,
Steffen

Subject: Re: Is it possible to use Core/Rpc in non blocking mode?
Posted by nlneilson on Tue, 23 Apr 2013 19:25:57 GMT
View Forum Message <> Reply to Message

steffen wrote on Tue, 23 April 2013 09:57To let the server send notifications to the client,

Have you considered having a server and client at each end?

This is my client code:

```
String snd(String r, int a){
 TcpSocket s;
 if(!s.Connect("127.0.0.1", 11811)) {
    return "x";
 }

 s.Put(r + "\n\0");
 if(a==1){
    String st = s.GetLine();
    return st;
 }
 return "y";
}
```

so it is just
snd(data, 0); // or 1
If there is no immediate response expected from the server it's 0
If there to be a response it's 1 for the GetLine()

That way there is no time lost.

AFAIK there is almost no overhead for the threads, servers and clients when no data is being transferred.

polling, timeout, etc. I like to stay away from if possible.

Subject: Re: Is it possible to use Core/Rpc in non blocking mode?
Posted by steffen on Wed, 24 Apr 2013 08:56:16 GMT
View Forum Message <> Reply to Message

Hi nlneilson,

Thank you for you input. I appreciate the feedback.

It is not possible with a server in both ends because all GPRS connections are automaticly put behind NAT routers.
The server cannot connect to the clients unless I implement some peer to peer technology, and since it is a standard webserver that is not an option.

I will go on making my own JsonRpc implementation, where the transport layer is independent.
Then it will be much easier to use it across different socket types and web protocols.
It will also be easier to implement batch requests.

Here comes a bit of clarification on what I'm facing, just for information if you are interested:

JsonRpc is a very simple format which we have used for several years now, but I have first recently started to move our clients from Delphi running on WinXP Embedded to U++ on Linux.
Having completed the user interface design for our touch screens and all the serial communication with the trucks in place, it is now time to bring them all online again.
Now I have the chance to get rid of the few caveats I know about in our current design.

We use JsonRPC for all communications, and normal HTTP for file transfers, like updates, bug reports and so on.

99% of my clients are placed in trucks and they sometimes move into zones without GSM/GPRS coverage. Causing all kinds of weird connection problems. And our error and reconnect handling is a very big part of the current implementation. Sometimes we even have to kill the power to the modems to bring them back online.

We currently have three threads for communications, Client->Server, Server->Client and a Server->Client download thread for updates.
The last one is only active when needed, but it takes 7-10 minutes to download an update, so it gets its own thread and runs in the background.
Bug reports never exceeds 100kB so they are just send through our normal outgoing queue.
The queue holds both jsonrpc requests as well as direct http requests. It is even possible to overwrite items in the queue, so stuff like fuel level and current GPS coordinates only reside once in the queue. There is no need to send outdated data to the server
When a connection dropout occurs it influences all connections and we sometimes have to abort all connections, repowering the modem, redial and restart all the threads.
Very complicated and the reason I would like to try a different approach.
The dial part is actually done using the Windows RAS api, and occasionally it simply refuses to dial until we reboot the entire system. This is one of the motivations for moving everything to U++ on Linux.
Another is the wish to move to an ARM based platform we currently have at the first prototype stage.

Regards,
Steffen

Subject: Re: Is it possible to use Core/Rpc in non blocking mode?
Posted by nlneilson on Mon, 29 Apr 2013 01:49:05 GMT
View Forum Message <> Reply to Message

steffen wrote on Wed, 24 April 2013 01:56
1. It is not possible with a server in both ends because all GPRS connections are automaticly put

behind NAT routers.

2. The server cannot connect to the clients

3. Sometimes we even have to kill the power to the modems to bring them back online.

4. Very complicated

2. That is correct that is why you need another client to connect with another server.

1. I don't know much about how you you are using GPRS and NAT routers. You can have a Server and Client on each end just using a different port, socket ow whatever. Using some sort of timer or whatever to get a response from the client is looking for problems as you are finding out. Having to turn the power off is not a viable solution.

After wasting more time on serial connections, ports, sockets, etc. I just did more reading on the specifics and found the most error free. Closing GPS ports was one of my big problems, if it is opened and not closed correctly it requires pulling the plug.

Good luck on your communication code.
