Subject: Socket review and reimplementation
Posted by fallingdutch on Mon, 02 Apr 2007 12:37:29 GMT
View Forum Message <> Reply to Message

At the moment i am having a look at the current implementation of Sockets in Ultimate++.

I am thinking of rewriting the Sockets but not without asking you about your ideas.

I have talked to Tom and Mirek and the current plan looks like this:

a basic socket class with minimal function, derived classes like TcpSocket and UdpSocket with extended functionality.

When using CtrlCore (Gui Applications) Ultimate++ can handle (if wished) receiving and sending of data and will call a callback when data arrived or when data was sent.
So you won't have to take care about select or WaitForMultipleObject yourself.

There are also thought about adding Sockets to Core.

Any Ideas are very welcome
Bas

Subject: Re: Socket review and reimplementation
Posted by mirek on Mon, 02 Apr 2007 12:45:11 GMT
View Forum Message <> Reply to Message

On related note, there is quite a lot going on with Core now...

As Core grows, I consider dividing it into two packages, "HardCore" and "Core", former intended for lower-level stuff and OS encapsulation (Streams, Heap, String etc...), later for higher-level (Date/Time, Value, Callbacks).

Mirek

Subject: Re: Socket review and reimplementation
Posted by mirek on Mon, 02 Apr 2007 13:00:14 GMT
View Forum Message <> Reply to Message

fallingdutch wrote on Mon, 02 April 2007 08:37
a basic socket class with minimal function, derived classes like TcpSocket and UdpSocket with extended functionality.

I think this is a little bit problematic if you take into account SSL issue. It is perhaps better to have single complex Socket class that can be opened both in SSL and non-SSL mode (2 implementations vs 4 implementations).

Mirek

Subject: Re: Socket review and reimplementation
Posted by zsolt on Mon, 02 Apr 2007 19:23:09 GMT
View Forum Message <> Reply to Message

This is a good idea.

Before you start implementation, you should check libevent, I think. It provides a transparent API to poll(), select() and event based APIs of different platforms, such as epoll, kqueue, etc. Using select() is very archaic on some unixes. These kernel services are much powerful compared to the old select().

libevent is BSD licensed, very stable and very easy to use.

Subject: Re: Socket review and reimplementation
Posted by fallingdutch on Mon, 02 Apr 2007 20:19:18 GMT
View Forum Message <> Reply to Message

luzr wrote on Mon, 02 April 2007 15:00fallingdutch wrote on Mon, 02 April 2007 08:37
a basic socket class with minimal function, derived classes like TcpSocket and UdpSocket with extended functionality.

I think this is a little bit problematic if you take into account SSL issue. It is perhaps better to have single complex Socket class that can be opened both in SSL and non-SSL mode (2 implementations vs 4 implementations).

Well this basic class includes the current Data subclass which can be changed when using sslsockets so the basic class is a minimal class hiding the socket by using its own subclass (at the moment Data)
Which makes it possible to use any underlying connection as long as it is able to acces it with read and write ...

Bas

Subject: Re: Socket review and reimplementation
Posted by fallingdutch on Mon, 02 Apr 2007 20:53:35 GMT

zsolt wrote on Mon, 02 April 2007 21:23This is a good idea.

Before you start implementation, you should check libevent, I think. It provides a transparent API to poll(), select() and event based APIs of different platforms, such as epoll, kqueue, etc. Using select() is very archaic on some unixes. These kernel services are much powerful compared to the old select().

libevent is BSD licensed, very stable and very easy to use.

Thank you for that link, browsing on it at the moment.
Correct me if i am wrong, but Ultimate++ uses in its main loop MsgWaitOnMultipleObjects for windows and select for linux (to listen on the socket for incomming X11 events) so this Wait/Select will be used for the sockets to wait for an event, too.

Bas

Subject: Re: Socket review and reimplementation
Posted by fallingdutch on Mon, 23 Apr 2007 20:40:30 GMT

After doing some research i thought of implementing a base class not only for sockets but for everything you might get a callback for called "device" which can be eg a serial line, parallel port, a file or a network connection.

the device class will be using poll on posix (had a look on aio but didn't like it and seems not very complete to me)
and completion ports on windows, which has the drawback that Win98 won't be supportet (only nt supports it so nt4, 2000, XP etc will support it)

If you have any suggestions/questions please let me know.

PS: then two Socket classes will be derived one for tcp and one for udp both will (udp later) support ssl

Bas

Subject: Re: Socket review and reimplementation
Posted by zsolt on Mon, 23 Apr 2007 21:11:42 GMT

Thanks for working on that.
May I suggest to implement the loop using some pluggable interfaces?

I mean, that in some situations (e.g. web servers), a fully featured mainloop is not needed, but performance is much more important. An other aspect is, that Unixes don't have a generic event handling interface, that can be used for every type of events, like MsgWaitForMultipleObjects on Windows.

I would prefer an extensible API, usable like this or something similar:

MainLoop<StdLoop> mainloop;
or
MainLoop<LibeventLoop> mainloop;

For simple TCP servers on unixes, using libevent as an option some way would be a killer solution because of it's performance.

Subject: Re: Socket review and reimplementation
Posted by fallingdutch on Mon, 23 Apr 2007 21:44:19 GMT
View Forum Message <> Reply to Message

Well i can't follow your idea, i guess i think to much in the way i wanted to implement it. Please give me some more informations about your pluggable main Loop.

I though of a static method with a timeout eg "void device::process(int timeout)" which just checks for incomming "events" and if some events occured calls the callback then quits or if none event occured quits after a timeout.

And aother static method for adding devices to the list of processed devices.

device::process will only handle the devices, not the Windows messages or X11 Events, it will be added to the upp main loop by using PostCallback.

It is possible to process all (correct me if i am wrong) "device" events (including files) using poll!

so a typical non-Gui app would have three steps:
1. initialize all devices
2. add devices to the process list
3. call device::process in a loop (then your callbacks set in step 1 will be called)

and a typical Gui ap would be nearly the same, just call another static method eg device::EnableProcessing which would add device::process to the MainLoop of Upp

Bas

Subject: Re: Socket review and reimplementation

Posted by lundman on Tue, 24 Apr 2007 01:04:54 GMT
View Forum Message <> Reply to Message

Having written my own network/event/ssl library, there are some finer things to be aware of, but no-show-stoppers.

Using select() is fine, even under windows, but be aware that Windows don't like an empty select() call. Using poll() instead, if it exists is trivial.

Non-blocking was fine, but socketpair do not work in Windows, had to use a localhost-tcp connection (same for NetBSD in non-blocking mode). Non-blocking files in Windows are tricky, if you want to be compatible with all of the versions of Windows. The only way I could see to do it, using one code base, was to start read/write threads. (I dislike using threads, but 95/98/Me did not support any ASync calls, and I wanted to support all versions of Windows, at the time).

Anyway, when it comes to Networking, and SSL, I consider myself to be close to an expert, but alas, complete newbie when it comes to U++

All sources etc are available for peekage:

http://www.lundman.net/wiki/index.php/LiON

Sample code:

http://www.lundman.net/unix/lion-example.c

Subject: Re: Socket review and reimplementation
Posted by fallingdutch on Tue, 24 Apr 2007 05:36:19 GMT
View Forum Message <> Reply to Message

thank you Lundman

Bas

Subject: Re: Socket review and reimplementation
Posted by fallingdutch on Tue, 24 Apr 2007 07:16:10 GMT
View Forum Message <> Reply to Message

I am just browsing your code, Lundman, but i am unable to find the lion_open for Windows.

Bas

Subject: Re: Socket review and reimplementation
Posted by lundman on Tue, 24 Apr 2007 10:46:34 GMT
View Forum Message <> Reply to Message

lion_open() for files should be in file_win32.c

Get liblion-1.2.tar.gz, or CVS.