NetEvents

NetEvents is a scriptable TCP/IP application that allows script authors to use Frontier or AppleScript to create TCP/IP connections to other server, accept TCP/IP connections, read and write data, and perform domain name operations.

NetEvents is "self-documenting" in that the AppleScript Dictionary documents the calls supported by NetEvents and the sample scripts show detailed examples of how to use NetEvents.

More detailed information on NetEvents can be found on-line at:

http://www.biap.com/downloads/netevents.html

Legal Stuff

NetEvents Release Notes

NetEvents 1.0b6, 2/28/97

Corrected a few bugs. Beefed up error detection.

• Listens that complete successfully now properly send the LISC event. This was broken under b5 and only worked for connections that were opened with no pending data.

• Quitting NetEvents while TCP/IP operations are in progress doesn't crash now. NetEvents waits until the operations terminate. Pending AppleEvents will fail and callers will be notified accordingly.

• Requesting the status of stream that no longer exists (or never existed) returns INACTIVE instead of CLOSED as with previous versions. This is a more appropriate response since the connection is invalid and not just closed.

• Added a key to the connection status colors in the About Box. About Box now doesn't block TCP/IP operations.

NetEvents 1.0b5, 2/16/97

One new event, spiffy new interface, minor bug fixes.

• corrected a problem where Listens that were aborted still resulted in a Listen Complete event being sent by NetEvents. Aborting a listen now results in an "event not handled" error being returned to the caller and the stream being discarded.

• added a new event, MyIPAddress, that looks like: return (appleEvent (netEvents.id, 'WAPI', 'MYIP')) and returns a long. This event returns the TCP/IP address of the local host.

• reduced the amount of debugging output and cleaned up the status window. Added indicator lights for each connection that has been created. Colors are as follows:
gray - inactive or new
red - closed
orange - closing
yellow - open
blue - data pending
green - listening

NetEvents 1.0b4, 2/15/97
New events, better closing behavior (I think).

• Added AbortStream event. In UserTalk: return (appleEvent (netEvents.id, 'WAPI', 'ABOR', '----', long (stream)))
Kills any stream, regardless of the state it's in.

• Added AddrToName event. In UserTalk: dns = appleEvent (netEvents.id, 'WAPI', 'AD2N', '----', long (adr))
Takes a 4 byte integer address as returned from ListenComplete or NameToAddr and converts it into a domain name.

• Changed CloseStream code to consume any pending data on the connection before trying to close it. This keeps OT much happier and should eliminate the early connection termination problems with Netscape, etc.

NetEvents 1.0b3, 2/13/97
Listens are now implemented correctly! I've corrected the event definitions below in the 1.0b2 notes. Notice that the ListenComplete event now has two parameters that can indicate failure. In general, the stream parameter will be the same value as was returned from ListenStream. Usually, a failure will appear as a client address of zero.

• corrected the AETE for ListenStream.

NetEvents 1.0b2, 2/12/97
Listens are now implemented. I think. This needs serious testing as I cannot seem to get the response to be received and it has to do with the limits of my testing environment. Frontier will provide a better framework for figuring out what's wrong (or right.)

The listen works, the incoming connection is connected, but I can't tell if the LISC event is being addressed and sent properly. If someone can add the appropriate code to the NetEvents suite and send it to me, it would help immensely.

NetEvents 1.0b1, 2/2/97
This release is quite a bit beefier than the previous, limited version. Here's what has changed:

• Added threading for individual IP commands, reduced default thread stack size to 8k. This should allow up to about 16 simultaneous connections before the memory partition (384k) needs to be increased. Each connection takes the 8k for the thread plus another 12k or so for TCP/IP buffers and data structures.

• Added a minimal connection status display, updated every 5 seconds with the current connection load and the total connections processed (opened).

• Added clean-up code to close connections that are idle for longer than 60 seconds. To keep a connection active, make StreamStatus calls occasionally.

• Added code to close all active connections if the application is quit.

• Added an AETE resource.

• Additional StreamStatus return values including:
"CLOSED" - Stream has been closed by both ends. Not likely you'll ever see this one as it exists only briefly.
"OPEN" - Stream has been opened successfully and there is a live host on the other end.
"DATA" - Stream is open and there is pending data to be read.
"LISTENING" - Stream is waiting for an incoming connection (not implemented yet).
"INACTIVE" - Stream is no longer active (you've accessed a stream that timed out or is no longer valid. Discard it.)
"CLOSING" - A close request has been received from the other end of the connection, but there may still be data to read.

Still to come (soon!):
- Listening for incoming connections
- status window improvements