Subject: HttpClient Execute vs ExecuteRedirect
Posted by nixnixnix on Fri, 24 Sep 2010 18:36:49 GMT
View Forum Message <> Reply to Message

Hi,

I have a full trial version of my software that calls home as part of its weak protection scheme (clients know). However, one of my customers has some sort of fancy firewall that blocks requests from my app but not the same request when from a browser.

In order to get around this I tried using ExecuteRedirect instead of Execute but that makes my software hang on their system.

What does ExecuteRedirect do please? Also, does that sound right that it just hangs like that? Is there something I can do that will get around this kind of packet filtering or failing that will return and not cause the software to hang please?

For now I am going to just use Execute instead as it appears that if it can't get through then there is no getting through. Perhaps I am using ExecuteRedirect wrong?

BTW I did look for forum posts and documentation on this class but couldn't find any.

Obviously it is pointless trying to submit a test case for this.

Thanks,

Nick

EDIT: what happens when the packets are just swallowed and nothing comes back? Is there a timeout?

Subject: Re: HttpClient Execute vs ExecuteRedirect
Posted by rylek on Tue, 05 Oct 2010 15:37:34 GMT
View Forum Message <> Reply to Message

Hi there!

The main purpose of ExecuteRedirect is to jump over redirection responses. A HTTP server can respond with a result code in the range 300 .. 399 to indicate that the requested resource is available at a different URI. ExecuteRedirect detects these cases and follows the redirection chain to (hopefully) obtain the 'true' final resource. The maximum number of times to hop these redirections is the first parameter in call to ExecuteRedirect and by default is set to the symbolic constant HttpClient::DEFAULT_MAX_REDIRECT ::= 5. Each hop is a separate request and as such it should timeout whenever the server becomes inactive for HttpClient::DEFAULT_TIMEOUT_MSECS ::= 120000 msecs (the default is 2 minutes, see also HttpClient::TimeoutMsecs()).

Note that the timeout check is not entirely reliable: if the server was able to produce, say, 1 byte a minute, the HttpClient would never proclaim it inactive even if dowloading a single web page took three days. On the other hand, you can use the 'progress' parameter in calls to Execute, ExecuteRedirect or HttpClientGet, to supply a custom callback to detect additional reasons for aborting a pending web request (e.g. maximum total duration) and to tell the Http client to stop communicating with the server and return immediately with the appropriate error code.

Regards

Tomas

Subject: Re: HttpClient Execute vs ExecuteRedirect
Posted by nixnixnix on Sun, 17 Oct 2010 02:43:35 GMT
View Forum Message <> Reply to Message

Hi Tomas,

Thanks for that. I haven't forgotten about your ray-tracing project - I've just been super busy with other work. I will come back to it soon.

Thanks again,

Nick
