Subject: Re: AsyncWork
Posted by Oblivion on Mon, 16 Oct 2017 07:50:27 GMT
View Forum Message <> Reply to Message

Quote:Dear Mirek, dear Oblivion

For a standard GUI program to have a more responsive user interface, do you propose AsynWork to launch all jobs?
(And of course, sorry for this reference, to forget forever any Ctrl::ProcessEvents() call Sad ).

Hello Koldo,

I agree with Mirek.
You should use dedicated Thread(s) (in contrast to worker threads).
While AsyncWork, hence CoWork, does a decent job here (considering that it has cancellation feature and -"experimental"- pipe support), disadvantages of thread pools in general comes into play in such scenarios.

I don't want to go into details here, so just to keep it short:

You'll usually have no control over the states and priorites of workers, or exact execution time, or sometimes, if the job will be executed in worker threads at all. Therefore building a UI on AsyncWork, or thread pools in general, is not a very good idea. UI or any other time consuming operation can easily take advantage of it, but IMO it should not be built upon it.
(See the new SSH package, for a concrete -experimental- example of AsyncWork, and how it makes things simpler (and faster, up to 2-4 times).)

P.s. But there is a middle ground if you really need something like that. An attempt build a hybrid of worker thread concept, and dedicated threads, with a reasonble control over the given thread's states and priority: Job (the one I wrote). You may also want to try that.

Best regards,
Oblivion.