Subject: Re: RE: Job package: A scope-bound worker thread for non-blocking operations.
Posted by Oblivion on Sat, 23 Sep 2017 12:17:54 GMT
View Forum Message <> Reply to Message

Hello Mirek (and Upp community),

Job package is updated in accordance with the above discussion.

To sum up, it is heavily improved (tested on GCC 7.2.0, MinGW (supplied with U++), and latest MinGWx64, and MSC 2017):

- Void type instantiation is re-introduced for good (does not rely on anything other than plain template specializaton rules.)

- Value return semantics is re-introduced for good.

- Constant reference access operator is added. (This is especially useful when using Job with containers such as Vector, or array (e.g. Job<Vector<int>>) in a loop. See JobExample test app provided)

- Exception propagation mechanism is added. Workers will propagate raised exceptions to their caller. (when one of the GeResult(), operator~(), or IsError() methods is called.)

- JobExample test application is added to the package. This example demonstrates the above mentioned traits.

I believe with the recent update Job is now slightly superior to async/future/promise trio (not shared_future), in that it has full control over its thread (which is a proper worker thread) worker cancelleation/abortion mechanics,
and optional container-like reference access to its result which can reduce copying/moving where unnecessary. (Except Job doesn'T support reference specialization. Job<T&> is not possible for now. But that is not a big deal, really) :)

Bug reports, criticism and feedback is deeply appreciated.

[See below message for new version]

Best regards.
Oblivion