
Subject: Re: RE: Job package: A scope-bound worker thread for non-blocking operations.

Posted by [Oblivion](#) on Sat, 07 Oct 2017 13:03:44 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello,

Job package is now compatible with single-threaded U++ environment. (Yet -almost- all methods and global functions retain their functionality under ST env.)
Documentation and provided example is updated accordingly.

The code below is a part of JobExample.cpp and demonstrates both non-blocking behaviour and Job cancelling feature in a single-threaded environment:

```
void CancelJob()
{
    // Singlethreaded version. (non-blocking operation)
    // Below example is one of the simplest ways to achive non-blocking operations with Job in a
    // single-threaded environment. A finer-grained operation would involve handling of return
    // codes. (e.g. using Job<int>)

    Job<void> job;
    auto work = [=] {
        if(IsJobCancelled()) {
            Cout() << "Worker #" << GetWorkerId() << " received cancel signal. Cancelling job...\n";
            throw JobError("Operation cancelled.");
        }
    };

    Cout() << "Worker #" << GetWorkerId() << " started. (Waiting the cancellation signal...)\n";
    const int timeout = 5000;
    auto start_time = msec();
    while(!job.IsError()) {
        job.Start(work);
        if(msecs(start_time) >= timeout)
            job.Cancel(); // Or if you have more than one non-blocking operation in progress,
                // you can call CancelJobs() global function to cancel all running jobs.
    }
}
```

As always, reviews, criticism, bug reports, etc. are deeply appreciated.

Best regards,
Oblivion

File Attachments

1) [Job Package and Examples.zip](#), downloaded 453 times
