GemBox Support Center

Support for concurrent mail merging Declined

- LP Lukáš Pečiva
- **Forum name:** #GemBox.Document Feature Request

Current GemBox.Document class is not thread-safe. If you start mail merging in multiple threads it will probably end up with an error.

Since merging of huge documents could be time consuming and since most of the processors nowadays contain two or more cores it would be nice if the merging could utilize the full power of the CPU. That means to

- split the merge into several threads, every working with its own document model

- run them in parallel

- wait till all threads are complete

- join all document models into the final one

However, this is not possible now. At least not when you have custom Document.MailMerge.FieldMerging event.

Comments (5)

MG **Mario - GemBox**
11 years ago
Hi, Mail merge in large documents can be optimized so that a small region of a document (instead of an entire document) is duplicated and filled with data for each record in the data source. For an example visit the following link: https://www.gemboxsoftware.com/document/examples/mail-merge-ranges/903 Running mail merge in parallel would require the following steps: 1. Partitioning of the original data source for each thread. 2. Loading or cloning the original template document in each thread. 3. Executing mail merge on the clone of the original template document with the partition of the original data source in each thread. 4. Wait for all threads to finish and merge the resulting documents of each thread into one resulting document. As opposed to regular mail merge, concurrent mail merge has a lot of redundant operations required to enable parallelism without blocking. These operations are partitioning the data source, cloning template documents, and merging resultant documents. These extra operations make concurrent mail merge more inefficient than regular mail merge. The conclusion is that mail merge in GemBox.Document is naturally a sequential operation because housekeeping operations required to achieve parallelism outweigh the actual parallel execution. Regards, Mario
A **Andrew**
10 years ago
Have each thread create it's own document and then merge them all once the processes are complete.
LP **Lukáš Pečiva**
9 years ago
(Sorry for a long time of my response). In general I could agree with your conclusion for simple mail merge. But I can't agree for our case. In our company we are using mail merging for exporting a very complex documents for construction business area (like tunneling) where several hundreds of pages are being printed. The data source is hierarchically structured and allows fast partitioning by calling GetChildRows(). Overhead of cloning original template and merging results is acceptable for us because sequential operation is anyway taking 2-3 minutes on a very fast hardware (i7, 16GB). And our customers don't have such a good machines. BTW: The link to your "concurrent mail merge" is not working anymore. Could you, please, fix it? Since you prepared an example does it mean the feature had been implemented?
MG **Mario - GemBox**
9 years ago
Hi, First no, this feature has not been implemented yet. Second, can you try that link again, unfortunately, it seems our CRM has some difficulties with formatting that comment? Note that the text of that link is correct and please make sure that the link to which it's pointing must be the same as the text. Regards, Mario
SG **Stipo - GemBox**
4 years ago

Hi, preliminary analysis based on the proposals listed here showed that performance gain would be minimal on none. Based on that we have decided to close this request. Regards, Stipo