In-memory dataset leak testing

Synopsis

A TClientDataset should never be used for long running processes such as a Windows Service, because it has a very bad memory leak. Short lived use of TClientDatasets are fine even though the memory leak is still there. I say memory leak, but this one is a little different. The leak only exists’ while the TClientDataset object is still valid and not freed. As soon as you free it, you get the leaked memory back. When presenting my findings to other developers that I work with, I was given a list of things to check before concluding that there was in fact a leak. The result of the research is a test harness which allows mem leak testing against TClientDataset and FDMemTable. FDMemTable was added as a baseline. The results are very clear. There is a leak in TClientDatasets. The following chart shows a summation of leak testing of the TClientDataset. It should be mentioned that during testing, FDMemTable passed all test and did not leak any mem. On startup, the test harness reads its WorkingSet memory then after each test, it re-reads the WorkingSet memory to reveal the leak. The “Current Memory Usage” on the main form shows the current WorkingSet memory which matches what the Windows Task Manager shows. During testing in the chart below, the WorkingSet memory steadily climbs on each test which resulted in a leak.

Test params:

Record Count=1000. Field count=8. Consecutive test count=1000. Random Edit count=50.

Random Delete count=50. LogChanges=FALSE

|  |  |  |  |
| --- | --- | --- | --- |
| Delete 50 random records | Edit 50 random records | Go to first record after test | Results |
| FALSE | **FALSE** | **FALSE** |  |
| FALSE | **FALSE** | **TRUE** | **LEAK** |
| FALSE | **TRUE** | **FALSE** |  |
| FALSE | **TRUE** | **TRUE** | **LEAK** |
| TRUE | **FALSE** | **FALSE** | **LEAK** |
| TRUE | **FALSE** | **TRUE** | **LEAK** |
| TRUE | **TRUE** | **FALSE** | **LEAK** |
| TRUE | **TRUE** | **TRUE** | **LEAK** |

Test harness

The MemDatasetTest.exe test harness was originally built in Delphi XE7. It was ported to Delphi 2007 and Delphi 10.2 Tokyo. 32bit builds from all above versions were tested and all show the same behavior. The test harness allows you to optionally show the data in a grid or not. This lets you rule out grid internal mem storage as the culprit.

Test algorithm:

* Append “Max Records” count to the dataset with randomized data
* If “Delete random records” is checked, it will delete a random “Delete Count” of records.
* If “Edit random records” is checked, it will edit a random selection of “Edit Count” records with randomized data.
* If “Jump to first record after each test” is checked, it will simply go to the first record at the end of each test.
* At end of test, random records are added back to fill any empty records slots from prior random deletes. If it deletes 5 records in the delete step above, this step adds back 5 records to keep record count at “Max Records”.

Test Types:

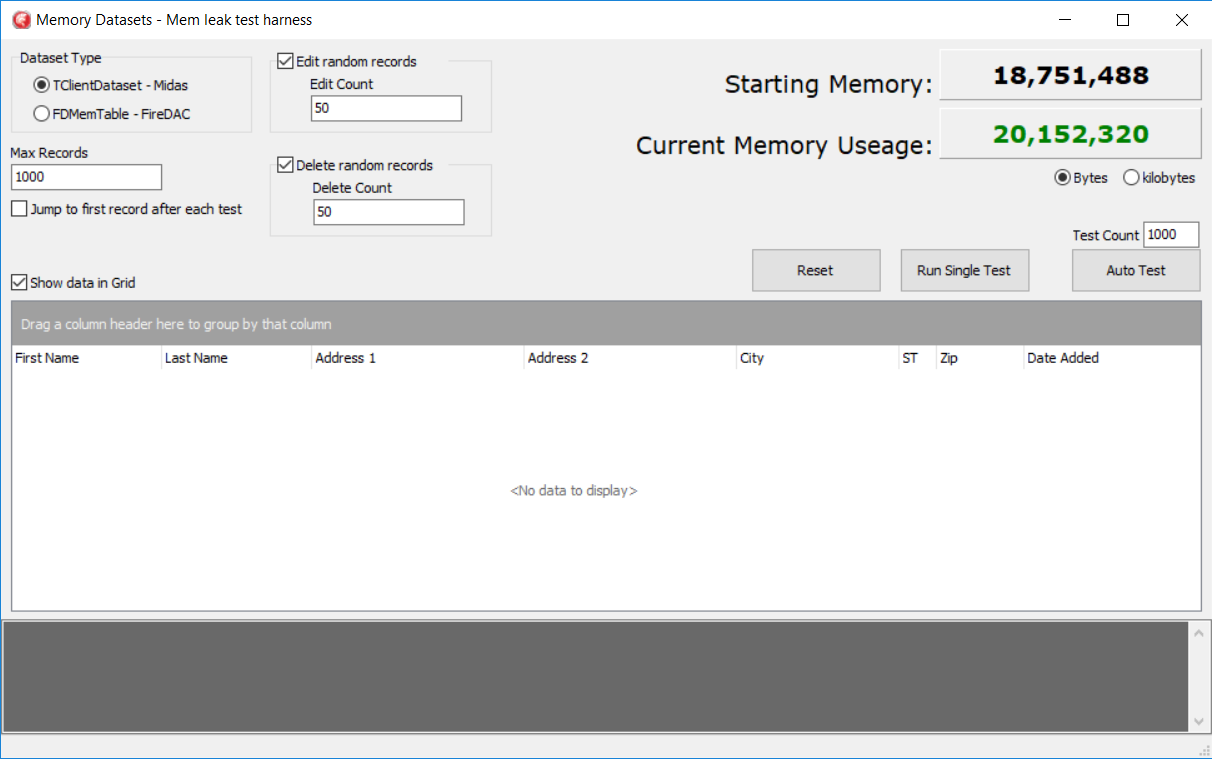
**Auto Test** – Automatically runs “Test Count” number of test for you

**Single Test** – Runs 1 test

Data set fields:

|  |  |  |
| --- | --- | --- |
| Name | Type | Size |
| FirstName | **String** | **50** |
| LastName | **String** | **50** |
| Address1 | **String** | **120** |
| Address2 | **String** | **120** |
| City | **String** | **30** |
| State | **String** | **2** |
| Zip | **Integer** | **10** |
| RecordDate | **Date** | **10** |

Test harness Screenshot:



Testing

The following TClientDataset test shows a gradual WorkingSet memory increase using the following test harness settings. The memory never releases during the test.

Settings:

Max Records = 1000

Jump to first record after each test = TRUE

Edit random records = TRUE

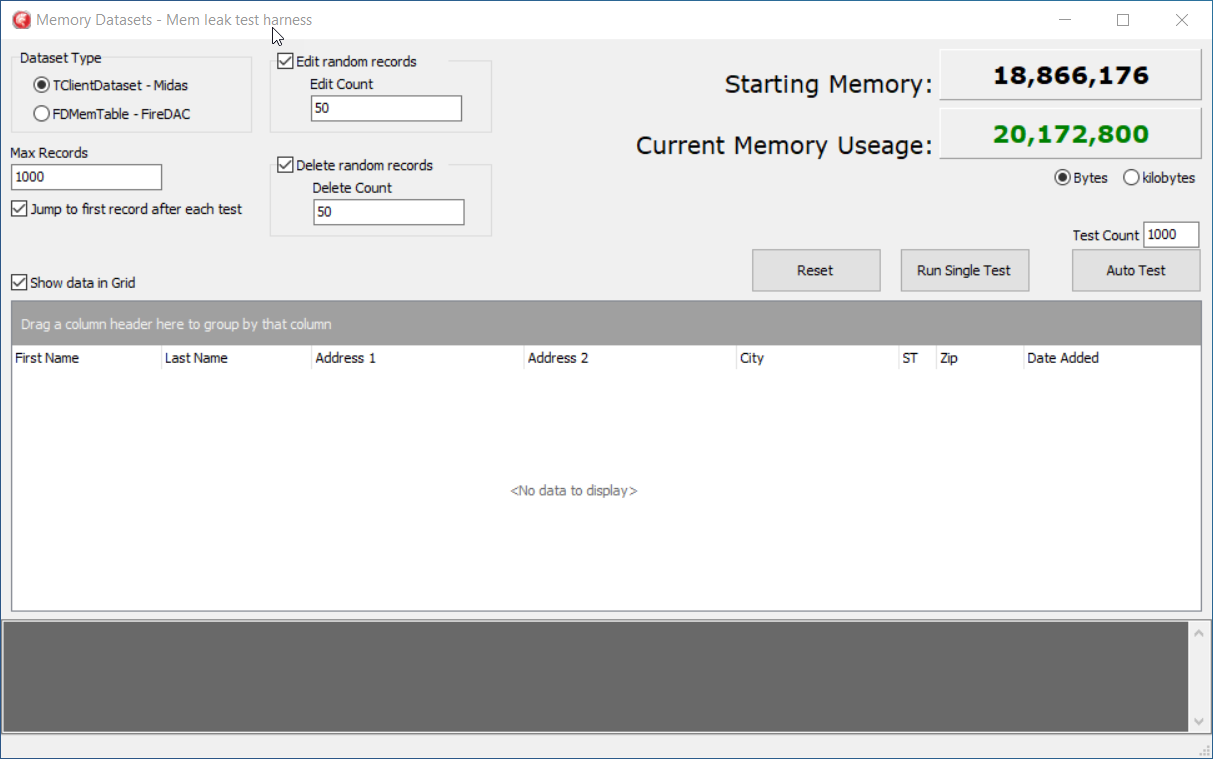
* Edit count = 50

Delete random records = TRUE

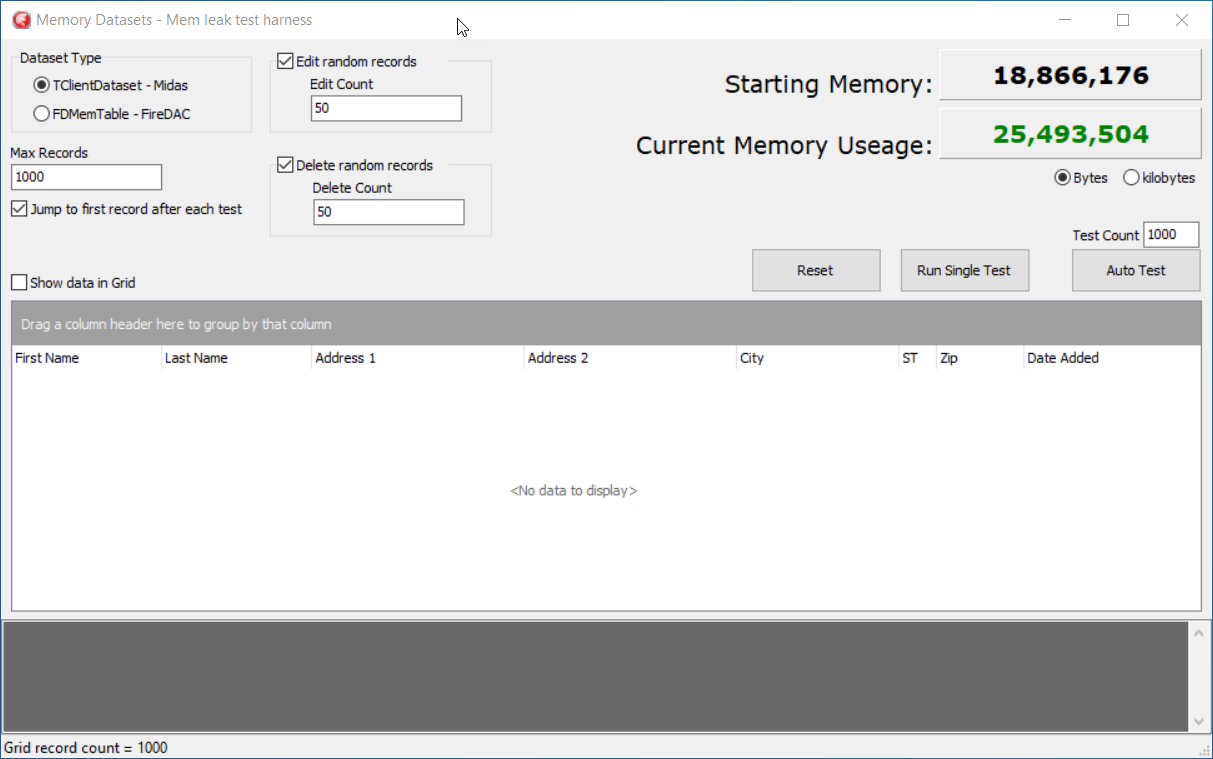
* Delete count = 50

Concurrent test count = 1000

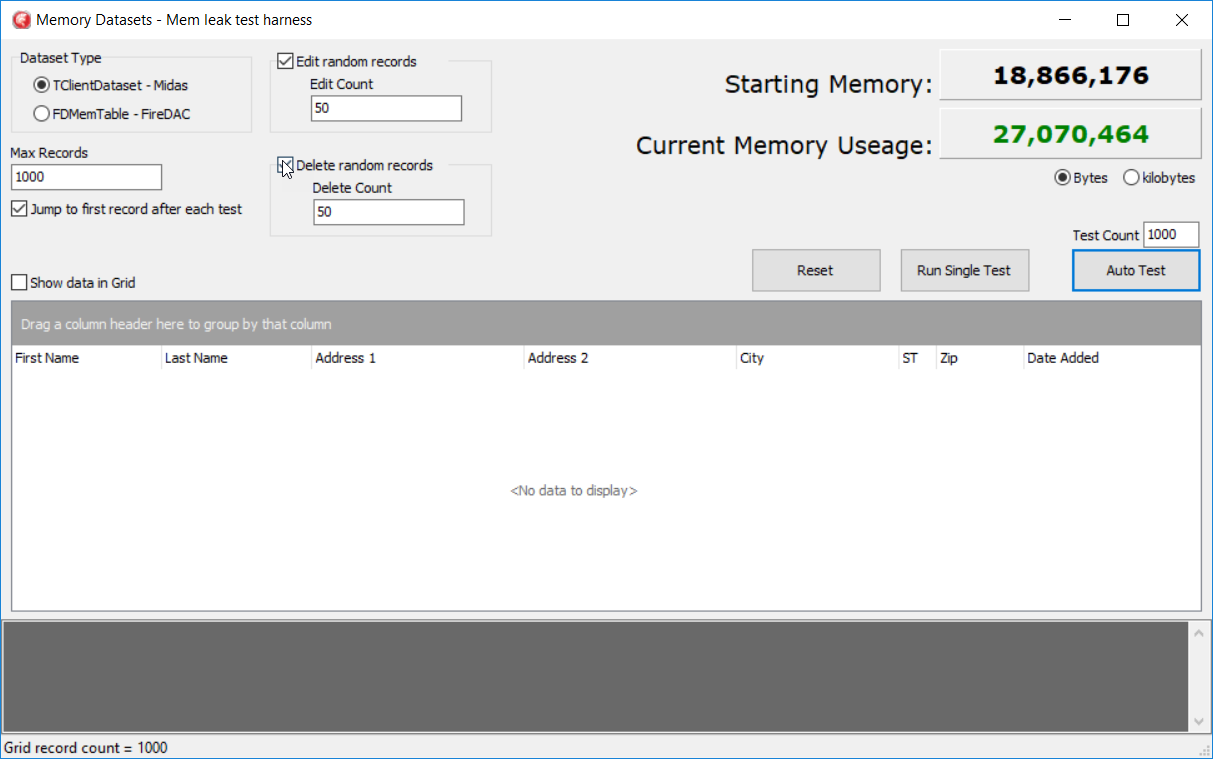
Test frame 1: **0 tests**



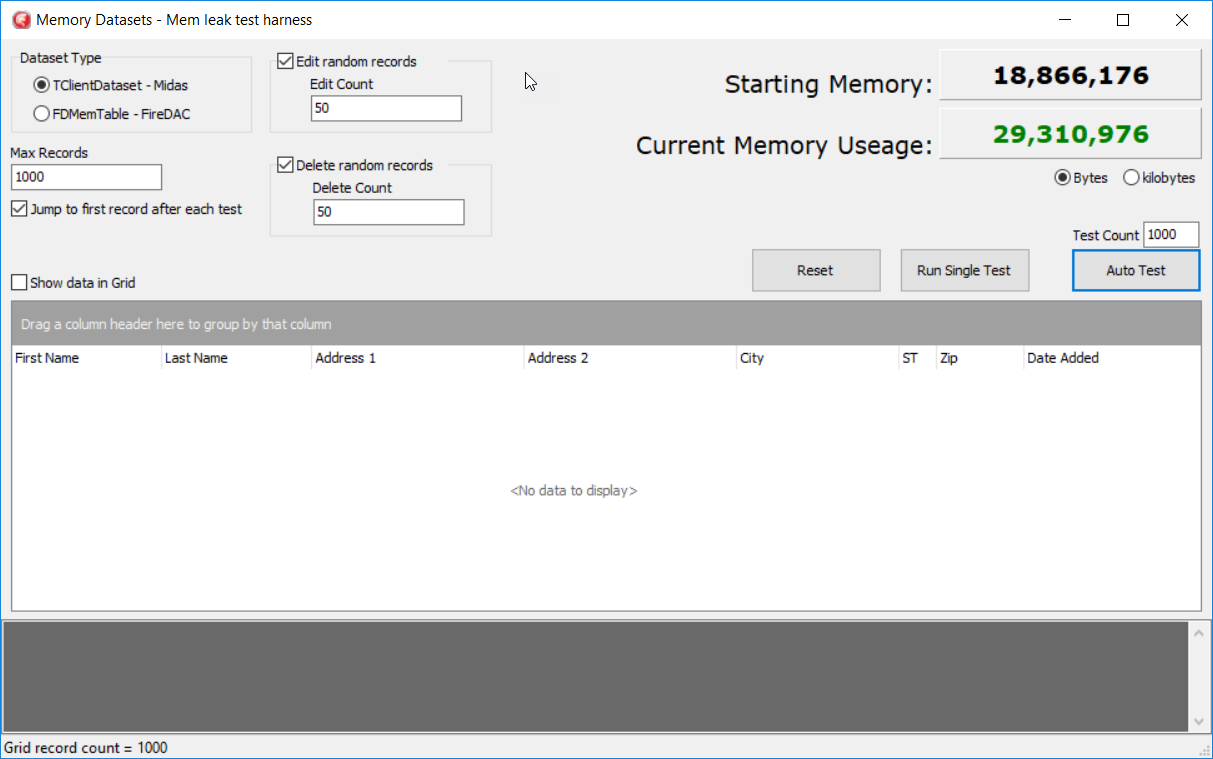
Test frame 2: **3000 tests**



Test frame 3: **6000 tests**



Test frame 4: **9000 tests**



Test Conclusion:

Test frame 1 starts with a WorkingSet memory of 20.1mb. Test frame 2 brings the WorkingSet mem up to over 25.4mb. Test frame 3 brings the mem up to 27mb. And by the end of the testing with 9000 test iterations, the WorkingSet mem is up to 29.3mb.

\*NOTE: The same test was performed with FDMemTable with absolutely no mem leak. Once the FDMemTable is loaded with the 1000 records, the WorkingSet memory never moved. It’s very stable.