
Diagnosing Memory Leaks

The easiest way to avoid managed memory leaks is to proactively monitor memory consumption as an application is written. You can obtain the current memory consumption of a program's objects as follows (the `true` argument tells the GC to perform a collection first):

```
long memoryUsed = GC.GetTotalMemory (true);
```

If you're practicing test-driven development, one possibility is to use unit tests to assert that memory is reclaimed as expected. If such an assertion fails, you then have to examine only the changes that you've made recently.

If you already have a large application with a managed memory leak, the *windbg.exe* tool can assist in finding it. There are also friendlier graphical tools such as Microsoft's CLR Profiler, SciTech's Memory Profiler, and Red Gate's ANTS Memory Profiler.

The CLR also exposes numerous Windows WMI counters to assist with resource monitoring.