User Level
New/Casual — Advanced

Getting the Most Out of ArcView GIS

Garbage Collection in Avenue

By Todd Stellhorn, ArcView GIS Development Lead

Avenue cleans up memory in ArcView GIS using a process called garbage collection. Understanding just how garbage collection operates lets programmers more effectively develop applications.

Garbage Collection Described

Avenue, the programming language and development environment that is part of ArcView GIS, has capabilities that make it easy for programmers to develop custom applications. Avenue manages memory and objects so that the programmer usually doesn't have to worry about them. Garbage collection returns memory that is no longer being used by the application to the system.

Garbage collection, as described under the entry in the ArcView GIS online help system, is "...the process of removing unreferenced objects and recovering the memory they consume." How does Avenue figure out which objects are unreferenced? More importantly, how does this affect the application developer? A deeper understanding of how garbage collection works is needed to answer both these questions.

There are many different approaches to memory management and garbage collection. ArcView GIS employs an algorithm called mark and sweep that visits each currently valid object in the system and marks it as visited or valid. After all the objects have been visited, invalid (nonmarked) objects are removed (swept). The mark and sweep approach assumes that the application or programming environment maintains a reference to each object in the system. This reference allows the algorithm to find all of the objects for marking and possible removal. Since all objects in ArcView GIS are created by Avenue, maintaining a reference to them is not a problem.

Mark and Sweep

When Avenue triggers a garbage collection pass, ArcView GIS starts its mark phase. Root objects are the first to be marked. Only a handful of objects are root objects. The Project object is the first root object. Other objects are root objects if they must remain in memory during the life of the system (e.g., the Application object, the Printer object, and Avenue global variables).

After marking the Project object, ArcView GIS performs a depth-first search. It visits and marks all of the objects connected to the Project. The vast majority of objects are visited during this traversal. For example, a Project that contains three View documents with three or four Themes will have Legends associated with each Theme and other associated properties. The Table documents will have a similar hierarchy of associated objects and properties. Each of these objects and the objects that are associated or connected with them are traversed.

Once the Project object hierarchy has been exhausted, the depth-first search ends and the next root object is traversed. When all the root objects have been traversed, the mark phase is complete. Sweeping is the second phase of the garbage collection. The sweep phase simply removes all of the objects that were not marked (i.e., all objects that are not root objects or connected to a root object).

Timing Is Everything

When does the garbage collection process occur? It can be an automatic event or programmatically triggered. The entire process from mark to sweep including the actual release of the garbage objects can occur at regular intervals, typically every 20 seconds. It can also occur when the number of objects reaches a certain threshold. The threshold, previously 800 objects, is now 2,400 objects. Garbage collection can also be triggered when at least one very large object is present such as a large polygon shape. The programmer can also trigger the process using the Avenue request av.PurgeObjs. Note that only the programmer-triggered process occurs immediately.

When ArcView GIS initiates the garbage collection process, it is scheduled to occur the next time the program is in a nonbusy state. This is an important aspect of garbage collection processing to remember for programmers. Garbage collection will not occur until ArcView GIS finishes processing the current request, whether that request is refreshing the display or running an Avenue script. The programmer needs to be aware of the number and size of the objects that are created while an Avenue script is processing because ArcView GIS will not interrupt the script to do a garbage collection pass. It will wait until the system is no longer busy.

Also, depending on the type of object, the release process may cause additional operations to be performed. In the case of File objects, if the programmer does not explicitly close the file, the release will cause the file to be closed. Knowing when a file is closed becomes extremely important when passing files to external applications. If the programmer does not explicitly close the file, choosing instead to rely on garbage collection, the file may not get closed when the programmer expects because ArcView GIS will wait for a nonbusy state to perform garbage collection.

Similarly, ArcView GIS does not keep Table Document VTabs open. These VTabs are only open while the Table display is being refreshed. However, user-created VTab objects are open while they are active and have not been terminated by a VTab.DeActivate request.

To avoid problems, the programmer should explicitly use a File.Close request to close files or a VTab.DeActivate request to deactivate a VTab instead of relying on a garbage collection pass to perform these operations.

Listing Open Files

With the release of ArcView GIS 3.1, a new request to the File class was added to help developers in cases where they need to know either how many files or which files are open. The File.ReturnOpenFilesAsList request returns a list containing the path names to all of the files currently open by ArcView GIS. An example of a simple script to call this request is

```
MsgBox.List(File.ReturnOpenFilesAsList,
"Current Open Files", "Open Files")
```

The Moral of the Story

Even though ArcView GIS automatically performs garbage collection, the programmer still needs to be aware of memory issues. If a script creates many objects, the programmer may need to trigger a garbage collection pass so that ArcView GIS will release memory back to the system that is no longer needed by the application. Also, even though a garbage collection pass will close files, the programmer should explicitly close them to avoid problems, particularly when passing files to an external application. Use the File.ReturnOpenFilesAsList request to find out how many or which files are open.