Kepler - Bug #7189

memory leak in Manager

05/11/2017 04:38 PM - Daniel Crawl

Status:	New	**Start date:**	05/11/2017
Priority:	Normal	**Due date:**	
Assignee:	Christopher Brooks	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour
Bugzilla-Id:			

Description

The Manager registers a shutdown thread with the JVM, but does not unregister it after execute() is finished, which prevents the Manager from being garbage collected. run() does unregister it.

It also seems odd that the thread is registered only once (in the constructors), but never registered again after it is unregistered. The purpose of the thread is to "gracefully stop the execution of a model if the JVM is shut down (by control-C, the user logging out, etc.)", so I would expect the thread to be registered every time execution takes place.

History

#1 - 05/12/2017 12:53 PM - Christopher Brooks

BTW - the purpose of the shutdown hook is:

> Added a shutdown hook so that model is gracefully stopped when the JVM shuts down due to control-C, the user logging out, etc. This allows 30 seconds for the shutdown, then shuts down anyway.

About leaks in Manager concerning the shutdown hook, see
https://chess.eecs.berkeley.edu/ptexternal/wiki/Main/MemoryLeaks#ManagerShutdownHooks

Edward suggested:

> The solution is that Manager should have a static code body and register exactly one shutdown hook.
> Then Manager._registerShutdownHook should create a static list of weak references to Managers that have been created.
> The one shutdown hook should check if the weak reference is valid and whether the thread is running, and if it is, call finish() and attempt a Thread.join().

I'd accept a patch that implements a fix, but I probably won't get around to fixing this myself any time soon.

#2 - 05/12/2017 01:22 PM - Daniel Crawl

Ignoring the memory leak issue, the shutdown hook does not appear to be correctly implemented since it is unregistered after the first execution (sometimes) and control-c during subsequent executions won't call it. A simple solution is to register the hook at the beginning of each model execution, and unregister when execution finishes.... this also fixes the memory leak.

#3 - 05/12/2017 01:26 PM - Christopher Brooks

Good point. I'll keep this bug in my queue.

#4 - 05/18/2017 10:28 AM - Christopher Brooks

I can take a look at this next week, after the dust settles from the other clone() bug.

#5 - 05/22/2017 04:56 PM - Christopher Brooks

I've made adjustments for how we create and delete the shutdown hook. Let's wait a bit to see what happens before closing this one.