Adding a New GC

Overview

This document describes how to add a new garbage collector to Jikes RVM. We don't address how to design a new GC algorithm, just how to add a "new" GC to the system and then build it. We do this by cloning an existing GC. We leave it to you to design your own GC!

Prerequisites

Ensure that you have got a clean copy of the source (either a recent release or the hg tip) and can correctly and successfully build one of the base garbage collectors. There's little point in trying to build your own until you can reliably build an existing one. I suggest you start with MarkSweep, and that you use the buildit script:

```
$ bin/buildit <targetmachine> BaseBase MarkSweep
```

Then test your GC:

```
$ bin/buildit <targetmachine> -t gctest BaseBase MarkSweep
```

You should have seen some output like this:

```
test:
     [echo] Test Result for [BaseBaseMarkSweep|gctest] InlineAllocation
(default) : SUCCESS
     [echo] Test Result for [BaseBaseMarkSweep|gctest] ReferenceTest
(default) : SUCCESS
     [echo] Test Result for [BaseBaseMarkSweep|gctest] ReferenceStress
(default) : SUCCESS
     [echo] Test Result for [BaseBaseMarkSweep|gctest] FixedLive (default)
: SUCCESS
     [echo] Test Result for [BaseBaseMarkSweep|gctest] LargeAlloc (default)
: SUCCESS
     [echo] Test Result for [BaseBaseMarkSweep|gctest] Exhaust (default) :
SUCCESS
```

If this is not working, you should probably go and (re) read the section in the user guide on how to build and run the VM.

Cloning the MarkSweep GC

The best way to do this is in eclipse or a similar tool (see here for how to work with eclipse):

1. Clone the *org.mmtk.plan.marksweep* as *org.mmtk.plan.**mygc***
   - You can do this with **Eclipse**:
     a. Navigagte to *org.mmtk.plan.marksweep* (within *MMTk/src*)
     b. Right click over *org.mmtk.plan.marksweep* and select "Copy"
     c. Right click again, and select "Paste", and name the target *org.mmtk.plan.**mygc*** (or whatever you like)
     d. This will have cloned the marksweep GC in a new package called *org.mmtk.plan.mygc*
   - or **by hand**:
     a. Copy the directory *MMTk/org/mmtk/plan/marksweep* to *MMTk/org/mmtk/plan/**mygc***
     b. Edit each file within *MMTk/org/mmtk/plan/**mygc*** and change its package declaration to *org.mmtk.plan.**mygc***
   - We can leave the GC called "MS" for now (the file names will all be *MMTk/org/mmtk/plan/**mygc**/MS*.java)*
2. Clone the BaseBaseMarkSweep.properties file as *BaseBase**MyGC**.properties*:
   a. Go to *build/configs*, and right click over *BaseBaseMarkSweep.properties*, and select "Copy"

   b. Right click and select "Paste", and paste as *BaseBaseMyGC.properties*
   c. Edit BaseBaseMyGC.properties, changing the text: "*config.mmtk.plan=org.mmtk.plan.marksweep.MS*" to "*config.mmtk.plan=org.mmtk.plan.**mygc**.MS*"
3. Now test your new GC:

```
$ bin/buildit <targetmachine> -t gctest BaseBase MyGC
```

You should have got similar output to your test of MarkSweep above.

That's it. You're done. 

Making it Prettier

You may have noticed that when you cloned the package *org.mmtk.plan.marksweep*, all the classes retained their old names (although in your new namespace; *org.mmtk.plan.**mygc***). You can trivially change the class names in an IDE like eclipse. You can do the same with your favorite text editor, but you'll need to be sure that you change the references carefully. To change the class names in eclipse, just follow the procedure below for each class in *org.mmtk.plan.**mygc***:

1. Navigate to the class you want changed (eg *org.mmtk.plan.**mygc**.MS)*
2. Right click on the class (MS) and select *"Refactor->Rename..."* and then type in your new name, (eg *MyGC*)
3. *Do the same for each of the other classes:#* MS -> MyGC#* MSCollector -> MyGCCollector*
   - *MSConstraints -> MyGCConstraints*
   - *MSMutator -> MyGCMutator*
   - *MSTraceLocal -> MyGCTraceLocal*
4. Edit your configuration/s to ensure they refer to the renamed classes (since your IDE is unlikely to have done this automatically for you)
   - Go to *build/configs*, and edit each file **MyGC.properties* to refer to your renamed classes

Beyond BaseBaseMyGC

You probably want to build with configurations other than just BaseBase. If so, clone configurations from MarkSweep, just as you did above (for example, clone *FastAdaptiveMarkSweep* as *FastAdaptive**MyGC***).

What Next?

Once you have this working, you have successfully created and tested your own GC without writing a line of code!! You are ready to start the slightly more tricky process of writing your own garbage collector code.

If you are writing a new GC, you should definitely be aware of the MMTk test harness, which allows you to test and debug MMTk in a very well contained pure Java environment, without the rest of Jikes RVM. This allows you to write unit tests and corner cases, and moreover, allows you to edit and debug MMTk entirely from within your IDE