**HOME**

**PAGES**
About Cog
About this blog
Building a Cog Development Image
Cog Projects
Collaborators
Compiling the VM
Downloads
Eliot Miranda
On-line Papers and Presentations

**CATEGORIES**
Cog
Spur

**BLOG ROLL**
3 + 4
Andreas Röver
Clément Béra's Smalltalk Tips 'n Tricks
Croqueteer
David Korten
Fun Script
HREF Considered Harmful
Jane Goodall Institute
Jon’s Place
Oh, meta…
Procrastineering
Room 101
RSA Blogs & Animates
Squeaking Along
thisContext
Thomas J. Peckish's Catfish

**COG**
0 Introduction
01 Closure Overview

# Building a Cog Development Image

The Cog VM source is in a Squeak Smalltalk Monticello package at http://source.squeak.org/VMMaker. Load the newest version of VMMaker.oscog, e.g. VMMaker.oscog-eem.2345 (*).

Building the image manually is tedious. Fortunately an automated build using the current Squeak release is provided in the image subdirectory of http://www.github.com/OpenSmalltalk/opensmalltalk-vm.

```
$ git clone http://www.github.com/OpenSmalltalk/opensmalltalk-vm
$ cd opensmalltalk-vm/image
$ cat README # this may be more up-to-date than this blog page
$ ./buildspurtrunkvmmakerimage.sh    #
or buildspurtrunkvmmaker64image.sh
```

This almost works perfectly; but…

- on Windows you’ll need to use Cygwin and install wget, e.g. from from http://gnuwin32.sourceforge.net/packages/wget.htm, probably as http://downloads.sourceforge.net/gnuwin32/wget-1.11.4-1-setup.exe
- the second part of the build loads VMMaker.oscog and the support packages. During this you’ll get prompted for your initials. Supply some to proceed.

Once you’ve built a VMMaker image, read the class comments of StackInterpreterSimulator and CogVMSimulator for running the simulator. Alternatively use the example expressions in VM Simulation Workspace.text. Slang test expressions are provided in Slang Test Workspace.text. Test expressions that run the JIT to produce machine code disassembly from methods in the image are provided in In-image Compilation Workspace.
Clément has done a lovely screen cast showing how the simulator is used to develop the VM. In this case he debugs a code generation error in the speculative inlining JIT. https://www.youtube.com/watch?v=hctMBGAXVSs. And here’s his accompanying blog post.

The VM source is generated from as VMMaker.oscog image whenever required. For the official source tree, which resides on github (see http://www.mirandabanda.org/cogblog/compiling-the-vm/) this is done by core developers on an as-needed basis. For one’s own development, generate the source as and when you see fit. For much of the time one can develop a VM using there simulator, but soon enough one will want to generate a real VM or have to debug the real VM because while the simulator is powerful, it is slow and of necessity incomplete (for example, the simulator cannot yet simulate FFI calls).

(*) The VMMaker package is for the Interpreter VM and the VMMaker-oscog package is an obsolete version of the Pharo version of Cog.

02 Closure Bytecodes
03 BlueBook Methods
04 Closure Compiler
05 Closure Stats
06 Simulate Out Of The Bochs
07 The Idée Fixe and the Perfected Profiler
08 Under Cover Contexts and the Big Frame-Up
09 Build me a JIT as fast as you can…
10 An Arranged Marriage – The Honeymoon

**DOWNLOADS**

Closure Bootstrap
Cog VMs

**LINKS**

CGO Conferrence
Eleazar & I in Concepcíon
Site Stats

**SPUR**

A Spur gear for Cog
Lazy Become
Lazy Become and Primitives

**RSS FEEDS**

Posts
Comments

**META**

Log in

**SEARCH**

Find