Present and Future of the Pharo VM

Headless and Beyond

Pablo Tesone

Pharo Consortium Engineer

Who I am!

Pablo Tesone

Pharo Consortium
Engineer

- 20 years trying to code
- 10 years of experience in industrial applications
- PhD in Dynamic Software Update
- Interested in improving development tools and the daily development process.
- Enthusiast of the object oriented programming and their tools.

Also, playing with me:

Guille Polito
CNRS Engineer
RMod Team

Esteban Lorenzano
Pharo Consortium
Engineer

Announcement

Still Alpha!
But improving fast!

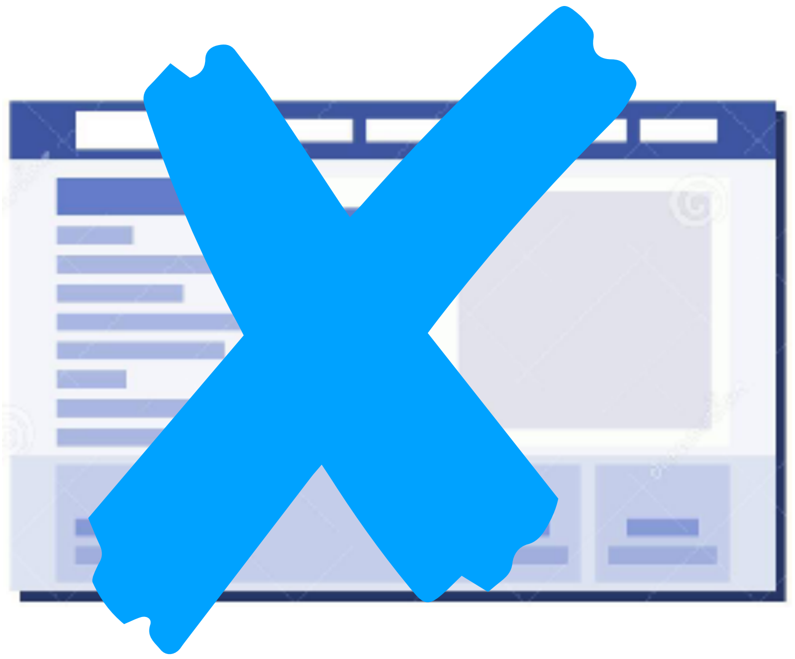
Pharo 8 Headless VM is out!

Get it now from zero-conf
http://get.pharo.org

```
$ wget -O - get.pharo.org/64/80+vmHeadlessLatest
$ ./pharo Pharo.image
```

What's Headless?

- More than a VM not showing the GUI
    - Remove window management
    - Remove event handling
    - Only running my program!

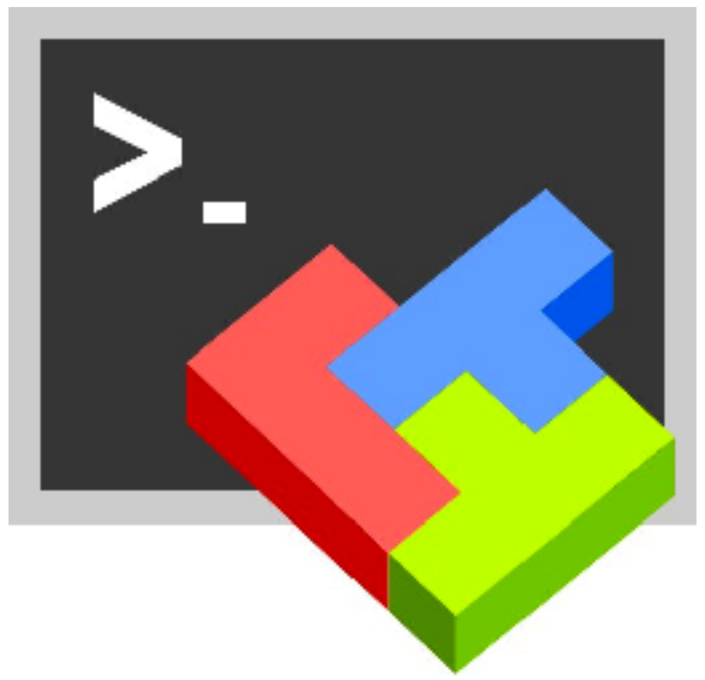


Why Headless?

- Command-line tools
- Scripting

UI:

- **Command-line arguments**
- **Files**
- **Piping standard input / output**



Why Headless? (II)

- Servers

- **No UI or Web UI**
- **Network communication**
  - **Sockets (TCP/UDP/Unix)**
  - **RPC**

Why Headless? (III)

- Services

  - No UI
  - Network communication
    - Sockets (TCP/UDP/Unix)
    - RPC
  - External control of lifecycle
  - Container dependent API
  - E.g: Window Services / launchd / xinetd / Lambda

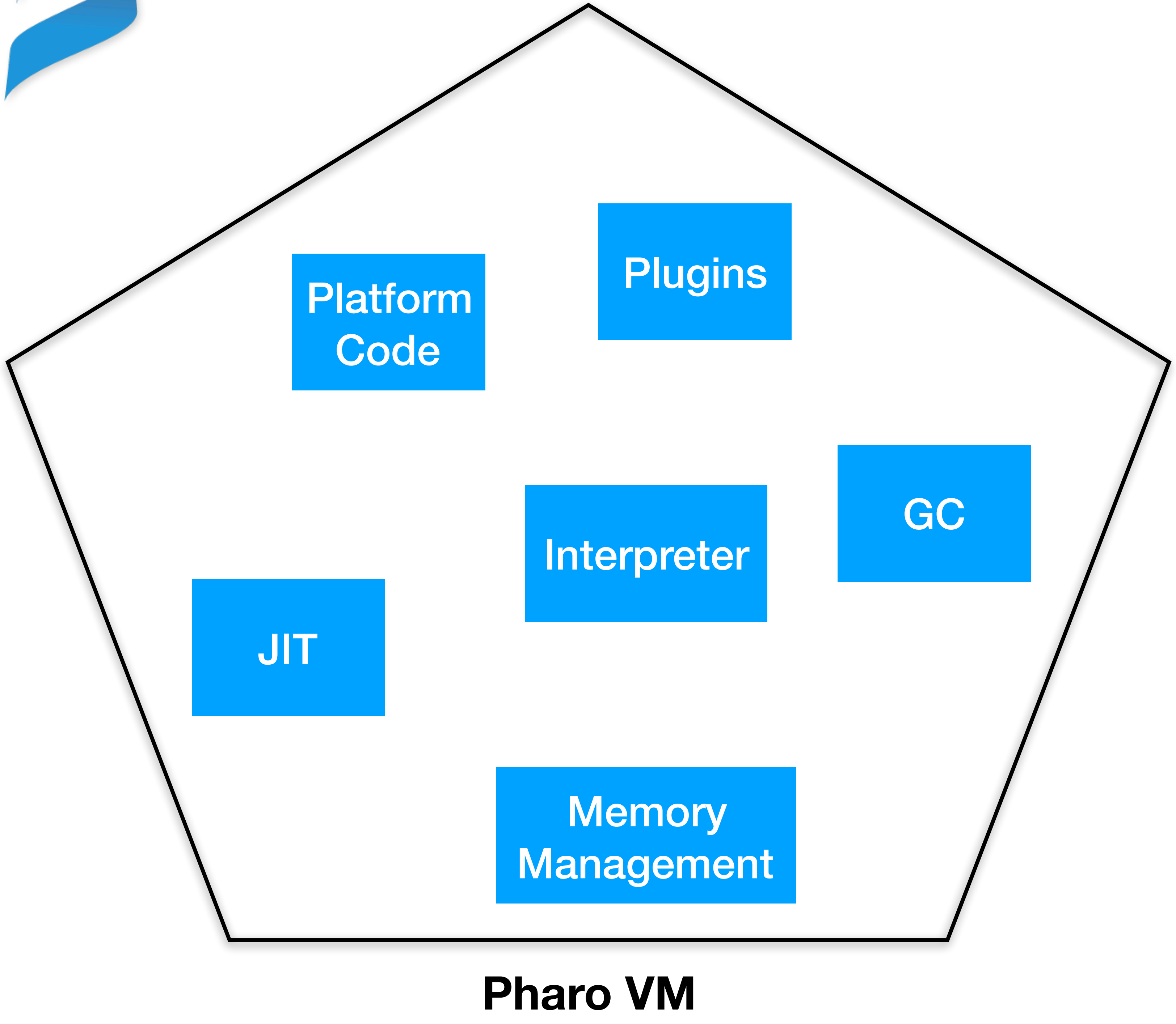
Why Headless? (IV)

- Stand-alone Desktop Applications

- Has GUI
- The GUI depends of the APP
- No Morphic
- E.g: GTK+3, OpenGL, WindowsForm, Cocoa
- All events and windows handled in the application (image side)

So…we want a headless VM!!!

Reality: a VM is Complex



We need tools to:

- Handle complexity
- Focus in the important stuff
- No fear of change

Slang ported to Pharo

Dr TEST APPROVED

- Taking advantage of the tools (Refactorings / Calypso / GT Inspector / Spotter / Iceberg) Build specific tools
- Tests for the translation
- Fix the incompatibilities of Slang with Pharo
- Ensure code-generation repeatability
- Generate the code in each build!!

Eat our own food!

Improved Build Process

- Simpler declarative building process
- Multi-platform (Windows / OSX / Linux)
- IDE integration (XCode / Eclipse / Visual Studio / …)
- Faster (Better dependency)

Easier to Understand and to modify!

Continuously Integrated

Dr TEST APPROVED

- Automatic build on each commit
- Building and testing branches & pull-requests
- Testing in three platforms: Windows / OSX / Linux
- Testing with VM tests, FFI tests, Pharo tests & benchmarks

I want a lot of tests!

Jenkins

Version Controlled

- All the code in a single Git repository (Slang + C)
- No generated code
- Available and visible in GitHub
- Tagging / Branching policies
- All modifications through PRs

We need to go back to any point in the past.

As we won…

- Safety
- Ability to change
- Early detection of errors & performance regressions
- Easier contribution

So,… without fear….
we touched the VM

Results (1/6)

Power to the Image

- Fully Backward Compatible Image
- The image open and controls the UI & Events
- Customisation of Window and Menus
- Two backends: Gtk+3 / SDL2
- Fully implemented with UFFI

Results (2/6)

Version Controlled

- Single GIT Repository
- VMMaker code in Tonel (Thanks feenk !)
- VMMaker in Pharo 7 & 8
- Source code restructuring
- Binary Dependencies Control

Results (3/6)

Much Simpler

- Removing Unused Plugin
- Cleanup of UUID, Socket & SSL plugins
- Source code restructuring
- Removing duplicated code
- Platform code minimisation

Results (4/6)

Towards Embedding

- All Plugins are External Dynamic Libraries
- The VM is a Dynamic Library
- The main executable is a thin customisable frontend
- Initial client API

Results (5/6)

Easier to Debug

- Improved Logging
- IDE Integration
- Improved Stack-dumps

Results (6/6)

General Improvements

- Unified Parameter handling
- Improved Module lookup
- Fixing Warnings and Type definitions!!!

What are we doing… now

- Experiments (CogMT, Running the VM outside the main thread, LibFFI backend, removing heartbeat……..)
- JIT Tests (Unicorn / LLVM)
- Interpreter & GC Tests
- VM Benchmarks
- GC policies

Changes are getting easier, we cannot stop.

Future

- Documentation / Tests / Comments
- Environmentally Friendly VM
  - Event-Based
  - Less idle-state CPU usage
  - Container friendly
  - Reduce battery consumption
- Embeddable
  - Integrating Pharo in other Apps.

NOW!

The most important concept of this presentation…

Pharo VM IT'S YOURS!

Try it!

Report Issues!

Break it!

Hack it!

Improve it!

Learn!

https://ci.inria.fr/pharo-ci-jenkins2/job/pharo-vm/

pharo-project/opensmalltalk-vm

Thanks!!!