Hazelnut

dynamically create a kernel
in a reflexive language

Benjamin Van Ryseghem

Introduction

Seed

- PineKernel: MicroSqueak portage
- Hazelnut: build a new kernel starting from Pharo kernel
- Both based on Micro-Squeak

Micro-Squeak

- John Maloney's project
- Done in 2004
- Released in september 2010
- Proof of concept: minimal kernel (47 Classes)

How to create a new image in 3 steps ?

Contents

Contents

Which classes need to be collected ?

- First approach: collect all the classes needed by **Object** to have an autonomous system
  - About 800 classes on 1800 (½ of the system)

- Second approach: provide a list of classes
  - Start from **Object**
  - Recursively analyze dependencies
  - About 200 classes on 1800 (1/9 of the system)

How to to build a new kernel structure ?

I. Mark objects

- Trace the objects and mark wanted ones
  - Based on SystemTracer2
- Filling up a list
- Easy process
- It works
- No living kernel

How to to build a new kernel structure ?

II. A new “namespace”

- Create a new System Dictionary
  - HazelSmalltalk

- Filling it up with copies of wanted classes
  - Perform a very deep copy
  - Take care of the class and metaclass hierarchy

- No recompilation
- Not handled by the system

Contents

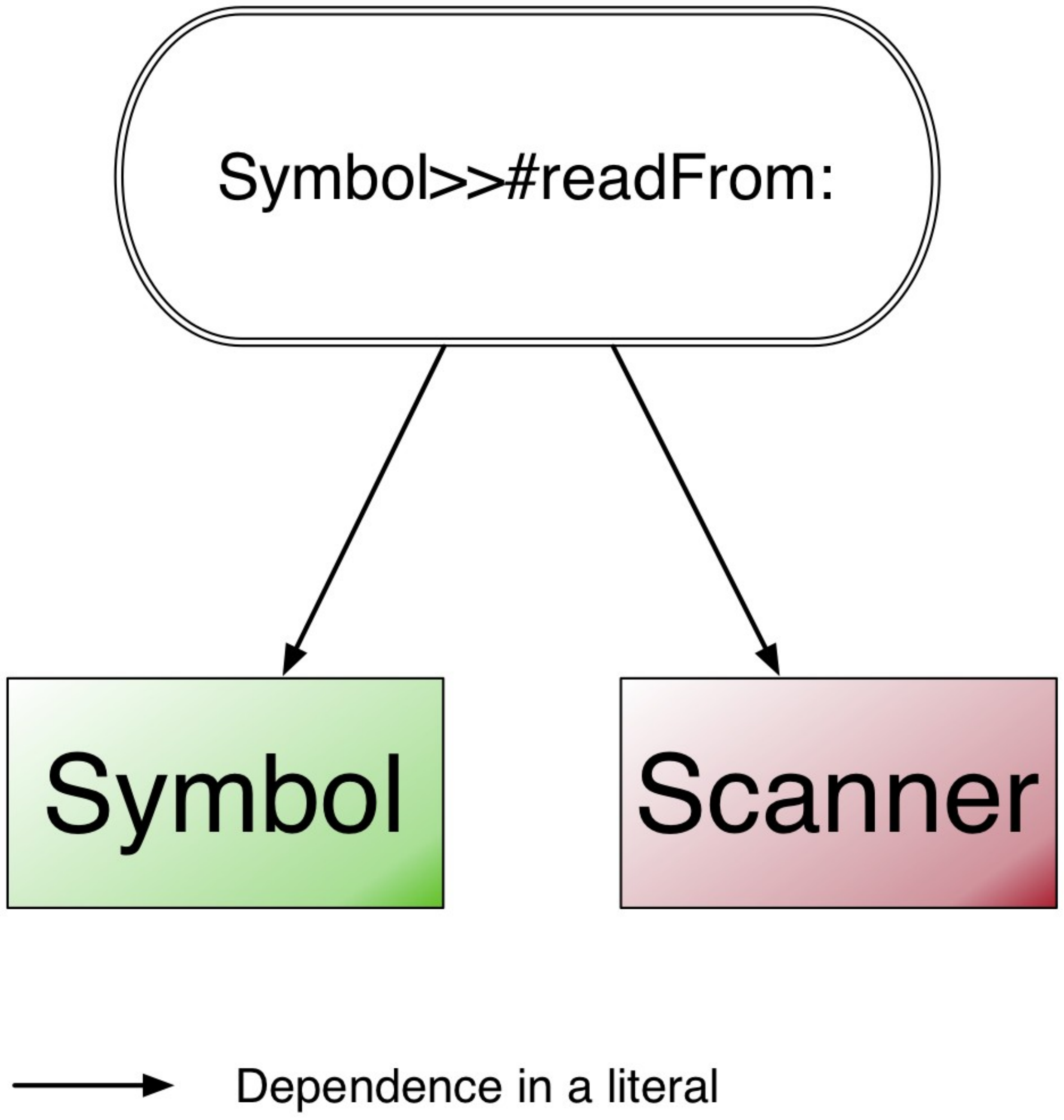
Remove dependencies to unneeded classes

I. HazelTracer2

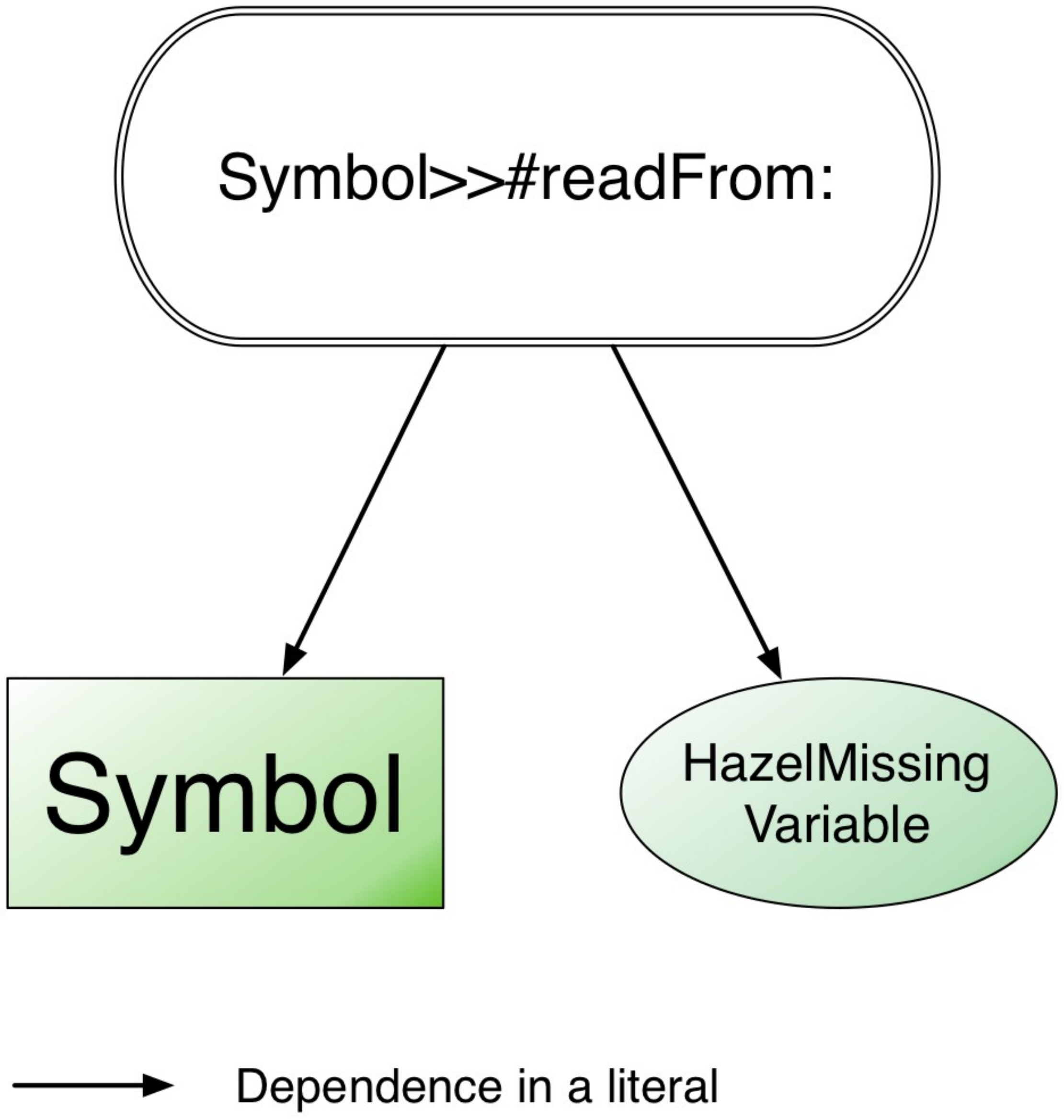
- Detect references to unwanted classes

- Fix them using a wrapper (HazelMissingVariable)

Remove dependencies to unneeded



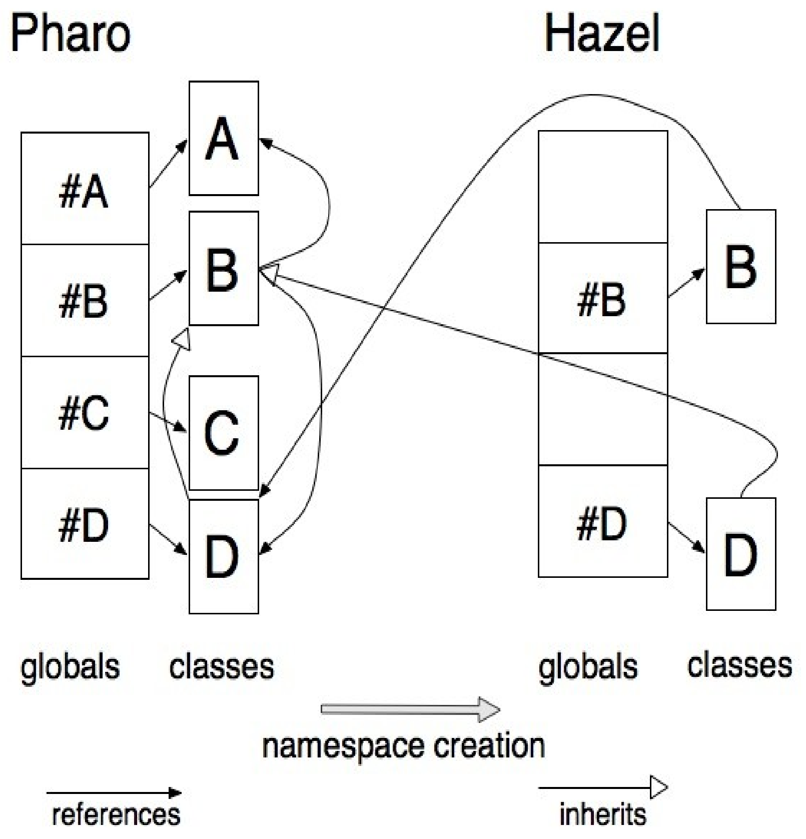
Remove dependencies to unneeded



Remove dependencies to unneeded classes

II. HazelKernelBuilder

- Detect references to the Pharo world in methods
- Fix them
  - A method: remove it
  - A class variable: set it to nil



Bootstrap the new kernel

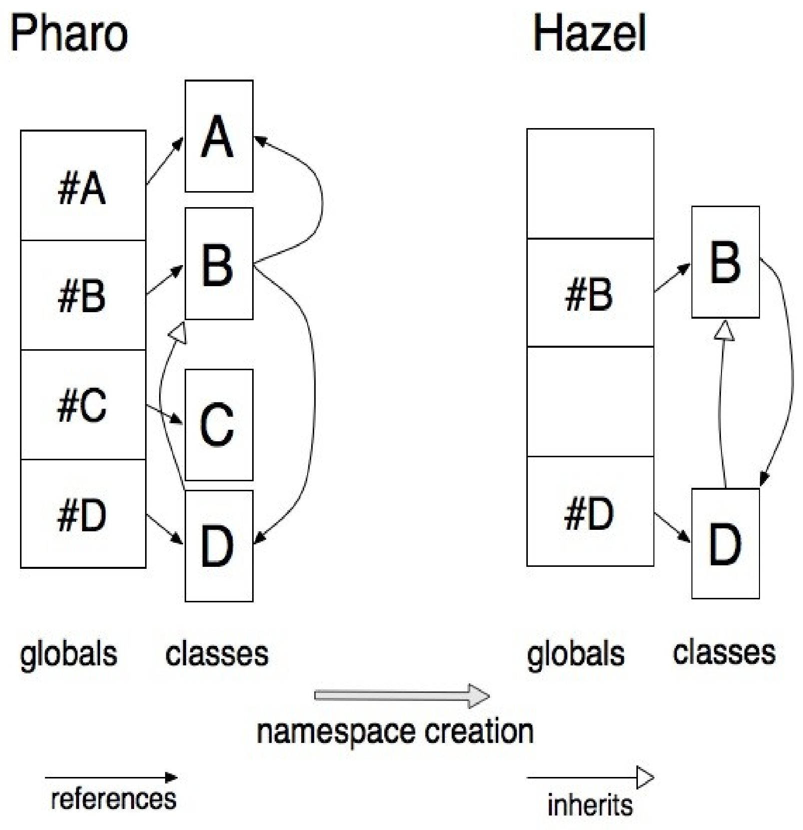
I. HazelTracer2

Nothing to do :)

Bootstrap the new kernel

II. HazelKernelBuilder

- Change Pharo references into Hazel dependencies
  - Fix methods literals
  - Change the class of HazelSmalltalk
  - Change the class of the HazelSmalltalk associations



Contents

SystemTracer2

- Tracing and writing process already coded
- Collect all the needed objects
- Check if there is in the granted list
- Finally serialize them in a binary stream

- Error if some objects change between traces
- Had to fix SystemTracer2 for Pharo

Micro-Squeak like serialization

- First approach
- Collect all the needed objects
- Parse them twice
- Finally serialize them in a binary stream

- Objects untraced
- Error during the serialization
- Code hard to debug or rewrite

What if we switch the SOA ?

- Change the SOA
- GarbageCollect unwanted objects
- Handled by the VM

- Can't switch some classes
- Modify method context during execution
- Hangs the VM

Next Steps

- Load code
  - What is the minimal image able to go back ?
  - How to load code without compiler ? (Fuel ?)
- Declarative kernel
- Get a better definition of the kernel

Conclusion

- Create a new structure
- Isolate it
- Create a new image