Polyglot Programming with GraalSqueak and GraalVM

Fabio Niephaus and Robert Hirschfeld

Oracle PIs: Mario Wolczko and Tim Felgentreff

Software Architecture Group
Hasso Plattner Institute
University of Potsdam, Germany

https://www.hpi.uni-potsdam.de/swa/
https://github.com/hpi-swa/graalsqueak/

November 25, 2019

Agenda

- Introduction
- GraalSqueak and GraalVM Demos
- GraalSqueak Implementation Details
- Polyglot Programming
- Conclusions

HPI

Ask me anything!

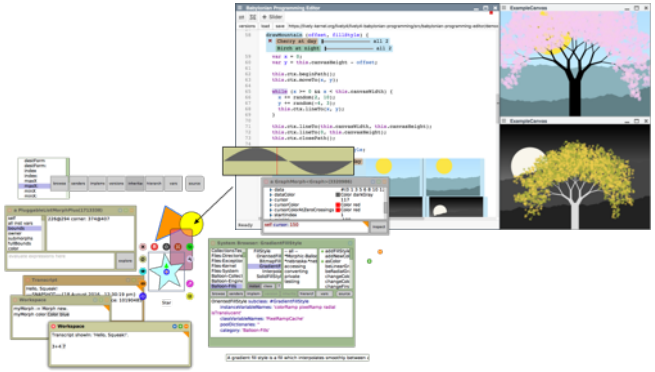
Agenda

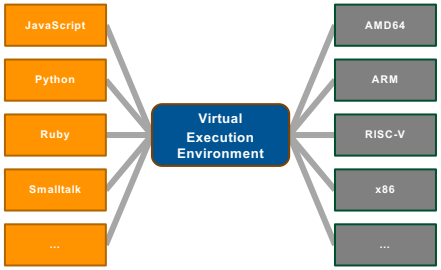
- Introduction
- GraalSqueak and GraalVM Demos
- GraalSqueak Implementation Details
- Polyglot Programming
- Conclusions

Prof. Dr. Robert Hirschfeld
Hasso Plattner Institute, University of Potsdam, Germany

Software Architecture Group

Development Tools	Programming Languages	Virtual Machines





https://www.hpi.uni-potsdam.de/swa/

Prof. Dr. Robert Hirschfeld
Hasso Plattner Institute, University of Potsdam, Germany

Software Architecture Group

Robert Hirschfeld
Sabine Wagner
Alan Borning
Tim Felgentreff
Bert Freudenberg
Richard P. Gabriel
Johannes Henning
Bastian Kruck
Jens Lincke
Nikolas Martens
Toni Mattis
Eliot Miranda
Fabio Niephaus
Tobias Pape
Stefan Ramson
Patrick Rein
Marcel Taeumel
Marcel Weiher

https://www.hpi.uni-potsdam.de/swa/

Personal Background

Research Topic

- Advancing the Polyglot Programming Experience

Research Project

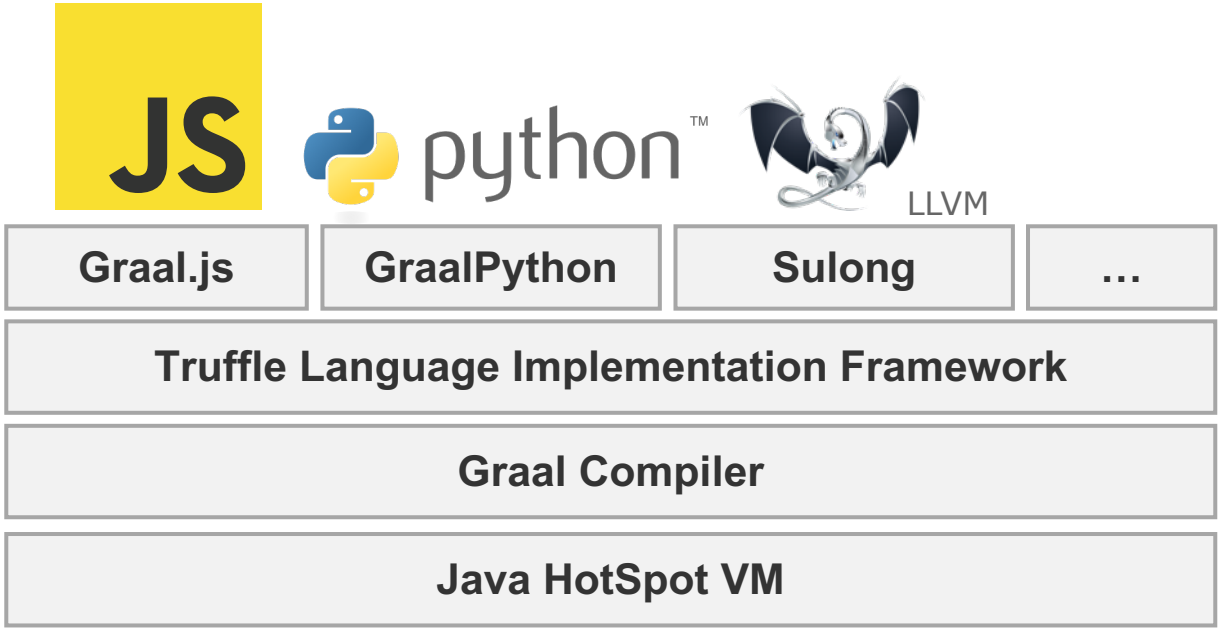
- GraalSqueak: Squeak/Smalltalk VM written in Truffle

GraalVM-related Publications

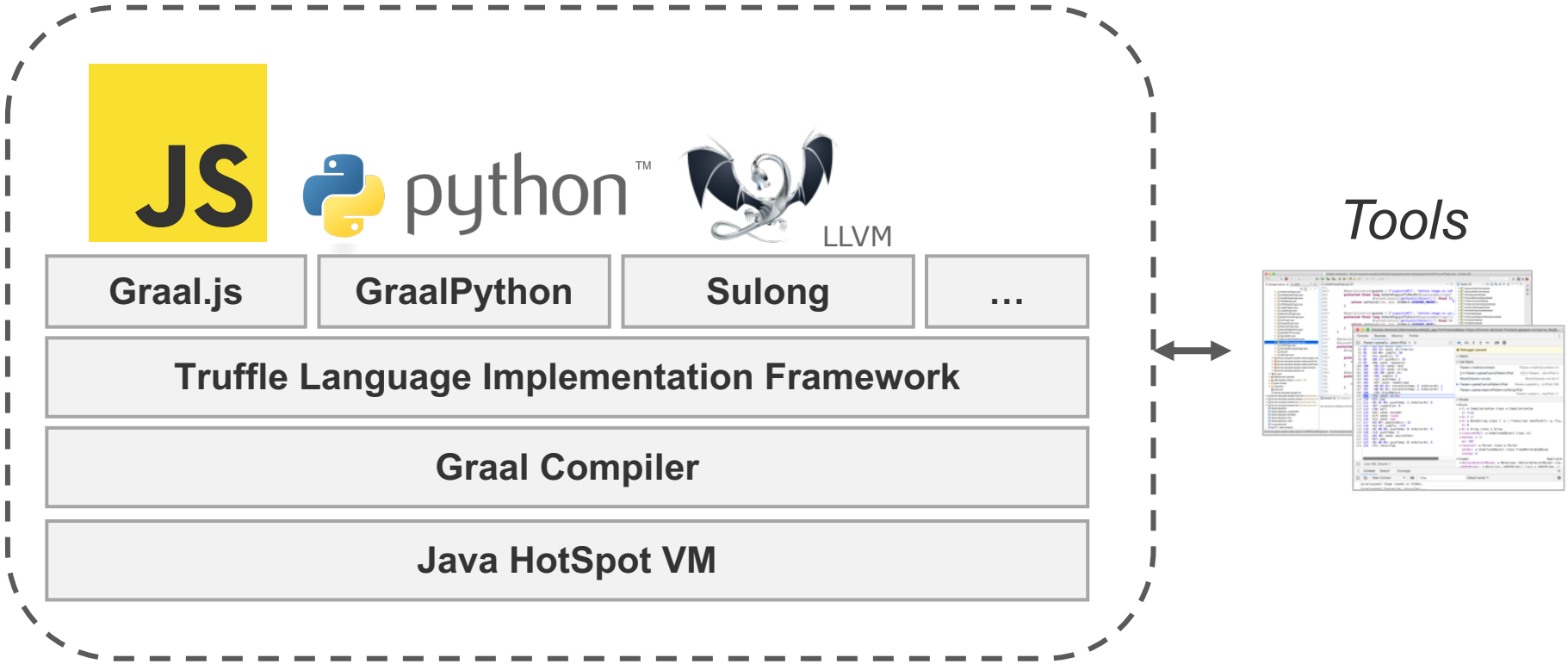
- *GraalSqueak: Toward A Smalltalk-based Tooling Platform for Polyglot Programming*. MPLR'19.
- *Language-independent Development Environment Support for Dynamic Runtimes*. DLS'19.
- *Efficient Implementation of Smalltalk Activation Records in Language Implementation Frameworks*. MoreVMs'19.
- *Polyglot Adapters*. ICW'19.
- *PolyJuS: A Squeak/Smalltalk-based Polyglot Notebook System for the GraalVM*. PX/19.
- *GraalSqueak: A Fast Smalltalk Bytecode Interpreter Written in an AST Interpreter Framework*. ICOOOLPS'18.
- *Live Multi-language Development and Runtime Environments*. ‹Programming›'18.

https://www.hpi.uni-potsdam.de/swa/publications/

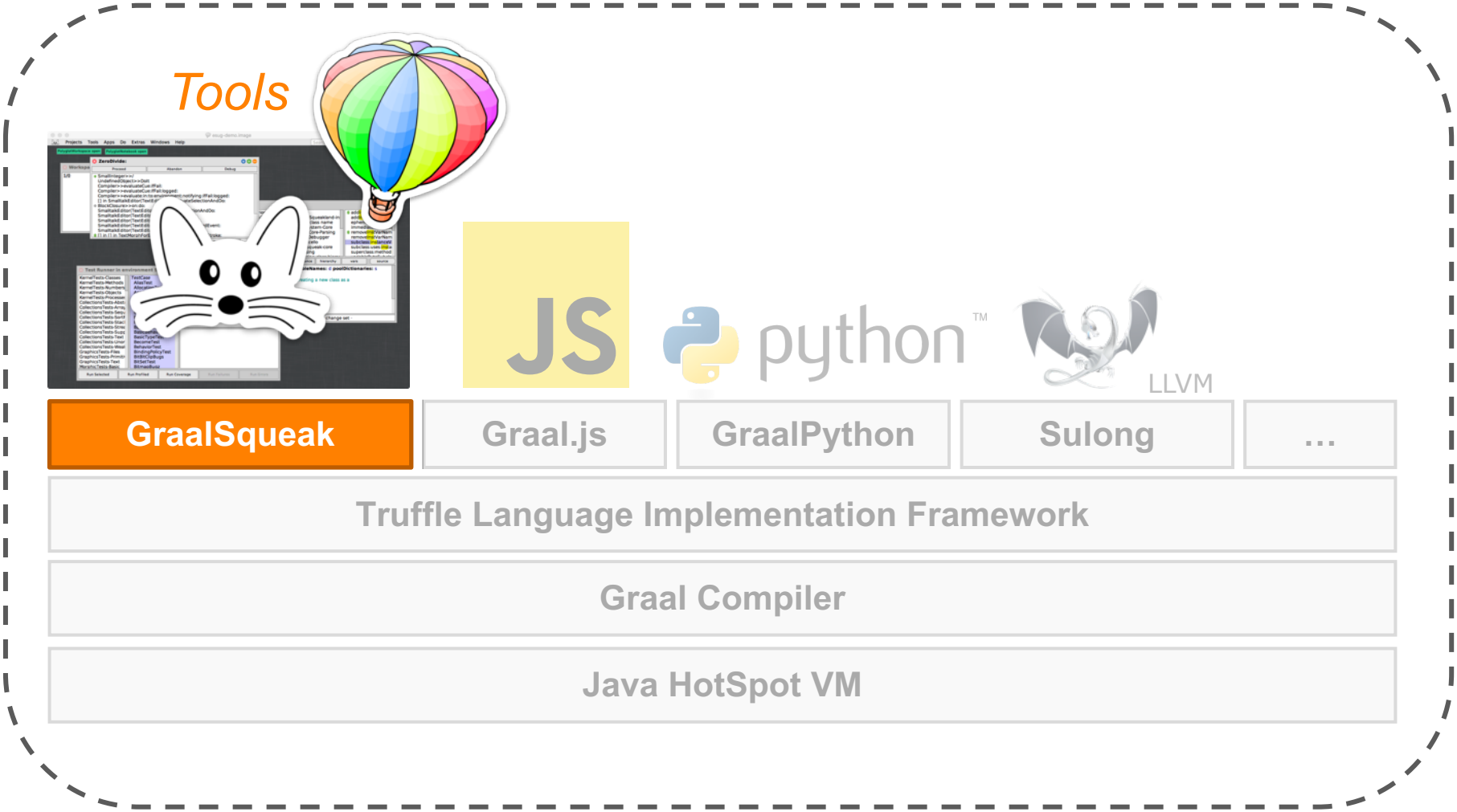
GraalVM Technology Stack 1/3



GraalVM Technology Stack 2/3



GraalVM Technology Stack 3/3



DEMOS

GraalSqueak Demo

Polyglot Notebook

Polyglot Notebook

Run all | Add cell | Load | Save

Markdown

Conference Contributors per Country
We are interested in how many people per country are contributing to <Programming> 2019. First, we download the `poeple-index` from the conference's website and extract the data from the `#results-table` using Ruby and its powerful Nokogiri HTML parsing library. The result is stored in the polyglot bindings as `rows`.

Ruby

```
require "nokogiri"; require "open-uri"
url = "https://2019.programming-conference.org/people-index"
doc = Nokogiri::HTML(open(url))
bindings["rows"] = doc.css("#results-table .row").map { |row| row.css("> div").map(&:content) }
```

root	ForeignObject[arraySize=277]
1	ForeignObject[arraySize=4]
1	ForeignString["Rodin Aarssen"]
2	ForeignString["CWI, Netherlands"]
3	ForeignString["Netherlands"]
4	ForeignString["Author of Concrete Syntax with Black Box Parsers within the Research P
2	ForeignObject[arraySize=4]

Markdown

Then, we use the Python library `pycountry` which provides a database of all country names to filter and transform the list of participants into a list of country names. This list is stored in `countries`.

Python

```
import pycountry
bindings["countries"] = [c.name for c in pycountry.countries
    for row in bindings["rows"] if c.name in str(row[2]) or c.name in str(row[1])]
len(bindings["countries"])
```

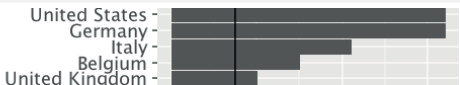
root	194
C2	'hexadecimal'

Markdown

Finally, we can use ggplot2, a data visualization package written in R, to visualize the number of contributors per country as a bar chart. For this, our notebook implementation supports a `%ggplot2` magic which provides convenient access to the visualization package. We instantiate a new `data.frame` object from the list of `countries`. Then, we aggregate this data before passing it into the `ggplot` function. Lastly, we can further configure the plot to display a sorted bar chart as well as a mean line.

R

```
%ggplot2
values <- data.frame(contributors = bindings["countries"])
data <- aggregate(x = values, by = list(countries = values$contributors), FUN = length)
ggplot(data, aes(x = reorder(countries, +contributors), contributors)) +
  geom_bar(stat = "identity") + xlab("") + ylab("") + coord_flip() +
  geom_hline(aes(yintercept = mean(contributors)))
```



bindings	ForeignObject[memberSize=3]
countries	ForeignObject[arraySize=194, memberSize=
pop	ForeignObject[memberSize=0]
extend	ForeignObject[memberSize=0]
count	ForeignObject[memberSize=0]
clear	ForeignObject[memberSize=0]
insert	ForeignObject[memberSize=0]
index	ForeignObject[memberSize=0]
copy	ForeignObject[memberSize=0]
sort	ForeignObject[memberSize=0]
reverse	ForeignObject[memberSize=0]
remove	ForeignObject[memberSize=0]
append	ForeignObject[memberSize=0]
1	'Albania'
2	'Argentina'
3	'Argentina'
4	'Argentina'
5	'Australia'
6	'Austria'
7	'Austria'
8	'Belgium'
9	'Belgium'
10	'Belgium'
11	'Belgium'
12	'Belgium'
13	'Belgium'
14	'Belgium'
15	'Belgium'
16	'Belgium'
17	'Belgium'
18	'Belgium'
19	'Belgium'
20	'Belgium'
21	'Belgium'
22	'Belgium'
23	'Canada'
24	'Canada'
25	'Canada'
26	'Canada'
27	'Canada'
28	'Switzerland'
29	'Switzerland'
30	'Switzerland'
31	'Switzerland'
32	'Switzerland'
33	'Switzerland'
34	'Switzerland'
35	'Switzerland'
36	'Switzerland'
37	'Switzerland'
38	'Colombia'
39	'Colombia'
40	'Germany'
41	'Germany'
42	'Germany'
43	'Germany'
44	'Germany'
45	'Germany'
46	'Germany'
47	'Germany'
48	'Germany'

IPolyglot

A GraalVM-powered Jupyter Kernel

https://github.com/hpi-swa/ipolyglot

jupyter Polyglot Jupyter Notebook Last Checkpoint: 07/10/2019 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Polyglot (GraalJS)

Polyglot Jupyter Notebook: Conference Contributions per Country

Scraping the conference data from a web table using `ruby's nokogiri` library

In [3]:
```
%polyglot ruby

require "nokogiri";
require "open-uri"
url = "https://2019.splashcon.org/people-index" # "https://2019.programming-conference.org/people-index"
doc = Nokogiri::HTML(open(url))
$rows = doc.css("#results-table .row").map { |row| row.css("div").map(&:content) }
$rows.length
```

Out[3]: 611

Compiling a list of all contributing countries using `python's pycountry` library

In [6]:
```
%polyglot python

import pycountry
countries = [c.name for c in pycountry.countries for row in rows if c.name in str(row[2]) or c.name in str(row[1])]
len(countries)
```

Out[6]: 386

Plotting the conference contributions per country using `R's ggplot2` library

In [9]:
```
%polyglot R

library(ggplot2)

values <- data.frame(contributors = countries)
data <- aggregate(x = values, by = list(countries = values$contributors), FUN = length)

svg()
plt <- ggplot(data, aes(x = reorder(countries, +contributors), contributors)) +
  geom_bar(stat = "identity") + xlab("") + ylab("") + coord_flip() +
```

Polyglot Inspector

```
▸rows
▾countries
  0: Argentina
  1: Australia
  2: Australia
  3: Australia
  4: Australia
  5: Australia
  6: Australia
  7: Austria
  8: Austria
  9: Austria
  10: Austria
  11: Austria
  12: Austria
  13: Austria
  14: Belgium
  15: Belgium
  16: Belgium
  17: Belgium
  18: Belgium
  19: Belgium
  20: Belgium
  21: Belgium
  22: Belgium
  23: Brazil
  24: Brazil
  25: Brazil
  26: Canada
  27: Canada
  28: Canada
  29: Canada
```

jupyter Polyglot Notebook Last Checkpoint: 13 minutes ago (unsaved changes)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Polyglot (GraalJS)

Run Code

Polyglot Jupyter Notebook: Conference Contributions per Country

Scraping the conference data from a web table using Ruby's `nokogiri` library

In [ ]:
```
%polyglot ruby

require "nokogiri"; require "open-uri"
url = "https://2019.splashcon.org/people-index"
doc = Nokogiri::HTML(open(url))
$rows = doc.css("#results-table .row").map { |row| row.css("div").map(&:content) }
$rows.length
```

Compiling a list of all contributing countries using Python's `pycountry` library

In [ ]:
```
%polyglot python

import pycountry
countries = [c.name for c in pycountry.countries for row in rows if c.name in str(row[2]) or c.name in str(row[1])]
len(countries)
```

Plotting the conference contributions per country using R's `ggplot2` library

In [ ]:
```
%polyglot R

library(ggplot2)

values <- data.frame(contributors = countries)
data <- aggregate(x = values, by = list(countries = values$contributors), FUN = length)

svg()
plt <- ggplot(data, aes(x = reorder(countries, +contributors), contributors)) +
  geom_bar(stat = "identity") + xlab("") + ylab("") + coord_flip() +
  geom_hline(aes(yintercept = mean(contributors)))
print(plt)
svg_data <- svg.off()
nchar(svg_data)
```

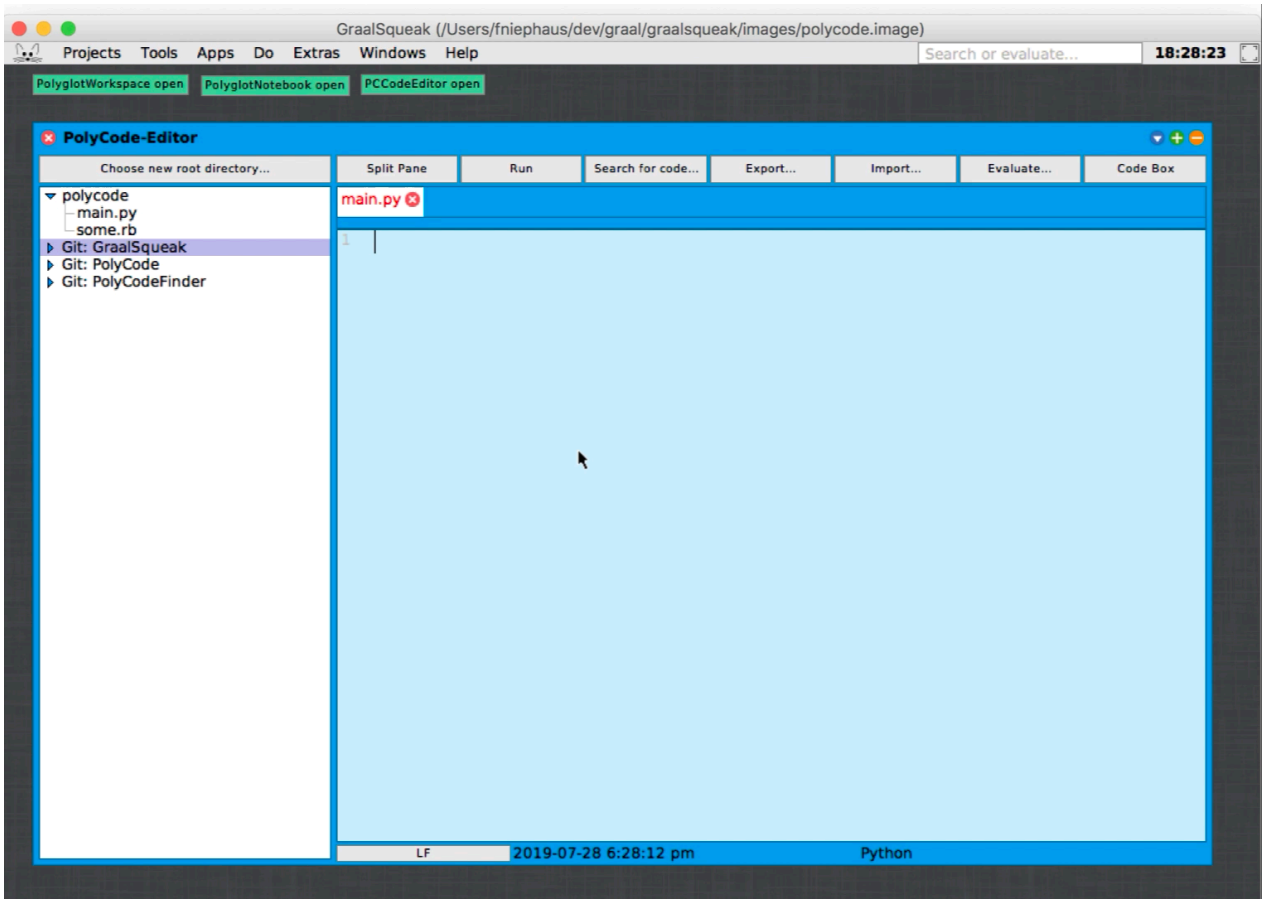
Render generated SVG

In [ ]:
```
$$.svg(svg_data)
```

Polyglot Inspector

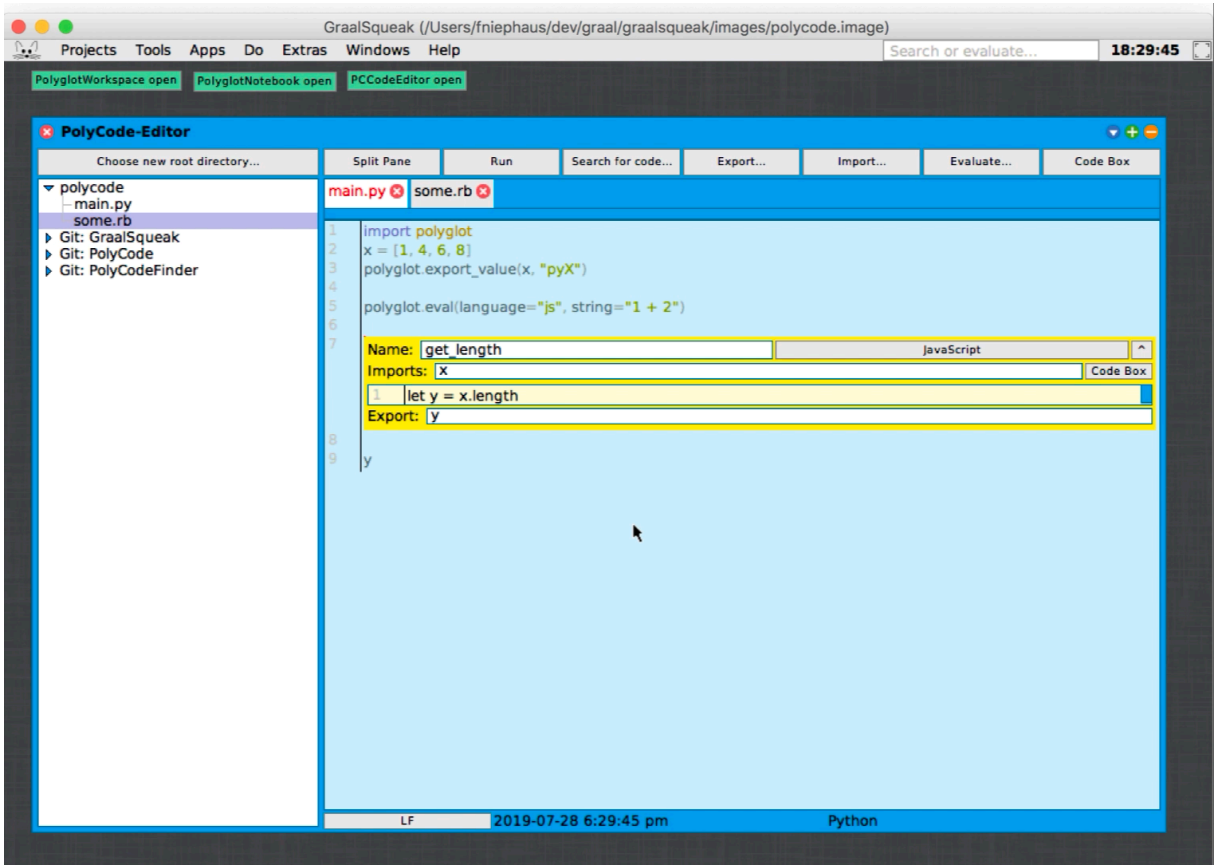
Polyglot Code Editor

- Goal: Help developers to create polyglot applications



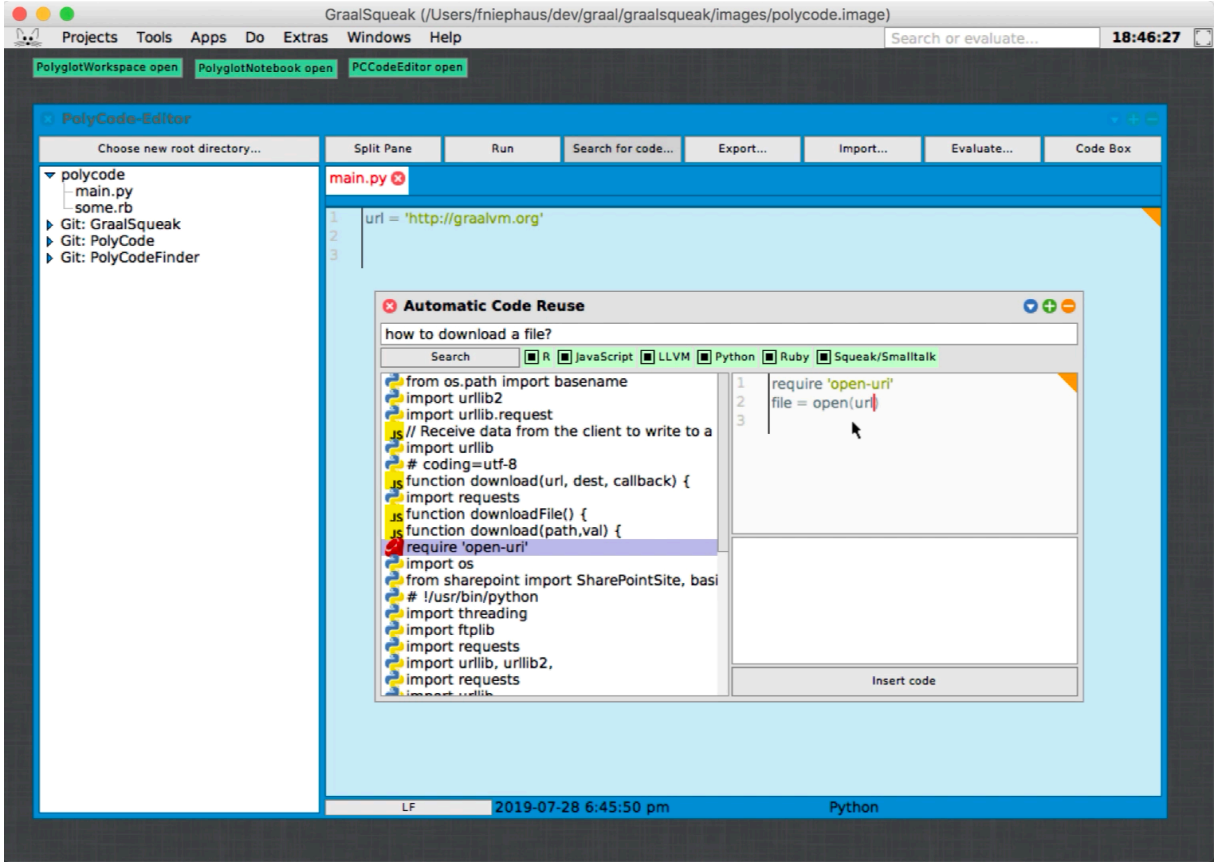
Polyglot Code Editor

- Goal: Help developers to write polyglot applications



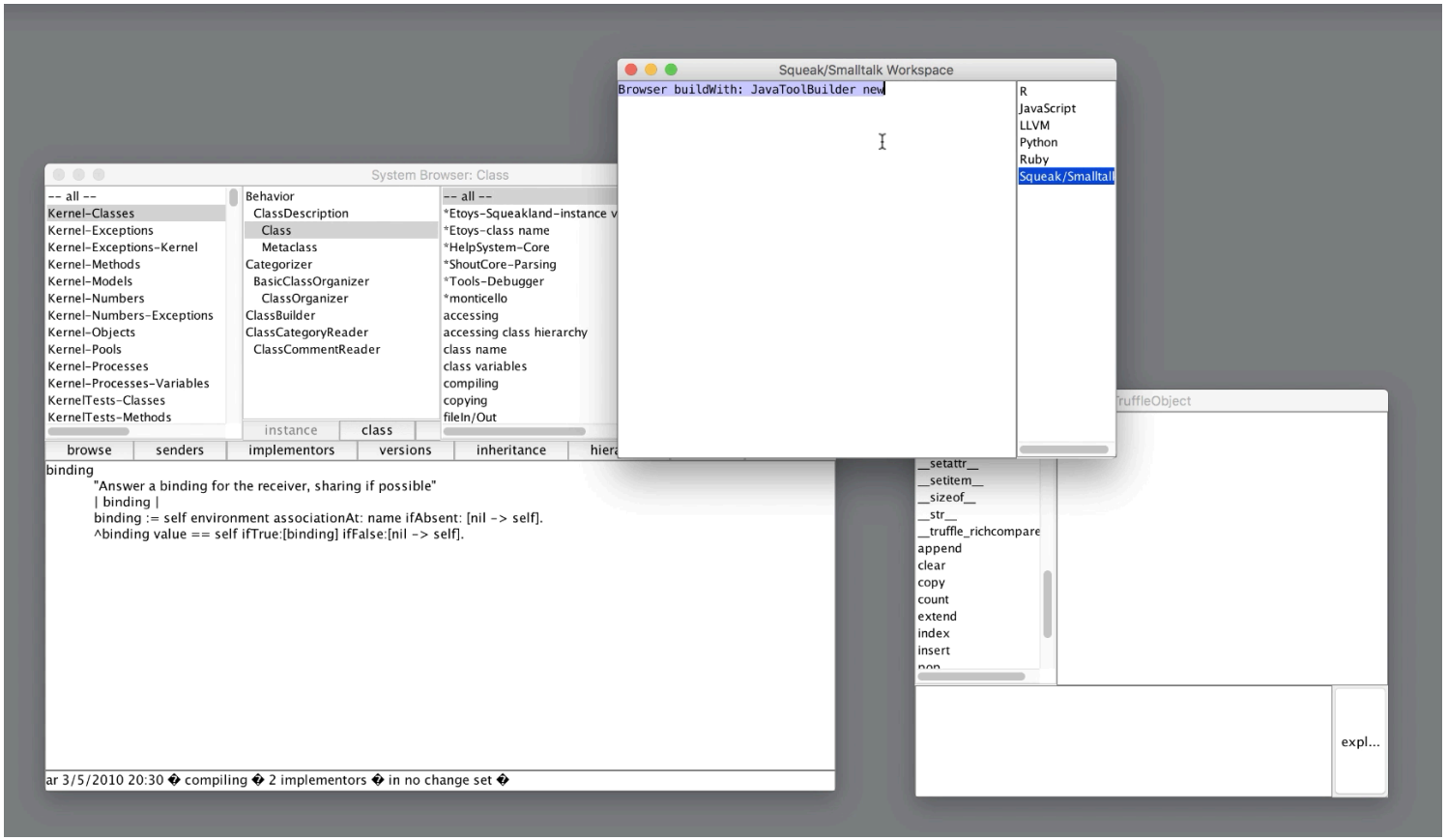
Polyglot Code Finder

- Goal: Help developers to find the right code for reuse



Polyglot User Interface

- Idea: Replace UI backend of a programming system

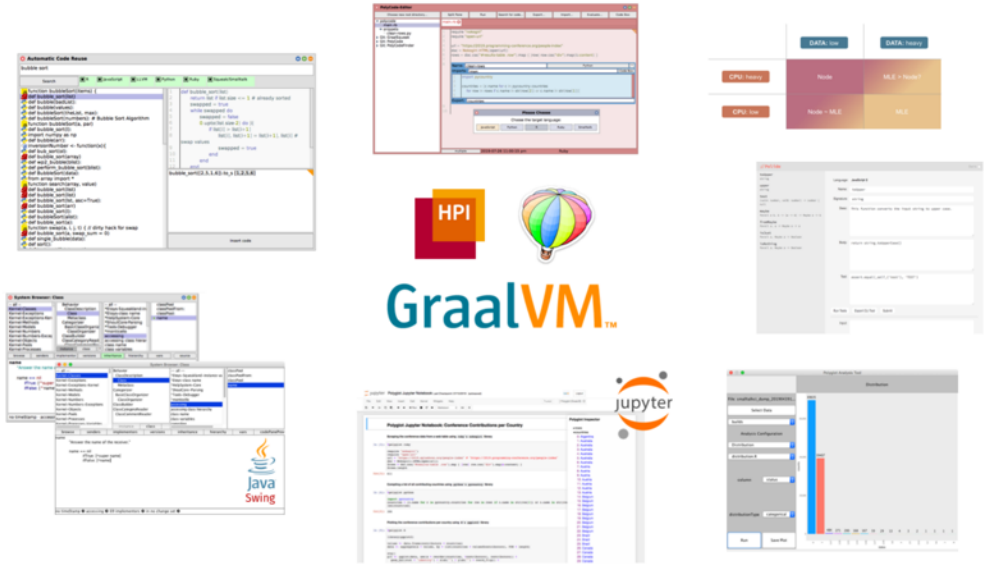


Sign in Get started

GraalVM ALL ARTICLES | LEARN MORE

HPI Polyglot Programming Seminar

Fabio Niephaus Follow
Oct 11 · 8 min read



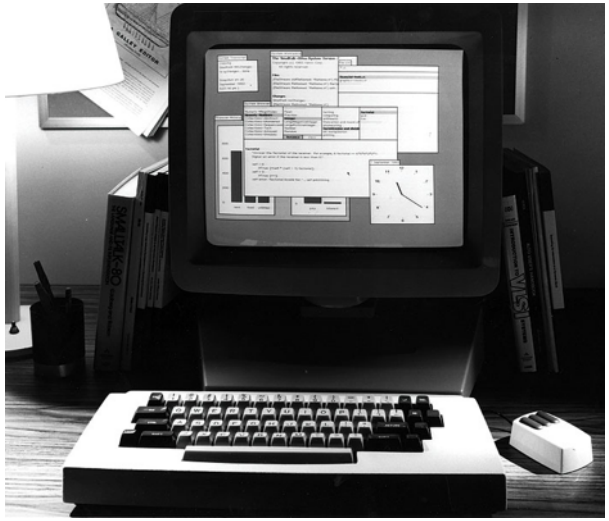
In the summer term 2019, we ran a seminar on Polyglot Programming as part of our graduate program at the Hasso Plattner Institute in Potsdam.

GraalSqueak

IMPLEMENTATION DETAILS

GraalSqueak Implementation Details

- Full support for Squeak/Smalltalk 5.2+ images
- Comprehensive object model
- 500+ primitives implemented
- Java AWT used for rendering
- BitBlt and Balloon plugins ported to Java (~9K SLOC)



- Interop API mapped to Smalltalk metaobject protocol
- 4464 of 5078 (~87.9%) SUnit tests passing

Supported Smalltalk Language Features

- Sender modifications and Spaghetti Stacks
  - Exception handling
  - *edit-and-continue* debugging
- *all instances* mechanism
- *become* mechanism
- …

Not supported (yet)

- Interrupt handler due to problems with PE
- Image snapshots (persisting polyglot object heaps)

→ see MPLR'19 paper

Integration into GraalVM Ecosystem

- Ships as component for GraalVM Updater (**gu** tool)
- Interop API directly exposed

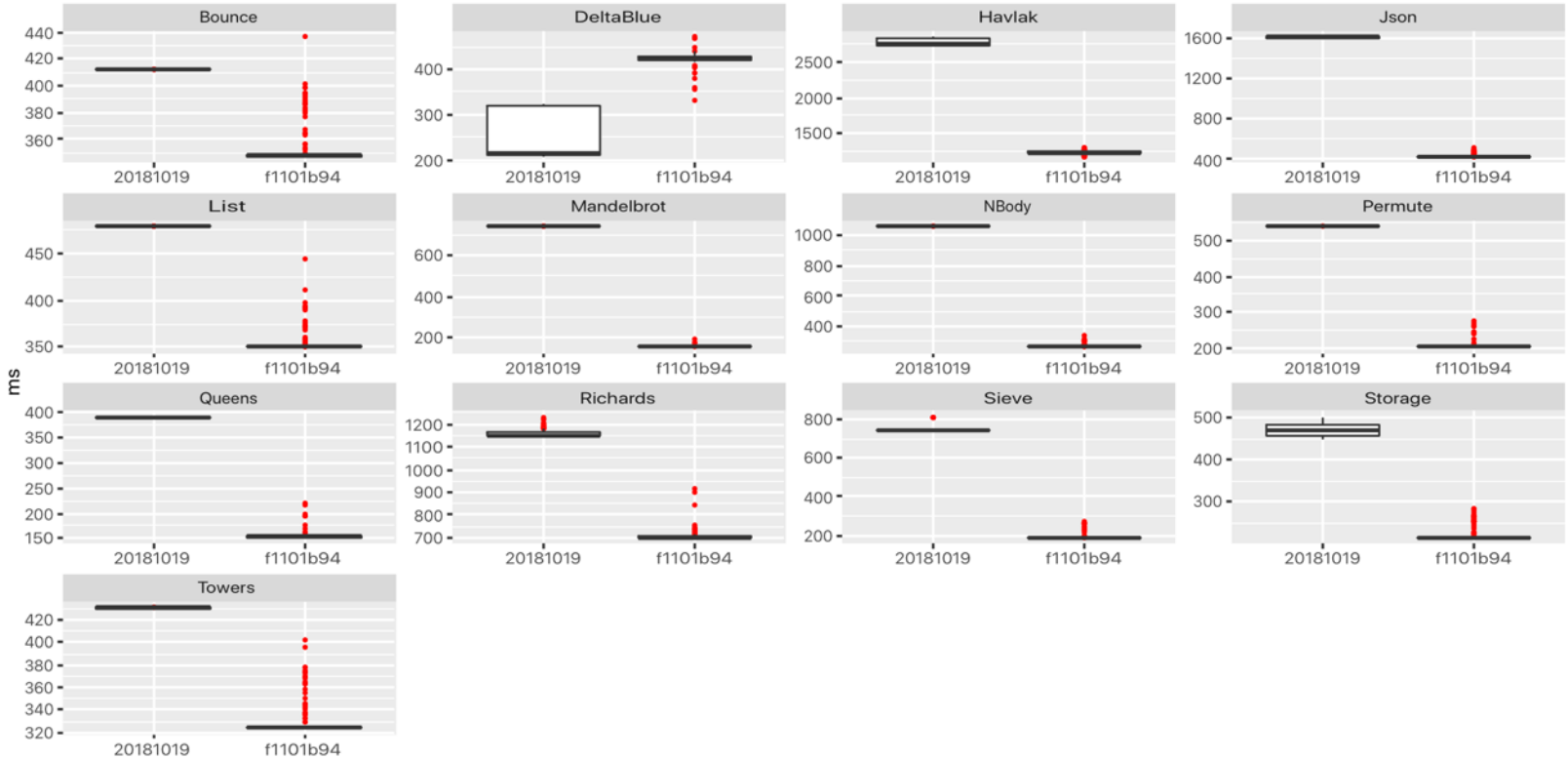
  (great for tools and experiments, integration on language level)
- AOT-compiled GraalSqueak (JFrame replaced by SDL2)
- Tested on OpenJDK8 and GraalVM by Travis CI

  (incl. coverage testing with JaCoCo)
- NFI-based FFIPlugin in the works

GraalSqueak vs. OpenSmalltalkVM

Summary: 20181019 (OpenSmalltalkVM) vs. f1101b94 (dev)

Summary | Bounce | CD | DeltaBlue | Havlak | Json | List | Mandelbrot | NBody | Permute | Queens | Richards | Sieve | Storage | Towers



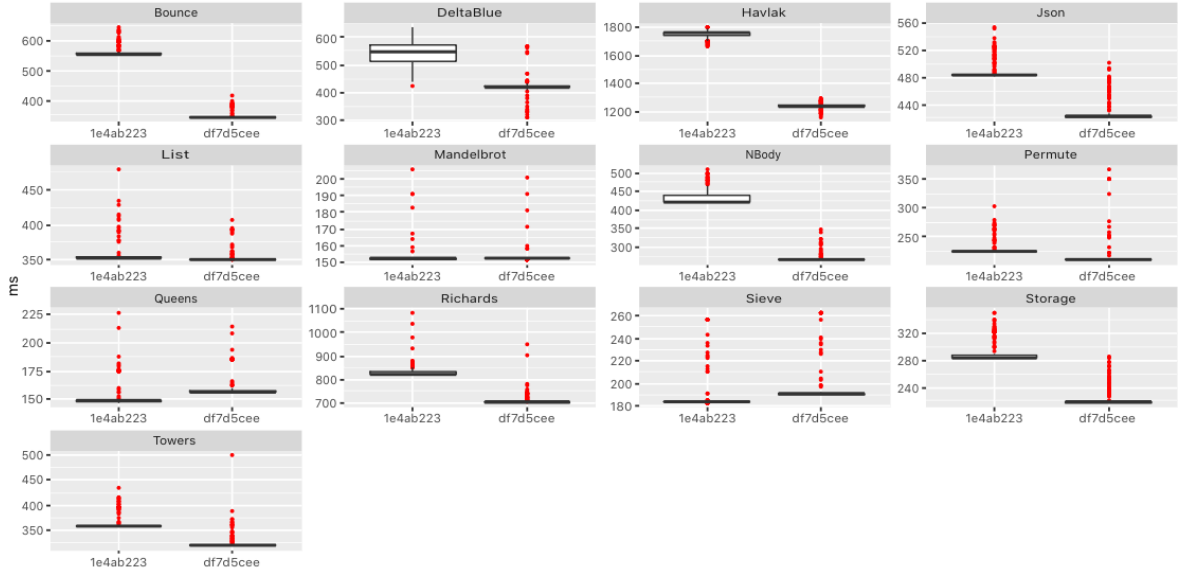
Commit	Commit Date	Commit Branch	Sum Min	Sum Geomean	Sum Median	Sum Mean	Sum Max	Total (ms)	Total (min)
20181019	2019-09-05.21:00:00	OpenSmalltalkVM	10908.596	11053.489	10980.127	11059.339	11413.151	2211867.757	36.864
f1101b94	2019-11-18.11:00:39	dev	4826	5036.263	4991	5039.985	6070	1007997	16.8
			6082.596	6017.225	5989.127	6019.354	5343.151	1203870.757	20.065

rebench-stats 1/2

172.16.64.123

Summary: 1e4ab223 (dev) vs. df7d5cee (object-layout)

Summary | Bounce | CD | DeltaBlue | Havlak | Json | List | Mandelbrot | NBody | Permute | Queens | Richards | Sieve | Storage | Towers

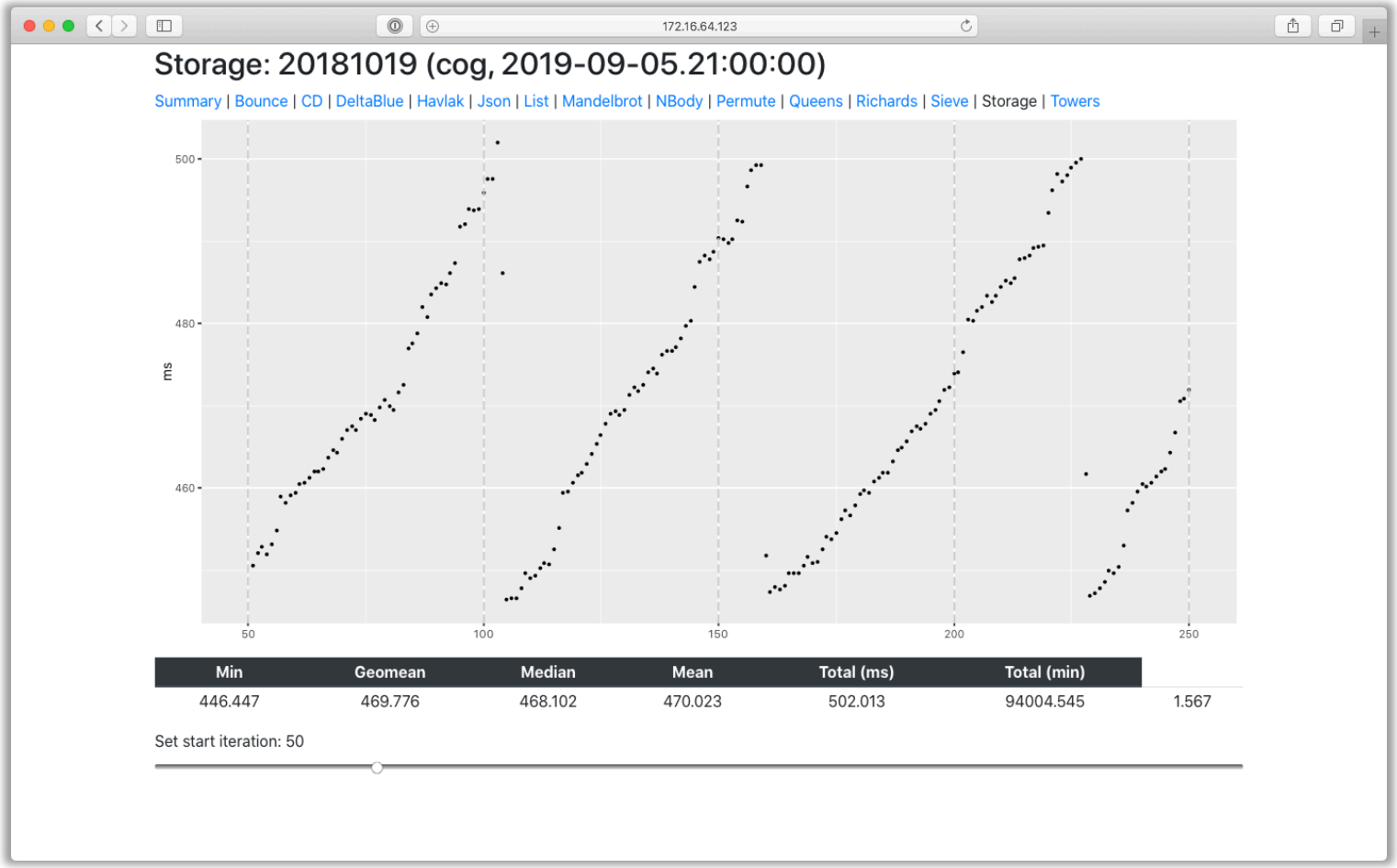


Commit	Commit Date	Commit Branch	Sum Min	Sum Geomean	Sum Median	Sum Mean	Sum Max	Total (ms)	Total (min)
df7d5cee	2019-10-31.16:00:11	object-layout	4789	5041.573	4993	5047.085	6318	1009417	16.824
1e4ab223	2019-10-07.11:30:37	dev	6067	6347.87	6296	6353.19	7498	1270638	21.177
			-1278	-1306.297	-1303	-1306.105	-1180	-261221	-4.354

Set start iteration: 50

https://github.com/fniephaus/rebench-stats

rebench-stats 2/2



https://github.com/fniephaus/rebench-stats

GraalSqueak
UI Performance Demo

UI Performance Benchmark

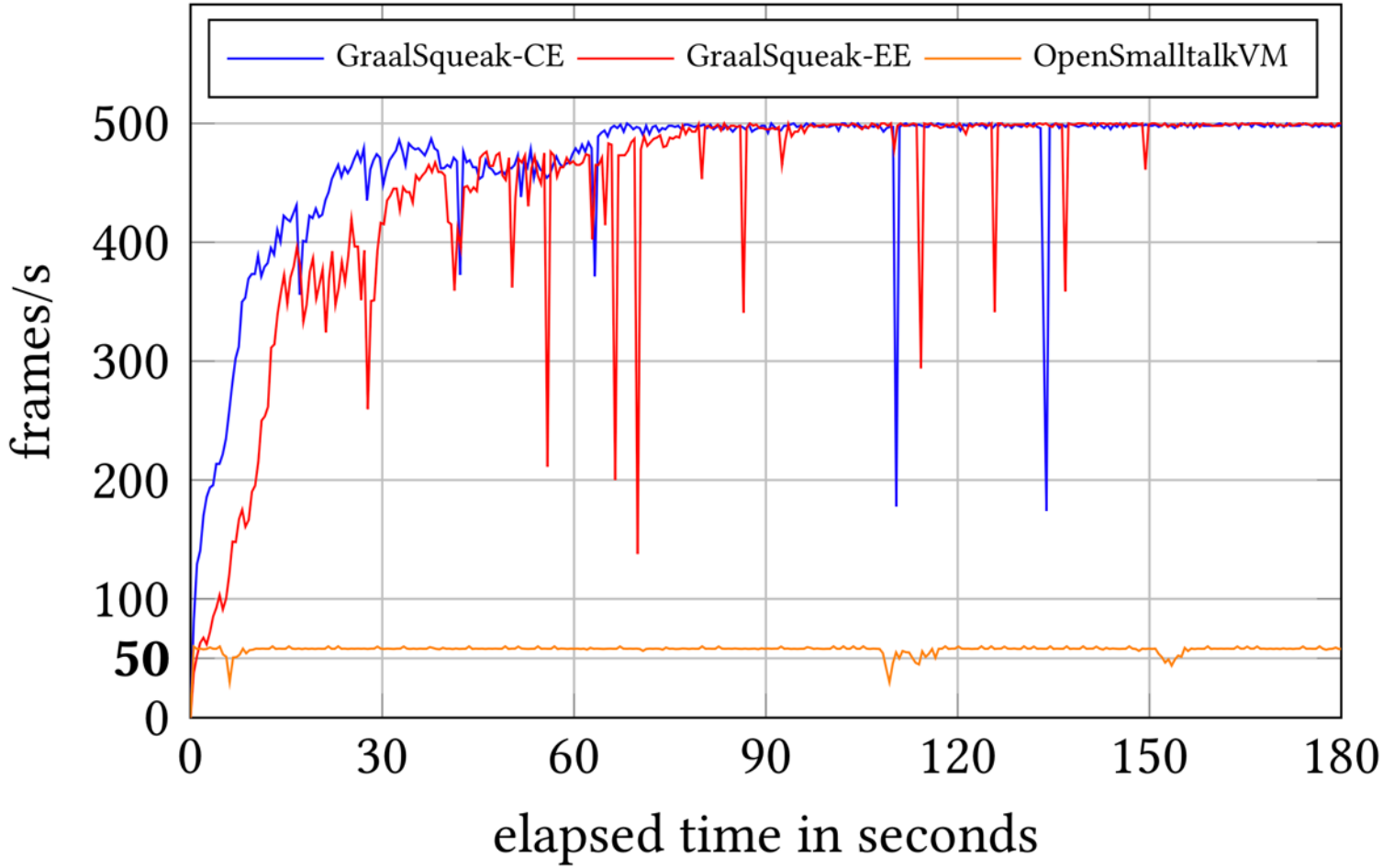


Fig. 5c, GraalSqueak paper, MPLR'19, https://doi.org/10.1145/3357390.3361024

POLYGLOT PROGRAMMING

Intuitive Guidelines for Polyglot Programming

- Modules written in different languages
  - (vs. functions, objects, expressions, …)
  - Reasonable granularity for separating concerns
    - Fewer cross-boundary interactions for developers (e.g. black-box frameworks)
    - API of module much more important than language
- Primitive values straightforward to share
- Easy to replace Foreign Function Interface (FFI) and Inter-Process Communication (IPC) calls with Polyglot Programming

Advantages of Polyglot Programming

- It's fast! (JIT optimizes across language boundaries)
- Allows sharing of arbitrary data and objects
- Retains data references and object graphs
- Does not break paradigms of languages
- Does not break tooling (e.g. debugger)
- Can provide a more consistent programming experience (e.g. LSP integration)
- …

Advantages of Polyglot Programming

- It's fast! (JIT optimizes across language boundaries)
- Allows sharing of arbitrary data and objects
- Retains data references and object graphs
- Does not break paradigms of languages
- Does not break tooling (e.g. debugger)
- Can provide a more consistent programming experience (e.g. LSP integration)
- …

So why is this still an interesting research topic and an engineering challenge?

PITFALLS OF POLYGLOT PROGRAMMING

$$\texttt{nil}_{\texttt{Smalltalk}} + \texttt{None}_{\texttt{Python}}$$

$$\texttt{nil}_{\texttt{Ruby}} + \texttt{None}_{\texttt{Python}}$$

$\texttt{nil}_{\texttt{Ruby}}\ +_{\texttt{Javascript}}\ \texttt{None}_{\texttt{Python}}$

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
  JavaScript version 19.3.0
  Python version 3.7.4
  Ruby version 2.6.2
js> result = (Polyglot.eval(‘ruby', 'None') +
  + Polyglot.eval(‘python', 'nil'))
0
js> typeof(result)
number
```

→ Type Coercion in Languages

Equality, Identity, Arithmetic, … in Languages

Equality in JavaScript

	true	false	1	0	-1	"true"	"false"	"1"	"0"	"-1"	""	null	undefined	Infinity	-Infinity	[]	{}	[[]]	[0]	[1]	NaN
true	=	≠	≅	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≅	≠
false	≠	=	≠	≅	≠	≠	≠	≠	≅	≠	≅	≠	≠	≠	≠	≅	≠	≅	≅	≠	≠
1	≅	≠	=	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≅	≠
0	≠	≅	≠	=	≠	≠	≠	≠	≅	≠	≅	≠	≠	≠	≠	≅	≠	≅	≅	≠	≠
-1	≠	≠	≠	≠	=	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
"true"	≠	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
"false"	≠	≠	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
"1"	≅	≠	≅	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≅	≠
"0"	≠	≅	≠	≅	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠	≠	≠	≠	≅	≠	≠
"-1"	≠	≠	≠	≠	≅	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
""	≠	≅	≠	≅	≠	≠	≠	≠	≠	≠	=	≠	≠	≠	≠	≅	≠	≅	≠	≠	≠
null	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	=	≅	≠	≠	≠	≠	≠	≠	≠	≠
undefined	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≅	=	≠	≠	≠	≠	≠	≠	≠	≠
Infinity	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠	≠
-Infinity	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	=	≠	≠	≠	≠	≠	≠
[]	≠	≅	≠	≅	≠	≠	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
{}	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
[[]]	≠	≅	≠	≅	≠	≠	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
[0]	≠	≅	≠	≅	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
[1]	≅	≠	≅	≠	≠	≠	≠	≅	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠
NaN	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠	≠

≠ Not equal

≅ Loose equality
Often gives "false" positives like "1" is true; [] is "0"

= Strict equality
Mostly evaluates as one would expect.

https://dorey.github.io/JavaScript-Equality-Table/unified/

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
  JavaScript version 19.3.0
  Python version 3.7.4
  Ruby version 2.6.2
js> let string1 = Polyglot.eval('python', '"Test"')
js> typeof(string1)
string
js> string1.length
4
js> string1.indexOf('s')
2
js> string1.__len__()
TypeError: (intermediate value).__len__ is not a function
```

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
  JavaScript version 19.3.0
  Python version 3.7.4
  Ruby version 2.6.2
js> let string2 = Polyglot.eval('ruby', '"Test"')
js> typeof(string2)
string
js> string2.length
null
js> string2.indexOf('s')
undefined method `indexOf' for "Test":String
Did you mean?  index (NoMethodError)
js> string2.index('s')
2
```

→ Mutability and Immutability

Semantical Differences

- Hard to reason about different language semantics at the same time
- Especially hard for novice developers
- **But:** Tools may be able to help

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
Python version 3.7.4
  Ruby version 2.6.2
python> import polyglot
python> rb_list = polyglot.eval(language='ruby', string='[]')
python> set(dir([])).intersection(set(dir(rb_list)))
```

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
Python version 3.7.4
  Ruby version 2.6.2
python> import polyglot
python> rb_list = polyglot.eval(language='ruby', string='[]')
python> set(dir([])).intersection(set(dir(rb_list)))
{'insert', 'pop', 'count', 'clear', 'index', 'reverse',
 'extend', 'append', 'sort'}
```

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
Python version 3.7.4
  Ruby version 2.6.2
python> import polyglot
python> rb_list = polyglot.eval(language='ruby', string='[]')
python> set(dir([])).intersection(set(dir(rb_list)))
{'insert', 'pop', 'count', 'clear', 'index', 'reverse',
 'extend', 'append', 'sort'}
python> set(dir([])).difference(set(dir(rb_list)))
```

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
Python version 3.7.4
  Ruby version 2.6.2
python> import polyglot
python> rb_list = polyglot.eval(language='ruby', string='[]')
python> set(dir([])).intersection(set(dir(rb_list)))
{'insert', 'pop', 'count', 'clear', 'index', 'reverse',
 'extend', 'append', 'sort'}
python> set(dir([])).difference(set(dir(rb_list)))
{'__name__', '__qualname__', '__doc__', '__new__', '__lt__',
 '__mul__', '__hash__', 'remove', '__delitem__', '__imul__',
 '__ge__', '__bool__', '__iter__', '__iadd__', '__eq__',
 '__gt__', '__rmul__', '__contains__', '__setitem__',
 '__le__', '__add__', '__len__', '__repr__', '__getitem__',
 '__init__', '__ne__', 'copy', '__str__', '__getattribute__',
 '__dict__', '__delattr__', '__class__', '__init_subclass__',
 '__setattr__', '__getattr__', '__truffle_richcompare__',
 '__format__', '__reduce__', '__reduce_ex__', '__sizeof__',
 '__dir__'}
```

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
Python version 3.7.4
  Ruby version 2.6.2
python> import polyglot
python> rb_list = polyglot.eval(language='ruby', string='[]')
python> set(dir([])).intersection(set(dir(rb_list)))
{'insert', 'pop', 'count', 'clear', 'index', 'reverse',
 'extend', 'append', 'sort'}
python> set(dir(rb_list)).difference(set(dir([])))
```

```
$ polyglot --shell --jvm
GraalVM MultiLanguage Shell 19.3.0
Copyright (c) 2013-2019, Oracle and/or its affiliates
Python version 3.7.4
  Ruby version 2.6.2
python> import polyglot
python> rb_list = polyglot.eval(language='ruby', string='[]')
python> set(dir([])).intersection(set(dir(rb_list)))
{'insert', 'pop', 'count', 'clear', 'index', 'reverse',
 'extend', 'append', 'sort'}
python> set(dir(rb_list)).difference(set(dir([])))
{'map', 'group_by', 'combination', 'instance_variable_set',
'unshift', 'remove_instance_variable', 'send', 'rotate!',
'instance_eval', 'singleton_methods', 'collect_concat', 'max',
'each_with_object', 'delete_at', 'swap', 'entries', 'sample',
'min_by', 'drop', 'take_while', 'difference', 'collect!',
'kind_of?', 'then', 'select', 'length', 'none?', 'taint',
'select!', 'hash', 'values_at', 'instance_exec', 'detect',
'|', 'flatten', 'to_h', 'fill', 'map!', 'eql?', 'each_cons',
'slice_when', 'each_index', 'rassoc', 'private_methods', 'take',
'reduce', 'each', 'slice!', 'inspect', 'join', … +126 others}
```

→ Mismatch of Interfaces

Another example:

Cannot just plot ***`pandas.DataFrame`*** *with R's* ***`ggplot2`***
(at the moment).

More Challenges

Tools for Polyglot Programming

Multi-language interfaces, inheritance, …

Mixing syntaxes and **semantics**

Multi-language testing

Multi-language performance (speed vs. readability vs. …)

Dependency management

Code documentation for polyglot app

Multi-language object heaps

Multi-paradigm programming

Multi-language debugging

???

Maintainability of polyglot code

Conclusions

- **GraalVM:** enables comprehensive interoperability between programming languages
- **GraalSqueak:** Smalltalk-based IDE platform for polyglot programming on top of GraalVM
- **Polyglot Programming:**
  - always use the right thing for the job
  - powerful technique to increase software reuse
  - interesting research topic (many open questions, different kinds of pitfalls, …)

hpi-swa / graalsqueak

Unwatch 20 | Unstar 52 | Fork 4

Code | Issues 18 | Pull requests 2 | Actions | Security | Insights | Settings

Branch: master | graalsqueak / README.md | Find file | Copy path

183 lines (155 sloc) | 10.3 KB | Raw | Blame | History

GraalSqueak

release v1.0.0-rc5 | build passing | code quality A | coverage 68% | license MIT

A Squeak/Smalltalk implementation for the GraalVM.

Getting Started

1. Find the latest GraalSqueak release and identify the supported version of GraalVM.
2. Download the corresponding GraalVM for your platform (if you are on Windows, please follow these instructions instead).
3. Install the pre-compiled GraalSqueak component attached to the release with the GraalVM Updater:

```
$GRAALVM_HOME/bin/gu install \
  -u https://github.com/hpi-swa/graalsqueak/releases/download/X.Y.Z/graalsqueak-component-X.Y.Z.jar
```

4. You should now be able to open the GraalSqueak image attached to the release:

```
$GRAALVM_HOME/bin/graalsqueak path/to/a/GraalSqueak-X.Y.Z.image
```

We need your help!

Please tell us about
your experience
building polyglot application on GraalVM!

More data points are
needed and greatly appreciated!

Polyglot Programming with GraalSqueak and GraalVM

Fabio Niephaus and Robert Hirschfeld

Oracle PIs: Mario Wolczko and Tim Felgentreff

Software Architecture Group
Hasso Plattner Institute
University of Potsdam, Germany

https://www.hpi.uni-potsdam.de/swa/
https://github.com/hpi-swa/graalsqueak/

November 25, 2019