On the Quest Towards
Fastest (Java) Virtual Machine
on the Planet!

@JaroslavTulach
Oracle Labs
Jaroslav Tulach

1. NetBeans Founder
2. NetBeans Initial Architect
3. Practical API Design book
4. Oracle Labs: Graal/Truffle
Program Agenda

1. The Vision
2. Graal VM
3. Truffle Languages
4. References & Discussion
Oracle Labs Vision

- Vision without speculation would be today's reality
- Radical innovation
- New approach to existing technologies
- Research + Engineering
- Hardware & Software
  - Engineered to work together
  - Java/bytecode/IR/assembly/processors/memory
“Create an extensible, modular, dynamic, and aggressive compiler using object-oriented and reflective Java programming, a graph-based and visualizable intermediate representation, and Java snippets.”

Thomas Würthinger
Graal Virtual Machine

- Modern alternative to HotSpot C2
  - Maintainable code base
  - Toolable, approachable
  - Ready for today's code
  - JEP 243: Java Compiler Interface

- Partial evaluation
- Aggressive speculations
- Smooth de-optimizations
## Truffle: “Write Your Own Language”

<table>
<thead>
<tr>
<th>Current situation</th>
<th>How it should be</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prototype a new language</td>
<td>Prototype a new language in Java</td>
</tr>
<tr>
<td>Parser and language work to build syntax tree (AST), AST Interpreter</td>
<td>Parser and language work to build syntax tree (AST)</td>
</tr>
<tr>
<td>Write a “real” VM</td>
<td>Execute using AST interpreter</td>
</tr>
<tr>
<td>In C/C++, still using AST interpreter, spend a lot of time implementing runtime system, GC, ...</td>
<td>Integrate with VM-building framework</td>
</tr>
<tr>
<td>People start using it</td>
<td>Integrate with Modular VM</td>
</tr>
<tr>
<td>People complain about performance</td>
<td>Add small language-specific parts</td>
</tr>
<tr>
<td>Define a bytecode format and write bytecode interpreter</td>
<td>People start using it</td>
</tr>
<tr>
<td>Performance is still bad</td>
<td>And it is already fast</td>
</tr>
<tr>
<td>Write a JIT compiler</td>
<td></td>
</tr>
<tr>
<td>Improve the garbage collector</td>
<td></td>
</tr>
</tbody>
</table>
A Spectrum of Programming Languages

Do you care about your code or your data?

Static typing
- Java, C#
- Visual Basic
- Python, Ruby
- JavaScript
- R

Faster execution
- Targeting larger projects
- More flexible
- Easier to use
  - Targeting exploratory programming, rapid prototyping, domain-specific programming, customization of existing applications

There is no single language that fulfills all needs
One VM for all languages means interoperability and being able to choose the best language for the task!

The goal:

[Bar chart showing the mean values for different programming languages]
You can execute any language on the JVM / CLR - as long as it looks like Java / C#.
JS

Ruby

C

Truffle

Graal VM

OS

VM Expert

OS Expert

Java

Java / C++

Unmanaged Language (typically C or C++)
\[ a + b \]

- int
- String
- Object
Node Rewriting for Profiling Feedback

Node Transitions

Uninitialized Integer
String Double Generic

AST Interpreter Uninitialized Nodes

Compiled Code

Compilation using Partial Evaluation

AST Interpreter Rewritten Nodes
Safe Harbor Statement

The preceding and following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Demo

The Speed(s) of Ruby
Multi, Multi, Multi, Multi, Multi VM

- Multi-language
- Multi-threaded
- Multi-tenant
- Multi-node
- Multi-tooling
Multi Language

• Single launcher for all
  
  tenant = PolyglotEngine.buildNew().config(...).build();
  
  fourtyTwo = tenant.eval("text/x-javascript", "32 + 10");
  
  twoAndFourty = tenant.eval("application/x-ruby", "10 + 32");

• Languages can export and import symbols
  
  window = tenant.findGlobalSymbol("window");
  
  // and JavaScript provides it by overriding
  @Override protected Object findGlobalSymbol(String name) {
      return "window".equals(name) ? jsWindowGlobal : null;
  }

Multi Tenant

• Create as many tenants as needed
  
  tenant1 = PolyglotEngine.buildNew().config(...).build();
  tenant2 = PolyglotEngine.buildNew().config(...).build();
  fourtyTwo = tenant1.eval("text/x-javascript", “x = this.x ? x + 1 : 42”);
  // sees previous value in x:
  fourtyThree = tenant1.eval("text/x-javascript", “x = this.x ? x + 1 : 42”);
  // no x defined, as this is different tenant:
  twoAndFourty = tenant2.eval("text/x-javascript", “x = this.x ? x + 1 : 42”);

• Isolation enforced by TCK
Multi Threaded

• **PolyglotEngine** is inherently single-threaded API
  - Access from a single (creator) thread enforced

• Easy to dispatch execution to different thread
  
  tenantAsync = PolyglotEngine.buildNew()
  .executor(Executors.newSingleThreadedExecutor())
  .build();

• Tenants can exchange messages
  - Orchestrate multiple tenants running in parallel from a single thread
Multi Node (under consideration)

- **PolyglotEngine** as a facade for remote object protocol
  
  ```javascript
  tenantAway = PolyglotEngine.buildNew()
  .nodeIP("192.168.34.65") // the execution will happen on different node
  .build();
  result = tenantAway.eval("text/javascript", "'Remote and slow ' + 42");
  ```

- Keeping the same API facade: **PolyglotEngine**
  
  - Get multi language, multi threaded, multi tenant
  - Multi tooling and multi node virtual machine!
Multi Tooling

• JPDA debugging always on
  - Attach Java IDE and all Truffle language details will be shown

• Turn on your debugger protocol
  ```java
tenantWithDebugger = PolyglotEngine.buildNew()
  .onEvent(new ExecutionEventHandler() {...
  .onEvent(new SuspendedEventHandler() {...
  .build();
```

• Other instrumentation based tools available as well
Demo

Polyglot Debugging
The Fastest (J)VM on the Planet

• Try it now!
  – Download jvm:  http://www.oracle.com/technetwork/oracle-labs/program-languages/
  – Open source:
    ● http://openjdk.java.net/projects/graal/
    ● https://github.com/graalvm
  – JRuby, Python, JavaScript, R, C

• Speed is Great!
  – In need of completeness

• We are hiring!
Hardware and Software
Engineered to Work Together