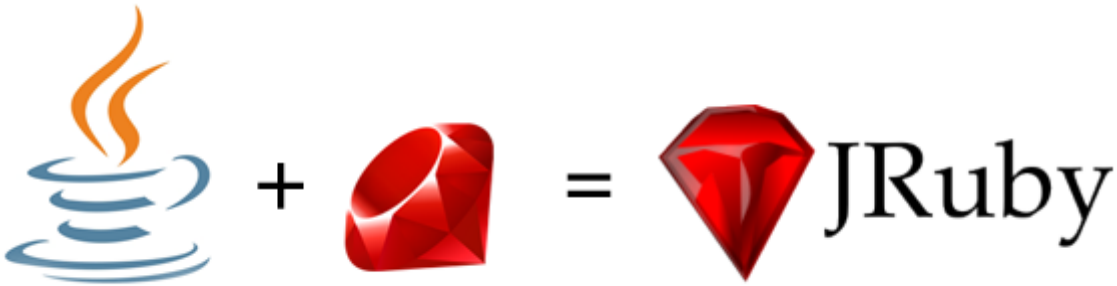
Implementacje języków na JVM
z JRuby w tle



by Arkadiusz Hiler

O JVM słów kilka

- Java Platform
- Java Virtual Machine
- Bytecode
- GC, JIT, Adaptive Optimization
- Kompletne (kompleksowe?) rozwiązanie
- Czemu ograniczać się do Javy?

Co już mamy?

- Znane i lubiane języki:
  - JRuby
  - Jython
  - Jacl (TCL)
- Nowości:
  - Scala (!)
  - Groovy
  - Clojure
- A ponadto:
  - Rhino (JS)
  - C na JVM w kilku wydaniach.
  - Quercus (PHP)

Jak to wygląda?
Java w JRuby

```
include Java
import javax.swing.JFrame

frame = JFrame.new("Hello Swing")
button = javax.swing.JButton.new("Klick Me!")

button.add_action_listener do |evt|
  javax.swing.JOptionPane.show_message_dialog(nil, <<EOS)
<html>Hello from <b><u>JRuby</u></b>.<br>
Button '#{evt.getActionCommand()}' clicked.
EOS
end

frame.get_content_pane.add(button)

frame.set_default_close_operation(JFrame::EXIT_ON_CLOSE)
frame.pack
frame.visible = true
```

```
verySophisticatedCamelCaseExample
ruby_is_using_this_naming_convention
```

Scala
naturalniej dla JVM

```
package main
import java.awt.event._
import javax.swing.{JFrame,JButton}

object ScalaSwing {
  def main(args: Array[String]) {
      val frame: JFrame = new JFrame("Hello world!")
      val button = new JButton("Klick Me!")

      button.addActionListener(new ActionListener() {
          override def actionPerformed(ae:ActionEvent) = {
              javax.swing.JOptionPane.showMessageDialog(null,
                      "<html>Hello from <b><u>Scala</u></b>.<br> Button "+
                      ae.getActionCommand() +  " clicked.")
          }
      })
      frame.getContentPane.add(button)
      frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE)
      frame.pack
      frame.setVisible(true)
  }
}
```

Jak to wygląda?
JRuby i .class

```
require 'java'
java_require 'my_foo'
class Foo
  java_signature 'void bar(int, int)'
  def bar(a,b)
    puts a + b
  end
end
```

```
PS my> jrubyc --javac my_foo.rb
Generating Java class Foo to my/Foo.java
javac  -d my -cp jruby-1.5.3/lib/jruby.jar:. my/Foo.java
```

Jak to wygląda?
Ruby z Javy

```
package redbridge;

import javax.script.ScriptEngine;
import javax.script.ScriptEngineManager;
import javax.script.ScriptException;

public class Jsr223HelloWorld {

    private Jsr223HelloWorld() throws ScriptException {
        ScriptEngineManager manager = new ScriptEngineManager();
        ScriptEngine engine = manager.getEngineByName("jruby");
        engine.eval("puts \"Hello World!\"");
    }

    public static void main(String[] args) throws ScriptException {
        new Jsr223HelloWorld();
    }
}
```

Właściwie po co?

- Język to tylko narzędzie.
- Twój ulubiony język na Twojej ulubionej platformie.
- Biblioteki Javy.
- Korporacje a nie-Java.
- Łączenie kodu w różnych językach.
- **DSL**
- Java assemblerem XXI wieku.

Domain-Specific Language

- Jaka Java jest, każdy widzi.
- RSpec + RBehave on Java

```
require 'java'
import 'java.net.ServerSocket'

describe "ServerSocket" do
 it "should know its own port" do
   server_socket = ServerSocket.new(5678)
   server_socket.localPort.should == 5678
 end
end
```

Domain Specific Language

```
require 'java'

describe java.util.ArrayList, " when first created" do
  before(:each) do
    @list = java.util.ArrayList.new
  end

  it "should be empty" do
    @list.should be_empty
  end

  it "should be able to add an element" do
    @list.add "content"
  end

  it "should raise exception when getting anything" do
    lambda{ @list.get 0 }.should \
                          raise_error(java.lang.IndexOutOfBoundsException)
  end
end
```

Podsumowanie

- Zestaw narzędzi.
- Dojrzała maszyna wirtualna.
- Piszemy kod szybciej.
- Integrowanie różnch języków.
- Wydajność.

Linki

- http://www.scala-lang.org/
- http://www.jython.org/
- http://jruby.org/
- http://rspec.info/