Implementacje języków na JVM

JAG

ETI PG

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?
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

```
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
```

Pytania?