

Parrot

"one bytecode to rule them all"

braga.pm Tech Meeting

Nuno 'smash' Carvalho <mestre.smash@gmail.com>

2007-02-03

Overview

Definition

Parrot is a virtual machine designed to compile and execute bytecode for dynamic languages.

Parrot Hacker Definition

Parrot is a register-based, bytecode-driven, object-oriented, dynamically typed, self-modifying, asynchronous interpreter.

Everyone should agree!

Parrot is **NOT** Perl6!

Overview

Definition

Parrot is a virtual machine designed to compile and execute bytecode for dynamic languages.

Parrot Hacker Definition

Parrot is a register-based, bytecode-driven, object-oriented, dynamically typed, self-modifying, asynchronous interpreter.

Everyone should agree!

Parrot is **NOT** Perl6!

Overview

Definition

Parrot is a virtual machine designed to compile and execute bytecode for dynamic languages.

Parrot Hacker Definition

Parrot is a register-based, bytecode-driven, object-oriented, dynamically typed, self-modifying, asynchronous interpreter.

Everyone should agree!

Parrot is **NOT** Perl6!

Software CPU

- register based (four type of registers)
- no operands limit to opcodes

Some Core Features

- object oriented
- threads
- exception handling
- events (signals)

Software CPU

- register based (four type of registers)
- no operands limit to opcodes

Some Core Features

- object oriented
- threads
- exception handling
- events (signals)

Some Advanced Features

- garbage collection
- MMD (Multi Method Dispatching)
- continuations
- coroutines

Core Design Principles

- speed
- abstraction
- stability

Some Advanced Features

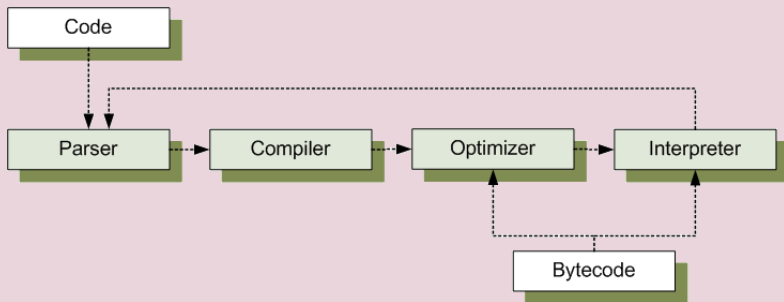
- garbage collection
- MMD (Multi Method Dispatching)
- continuations
- coroutines

Core Design Principles

- speed
- abstraction
- stability

Parrot's Architecture - Part I

Interpretation Flow



Parrot's Architecture - Part II

Parser

The parser is responsible for tokenizing source code and creating an AST (Abstract Syntax Tree).

Compiler

The compiler is responsible for generating the bytecode to feed the interpreter.

Optimizer

The optimizer generates bytecode that runs faster.

Interpreter

The engine that executes the generated bytecode.

Parrot's Architecture - Part II

Parser

The parser is responsible for tokenizing source code and creating an AST (Abstract Syntax Tree).

Compiler

The compiler is responsible for generating the bytecode to feed the interpreter.

Optimizer

The optimizer generates bytecode that runs faster.

Interpreter

The engine that executes the generated bytecode.

Parrot's Architecture - Part II

Parser

The parser is responsible for tokenizing source code and creating an AST (Abstract Syntax Tree).

Compiler

The compiler is responsible for generating the bytecode to feed the interpreter.

Optimizer

The optimizer generates bytecode that runs faster.

Interpreter

The engine that executes the generated bytecode.

Parrot's Architecture - Part II

Parser

The parser is responsible for tokenizing source code and creating an AST (Abstract Syntax Tree).

Compiler

The compiler is responsible for generating the bytecode to feed the interpreter.

Optimizer

The optimizer generates bytecode that runs faster.

Interpreter

The engine that executes the generated bytecode.

PASM vs PIR

PASM

- Parrot Assembly
- traditional low-level features
- also has advanced features
 - lexical, global variables, objects, *etc*

PIR

- Parrot Intermediate Representation
- high level features
 - named variables, *etc*
- it still isn't a HLL

PASM vs PIR

PASM

- Parrot Assembly
- traditional low-level features
- also has advanced features
 - lexical, global variables, objects, *etc*

PIR

- Parrot Intermediate Representation
- high level features
 - named variables, *etc*
- it still isn't a HLL

Get your feet wet

Checking Out

```
~$ svn co https://svn.perl.org/parrot/trunk \  
parrot
```

Building

```
~/parrot$ perl Configure.pl  
~/parrot$ make
```

Testing

```
~/parrot$ make test
```

Get your feet wet

Checking Out

```
~$ svn co https://svn.perl.org/parrot/trunk \  
parrot
```

Building

```
~/parrot$ perl Configure.pl  
~/parrot$ make
```

Testing

```
~/parrot$ make test
```

Get your feet wet

Checking Out

```
~$ svn co https://svn.perl.org/parrot/trunk \  
parrot
```

Building

```
~/parrot$ perl Configure.pl  
~/parrot$ make
```

Testing

```
~/parrot$ make test
```

The 'Hello world!' Example

hello.pir

```
.sub main :main
    print "Hello world!\n"
.end
```

linux x86

```
$ ./parrot hello.pir
Hello world!
```

hello.pbc

```
$ ./parrot --output-pbc --output=hello.pbc \
    hello.pir
$ ./parrot hello.pbc
```

The 'Hello world!' Example

hello.pir

```
.sub main :main
    print "Hello world!\n"
.end
```

linux x86

```
$ ./parrot hello.pir
Hello world!
```

hello.pbc

```
$ ./parrot --output-pbc --output=hello.pbc \
    hello.pir
$ ./parrot hello.pbc
```

The 'Hello world!' Example

hello.pir

```
.sub main :main
    print "Hello world!\n"
.end
```

linux x86

```
$ ./parrot hello.pir
Hello world!
```

hello.pbc

```
$ ./parrot --output-pbc --output=hello.pbc \
    hello.pir
$ ./parrot hello.pbc
```

Conclusion

- Parrot is a virtual machine
- Parrot Compiler Tools
 - target multiple languages (or brand new)
 - implement Perl6

It's a work in progress.

Conclusion

- Parrot is a virtual machine
- Parrot Compiler Tools
 - target multiple languages (or brand new)
 - implement Perl6

It's a work in progress.

Thanks

References

- the Parrot documentation
- "Perl6 And Parrot Essentials", by Allison Randal, Dan Sugalski & Leopold Tötsch

Find more

- <http://www.parrotcode.org>
- join the mailing-lists
- join #parrot @ irc.perl.org

Thanks

References

- the Parrot documentation
- "Perl6 And Parrot Essentials", by Allison Randal, Dan Sugalski & Leopold Tötsch

Find more

- `http://www.parrotcode.org`
- join the mailing-lists
- join #parrot @ irc.perl.org