Expressividade da linguagem enquanto programando

Jônatas Davi Paganini

15 de outubro de 2009

Resumo

Este trabalho busca trazer ao leitor um ambiente de programação mais próximo de sua realidade. Através de exemplos de programas de computador, será mostrado como a linguagem de programação pode ser expressiva e de fácil compreensão.

Sumário

1 Teoria da linguagem **2**

2 Evolução da linguagem de programação **2**
2.1 Exemplo 'Hello world' 2
2.1.1 Exemplo usando a linguagem de programação assembly . 3
2.1.2 Exemplo usando a linguagem de programação ruby 3
2.2 A simplicidade da linguagem 3

3 A linguagem ruby **4**
3.1 O compilador ruby 4
3.2 Ruby Interativo - IRB 4
3.3 Tipos de dados 5
3.3.1 Strings 5
3.3.2 Números 5
3.3.3 Entrada de dados 5
3.3.4 Classes e Módulos 6

1 Teoria da linguagem

Linguagem é qualquer e todo sistema de signos que serve de meio de comunicação de idéias ou sentimentos através de signos convencionais, sonoros, gráficos, gestuais etc., podendo ser percebida pelos diversos órgãos dos sentidos, o que leva a distinguirem-se várias espécies de linguagem: visual, auditiva, tátil, etc., ou, ainda, outras mais complexas, constituídas, ao mesmo tempo, de elementos diversos.12 Os elementos constitutivos da linguagem são, pois, gestos, sinais, sons, símbolos ou palavras, usados para representar conceitos de comunicação, idéias, significados e pensamentos. Embora os animais também se comuniquem, a linguagem propriamente dita pertence apenas ao Homem.

[1]

2 Evolução da linguagem de programação

A evolução dos computadores trouxe dispositivos menores e mais potentes. Na década de 80, usavam-se grandes computadores para realizar pequenos processos, 30 anos mais tarde estes dispositivos ganharam poder, velocidade, design e com baixo consumo de energia. Dispositivos que cabem na palma da mão, com apenas alguns toques ou cliques tornam acessível a informação desejada.

Da mesma forma como os computadores ganharam potência, as linguagens de programação se tornaram expressivas e humanas. Este dinâmismo não tem o objetivo de trazer conforto a máquina, mas sim ao seu manipulador - o homem.

A linguagem de computador, inicialmente era grosseira e de difícil compreensão, com o passar do tempo, as técnicas foram evoluindo e a linguagem, mesmo de computador, foi ganhando forma e expressão. Houve uma percepção de mudança, que tornaria a linguagem de programação uma auxiliadora do programador e não uma interpretadora.

2.1 Exemplo 'Hello world'

O programa Hello world é o programa mais conhecido no mundo inteiro e é o exemplo mais básico de uma linguagem de programação com o objetivo de imprimir a mensagem "Hello, world!"e guiar o iniciante em sua primeira compilação/execução de um programa de computador. Abaixo seguem dois exemplos "Hello, world"em duas linguagens de programação distintas: asssembly e ruby.

2.1.1 Exemplo usando a linguagem de programação assembly

Listing 1: Exemplo em assembly

```
variable:
    .message    db    "Hello world!\$"
code:
    mov   ah,9
    mov   dx,offset .message
    int   0x21
    ret
```

Analisando a listagem 1.1 vemos que possuem muitas palavras (comandos, declarações) desconhecidas, e com pouca aparência semântica. Isto acontece devido ao tipo de compilador (de linguagem compilada) e o nível de acesso as complexidades do hardware do computador.

2.1.2 Exemplo usando a linguagem de programação ruby

Listing 2: Exemplo em ruby

```
print "Hello, world!"
```

2.2 A simplicidade da linguagem

Foi necessário apenas uma linha de código para representar o mesmo exemplo na linguagem ruby. No exemplo mais simples o objetivo é apenas imprimir "Hello, world!"e é exatamente isso que está escrito. Diferente de assembly, ruby não é uma linguagem compilada e sim dinâmica. Assim, tudo acontece em tempo real, enquanto está sendo executado.

Listing 3: Tradução do programa ruby

```
imprima "Ola mundo!"
```

Apenas com uma linha de código é possível fazer exatamente o que está sendo proposto. Este programa de computador foi escrito de forma simples e humanamente legível, diferente da primeira instrução de máquina escrita em assembly. No decorrer deste artigo serão abordados outros exemplos de programação como este, que, com poucas palavras expressa exatamente o objetivo do software no domínio em questão. Exemplos como este, mostram o poder do homem de categorizar e generalizar as informações. Desta forma a percepção mudou de, programar para

o computador para programar para as outras pessoas. Anteriormente, com uma programação rígida e a escassez de processamento, a codificação de um software realmente fazia parte de um processo árduo e lento, aonde não era possível tornar agradável a leitura de uma instrução de computador.

3 A linguagem ruby

Ruby é uma linguagem dinâmica, open source com foco na simplicidade e produtividade. Tem uma sintaxe elegante de leitura natural e fácil escrita. Com um grande número de bibliotecas disponíveis gratuitamente (gems), tem atraído muita gente para desenvolver nesta linguagem.

Para usar no Mac OS X ou Linux, é necessário abrir o terminal e no Windows pode-se usar o Command-DOS e após isso digitar **ruby** para invocar o compilador, **irb** para iniciar uma interação.

3.1 O compilador ruby

O compilado pode ser invocado usando a linha de comando e receber uma código para compilar"

Listing 4: Usando o compilador na linha de comando

```
jonatas@xonatax-mac:~$ ruby -e "puts 1+2"
3
```

3.2 Ruby Interativo - IRB

Existe um programa chamado IRB, que vem instalado com compilador de ruby, que serve para interagir com a linguagem na forma de console.

Listing 5: Usando o compilador na linha de comando

```
jonatas@xonatax-mac:~$ irb
>> 1.class
=> Fixnum
>> 1 + 2
=> 3
```

3.3 Tipos de dados

3.3.1 Strings

As strings podem ser limitados por vários delimitadores, em geral são usadas aspas simples e duplas, mas existem outras formas.

Listing 6: Exemplos de uso de string

```
>> "␣outra␣string" + '␣mais␣uma' + %[ outra ] + %{outra} + %(uma strin
=> "␣outra␣string␣mais␣uma␣outra␣outrauma␣strings"
```

3.3.2 Números

Os números são objetos como qualquer outro e os operadores são simples métodos.

Listing 7: Exemplos de uso de números

```
>> 1.class
=> Fixnum
>> 1.class.ancestors
=> [Fixnum, Integer, Precision, Numeric, Comparable, Object, Kernel]
>> 1.0.class
=> Float
```

3.3.3 Entrada de dados

No exemplo abaixo será feito uma pergunta para o usuário e quando ele confirmar a resposta irá saudar o usuário.

Listing 8: Exemplo de entrada de dados

```
print "qual␣seu␣nome?"
nome = gets
print "oi␣#{nome}"
```

Na primeira linha, é impressa a pergunta. Na próxima linha é atribuido a uma variável nome a entrada de dados dada pelo método gets. Em seguinda é impressa a saudação oi juntamente com a variável nome. Notem que o uso de #{} embutido na string já concatena a expressão interna a string.

3.3.4 Classes e Módulos

Classe é a declaração que permite definir uma classe de elementos.

Módulos permitem dividir comportamentos de classe e de instância, permitindo assim a multipla extensão de classes. Ele pode ser usado em blocos e subblocos e todo o código gerado fica encapsulado na declaração.

Referências

[1] Wikipédia, Acesso em setembro de 2009.

`http://pt.wikipedia.org/wiki/Linguagem`