

Introdução ao Perl

Practical Extraction And Report Language



Tchelinix Ulbra Gravataí

Douglas Schilling Landgraf
<dougslan@redhat.com>

Maio 28, 2009.

Introdução ao Perl

- ▶ Nível: Básico
- ▶ Escopo: Perl para iniciantes
- ▶ Pré-requisitos: Lógica de Programação
- ▶ Sobre o palestrante

Introdução ao Perl

- ▶ Cebola ou Camelo?
- ▶ Origem: 1987 (shell script, C, awk e sed)
- ▶ Criador: Larry Wall
- ▶ Por que utilizar Perl?
- ▶ Por que Perl foi criado?
- ▶ Plataformas suportadas?
- ▶ Web? Shell? Orientação a objetos? GTK? QT?
- ▶ Linguagem interpretada
- ▶ O que vamos ver hoje?

Introdução ao Perl

- ▶ Como eu instalo o perl?

- Sources:

- CPAN - Comprehensive Perl Archive Network
 - <http://www.cpan.org>

- Distros

Introdução ao Perl

- ▶ Primeiro programa:

```
#!/usr/bin/perl  
print "Oi pessoal, eu não sou um camelo!\n";
```

- ▶ Linhas:

- #!
- /user/bin/perl
- print "Oi pessoal, eu não sou um camelo!\n";

- ▶ Precisa de extensão? .plx?
- ▶ Cadê o main()?
- ▶ Pode #!/usr/bin/env perl ?

Introdução ao Perl

- ▶ Permissão + Execução:

- shell> chmod +x programa

- shell> ./programa

- Oi pessoal, eu não sou um camelo!

- ▶ Outra alternativa:

- shell> perl ./programa

perl -w (Warning e Strict)

- ▶ Warning seu ajudante contra possíveis erros:

```
#!/usr/bin/perl -w
```

```
print "Oi pessoal, eu não sou um camelo!\n"
```

- ▶ Não funcionou? oops cadê a ; ?
- ▶ use strict;

CodingStyle

- ▶ CodingStyle
- ▶ O que é CodingStyle? (Codificação em estilo)
- ▶ Devemos nos preocupar?
- ▶ Os famosos números mágicos.....

```
#!/usr/bin/perl
```

```
                print  
    "Oi pessoal, "  
                "eu não sou um ...."  
    "camelo!"  
                . "...\\n"  
    ;
```

perl - Comentários

▶ Comentários

```
# Eu sou um comentário
```

```
print "Oi pessoal!\n" # Ele escreveu oq?
```

```
# Não sei :(
```

Variáveis

▶ Tipos de Variáveis apresentados hoje:

- Scalar

- Array

- Hash

- Filehandle

Variáveis: Scalar

- ▶ Representa: números, strings e uma ... (mais tarde você saberá)

- Associe em sua memória Scalar com o símbolo \$

- ▶ Exemplos:

```
$valor = 5;           $minhaStr = "Oi pessoal!\n";
```

```
$nroHexa = 0x45;     $valor++;
```

```
$soma = 2 + 2;       $float = 0.54;
```

```
$float = "Sim, eu virei string!\n";
```

Variáveis: Strings

- ▶ Declarando:

- `$minhaStr = "Oi, eu sou uma string";`

- ▶ Concatenando:

- O caracter `.`

- ▶ Caracteres especiais:

- `"\n"` - Nova linha
- `"\t"` - tab
- `"\r"` - Return
- ...

Variáveis: Strings

- ▶ Repetição:

- "string" x 4
- \$array[0] x 4

- ▶ Removendo "\n":

- `chomp($variavel);`

Comparação (if/elsif/else):

▶ Operadores:

Numérico	String	Significado
==	eq	Equal
!=	ne	Not Equal
<	lt	Less than
>	gt	Greater than
<=	le	Less than or equal to
>=	ge	Greater than or equal to
		Or
&&		And
...		

▶ Exemplos:

```
if ($variaval == 10) {  
  print "Sim, é 10!\n"  
}
```

```
if ($variaval eq "Linux") {  
  print "Sim, é Linux!\n";  
}
```

while() - Repita a operação até...

▶ while()

```
while($var != 5) {  
    print "Repitindo: O conteúdo de $var é diferente de 5\n";  
    $var++;  
}
```

for() - Repita a operação até...

▶ for()

```
for ($i = 0; $i < 10; $i++) {  
    print "Repitindo: O conteúdo de $i ainda é menor que 10!\n";  
}
```

foreach() - Repita a operação até...

- ▶ foreach()

```
foreach $elemento (@dogs) {  
    print $elemento;  
}
```

switch() - Escolha entre..

▶ switch()

```
use Switch;

switch($var) {
    case 15: {
        printf("Achei 15\n");
    }
    case 25: {
        printf("Achei 25\n");
    }
    case 30: {
        printf("Achei 30\n");
    }
    else {
        printf("Não achei nada :(\n");
    }
}
```

Executando um comando no sistema

▶ Chamadas:

- `$retval = `comando``
- `print $retval;`

ou

- `$retval = system("comando");`

Lendo teclado

▶ Tipos do saída:

- stdout = 0
- stdin = 1
- stderr = 2

```
$character = <STDIN>;  
chomp($character); # Remove "\n"  
  
if ($character eq "tchelinix") {  
    print "Você digitou tchelinix!\n";  
}
```

Funções (sem parâmetro e sem retorno)

▶ Criando:

```
sub nome_funcao {  
    print "Eu sou uma função";  
}
```

▶ Chamando:

```
&nome_funcao;  
ou  
nome_funcao;
```

Funções (com validação/retorno)

▶ Tipos de retorno:

- Positivos (sucesso: 0, 1, 2 ..)
- Negativos (erro: -1, -2, -3 ..)

▶ Criando função p/ retornar:

```
sub minha_funcao {  
    $ret = &Le_Dados_Webcam;  
    return ret;  
}
```

▶ Chamando a função:

```
$retval = &minha_funcao;  
if ($retval < 0) {  
    print "Erro ao chamar minha_funcao!\n";  
    exit;  
}
```

Funções (com parâmetro)

- ▶ Criando função p/ retornar:

```
sub minha_funcao {  
    $id          = $_[0];  
    $nome        = $_[1];  
    $sobreNome   = $_[2];  
    $idade       = $_[3];  
  
    print $nome  
    return ret;  
}
```

- ▶ Chamando a função:

```
$retval = &minha_funcao( 0, "douglas", "schilling", "landgraf", $idade );  
if ($retval < 0) {  
    print "Erro ao chamar minha_funcao!\n";  
    exit;  
}
```

Variáveis Locais

▶ `my $variavel = 5;`

Variáveis: Arrays

- ▶ O que é um array?

- ▶ Como criar um array?

- ▶ Exemplo:

```
@dogs = ("misca\n", "alemoa\n", "balof\n", 12, "clacla\n", 80 );  
Posição: 0          1          2          3          4          5
```

- ▶ Como acessar um elemento do array? (Aqui lembrar do Scalar)

```
print $dogs[0]; # posição 0 é a string misca  
print $dogs[1]; # posição 1 é a string alemoa  
print $dogs[2]; # posição 2 é a string balof  
print $dogs[3]; # posição 3 é o número 12  
print $dogs[4]; # posição 4 é a string clacla  
print $dogs[5]; # posição 4 é o número 80
```

Variáveis: Arrays

- ▶ Como imprimir o array inteiro?

- `print @dogs;`

- ▶ Ordenando alfabeticamente um array:

- `@var = sort("tchelinux", "linux", "Fedora", "Laptop", "amarok");`

- ▶ Invertendo a ordem do array..

- `@var = reverse("tchelinux", "linux", "Fedora", "Laptop", "amarok");`

- ▶ Pegando somente alguns campos:

- `@novaVariavel = @var[1..4];`

Variáveis: Arrays

- ▶ Quoted words ou Quoted whitespace

- `@dogs = qw(clara balofao scooby);`

- ▶ Split

- `$string = "Fedora::RedHat::Linux::Kernel";`

- `@meuArray = split(/::/, $string);`

- `print $meuArray[posicao];`

- ▶ Join

- `@dogs = ('balofa', 'tigor', 'SuperFera', 'SemNocao');`

- `@dogsComVirgula = join(', ', @dogs);`

Variáveis: Arrays

- ▶ Remove ou adiciona um elemento no **final** do array
- ▶ pop/push
 - `pop(@dogs);`
 - `push(@dogs, "ze");`
- ▶ Remove ou adiciona um elemento no **início** do array
- ▶ shift/unshift
 - `shift(@dogs);`
 - `unshift(@dogs, "clara");`

Variáveis: Arrays

- ▶ Como saber qual é o número total de elementos?

- `print scalar @dogs;`

- ▶ O último elemento?

- `print $dogs[-1];`

- ▶ Outros exemplos:

- `@dogs = ();`

- `@dogs = (1..100);`

Variáveis: Como lembrar operadores do array?

- ▶ Em outras palavras... associe:

`@` -> você vai declara uma variável do tipo array

- ▶ Para acessar uma posição utilize o caracter \$ (scalar) e a posição:

```
print $carros[0];  
print $carros[1];  
print $carros[2];  
print $carros[3];
```

- ▶ Imprimir um array inteiro:

```
print @carros;
```

Variáveis: Hash

- ▶ É um array onde seu índice é uma string

```
%carros = ();
```

```
$carros{'fiat'} = "palio";  
$carros{'volks'} = "parati";
```

```
print $carros{fiat};
```

ou

```
%idades = ('Douglas' => 26, 'Sharon' => 28);  
foreach $pegaldade (%idades)  
{  
    print "$pegaldade";  
    print "\n";  
}
```

Filehandle - Arquivos

▶ Filehandle

```
$file = meu-arquivo.txt
```

```
open (LOG, "<$file") or die "falhou";
```

```
foreach $linha <LOG> {  
    print $linha;  
}  
close(LOG);
```

▶ Escrevendo:

- print LOG \$str;

Filehandle - Arquivos

▶ Onde:

- < abre o arquivo somente p/ leitura
- > abre o arquivo p/ escrita, criando-o caso não exista;
- >> abre o arquivo em modo anexo (append);
- +> abre o arquivo p/ leitura e escrita

Perdoc

▶ Documentação:

```
shell> man perldoc
```

```
shell> perldoc -f funcao
```

```
shell> perldoc -q palavra do FAQ
```

Introdução ao Perl

Dúvidas e Sugestões ?

Douglas Schilling Landgraf
<dougslan@redhat.com>

<http://dougslan.livejournal.com>
<http://people.tchelinux.org>

Maio 28, 2009.