

Delphi

Object Pascal

Jocélio Passos

joceliopassos@bol.com.br

Object Pascal

- É uma linguagem Orientada a Objetos
- Não pura mas, híbrida
- Linguagem prática
- Versátil
- Extremamente estável
- Produto de grande complexidade
- Força o programador executar passos lógicos
- Consequentemente torna mais fácil o desenvolvimento :
- Ambiente Windows, aplicações livres ou que utilizam banco de dados do tipo Cliente/Servidor

2/6/2008

2

Object Pascal

- Trabalha com o uso de ponteiros para a alocação de memória
- Código totalmente compilável
- Possibilita a criação e reutilização de :
 - Objetos e
 - Bibliotecas dinâmicas (Dynamic Link Libraries - DLL).
- Há tipos de informações em modo run-time
- Manipulação de exceções
- Referências de classes.

2/6/2008

3

Object Pascal

- Suportar as facilidades de um baixo nível de programação, tais como:
 - Controle e acesso das subclasses do Windows (API - Application Program Interface)
 - Passar por cima das mensagens de loop do Windows
 - Mensagens semelhantes as do Windows
 - Código puro da linguagem Assembler.
- É a base de toda a programação Delphi

2/6/2008

4

Object Pascal - Palavras Reservadas

- | | |
|-------------|---------------|
| ■ ABSOLUTE | ■ CONSTRUCTOR |
| ■ ABSTRACT | ■ DEFAULT |
| ■ AND | ■ DESTRUCTOR |
| ■ ARRAY | ■ DIV |
| ■ AS | ■ DO |
| ■ ASM | ■ DOWNTON |
| ■ ASSEMBLER | ■ DYNAMIC |
| ■ AT | ■ ELSE |
| ■ BEGIN | ■ END |
| ■ CASE | ■ EXCEPT |
| ■ CDECL | ■ EXPORT |
| ■ CLASS | ■ EXPORTS |

2/6/2008

5

Object Pascal - Palavras Reservadas

- | | |
|------------------|------------------|
| ■ EXTERNAL | ■ INHERITED |
| ■ FAR | ■ INTERRUPT |
| ■ FILE | ■ INITIALIZATION |
| ■ FINALLY | ■ INLINE |
| ■ FOR | ■ INTERFACE |
| ■ FORWARD | ■ IS |
| ■ FUNCTION | ■ LABEL |
| ■ GOTO | ■ LIBRARY |
| ■ IF | ■ MESSAGE |
| ■ IMPLEMENTATION | ■ MOD |
| ■ IN | ■ NAME |
| ■ INDEX | ■ NEAR |

2/6/2008

6

Object Pascal - Palavras Reservadas

- | | |
|-------------|-------------|
| ■ NIL | ■ PROPERTY |
| ■ NODEFAULT | ■ PROTECTED |
| ■ NOT | ■ PUBLIC |
| ■ OBJECT | ■ PUBLISHED |
| ■ OF | ■ RAISE |
| ■ ON | ■ READ |
| ■ OR | ■ RECORD |
| ■ OVERHIDE | ■ REPEAT |
| ■ PACKED | ■ RESIDENT |
| ■ PRIVATE | ■ SET |
| ■ PROCEDURE | ■ SHL |
| ■ PROGRAM | ■ SHR |

2/6/2008

7

Object Pascal - Palavras Reservadas

- | | |
|-----------|---------|
| ■ STORED | ■ WITH |
| ■ STRING | ■ WRITE |
| ■ THEN | ■ XOR |
| ■ TO | |
| ■ TRY | |
| ■ TYPE | |
| ■ UNIT | |
| ■ UNTIL | |
| ■ USES | |
| ■ VAR | |
| ■ VIRTUAL | |
| ■ WHILE | |

2/6/2008

8

Object Pascal

■ Constantes

- Quando há o uso do mesmo valor repetidamente ...
- Deve-se usar constantes que são declaradas na seção const
- Vantagens :
 - Torna o número mais fácil de ser lembrado
 - É muito mais fácil para alguém manter seu código.
 - Clareza e a facilidade de manutenção.

2/6/2008

9

Object Pascal

■ Constantes

- Podem assumir muitas formas e abranger muitos tipos de dados.
- CONST RCPMF = 0.0038;
- CNOOME = 'João da Silva';
- BIDADE = 32;
- RPESO = 6.50;
- CENDERECO = 'Rua das Laranjeiras, 52';

2/6/2008

10

Object Pascal

■ Constantes

- No momento da compilação, o compilador olha sua declaração de constante RCPMF = 0.0038, localiza cada ocorrência de RCPMF no aplicativo e a substitui por 0.0038.
- A questão é que ele faz isso apenas **durante a compilação** e **não muda seu código-fonte**.
- As constantes são apenas para seu proveito e não do Delphi/pascal.
- Suas declarações de constantes são simplesmente uma grande tabela de substituição

2/6/2008

11

Object Pascal

■ Constantes

- Funções que podem ser utilizadas para a declaração das constantes :
- | | | | |
|----------|------|------|-------|
| ■ AB | CHR | HI | HIGH |
| ■ LENGTH | LO | LOW | ODD |
| ■ ORD | PRED | PTR | ROUND |
| ■ SIZEOF | SUCC | SWAP | TRUNC |

2/6/2008

12

Object Pascal - Tipos Inteiros

Tipo	Intervalo de Valores	Bytes de Memória
Byte	0...255	1
Word	0...65 535	2
Shortint	-128...127	1
Small Int	-32 768...32 767	2
Integer	-2 147 483 648...2 147 483 647	4
Ordinal	0...2 147 483 647	4
LongInt	-2 147 483 648...2 147 483 647	4

VAR indice : INTEGER;

2/6/2008

13

Object Pascal - Tipos Reais

Tipo	Intervalo de Valores	Bytes de Memória
Real	$2.9 \times 10^{-39} \dots 1.7 \times 10^{38}$	06
Single	$1.5 \times 10^{-45} \dots 3.4 \times 10^{38}$	04
Double	$5.0 \times 10^{-324} \dots 1.7 \times 10^{308}$	08
Extenden	$3.4 \times 10^{-4932} \dots 1.1 \times 10^{4392}$	10
Comp	$-2^{63} \dots 2^{63-1}$	08

VAR RPAGAMENTO : REAL;

2/6/2008

14

Object Pascal

■ Tipo Currency (Moeda)

- Novo tipo de dados
- É um tipo de **ponto flutuante**
- Compatível em relação à atribuição com todos os outros tipos de ponto flutuante, incluindo o tipo Variant
- Tem uma precisão de **quatro casas decimais**
- É armazenado como um inteiro de 64 bits
- Tem uma precisão maior para conter números grandes
- É usado no **Currency Field** e em outros componentes.
- Ele é compatível com os tipos de banco de dados que representam dinheiro

■ **VAR YPAGAMENTO : CurrencyY;**

2/6/2008

15

Object Pascal

■ Tipo Boolean

- Um dos mais simples e mais usados
- Representam TRUE e FALSE
- Aceitam os operadores and, or e not
- VAR LOK: BOOLEAN; // Lógico

2/6/2008

16

Object Pascal

■ Tipo Char (Caracter)

- Armazena apenas um caracter (um byte)

```

VAR CLETRA : CHAR;
...
CLETRA := 'A';
CLETRA := #65; // Letra A em ASCII
...

```

2/6/2008

17

Object Pascal

■ Tipo String

```

Array de caracteres
VAR  CNome : STRING [30];
    Cendereco : STRING;
...
CNome := 'Carlos José';
Cendereco := ''; // Vazia ...

```

2/6/2008

18

Object Pascal

Estrutura de Dados Array (Vetores)

- VAR ARNOME : ARRAY [1..5] OF REAL;
- Ou
- VAR ARNOME : ARRAY [6..10] OF REAL;

2/6/2008

19

Object Pascal

Estrutura de Dados Array Multidimensionais

- Matriz
- VAR ACNOME : ARRAY [1..3,1..3] OF STRING;

2/6/2008

20

Object Pascal

Estrutura de Dados Record (Registros)

- Ao contrário de um array, os itens de um registro não precisam ser do mesmo tipo
- TYPE RPessoal=RECORD
- CNAME : STRING [40];
- CFONE : STRING [11];
- BIDADE : BYTE;
- CSEXO : CHAR; // M ou F
- YSALA : CURRENCY;
- END;

2/6/2008

21

Object Pascal

Estrutura de Dados Record (Registros)

- VAR RTEMP : RPessoal;
- ARFUNCIONARIO : ARRAY [1..300] OF RPessoal;
- ...
- RTEMP.CNAME := 'Antônio Luiz';
- ARFUNCIONARIO [1].CFONE := '2547823398';
- WITH RTEMP DO
- BEGIN
- BIDADE := 18;
- CSEXO := 'M' ;
- YSALA := 495.45;
- END;

2/6/2008

22

Object Pascal

WITH ... DO ... :

- Delimita um determinado bloco de declarações para um identificador específico, evitando a declaração deste identificador.
- WITH <IDENTIFICADOR> DO
- BEGIN
- ATRIBUTO;
- FUNCTION;
- END;
- Equivale a :
- <IDENTIFICADOR>.ATRIBUTO;
- <IDENTIFICADOR>.FUNCTION;

2/6/2008

23

Object Pascal

BEGIN ... END:

- Envolve um conjunto de declarações em um bloco de comandos determinado.
- BEGIN
- ...
- BEGIN
- ...
- END;
- ...
- END;

2/6/2008

24

Object Pascal

Tipo Variant

- Contido também no Visual Basic
- Ou o **Usual** no Visual Objects
- Pode-se colocar nele praticamente qualquer valor, de um inteiro a um string.
- É uma estrutura de 16 bytes
- Contém não apenas o valor, mas também as informações do tipo.
- VAR VMEUVALOR : VARIANT;
- ...
- VMEUVALOR := 3.5555;
- VMENUVALOR := 'Carlos Magno';

2/6/2008

25

Object Pascal - Operadores Aritméticos

Operador	Operação
+	Adição
-	Subtração
*	Multiplicação
/	Divisão
Div	Divisão Inteira
Mod	Resto da Divisão

2/6/2008

26

Object Pascal - Operadores Booleanos

Operador	Operação
not	Negação
and	and
or	or
xor	xor

2/6/2008

27

Object Pascal - Operadores Relacionais

Operador	Operação
=	Igual
<>	Diferente de
<	Menor que
>	Maior que
<=	Menor ou igual
>=	Maior ou igual a
In	Membro de (∈)

2/6/2008

28

Object Pascal - Precedência de Operadores

Operadores	Categorias
@, not	unários
*, /, div, mod, and, as	multiplicação
+, -, or, xor	adição
=, <, >, <=, >=, in, is	relacionais

2/6/2008

29

Object Pascal

Comentários

- {Comentário em mais de uma linha }
- COMANDO // Comentário na mesma linha

2/6/2008

30

Object Pascal

■ IF

```
■ If <Expressão> then <Expressão2> else <Expressão3>;  
  
■ If <Expressão> then  
  BEGIN  
    | <Expressão2>  
  END  
■ else  
  BEGIN  
    | <Expressão3>  
  END;  
■ END;
```

2/6/2008

31

Object Pascal

■ Case

```
■ CASE <SELETOR> OF  
  OPC1 : BEGIN  
    END;  
  OPCN : BEGIN  
    END;  
  ELSE  
    BEGIN  
    END;  
■ END;
```

2/6/2008

32

Object Pascal

■ Case

- Se você usar mais de uma constante em uma linha de comparação, elas devem ser separadas por vírgulas, a não ser que você esteja denotando um intervalo de valores, como ("a" até "z").
- A expressão ou a constante que está sendo comparada com a expressão devem ser tipos originais de byte ou do tamanho de Word, de modo que você não pode usar um tipo String ou LongInt como argumentos.

2/6/2008

33

Object Pascal

■ Repeat ... Until

- ```
■ REPEAT
 ...
■ UNTIL <Expressão>
```
- As instruções entre repeat e until são repetidas até que a condição definida após a palavra reservada Until seja avaliada como TRUE
  - Nenhum begin/end é necessário em torno das instruções que estão sendo executadas, pois o repeat e o until servem como marcadores para mostrar que código está sendo executado no loop.

2/6/2008

34

## Object Pascal

### ■ While ... Do

```
■ WHILE <Expressão> DO
 BEGIN
 ...
■ END;
```

2/6/2008

35

## Object Pascal

### ■ For ... Do

```
■ FOR <CONTADOR> := N1 TO N2 DO
 BEGIN
 ...
■ END;

■ FOR <CONTADOR> := N2 DOWNTON N1 DO
 BEGIN
 ...
■ END;
```

2/6/2008

36

### Object Pascal - Estrutura dos Programas

```

PROGRAM ... ;
USES ... ; // Novas unidades de código
CONST ... ;
TYPE ... ;
VAR <VARIÁVEL> : TIPO ;

BEGIN
...
END.

```

2/6/2008

37

### Object Pascal - Procedures

```

PROCEDURE PROC1 ([VAR]<PARAMETRO> : TIPO;...) ;
CONST ... ;
TYPE ... ;
VAR ... ;
 PROCEDURE PROC2 ;
 CONST ... ;
 TYPE ... ;
 VAR ... ;
 BEGIN
 ...
 END;
BEGIN
...
END;

```

2/6/2008

38

### Object Pascal - Functions

```

FUNCTION FUNC1 ([VAR]<PARAMETRO> : TIPO;...):TIPO ;
CONST ... ;
TYPE ... ;
VAR ... ;
 FUNCTION FUNC2 : TIPO ;
 CONST ... ;
 TYPE ... ;
 VAR ... ;
 BEGIN
 ...
 RESULT := ... ;
 END;
BEGIN
...
RESULT := ... ;
END;

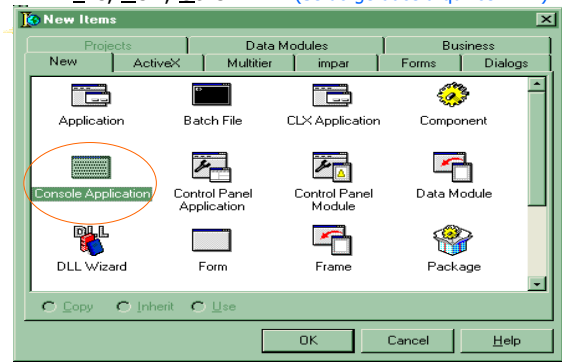
```

2/6/2008

39

### Object Pascal - Console Application

File; New; Other (Serão gerados arquivos .DPR)



2/6/2008

40

### Object Pascal - Exemplo (Console)

```

program Console;
{$APPTYPE CONSOLE}
uses SysUtils;
var CNome:String;
 NA, NB : INTEGER;
begin
 Writeln ('ola, Qual e o seu nome ?');
 Readln (CNome);
 Writeln (CNome, ', gostei de voce !');
 Writeln (Copy (CNome, 1,3));
 Write ('Digite um numero : ');
 Readln (NA);

```

2/6/2008

41

### Object Pascal - Exemplo (Console)

```

 Write ('Digite outro numero : ');
 Readln (NB);
 Writeln ('A soma e ', NA + NB);
 Writeln ('A media e ', ((NA + NB) / 2):6:2);
 Writeln ('CHR (128) ',CHR (128));
 Writeln ('ORD (a) ',ORD ('a'));
 Writeln ('CONCAT (JOAO,MARIA) ', CONCAT ('JOAO',
 ' MARIA'));

 Writeln ('ABS (-256) ', ABS (-256));
 Writeln ('INT (4.25) ', INT (4.25):6:2);
 Writeln ('FRAC (4.25) ', FRAC (4.25):6:2);
 Writeln ('ROUND (4.25) ', ROUND (4.25));

```

2/6/2008

42

### Object Pascal - Exemplo (Console)

```
■ WRITELN ('ODD (4) ', ODD (4));
■ WRITELN ('ODD (5) ', ODD (5));
■ WRITELN ('SQR (12) ', SQR (12));
■ WRITELN ('SQRT (144) ', SQRT (144):6:2);
■ WRITELN ('PI : ', PI:6:4);
■ WRITELN ('POS (A, MARIANA) ', POS ('A', 'MARIANA'));
■ WRITELN ('UPCASE (a) ',UPCASE ('a'));
■ STR (10,CNOME);
■ WRITELN ('STR (10)', CNOME);
■ VAL ('1',NA, NB);
■ WRITELN ('VAL ("1") ', NA);
■ READLN;
■ end.
```

2/6/2008

43

### Object Pascal - Exemplo (Impar)

```
■ program Impar;
■ {$APPTYPE CONSOLE}
■ uses SysUtils;
■ var inum:integer;
■ begin
■ WRITELN ('Mostra se um numero e par ou impar !');
■ write ('Digite um numero ? ');
■ readln (inum);
```

2/6/2008

44

### Object Pascal - Exemplo (Impar)

```
■ while (inum <> 0) do
■ begin
■ WRITE ('O numero informado e ');
■ IF (odd (inum)) THEN
■ WRITELN (' impar !')
■ ELSE
■ WRITELN (' par !');
■ write ('Digite um numero ? ');
■ readln (inum);
■ end;
■ end.
```

2/6/2008

45

### Object Pascal - Exemplo (Fibonacci)

```
■ program Fibonacci;
■ {Projeto ou programa principal}
■ {$APPTYPE CONSOLE}

■ uses
■ SysUtils;

■ var bnum:byte;
```

2/6/2008

46

### Object Pascal - Exemplo (Fibonacci)

```
■ PROCEDURE FIB (BNUM:BYTE);
■ VAR bcont:byte;
■ ti, tj : integer;
■ {Procedure que mostra a sequência}
■ BEGIN
■ ti := 0;
■ tj := 1;
■ write (ti, ', ');
```

2/6/2008

47

### Object Pascal - Exemplo (Fibonacci)

```
■ for bcont := 2 to bnum do
■ begin
■ write (tj, ', ');
■ tj := ti + tj;
■ ti := tj - ti;
■ end;
■ //readln; //Pausa
■ END;
```

2/6/2008

48



### Object Pascal - Exemplo (Fibonacci)

```
begin
write ('Quantos termos ? ');
readln (bnum);
while (bnum <> 0) do
BEGIN
FIB (BNUM);
write ('Quantos termos ? ');
readln (bnum);
END;
end.
```

2/6/2008

49

### Exercícios Propostos

- 1) "Object Pascal contém todo o conceito da orientação a objetos, incluindo encapsulamento, herança e polimorfismo". Verdadeiro ou Falso ?
- 2) "Programa de ponto de vendas. Tem-se uma taxa de imposto de vendas local várias vezes em seu aplicativo. Para resolver isto simplesmente coloca 0,075 (7,5%) cinco vezes em seu aplicativo. O problema é quando eles mudam a taxa do imposto. Você precisa se lembrar de mudá-la em cinco lugares. Isso também significa que você tem uma alta probabilidade de esquecer uma mudança ou cometer um erro tipográfico no valor 0,075." Como resolver este problema ?

2/6/2008

50

### Exercícios Propostos

- 3) "É importante que seus nomes de variável de constante sejam o mais descritivos possível. Os nomes de identificador podem ter qualquer comprimento, mas somente os 63 primeiros caracteres são reconhecidos. O primeiro caractere do nome de um identificador deve ser um sublinhado ou uma letra. Os caracteres subsequentes devem ser letras, números ou um sublinhado." Verdadeiro ou Falso ?
- 4) "O valor da constante pode mudar durante a execução do programa." Verdadeiro ou Falso ?

2/6/2008

51

### Exercícios Propostos

- 5) "O inteiro é um dos tipos genéricos no Delphi. Os tipos genéricos são aqueles que são afetados pela CPU ou sistema operacional em particular em que o compilador está implementado. Em um sistema operacional de 32 bits, como o Windows 9x, os tipos genéricos assumem suas respectivas capacidades de armazenamento com base no sistema operacional." Verdadeiro ou Falso ?
- 6) "O tipo Comp é, na verdade, um grande inteiro e não um número real. A razão de estar incluído na tabela de reais é que ele é implementado no mesmo estilo dos tipos de ponto flutuante. Na verdade, trata-se de um inteiro de 64 bits." Verdadeiro ou Falso ?

2/6/2008

52

### Exercícios Propostos

- 7) "2<sup>8</sup> (um byte) tem 256 diferentes caracteres que poderiam ser colocados em uma variável de tipo **char**. Na tabela ASCII existem caracteres com códigos variando de 0 a 255". Verdadeiro ou Falso ?
- 8) "O tamanho de uma String pode ir até de 3GB - suporte para String longo. Isto é possível com o uso da diretiva de compilador \$H+. Essa diretiva fica ligada por padrão." Verdadeiro ou Falso ?
- 9) "O problema do tipo Variant no VB é a memória exigida para uma variável desse tipo. A Microsoft sugeriu o uso desse tipo apenas quando necessário (devido às exigências de recursos)." Verdadeiro ou Falso ?

2/6/2008

53

### Exercícios Propostos

- 10) "O Delphi 1 não tinha tipo Variant, pois, segundo a visão do autor, o Object Pascal é uma linguagem fortemente tipada, e o tipo Variant se presta a algumas práticas de programação imprecisas. Mas, como resultado da demanda popular, e graças a alguns requisitos novos, o Delphi inclui um tipo Variant." Verdadeiro ou Falso ?
- 11) "A procedure é um tipo de miniprograma e tem o mesmo formato. Após o cabeçalho, você pode ter uma seção Const, Type Var, ou mesmo outra procedure. Uma procedure pode ter outra, aninhada dentro dela. Após isso tudo, a procedure, é claro, tem uma seção executável identificada pelo parâmetro de instruções begin. . .end;". Verdadeiro ou Falso ?

2/6/2008

54

### Exercícios Propostos

- 12) “As variáveis só são visíveis, o que significa que você pode acessar essa variável, no bloco em que elas são definidas. Em procedures ou funções aninhadas, você sempre pode ver de dentro para fora. Isso significa que as procedures mais profundamente aninhadas podem ver as variáveis definidas em sua procedure-mãe e as que estão dentro desta etc.” Verdadeiro ou Falso ?
- 13) Crie um programa Console que calcule o valor de Delta da equação do 2º grau. Use uma função.

2/6/2008

55

### Bibliografia

- Cantù, Marco  
Dominando o Delphi 6,  
Makron Books, 2002
- Osier, Dan  
Aprenda em 21 dias Delphi  
Campus

2/6/2008

56