Web:
http://inf.unisinos.br/~osorio/jogos-gt.html

DarkBasic Pro

IDE = Integrated Development Environment

- Gerência e Configuração do Projeto (.DBPRO)
- Editor de Textos (programas .DBA)
- Compilador (.DPPRO, .DBA => .EXE)
- Depurador de Código (Debug, Passo-à-Passo, CLI)
- Help e Documentação On-Line
- Interface Integrada: Edit/Compile/Exec/Debug/Help
- Ferramenta de Desenvolvimento Rápido de Jogos 2D e 3D (RAD = Rapid Application Development)

Site: http://darkbasic.thegamecreators.com/

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

DBPro – Desenvolvimento de um Jogo

- Alteração de jogos já existentes (MODs) devido a disponibilidade do programa fonte!
- Criação de Jogos:
  - Programação em Basic
  - Estrutura dos Jogos: Laço Principal
  - Aprender a partir dos exemplos disponíveis
  - Disponibilidade de “medias”: Sons, Músicas, Objetos, Texturas, etc

(Diretório: C:\Program Files\Dark Basic Software\Dark Basic Professional\Media)

DarkBasic Pro

DBPro – Desenvolvimento de um Jogo

- Estrutura de um Jogo
  - Abertura
  - Definir as configurações iniciais
  - Carregar objetos e mídias
  - Laço principal do Jogo (**seqüência** ≠ processos concorrentes)
    - Controla jogador / veículo
    - Controla eventos: tiros, colisão, etc
    - Controla inimigos
    - Atualiza tela
    - Verifica estado do jogo: fases / final
  - Fim do Laço
  - Encerramento do Jogo / Exibir créditos

DarkBasic Pro

Programação no Dark Basic Pro

- **Linguagem Basic**:
  Beginner's All-Purpose Symbolic Instruction Code
- Programas fonte: <arquivo>.DBA
- DBPro Basic é um "*flavor*" de Basic (tipo específico) que guarda similaridade com o AppleSoft Basic, MSBasic, Basic MSX, VisualBasic, ...
- *Vantagem*: Linguagem muito simples de usar
- *Desvantagem*: Pouco estruturada, e conseqüentemente, permite criar códigos pouco claros (exige disciplina!)

DarkBasic Pro

Linguagem Basic – Componentes e Sintaxe

- Tipos de Dados, Variáveis e Constantes
- Operadores (Aritméticos, Lógicos e Relacionais)
- Comandos (núcleo da linguagem):
  - Entrada e Saída: comandos básicos [**Print, Input, Data, Read, Restore**]
  - Comandos: Desvio Incondicional [**Goto**] e Condicional [**If/Then/Else/EndIf**]
  - Sub-rotinas [**Gosub/Return**]
  - Comandos de Laço / Repetição [**For/Next, Do/Loop, Repeat/Until, While/EndWhile**]
  - Controle de execução [**Exit, End**]
- Comentários / Documentação [**Rem**]
- Funções (e.g. aritméticas, teclado, tela, arquivos, ...)
- Funções do usuário [**Function/EndFunction**]
- Extensões DBPro... Manipulação de Estruturas de Dados, Interação, Multimídia, Rotinas de Rede e Multiplayer, Gráficos 2D e 3D.

DarkBasic Pro

Linguagem Basic – Tipos de Dados, Variáveis e Constantes

- Não há distinção em Basic de nomes de variáveis e comandos escritos em maiúsculas ou minúsculas.

- As variáveis em Basic não precisam ser declaradas de forma explícita, onde podemos “sair usando” uma variável sem ter que declará-la.

- As variáveis em Basic possuem seu tipo usualmente definido pelo próprio nome da variável. Exemplos:

  A => Variável do tipo inteira
  A# => Variável do tipo real (ponto flutuante)
  A$ => Variável do tipo string (texto)
  A(x), A#(x), A$(x) => Variáveis do tipo vetor (array)

DarkBasic Pro

Linguagem Basic – Tipos de Dados, Variáveis e Constantes

Valores Constantes:

- Números Inteiros – Exemplos: 10, -125, -233000 (atenção para a precisão)
- Números Reais – Exemplos: 20.0005, 99.9, -5000.12, -9999.9991
- Strings – Exemplos: “A”, “Hello World!”, “1.23” , “” (max. 255 caracteres)

Tabela da Precisão dos Tipos de Dados:

INTEGER Range : –2,147,483,648 to 2,147,483,647
REAL Range : 3.4E +/- 38 (7 digits)
BOOLEAN Range : 0 to 1
BYTE Range : 0 to 255
WORD Range : 0 to 65535
DWORD Range : 0 to 4,294,967,295
DOUBLE INTEGER Range : –9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
DOUBLE FLOAT Range : 1.7E +/- 308 (15 digits)

GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Tipos de Dados, Variáveis e Constantes

Variáveis:

- Variáveis simples: inteiro (x), real (x#), string (x$)
  Não precisa declarar... pode sair usando. Exemplo:
  CoordX = 100 : A = 1 : A = A+1 : B = A + 10 : Arq$ = “Meu.X” : Versao# = 1.01
- Variáveis do tipo vetor/array: x(1), x(2), x(3), ...
  Precisa declarar o tamanho do array, usando DIM. Exemplo:
  DIM Dados(10) : Dados(1) = 10 : Dados(2) = 20 : ... : Dados(10) = 100
  DIM Nome$(3) : Nome$(1)=“Fulano” : Nome$(2)=“Beltrano” : Nome$(3)=“Ciclano”
  DIM Tabuleiro_Jogo_da_Velha(3,3)
- Variáveis com tipos definidos pelo usuário: “novos tipos”
  Podemos criar tipos novos simples ou mesmo agrupando dados, através do uso dos comandos “AS” e “TYPE”

DarkBasic Pro

Linguagem Basic – Tipos de Dados, Variáveis e Constantes

Variáveis – Novos Tipos de Dados. Exemplos:

```
BigNumber AS Double Float
BigNumber = 0.00000000000000000000000000000000000000000001
print BigNumber

TYPE DadosFunc
    Nome AS String
    Idade AS Integer
    Salario AS Float
ENDTYPE

DIM Funcionario(10) AS DadosFunc
Funcionario(1).Nome = "Fulano" : Funcionario(1).Idade = 45 : Funcionario(1).Salario = 10000.00
```

- Portanto, é possível definir tipos novos a partir dos tipos já existentes...
- Criar novos tipos agrupando tipos já existentes...
- E até mesmo criar novos tipos agrupando novos tipos!

GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Tipos de Dados, Variáveis e Constantes

Variáveis – Ponteiros. Exemplos:

- Variáveis do tipo ponteiro servem para armazenar endereços da memória
  Exemplo:

```
PtrMem AS Dword
Ptr = 0x00FF8820
print "Endereço: ", Ptr, " = ", *Ptr
```

Atenção: Você não deve ler, escrever ou alterar endereços da memória, sem ter um total controle do que está fazendo... O exemplo acima, pode gerar um erro (acesso indevido a memória) caso o referido endereço não seja de uso livre do usuário que está executando o programa!

- Existem comandos que irão permitir obter endereços válidos de memória como:
  GET BACKBUFFER PTR ()
  MAKE MEMORY (NroBytes)

DarkBasic Pro

Linguagem Basic – Tipos de Dados, Variáveis e Constantes

Variáveis – Criando um conjunto de dados com Data/Read.

- O comando Data permite que seja criada uma lista de dados constantes que poderão ser recuperados (lidos) através do comando *read*. Exemplos:

```
DATA 9, "nove", 9.0
READ A, A$, A#

Dias_Semana:
DATA "Seg", "Ter", "Qua", "Qui", "Sex", "Sab", "Dom"
Tam_Turmas:
DATA  30, 42, 12
DATA  23, 8, 32
DATA  10, 11, 12

RESTORE Dias_Semana
For C=1 to 7 : READ D$ : Print D$ : Next C
RESTORE Tam_Turmas:
For C=1 to 9 : READ TT : Print TT," "; : Next C
```


DarkBasic Pro

Linguagem Basic – Operadores: Aritméticos, Relacionais e Lógicos

Operadores Aritméticos, Relacionais e Lógicos:

- Aritméticos: + - * /
  Exemplo: b = a + 10 : c = a * b
- Relacionais: > < >= <= <> (=>, =<)
  Exemplo: a < 10
- Lógicos: and or not xor
  Exemplo: (a < 10) and (a > 1)
- Precedência: (e)

Funções matemáticas...

Sin, Cos, Sqrt, Exp, Abs, Int, entre outras
Vide Help => "Core Commands"

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos:

- Entrada e Saída: read, data, restore (visto anteriormente) // **print, input** // **wait**
  Exemplos:

```
input "Digite seu nome: ",Nome$
print "Olá ",Nome$, "!"
print "Como "; : print "vai "; : print "você ?" : input Status$
print "Pressione uma tecla para continuar.." : wait key
print "Clique no mouse para terminar!"       : wait mouse
```

  * Nota: Existem comandos específicos para leitura de teclas, mouse e mesmo joystick
  Ver no *help – core* e *input commands* (Consulte: cls, set cursor, inkey$(), scancode(), upkey(), shiftkey(), ...)

- Comandos de Arquivos: **open** to read/write, **write** file/*data*, **read** file/*data*, **close** file
  Exemplo:

```
Numarq =1, Nomarq$ = "teste.txt"
open to write Numarq, Nomarq$
write string Numarq, "Nome do usuário: " : write string NumArq, Nome$
close file NumArq
```

  * Nota: Os comandos save array e load array podem ser bastante práticos de usar (core)

GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Desvio – Incondicional, Condicional, Sub-Rotinas

- Desvio do Fluxo de Execução – Desvio Incondicional: **goto**
  Exemplo:

```
goto pula_prox_linha
  print "Não deve exibir esta mensagem!!"
pula_prox_linha:
  print "Esta mensagem deve ser exibida!"
```

- Desvio Condicional: **if / then / else / endif**
  Exemplos:

```
if  Idade >= 18  then  write "Pode Jogar..."
if  Idade <  18  then  write "Vá para casa!"
if  Idade >= 18
    print "Pode jogar..."
else
    print " Vá para casa"
endif
```

DarkBasic Pro

Linguagem Basic – Comandos (“*core*”)

Comandos Básicos: Desvio – Incondicional, Condicional, Sub-Rotinas

• Desvio Condicional: if / then / else / endif => Substituindo por **select / case**
Exemplo:

```
if Nivel_Jogo =1
  print “Nível Fácil selecionado”
  Speed = 10
endif  ... Substituir por Select / Case!

select Nivel_Jogo
    case 1 : print “Nível Fácil selecionado” : Speed = 10 : endcase
    case 2 : print “Nível Médio selecionado” : Speed = 20 : endcase
    case 3 : print “Nível Intermediário selecionado” : Speed = 40 : endcase
    case 4 : print “Nível Avançado selecionado” : Speed = 100 : endcase
    case 5 : print “Nível Expert selecionado” : Speed = 500 : endcase
    case default: print “Nível NULO selecionado” : Speed = 1 : endcase
endselect
```


GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Comandos (“*core*”)

Comandos Básicos: Desvio – Incondicional, Condicional, Sub-Rotinas

• Sub-rotinas: **gosub / return**
Exemplo:

```
goto Abertura:
REM Carrega e toca uma música em MP3
Inicio_Musica:
   load music “meu.mp3”,1
   play music 1  :  loop music 1
return
REM Termina a execução da Música
Fim_Musica:
    stop music 1
return
Abertura:
gosub Carrega_Tela_Abertura
gosub Inicio_Musica
rem gosub Executa_Jogo
gosub Fim_Musica
```

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Laços / Repetição

• Repetição sem condição / sem fim pré-definido: **do / loop**

Exemplo:

```
do
  input "Entre seu nome: ",Nome$
  print "Olá : ",Nome$
loop
```

• Laço com um número definido de repetições: **for / to / step / next**

Exemplos:

```
for n=1 to 100 : print n : next n
for contador = 1 to 10 step 2 : print contador : next contador
for contador = 10 to 1 step –1
    print contador
    if contador = 1 then print "FIM!"
next contador
```


GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Laços / Repetição

• Laço com uma condição de repetição: **while / endwhile, repeat / until**

Exemplo:

```
while x  < 10
      x=x+1
      print "X=",x
endwhile
```

```
repeat
   x=x+1
   print "X=",x
until  x >= 10
```

• Interrupção do laço de repetição: **exit / end / break**

Exit = Sai de um laço do tipo Do/Loop, While/EndWhile ou Repeat/Until (vai para "fora" do laço)
End = Termina imediatamente a execução do programa (volta ao sistema operacional)
Break = Interrompe a execução de um programa em modo de deputação (como um *breakpoint*)

Exemplo:

```
REM Após o exit vai para fim:
do
   input "Entre um valor: ",x
   if x = -1 then exit
loop
fim:
```

```
REM Termina execução no end
do
   input "Valor: ",x
   if x = -1 end
loop
```

```
REM Usar no modo DEBUG
do
   input "Valor: ",x
   if x = -1 then break
loop
```

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Funções Básicas / Funções Definidas pelo Usuário

• Funções Básicas: Executa um comando (sub-rotina) que usualmente retorna um valor.
Exemplos:

```
valorint = int( 5.1234 ) : print int ( 5.1234 ) : a = 5 * abs ( b ) : print rnd(100)
cls : print get time$() : if cl$() <> "" then print "Parâmetro: ", cl$() else print "Sem parâmetros"
```

• Funções Definidas pelo Usuário: Executa uma sub-rotina do usuário com retorno
Exemplo:

```
Function ExibeMsg (cx,cy,msg$)
    Set Cursor cx,cy
    print msg$
EndFunction
```

* Notas: **(1) Variáveis usadas nas funções são locais (isoladas do resto do programa);**
(2) Possuem de 0 a 255 parâmetros de entrada;
(3) Retornam um valor (funções podem retornar usualmente um inteiro, real, string) ou podem não retornar nada. Sempre vem acompanhadas de um "( )" independente do que entra e sai.

GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Funções Definidas pelo Usuário (cont.)

•Funções Definidas pelo Usuário: Executa uma sub-rotina do usuário com retorno
Exemplos:

```
Function Adivinha(valor as integer)
    if valor = rnd(1)
        result$ = "Acertou!"
    else
        result$ = "Errou!"
    endif
EndFunction result$

Function EsperaX
    cont = 0
    do   Tecla$ = Inkey()
         if  Tecla$ = "X" then ExitFunction
         print ".";  : cont = cont + 1
         if cont = 79 then print : cont = 0
     loop
 EndFunction
```

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Estruturas de Dados – Array: Listas, Filas, Pilhas

• Arrays – Listas: Insere dados no topo ou na base do array (lista)
Comandos:

Empty Array
Array Index to Bottom
Array Index to Top
Next Array Index
Previous Array Index
Array Insert at Top
Array Insert at Botton
Array Insert at Element
Array Delete Element
Array Count()
Array Index Valid()

GTJedi – Ferramentas RAD / RGD
Prof. Osório / Unisinos 2005-1

Tópico:
Ferramenta DarkBasicPro / Prog. em Basic

DarkBasic Pro

Linguagem Basic – Comandos ("*core*")

Comandos Básicos: Estruturas de Dados – Array: Listas, Filas, Pilhas

• Arrays – Filas: Insere dados no final da fila e retira do início da fila (QUEUE)
FILAS = FIFO (First In, First Out)
Comandos:

Array Index to Queue
Add to Queue
Remove from Queue

•Arrays – Pilhas: Insere dados no topo da pilha e retira do topo da pilha (STACK)
PILHAS = LIFO (Last In, First Out)
Comandos:

Array Index to Stack
Add to Stack
Remove from Stack