



Programando Jogos com Delphi (LÓGICA, ANIMAÇÃO, CONTROLE)

Autor: Antônio Sérgio de Sousa Vieira

Email: sergiosvieira@hotmail.com

Introdução

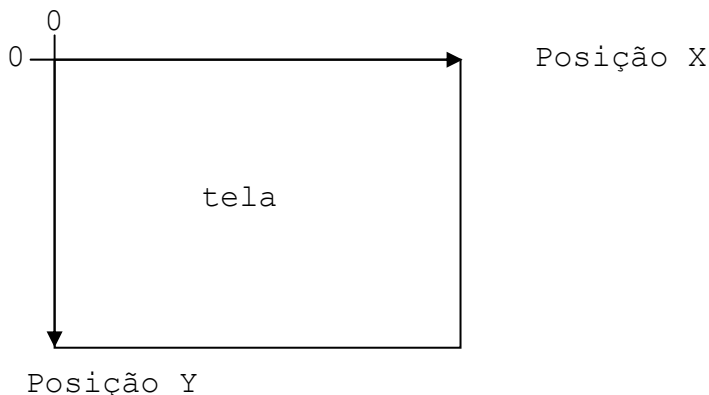
Este texto foi escrito para pessoas que se interessam pela arte da programação de jogos, uma das mais completas, pois utiliza a parte visual, sonora e interativa. Esse texto não tem como objeto de estudo a criação do enredo do jogo e sim a parte lógica da programação, utilizado-se das facilidades do Delphi. É de suma importância o leitor ter um bom nível técnico em programação e algum conhecimento de ciências exatas, pois serão vistos assuntos específicos que utilizam lógica, matemática e física.

No desenrolar da leitura serão abordadas questões relacionadas a manipulação de resoluções do vídeo, utilização de músicas e performance dos controles do jogo. Também será visto a parte lógica do comportamento do personagem principal e seus inimigos.

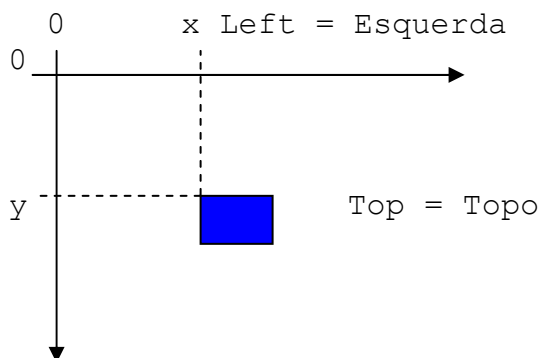
No primeiro exemplo, será criado um jogo de nave com personagem principal, inimigos, tiro, colisão etc. No segundo exemplo será explicado o desenvolvimento de um jogo estilo Mario Brothers, baseado em Blocos(tile based game).

1. Entendendo o Posicionamento do Personagem em Relação a Tela.

O cenário está dividido da mesma forma de um plano cartesiano invertido. Por ele podemos localizar a posição do personagem. Veja abaixo:



Para descobrir em qual posição está o personagem basta verificar as suas cordenadas X e Y.



Estas cordenadas também são responsáveis pela movimentação. Para movimentar basta incrementar ou decrementar do valor de x para movimentar na horizontal ou do valor y na vertical.

Exemplo:

```
Const  
  Velocidade = 1;  
  PosicaoX, PosicaoY: Integer;  
Begin  
  PosicaoX:=0; PosicaoY:=0;  
  Inc(PosicaoX, Velocidade); // Movimento para a direita  
  Dec(PosicaoX, Velocidade); // Movimento para a esquerda  
  Inc(PosicaoY, Velocidade); // Movimento para baixo  
  Dec(PosicaoY, Velocidade); // Movimento para cima  
End;
```

2. Entendendo o conceito de OffScreen ou BackBuffer.

É uma técnica utilizada para eliminar o "flick" (tremor) da animação, deixando o jogo com uma aparência melhor. Consiste em criar uma tela virtual na memória e depois desenhar todo o cenário (fundo, personagens etc) nela. Ao final, envia o cenário completo para o vídeo.

Para criar o OffScreen basta definir uma variável global do tipo TBITMAP, depois de instanciá-la (coloque o código no ONCREATE do formulário principal) defina valores para sua largura e altura, nunca esquecendo de destruí-la ao final do programa.

Exemplo:

```
(...)  
var OFFScreen: TBitmap;  
(...)  
  
procedure TForm1.FormCreate(Sender: TObject);  
begin  
    OFFScreen:= TBitmap.create;  
    OFFScreen.Width:= 320;  
    OFFScreen.Height:= 240;  
end;
```

3. Estrutura Principal do Jogo.

O código principal que será utilizado deverá estar dentro de um laço, para que seja possível controlar as animações e movimentação dos personagens. Veja abaixo um exemplo do código acima descrito:

Exemplo:

```
procedure TForm1.Button1onClick(sender:TObject);  
begin  
    while not Application.Terminated;  
    // Em quanto jogo não termina faça  
        begin  
            {CODIGO PRINCIPAL DO JOGO}  
            Application.ProcessMessages;  
            // utilizado para liberar processamento atual  
        end;  
end;
```

Descrição do Código: o código será iniciado ao clicar em um botão. Um laço funcionará até que o aplicativo seja fechado. Dentro do laço deve-se colocar os comandos de manipulação do jogo e liberar o processamento através do comando **Application.ProcessMessages;**

4. Definindo Características do Jogo.

Para facilitar a programação, deve ser criado alguns tipos que auxiliarão no desenvolvimento. É criado um tipo para o **PERSONAGEM** e outro para os **INIMIGOS**. Para os personagens atribui-se: Posição X (controla o movimento na horizontal), Posição Y (controla o movimento na vertical),

Vivo (tipo lógico, controla a existência dos personagens) e Sentido (cima,baixo,esquerda,direita e parado). Outros tipos podem ser criados conforme as características do jogo.

Exemplo:

Type

```
TSentido = (cima,baixo,esquerda,direita,parado);
```

Type

```
TNave = Record
```

```
  PosX: Integer; //Posição horizontal
```

```
  PosY: Integer; //Posição vertical
```

```
  Sent: TSentido; //Sentido do movimento
```

```
  Vivo: Boolean; //Controle de existência
```

```
End;
```

5. Movimentando o Personagem.

A movimentação deverá ser dividida em duas partes: a primeira busca capturar as teclas pressionadas e informar ao personagem em que sentido se movimentar, na segunda parte lê o sentido armazenada na variável "SENT" e INCREMENTA-SE ou DECREMENTA-SE um Valor(Velocidade) da Posição X (horizontal) ou Y (vertical).

Exemplo:

```
(...)  
Var Nave: TNave; //Definindo uma variável global do tipo  
TNAVE  
(...)  
  
procedure Ler_Teclado;  
begin  
  if GetKeyState(vk_left)<0 then //se tecla "seta  
esquerda" pressionada  
    Nave.Sent:= Esquerda // Movimento para esquerda  
  else  
    if GetKeyState(vk_right)<0 then // se "seta direita"  
pressionada  
      Nave.Sent:= Direita // Movimenta para direita  
    else  
      Nave.Dir:= Parado; // caso contrário, personagem  
parado  
  
  If GetKeyState(vk_up)<0 then //Idem (cima)
```

```

    Nave.Dir:= Cima //Idem
Else
    If GetKeyState(vk_down)<0 then //Idem (baixo)
        Nave.Dir:= Baixo; //Idem
end;

procedure Movimenta_Nave(Velocidade: Integer);
begin
    with Nave do
        begin
            if Sent = Esquerda then // Se SENT é igual
esquerda então
                Dec(PosX, Velocidade)// POSX igual POSX -
VELOCIDADE
            Else
                If Sent = Direita then // Se SENT é igual
direita então
                    Inc(PosX, Velocidade)// POSX igual POSX +
VELOCIDADE

                    if Sent = Cima then // Se SENT é igual cima então
                        Dec(PosY, Velocidade)// POSY igual POSY -
VELOCIDADE
                    Else
                        If Sent = Baixo then // Se SENT é igual baixo
então
                            Inc(PosY, Velocidade)// POSY igual POSY +
VELOCIDADE

                        end;
                    end;
        end;
end;

```

Observação: Não esqueça de colocar os dois procedimentos (ler_teclado, movimenta_nave) dentro do laço para que a movimentação ocorra corretamente.

6. Criando seu primeiro jogo no Delphi.

Após termos visto os conceitos iniciais podemos agora desenvolver um jogo de Nave. Note que ainda não existirá inimigos (será visto no próximo capítulo) e nem sprites. Nosso jogo será apenas um quadrado movimentando pela tela.

Siga os passos descritos abaixo para criar seu primeiro jogo.

- Crie tipo para controlar o sentido da movimentação.

```
Tsentido = (cima,baixo,esquerda,direita,parado);
```

- Crie um tipo para controlar a Nave, veja abaixo:

```
TNave = Record
  PosX: Integer; // Controla movimento horizontal
  PosY: Integer; // Controle movimento vertical
  Sentido: Tsentido;
  Vivo: Boolean;
End;
```

- Defina algumas constantes:

```
Const
  LARGURA = 320; // largura da tela
  ALTURA = 240; // altura da tela
  TAMANHO = 5; // Tamanho da Nave
```

- Defina uma variável global para o BackBuffer e uma do tipo TNAVE:

```
Var
  BackBuffer: Tbitmap;
  Nave: Tnave;
```

- No ONCREATE do formulário Crie o BackBuffer:

```
BackBuffer:= Tbitmap.Create;
BackBuffer.Width:= LARGURA;
BackBuffer.Height:= ALTURA;
```

- No ONDESTROY libere o BackBuffer:

```
BackBuffer.Destroy;
```

- Depois de definirmos todos os procedimentos iniciais vamos começar a programação da lógica do jogo. Primeiramente no ONDBLCLICK do formulário digite o laço principal do jogo:

```
while not Application.Terminated do
  begin
    Application.ProcessMessages;
  end;
```

- Defina agora alguns procedimentos (OBS: Coloque os procedimentos antes do BEGIN do procedimento ONDBLCLICK. Veja abaixo).

```
procedure TForm1.FormDbClick(Sender: TObject);
{ AQUI }
begin
```

- procedimento para ler o teclado;

```
procedure Ler_Teclado;
begin
  if GetKeyState(vk_left)<0 then
//se tecla "seta esquerda" pressionada
    Nave.Sent:= Esquerda
// Movimento para esquerda
  else
    if GetKeyState(vk_right)<0 then
// se "seta direita" pressionada
      Nave.Sent:= Direita
// Movimenta para direita
    else
      Nave.Sent:= Parado;
// caso contrário, personagem parado

If GetKeyState(vk_up)<0 then //Idem (cima)
  Nave.Sent:= Cima //Idem
Else
  If GetKeyState(vk_down)<0 then //Idem (baixo)
    Nave.Sent:= Baixo; //Idem
end;
```

- procedimento para movimentação:

```
procedure Movimenta_Nave(Velocidade: Integer);
begin
  with Nave do
    begin
      if Sent = Esquerda then
// Se SENT é igual esquerda então
        Dec(PosX, Velocidade)
// POSX igual POSX - VELOCIDADE
      Else
        If Sent = Direita then
// Se SENT é igual direita então
          Inc(PosX, Velocidade)
// POSX igual POSX + VELOCIDADE
```

```

        if Sent = Cima then
            // Se SENT é igual cima então
            Dec(PosY, Velocidade)
            // POSY igual POSY - VELOCIDADE
        Else
            If Sent = Baixo then
                // Se SENT é igual baixo então
                Inc(PosY, Velocidade)
                // POSY igual POSY + VELOCIDADE
            end;
        end;
end;

```

- procedimento para desenhar a nave:

```

procedure Desenha_Nave(x,y: Integer);
var i,ii: Integer;
// variáveis para controlar o desenho do quadrado
begin
    BackBuffer.Canvas.FillRect(Rect(0,0,LARGURA,ALTURA));
    // Limpando o BackBuffer
    for i:= 0 to TAMANHO do
        //desenhando os pixels da esquerda para a direita
        for ii:= 0 to TAMANHO do
            //desenhando os pixels de cima para baixo
            BackBuffer.Canvas.Pixels[i+x,ii+y]:= clBlue;
            // a posição horizontal do pixel será o valor de i + x
            // a posição vertical do pixel será o valor de ii + y
        end;
    end;

```

- Coloque os procedimento dentro do laço na ordem abaixo:

```

(...)
    Ler_Teclado;
    Movimenta_Nave(1);
    Desenha_Nave(Nave.PosX,Nave.PosY);
(...)

```

- Envie agora a imagem gravada no BackBuffer para a Tela (Dentro do Laço):

```

Canvas.Draw(0,0,BackBuffer);

```

- Finalmente inicialize as variáveis POSX, POSY desta forma:

```
(...)  
Initialization  
  Nave.PosX:= 0;  
  Nave.PosY:= 0;
```

```
End.
```

TCHARAMMMM!!! Seu primeiro jogo no Delphi. (quase)...