

Curso: Desenvolvendo Jogos 2d Com C# E Microsoft XNA

Conteudista: André Luiz Brazil

Aula 6: EXIBINDO E MOVIMENTANDO A SUA ESPAÇONAVE

META

Exibir e movimentar a espaçonave do jogador na tela do jogo.

OBJETIVOS

Ao final da aula, você deve ser capaz de:

1. Exibir a espaçonave na tela do seu projeto de jogo;
2. Movimentar a espaçonave pela tela do seu projeto de jogo.

PRÉ-REQUISITOS

1. Conhecer a ferramenta XNA Game Studio, conceito abordado na aula 2;
2. Possuir um computador com as ferramentas Visual C# e XNA Game Studio instaladas, conforme explicado na aula 3.
3. Conhecer os conceitos de classe e objeto, abordados na aula 4.
4. Ter iniciado o seu projeto de jogo com a ferramenta Visual C# e acrescentado a classe Espaçonave ao jogo, conforme visto na aula 5.

Introdução

Na aula anterior, iniciamos o nosso projeto de jogo espacial dentro da ferramenta Visual C#. Também incorporamos ao nosso projeto a classe espaçonave, vista na aula 3.

Já criamos a espaçonave do jogador. Ela possui um nome: “Falcão Vingador” e os atributos de energia e velocidade também já foram configurados. Precisamos agora fazer a espaçonave aparecer na tela do jogo.

Vamos desenhar a nossa espaçonave na tela utilizando o recurso de **texturas**.

Caixa de Ênfase

Textura é uma imagem utilizada para revestir os objetos criados em duas ou três dimensões dentro de um projeto de jogo.



Fim da Caixa de Ênfase

Veja na Figura 6.1 alguns exemplos de imagens que podemos utilizar como texturas para as espaçonaves do jogador e as naves inimigas:



Figura 6.1 – Imagens de aviões e espaçonaves

Fonte: Desconhecida

Curiosidade

As texturas também podem ser utilizadas para dar mais realismo a outros objetos criados dentro do jogo. Na Figura 6.2 podemos observar alguns exemplos de texturas que podemos usar para revestir objetos como paredes, portas e pisos criados dentro do jogo:

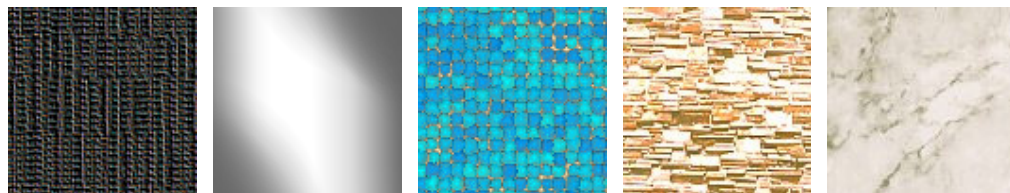


Figura 6.2 - Texturas de tecido, metal, pastilhas de vidro, pedras e mármore

Fonte: Desconhecida

Fim da Curiosidade

Exibindo a espaçonave na tela do jogo

Vamos rever a nossa classe `Espaçonave`.

```
public class Espaçonave
{
    private string _nome;
    private int _energia;
    private int _velocidade;
    private int _potencia_tiro;
    private int _velocidade_tiro;
    private int _escudo;

    public Espaçonave()
    {
        _energia = 50;
        _velocidade = 5;
        _velocidade_tiro = 5;
    }

    // Método de atualização do atributo _nome
    public void Nome(string p_nome)
    {
        _nome = p_nome;
    }

    // Método de acesso ao valor do atributo _nome
    public string Nome()
    {
        return _nome;
    }

    public void Mover(int direcao)
    {
        // Move a espaçonave na direção especificada
    }

    public void Atirar()
    {
        // Cria um tiro
    }
}
```

Passo 1: Analise agora os atributos da classe `Espaçonave`. Repare que ainda não temos nenhum atributo que possa armazenar uma textura para a nave. Criaremos então o atributo `_textura`.

Passo 2: Também precisamos armazenar a posição da espaçonave dentro da tela do jogo. Como estamos projetando um jogo em duas dimensões (x e y), esta posição pode ser definida por dois novos atributos: um para controlar a posição horizontal da nave (x) e outro para controlar a posição vertical (y). Podemos chamá-los de `_posicao_x` e `_posicao_y`.

Após acrescentarmos os novos atributos, a classe `espaçonave` ficará assim:

```
public class Espaçonave
{
    private string _nome;
    private int _energia;
```

```

private int _velocidade;
private int _potencia_tiro;
private int _velocidade_tiro;
private int _escudo;

// Novos atributos
private int _posicao_x;
private int _posicao_y;
private Texture2D _textura;

```

Passo 3: Observe agora o método de criação da classe `Espaçonave`. Tente agora definir a posição inicial da espaçonave na tela. Podemos fazer isso colocando valores para os atributos `_posicao_x` e `_posicao_y` que acabamos de criar. Veja como ficou o código:

```

public Espaçonave()
{
    _energia = 50;
    _velocidade = 5;
    _velocidade_tiro = 5;
    _posicao_x = 10;
    _posicao_y = 10;
}

```

Ok. Já criamos os três novos atributos e definimos valores iniciais de posição da nave. Como faremos para carregar a imagem da espaçonave como uma textura?

Passo 4: Vamos criar um novo método para a classe `Espaçonave`, chamado `CarregarTextura`. Veja a seguir, como ficou o código:

```

// Carrega uma textura a partir de um arquivo de imagem
public void CarregarTextura(GraphicsDevice p_dispositivo, string
p_local)
{
    _textura = Texture2D.FromFile(p_dispositivo, p_local);
}

```

Repare que o método `CarregarTextura` recebe dois parâmetros: **`p_dispositivo`** e **`p_local`**. O parâmetro **`p_dispositivo`** é o dispositivo gráfico onde a textura será exibida e o parâmetro **`p_local`** indica o caminho completo onde a imagem está localizada no computador. Se a imagem da nave do jogador estiver na pasta “imagens”, por exemplo, o caminho poderá ser “**`imagens\espaçonave_jogador.jpg`**”.

Note também que ele chama o método **`Texture2D.FromFile`**, para realizar a carga da textura de duas dimensões. A imagem localizada em **`p_local`** e será carregada como textura e ficará associada ao dispositivo gráfico **`p_dispositivo`**, para que possa ser exibida mais tarde.

O primeiro deles é um método para carregar a textura a partir de um arquivo de imagem, chamado `CarregarTextura`. Repare que utilizamos o comando

Texture2D.FromFile, de forma a buscar a textura a partir de um caminho de arquivo que foi informado, como por exemplo "..../Content/Imagens/nave.PNG".

Passo 5: Já temos a textura carregada e uma posição inicial (x e y) onde a textura aparecerá na tela. Vamos criar agora o método **Desenhar** da classe **Espaçonave** para desenhar a textura na tela do computador, na posição (x e y) especificada.

Para que possamos desenhar a textura da espaçonave na tela, precisamos delinear uma área na tela que será ocupada por essa textura. Esta área é chamada de **Sprite**.

Caixa de Ênfase

Sprite é a definição de uma área de tela para o carregamento de texturas.

Fim da Caixa de Ênfase

Veja como ficou o nosso método desenhar:

```
// Desenha a nave na tela utilizando um sprite
public void Desenhar(SpriteBatch p_sprite)
{
    p_sprite.Begin();
    p_sprite.Draw(_textura, new Rectangle(_posicaoX, _posicaoY,
    _textura.Width, _textura.Height), Color.White);
    p_sprite.End();
}
```

Este método recebe o parâmetro de entrada **p_sprite**, que é do tipo **SpriteBatch**. **SpriteBatch** significa “conjunto de *sprites*”. Dentro desses *sprites* colocaremos todas as texturas que serão exibidas dentro do nosso jogo espacial.

Os métodos **p_sprite.Begin()** e **p_sprite.End()**, iniciam e finalizam o processo de desenhar a espaçonave na tela.

Analisemos agora o método **p_sprite.Draw**. Este método irá desenhar a textura na tela. Ele recebe como parâmetros a textura da espaçonave, que é **_textura** e também a definição de um retângulo onde ficará desenhada a textura. O retângulo será desenhado nas posições **_posicaoX** e **_posicaoY** da espaçonave e terá a altura e a largura iguais às da textura que está sendo desenhada. Além disso, também é estabelecida uma cor de fundo branca (**Color.White**) para o desenho.

Atividade Prática 1 – Atende ao Objetivo 1

Seguindo os cinco passos descritos anteriormente, abra o seu projeto de jogo espacial na ferramenta Visual C# e altere o código da classe **Espaçonave** dentro do seu projeto de jogo para acrescentar a ele os atributos e os métodos necessários para carregar e desenhar as texturas.

Fim da Atividade Prática 1



Nossa classe Espaçoave já está preparada para exibir a espaçonave do jogador na tela do jogo. Precisamos agora encontrar uma imagem para a espaçonave do jogador. Vamos agora incluir a imagem da espaçonave dentro do nosso projeto de jogo.

Passo1: Dentro do **Solution Explorer**, localizado no canto direito do seu projeto de jogo, procure pelo item **Content** (Conteúdo). Em seguida, clique com o botão direito do mouse em cima desse item e selecione a opção **Add/New Folder**, para criar uma nova pasta chamada **Imagens**, conforme a Figura 6.3.

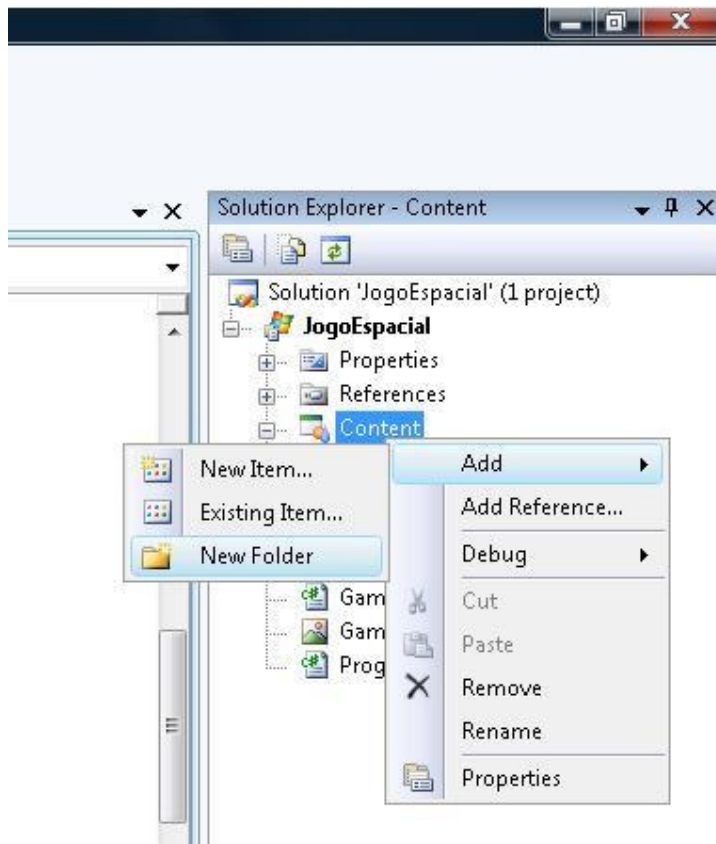


Figura 6.3 – Adicionando a pasta Imagens ao seu projeto de jogo

Fonte: Visual C#

Passo 2: Clique com o botão direito do mouse em cima da pasta **Imagens**, que você acabou de criar e selecione a opção **Add/Existing Item**, conforme a Figura 6.4.

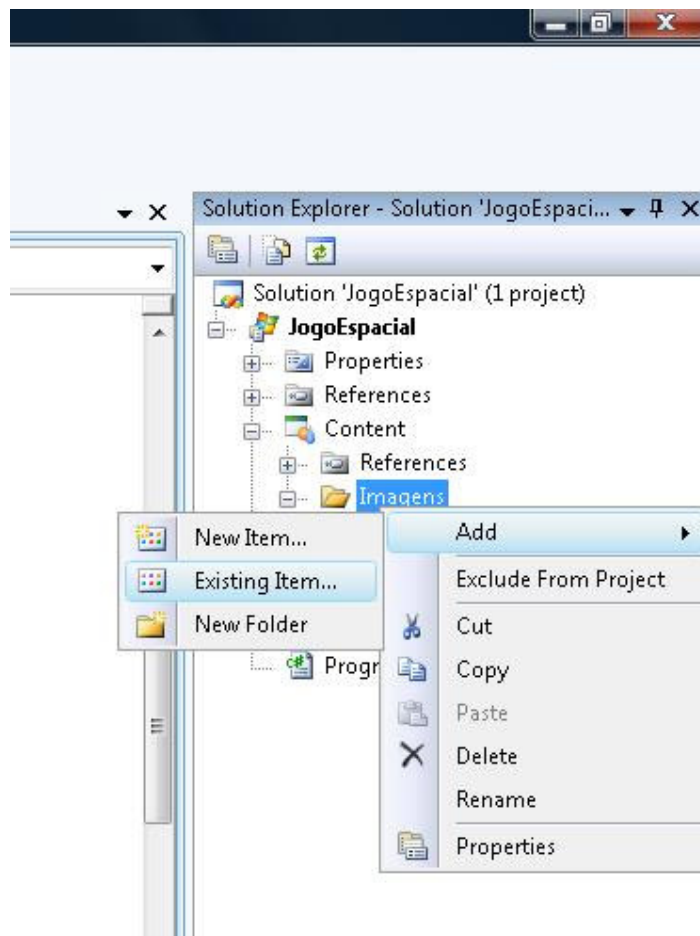


Figura 6.4 – Acrescentando a imagem da espaçonave na pasta Imagens

Fonte: Visual C#

Passo 3: Procure pelo arquivo de imagem da espaçonave, selecione este arquivo e clique no botão **Add**. Veja a Figura 6.5.

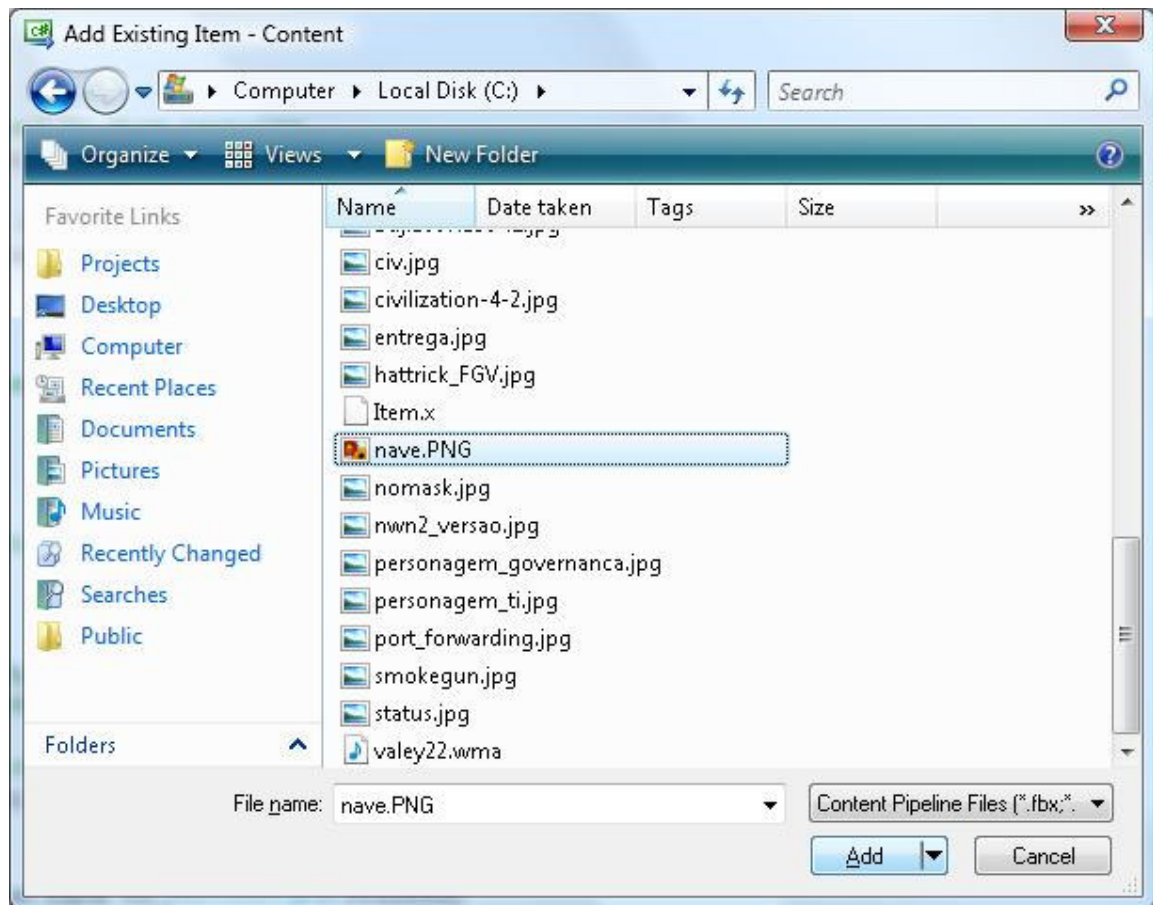


Figura 6.5 – Procurando e selecionando o arquivo de imagem da espaçonave

Fonte: Visual C#

Atividade Prática 2 – Atende ao Objetivo 1

Procure pela internet uma imagem para a espaçonave do jogador e acrescente-a ao seu projeto de jogo, seguindo os três passos descritos anteriormente. O ideal é esta imagem seja do tipo jpeg (.jpg) ou *portable network graphics* (.png). **Dica:** Para encontrar mais facilmente a imagem, utilize o aplicativo Cooliris: <http://www.cooliris.com/>.

Fim da Atividade Prática 2

Agora que já temos a imagem da espaçonave no projeto de jogo, vamos alterar o código do projeto de jogo para carregar essa imagem e exibir a espaçonave na tela.

Passo 1: Para carregar a imagem da espaçonave no projeto do jogo, chamaremos o método **CarregarTextura** da classe **Espaçonave**. Isto mesmo, este que acabamos de criar!

O melhor lugar para colocar essa chamada é dentro do método **LoadContent** do projeto de jogo, onde todo o conteúdo do jogo é carregado. Vejamos como ficou:


```

protected override void LoadContent()
{
    // Create a new SpriteBatch, which can be used to draw textures.
    spriteBatch = new SpriteBatch(GraphicsDevice);

    // TODO: use this.Content to load your game content here
    NaveJogador.CarregarTextura(graphics.GraphicsDevice,
    "../../../Content/Imagens/nave.PNG");
}

```

Passo 2: O último passo é fazer a espaçonave aparecer na tela, desenhando-a. Para isso, utilizaremos o método **Desenhar** da classe **Espaçonave**, que também criamos há pouco.

O local ideal para colocarmos esta chamada é dentro do método **Draw** do projeto de jogo, onde todos os objetos do jogo são desenhados. Veja como ficou:

```

protected override void Draw(GameTime gameTime)
{
    GraphicsDevice.Clear(Color.Black);

    // TODO: Add your drawing code here
    NaveJogador.Desenhar(spriteBatch);


    base.Draw(gameTime);
}

```

Atividade Prática 3 – Atende ao Objetivo 1

Abra o seu projeto de jogo espacial na ferramenta Visual C# e altere o código do seu projeto de jogo acrescentando a ele as chamadas para os métodos **CarregarTextura** e **Desenhar** da classe **Espaçonave**, seguindo os dois passos já descritos nesta aula.

Fim da Atividade Prática 3

Quer ver a espaçonave aparecer na tela do seu jogo? Para executar o código do seu projeto de jogo na ferramenta Visual C#, basta clicar no botão , localizado na barra superior de ícones, ou então apertar a tecla F5. A tela de jogo que vai aparecer será semelhante à Figura 6.6.

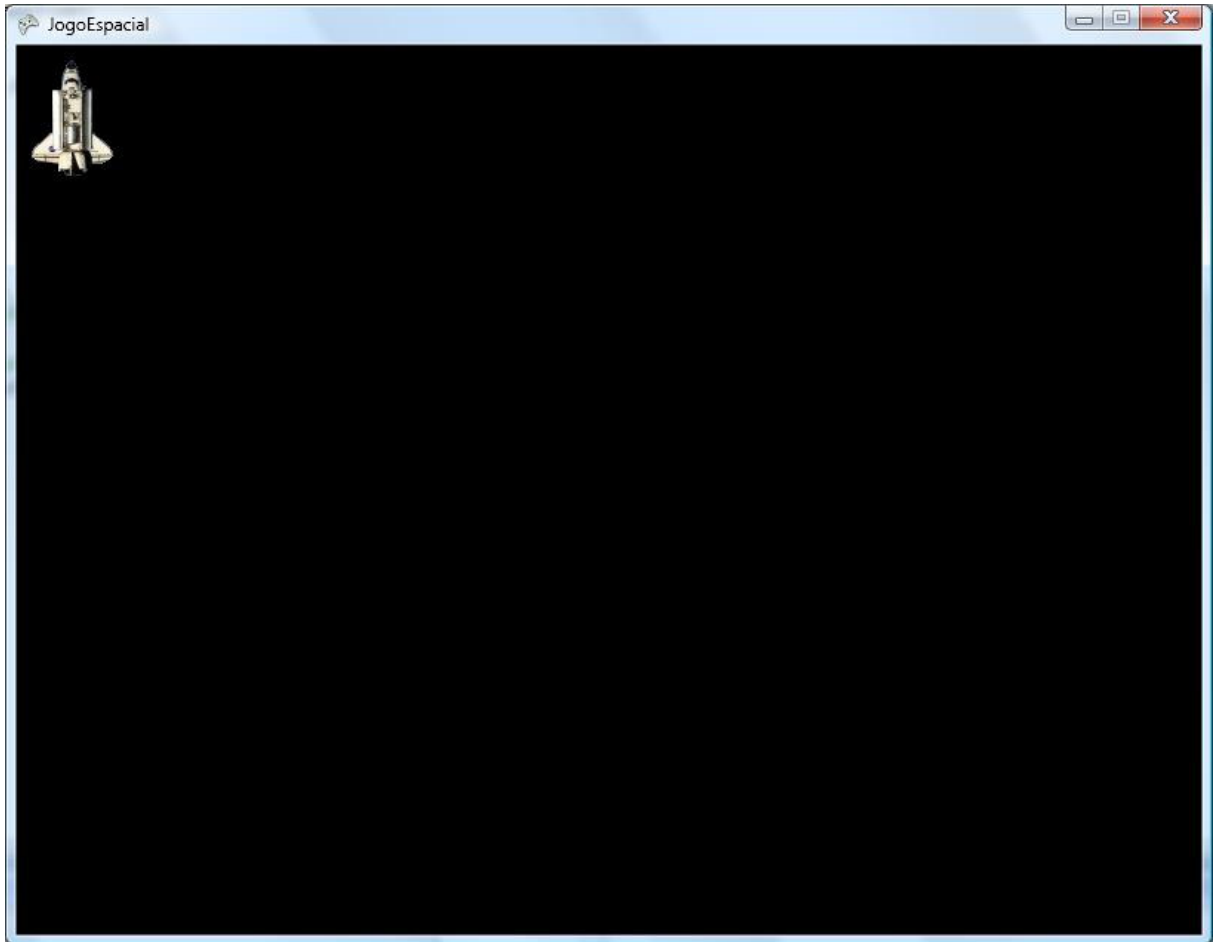


Figura 6.6 – Tela do jogo após a execução do código do projeto de jogo

Fonte: Visual C#

Atividade Prática 4 – Atende ao Objetivo 1

Abra o seu projeto de jogo espacial na ferramenta Visual C# e execute-o para ver se a espaçonave vai aparecer na tela do jogo. Caso apareça, altere o código do jogo para fazer a espaçonave do jogador aparecer em outra posição da tela, que não seja no canto superior esquerdo. Caso a espaçonave não apareça na tela, reveja o seu código do projeto de jogo e procure ajuda no fórum.

Fim da Atividade Prática 4

Movimentando a espaçonave pela tela do jogo

A espaçonave do jogador já aparece na tela do jogo? Excelente! Mas ela ainda permanece imóvel. Precisamos cuidar agora da movimentação da espaçonave do jogador.

Lembra-se do método **Mover**, que já existia na nossa classe **Espaçonave**?

```
public void Mover(int direcao)
{
    // Move a espaçonave na direção especificada
}
```

Exatamente, precisamos programá-lo para que a nossa espaçonave saia da inércia e ganhe movimentação pela tela do jogo.

Repare que o método **Mover** recebe um parâmetro, que é a direção para a qual a espaçonave deve se movimentar. Vejamos na Figura 6.7 quais são as possíveis direções para onde a espaçonave pode se mover:

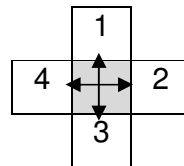


Figura 6.7 – Direções de movimentação

Fonte: Microsoft Word

A direção 1, por exemplo, significa ir para cima. A direção 2, para a direita, 3 para baixo e 4 para a esquerda.

Passo 1: Para realizar a movimentação da espaçonave, estaremos criando dentro do método **Mover** duas variáveis temporárias, que são o **deslocamento_x** e o **deslocamento_y**.

Caixa de Ênfase

Variável é um valor associado a um nome que criamos para identificar este valor.



Fim da Caixa de Ênfase

As variáveis temporárias **deslocamento_x** e **deslocamento_y** irão armazenar temporariamente números que significam o quanto a espaçonave irá se deslocar dentro da tela do jogo nos eixos x (horizontal) e y (vertical).

Vamos supor que desejemos deslocar a espaçonave para baixo, ou seja, na direção 3. Quais serão os valores que as variáveis **deslocamento_x** e **deslocamento_y** irão armazenar para executarmos esse deslocamento?

Analisando o desenho, podemos ver que o deslocamento para baixo (direção 3), é um deslocamento vertical. Desta forma, a variável **deslocamento_x**, neste caso irá ficar com o valor 0, pois não haverá deslocamento horizontal. Já a variável **deslocamento_y**, irá assumir o valor correspondente à velocidade atual da espaçonave, que é +5. Se fosse um deslocamento para cima, a variável **deslocamento_y** assumiria um deslocamento negativo, ou seja, -5. Vejamos então como ficará o código do método **Mover** da classe **Espaçonave**:

```
public void Mover(int direcao)
{
    // Variáveis temporárias
    int deslocamento_x, deslocamento_y;
    deslocamento_x = 0;
    deslocamento_y = 0;

    // Deslocamento para cima
    if (direcao == 1)
    {
        deslocamento_y = - _velocidade;
    }

    // Deslocamento para a direita
    if (direcao == 2)
    {
        deslocamento_x = + _velocidade;
    }

    // Deslocamento para baixo
    if (direcao == 3)
    {
        deslocamento_y = + _velocidade;
    }

    // Deslocamento para a esquerda
    if (direcao == 4)
    {
        deslocamento_x = - _velocidade;
    }

    // Executando o movimento
    _posicao_x = _posicao_x + deslocamento_x;
    _posicao_y = _posicao_y + deslocamento_y;
}
```

Após os comandos **if**, o movimento é executado, ou seja, os valores das posições x e y da espaçonave são alterados conforme o deslocamento.

Repare que essas movimentações todas não verificam algo bastante importante: se a espaçonave saiu da tela!

Passo 2: Podemos ajustar o método **Mover** para realizar esta verificação também. Veja como fica:

```
public void Mover(int direcao, int p_posicao_x_maxima, int
    p_posicao_y_maxima)
{
    // Variáveis temporárias
    int deslocamento_x, deslocamento_y;
```

```

deslocamento_x = 0;
deslocamento_y = 0;

// Deslocamento para cima
if (direcao == 1)
{
    deslocamento_y = - _velocidade;
}

// Deslocamento para a direita
if (direcao == 2)
{
    deslocamento_x = + _velocidade;
}

// Deslocamento para baixo
if (direcao == 3)
{
    deslocamento_y = + _velocidade;
}

// Deslocamento para a esquerda
if (direcao == 4)
{
    deslocamento_x = - _velocidade;
}

// Validando o movimento da espaçonave
if (_posicao_y + deslocamento_y >= 0 &&
    _posicao_x + deslocamento_x >= 0 &&
    _posicao_x + deslocamento_x <= p_posicao_x_maxima &&
    _posicao_y + deslocamento_y <= p_posicao_y_maxima)
{
    // Executando o movimento
    _posicao_x = _posicao_x + deslocamento_x;
    _posicao_y = _posicao_y + deslocamento_y;
}
}

```

Para realizar esta verificação, foram acrescentados mais dois parâmetros de entrada ao método, que são a posição x e a posição y máximas da tela. Desta forma, antes de executar o movimento, o código testa se o deslocamento irá ultrapassar os valores mínimos e máximos de x e y da tela e apenas realiza a movimentação da espaçonave se o movimento estiver correto, impedindo que a nave saia da tela do jogo.

Atividade Prática 5 – Atende ao Objetivo 2

Seguindo os dois passos descritos anteriormente, abra o seu projeto de jogo espacial na ferramenta Visual C# e altere o código da classe *Espaçonave* dentro do seu projeto de jogo para acrescentar a ele o código do método **Mover**.

Fim da Atividade Prática 5

A movimentação já está programada dentro da classe `Espaçonave`. Precisamos agora programar o jogo para que a movimentação seja realizada quando apertarmos as setas do teclado.

Passo 1: Vamos acrescentar aos nossos atributos do projeto de jogo um leitor para as teclas do teclado, o `KeyboardState`.

```
public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;

    // Definindo a espaçonave do jogador no jogo
    Espaçonave NaveJogador;
```

```
// Definindo um leitor de teclas do teclado
KeyboardState teclado;
```

Passo 2: Alteraremos também o método de atualização do jogo (**Update**). Dentro dele colocaremos toda a lógica de movimentação da espaçonave conforme as setas do teclado forem sendo pressionadas. Vamos utilizar o comando **Keyboard.GetState** para verificar o estado do teclado, ou seja, recuperar quais teclas foram pressionadas pelo jogador. Veja como ficou:

```
protected override void Update(GameTime gameTime)
{
    // Allows the default game to exit on Xbox 360 and Windows
    if (GamePad.GetState(PlayerIndex.One).Buttons.Back ==
        ButtonState.Pressed)
        this.Exit();

    // TODO: Add your update logic here
    teclado = Keyboard.GetState();
    if (teclado.IsKeyDown(Keys.Up))
    {
        NaveJogador.Mover(1, this.Window.ClientBounds.Width,
            this.Window.ClientBounds.Height);
    }
    if (teclado.IsKeyDown(Keys.Right))
    {
        NaveJogador.Mover(2, this.Window.ClientBounds.Width,
            this.Window.ClientBounds.Height);
    }
    if (teclado.IsKeyDown(Keys.Down))
    {
        NaveJogador.Mover(3, this.Window.ClientBounds.Width,
            this.Window.ClientBounds.Height);
    }
    if (teclado.IsKeyDown(Keys.Left))
    {
        NaveJogador.Mover(4, this.Window.ClientBounds.Width,
            this.Window.ClientBounds.Height);
    }

    base.Update(gameTime);
}
```

Repare também que são realizados diversos testes (**if**) para verificarmos se cada uma das setas do teclado foi pressionada. Para testarmos se o jogador apertou a seta para cima, por exemplo, usamos o comando **IsKeyDown(Keys.Up)**. Caso alguma das setas do teclado tenha sido apertada, veja que o método **Mover** da espaçonave do jogador é acionado e a direção do movimento é enviada para ele junto com os valores máximos das posições x e y da tela. Estes valores máximos são obtidos com os comandos **Window.ClientBounds.Width** (largura da tela) e **Window.ClientBounds.Height** (altura da tela).

Atividade Prática 6 – Atende ao Objetivo 2

Abra o seu projeto de jogo espacial na ferramenta Visual C# e altere o código do seu projeto de jogo acrescentando a ele o atributo leitor das teclas do teclado (**KeyboardState**) e as chamadas para o método **Mover** da classe **Espaçonave**, seguindo os dois passos descritos. Depois, execute o seu projeto de jogo e pressione as setas para verificar se a espaçonave está se movimentando corretamente.

Fim da Atividade Prática 6

Atividade Prática 7 – Atende aos Objetivos 1 e 2

Com base no conhecimento adquirido para criar e desenhar espaçonaves, abra o seu projeto de jogo espacial na ferramenta Visual C# e crie dentro do seu jogo uma espaçonave inimiga chamada “Soldado do Espaço” e programe a sua movimentação sempre na direção 3 (andar para baixo).

Fim da Atividade Prática 7

Atividade 8 – atende ao objetivo 2

Existe uma pequena falha no código de movimentação da classe **Espaçonave**. Quando a espaçonave movimenta-se para o canto direito ou inferior da tela, ela ainda some. Tente melhorar o método **Mover** da classe **Espaçonave** para que isto não aconteça. **Dica:** Utilize a altura e a largura da textura da espaçonave (**_textura.Width** e **_textura.Height**).

Fim da Atividade 8

CAIXA DE FÓRUM [Informação sobre Fórum](#)



Figura 6.8

Fonte: http://www.stockxpert.com/browse_image/view/28331341/?ref=sxc_hu (Jefferson- favor redesenhar)

Você teve alguma dificuldade para exibir ou movimentar a sua espaçonave no jogo? Entre no fórum da semana e compartilhe suas dúvidas e experiências com os seus amigos.

FIM DE CAIXA DE FÓRUM

CAIXA DE ATIVIDADE Informação sobre Atividade on-line



Figura 6.9


Fonte: <http://www.sxc.hu/photo/1000794> (Jefferson : favor redesenhar)

Agora que você já está com o seu código de projeto do jogo ajustado para exibir e movimentar a espaçonave, vá à sala de aula virtual e resolva as atividades propostas pelo tutor.

FIM CAIXA DE ATIVIDADE

Resumo

- Para exibirmos as espaçonaves dentro do jogo espacial, utilizaremos o conceito de texturas. Textura é uma imagem utilizada para revestir os objetos criados em duas ou três dimensões dentro de um projeto de jogo. Neste caso estaremos revestindo um retângulo com a imagem de uma espaçonave no formato jpeg ou PNG.
- A imagem da espaçonave pode ser exibida na tela do jogo. Para tal, é necessário carregá-la dentro de um **SpriteBatch**, ou seja, um conjunto de *sprites*, que são áreas de tela específicas para o carregamento de texturas.
- Em seguida, será necessário criar os métodos **CarregarTextura** e **Desenhar** na classe **Espaçonave**, para que possamos carregar a imagem da espaçonave e exibi-la na tela do jogo posteriormente. Também será necessário alterar o método **Draw** do projeto de jogo para chamar o método **Desenhar** da classe espaçonave durante a execução do jogo.

- Podemos programar a movimentação da espaçonave pela tela. Para isso precisaremos alterar o método **Mover** da classe **Espaçonave**, de forma que ele modifique as posições x e y da espaçonave de acordo com a direção do movimento e a velocidade atual da espaçonave.
- Também será necessário acrescentar um leitor de teclas do teclado, o **KeyboardState**, para verificar quais teclas foram pressionadas durante o jogo. Além disso, o método **Update** do jogo precisa ser modificado para levar em conta quais setas do teclado foram pressionadas e chamar o método **Mover** da classe espaçonave para executar a movimentação.
- Você pode executar um projeto de jogo utilizando o ícone , localizado na barra superior de ícones, ou então apertar a tecla F5.

Fim do resumo

Informações sobre a próxima aula

Na próxima aula, veremos como acrescentar tiros e um fundo de tela ao jogo.