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

Conteudista: André Luiz Brazil

Aula 3: CRIANDO A CLASSE ESPAÇONAVE

META

Mostrar como funciona a programação orientada a objetos

OBJETIVOS

Ao final da aula, você deve ser capaz de:

1. Conceituar atributo, método, classe e objeto;
2. Criar e codificar em C# uma classe para o seu jogo, com métodos e atributos próprios.

PRÉ-REQUISITOS

1. Conhecer a ferramenta XNA Game Studio, conceito abordado na aula 2**.**
2. Possuir um plano de desenvolvimento do jogo, construído na aula 2.

Introdução

Na aula 2, idealizamos um jogo e geramos o seu plano de desenvolvimento, um passo essencial para evitar problemas na hora de escrever o código desse jogo.

Ok! Vamos rever o nosso plano de desenvolvimento do jogo, produzido na aula 2.

Nome do Jogo: A Vingança de Anakam : 1 – O Az do Espaço

Categoria de aprendizado: Simulação (espacial)

Época: Futurista

Personagens: Anakam Spacewalker (jogador), Zark MoonVader (inimigo principal e Soldados do Espaço (inimigos do jogador no espaço).

Cenário do jogo: Espaço sideral, ao fundo diversas estrelas e corpos estelares. Ocasionalmente aparecerão planetas, conforme a movimentação da espaçonave do jogador.

Estória do jogo: Anakam, ao retornar de suas férias no planeta Kibokan, percebe que sua terra natal foi invadida e devastada por um ataque muito poderoso. Após conversar com alguns sobreviventes do ataque, percebe que tudo foi obra do malicioso e já conhecido Zark MoonVader, vilão inveterado das galáxias mais próximas e com seu domínio localizado no anel externo da galáxia Makabro, no planeta Snistrom. Decide então pedir ajuda ao seu melhor amigo e ambos, após alguns dias, reconstroem a espaçonave de deu pai Oki Abu, que havia sido ferido fatalmente tentando defender o seu planeta natal. Agora, equipado com a espaçonave de seu pai, a poderosa B-Wang z232, Anakam e seu amigo rumam em direção às terras do poderoso vilão, para vingar a morte de seu pai e a destruição de seu planeta natal.

Regras do jogo: A Espaçonave do jogador (Anakam) possuirá 50 pontos de vida iniciais e uma velocidade de 5 nós estelares. A potência do tiro dessa espaçonave será de 10 megatrons. A velocidade do tiro da espaçonave será de 5 nós estelares.

As espaçonaves dos soldados do espaço terão 10 pontos de vida, uma velocidade de 3 nós estelares e uma potência de tiro de 3 megatrons. A velocidade do tiro das espaçonaves será de 2 nós estelares, ou de 4 nós estelares para os soldados experientes.

Ao ser atingida, a espaçonave do jogador perderá os pontos de vida correspondentes à potência do tiro da espaçonave inimiga. Se o total de pontos de vida atingir o valor de 0 (zero) pontos ou menos, a espaçonave explodirá e o jogador perderá uma vida. O jogador possui um total de 3 vidas durante cada partida do jogo.

E agora, como faremos para começar a escrever o código desse jogo?

Repare que nas regras do nosso plano de jogo espacial, podemos verificar que a idéia de uma **nave** aparece duas vezes, uma para a espaço**nave** do jogador e outra para as **nave**s inimigas dos soldados do espaço.

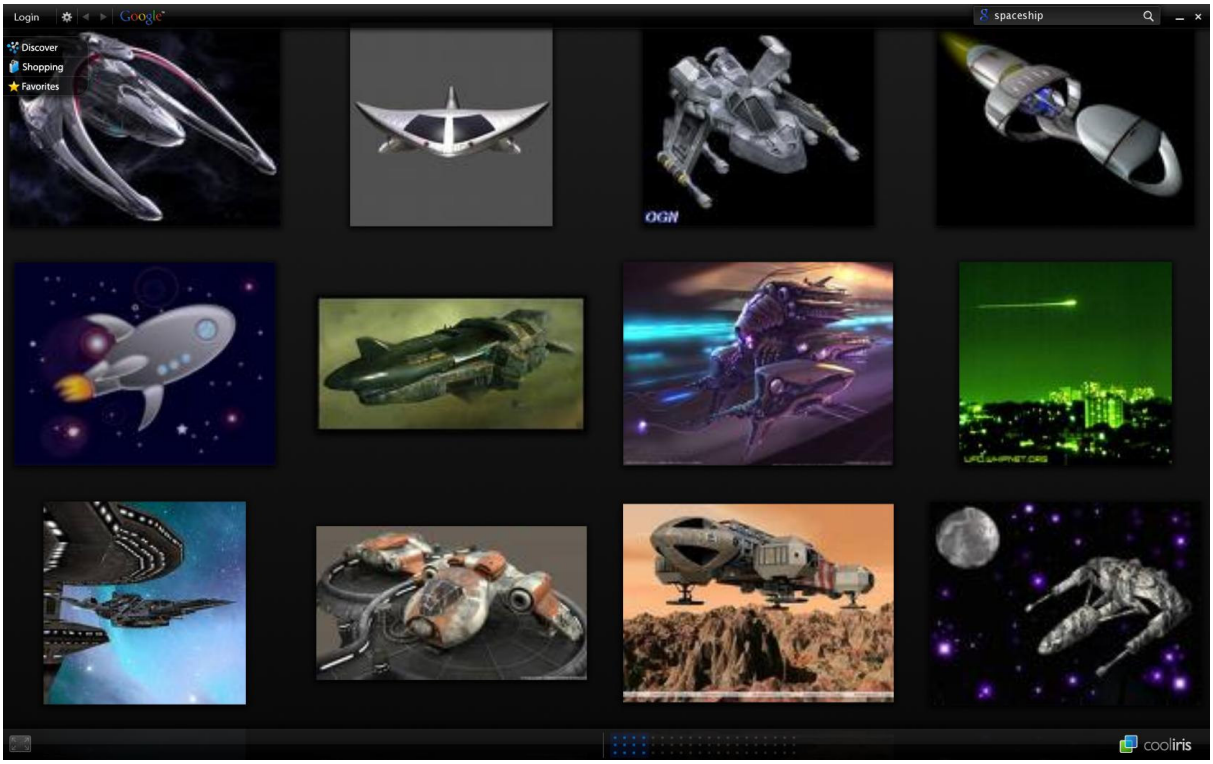

Figura 4.1 – Exemplos de Espaçonaves

Figura 4.1
Fonte: Cooliris
Legenda: Exemplos de Espaçonaves

Agora imagine como seriam essas espaçonaves. Ao comparar as duas espaçonaves, precisamos responder a algumas perguntas, como:

A espaçonave do jogador é mais forte ou mais fraca do que a dos soldados do espaço?

A espaçonave do jogador é mais rápida do que a dos soldados do espaço?

A espaçonave do jogador solta um tiro diferente daquela dos soldados do espaço?

A espaçonave do jogador e dos soldados do espaço são totalmente diferentes ou têm algo em comum?

Veja que ao respondermos as três primeiras perguntas, estamos “configurando” diferentes espaçonaves. A primeira pergunta fala sobre vida, a segunda sobre velocidade e a terceira sobre potência do tiro da espaçonave. Desta forma, observe que ambas as

espaçonaves possuem estas três características: vida, velocidade e potência do tiro. A estas características damos o nome de atributos.

Caixa de Ênfase

Atributo é uma característica encontrada em um ou mais objetos que desejamos acrescentar ao jogo.

Fim da Caixa de Ênfase

Perceba também, que além dos atributos, as espaçonaves irão executar ações durante o jogo. Pense em quais ações as espaçonaves poderiam executar: mover, atirar e levantar escudo são alguns exemplos. A estas ações executadas por ambas as espaçonaves damos o nome de métodos.

Caixa de Ênfase

Método é uma ação ou procedimento executado por um ou mais objetos que desejamos acrescentar ao jogo.

Fim da Caixa de Ênfase

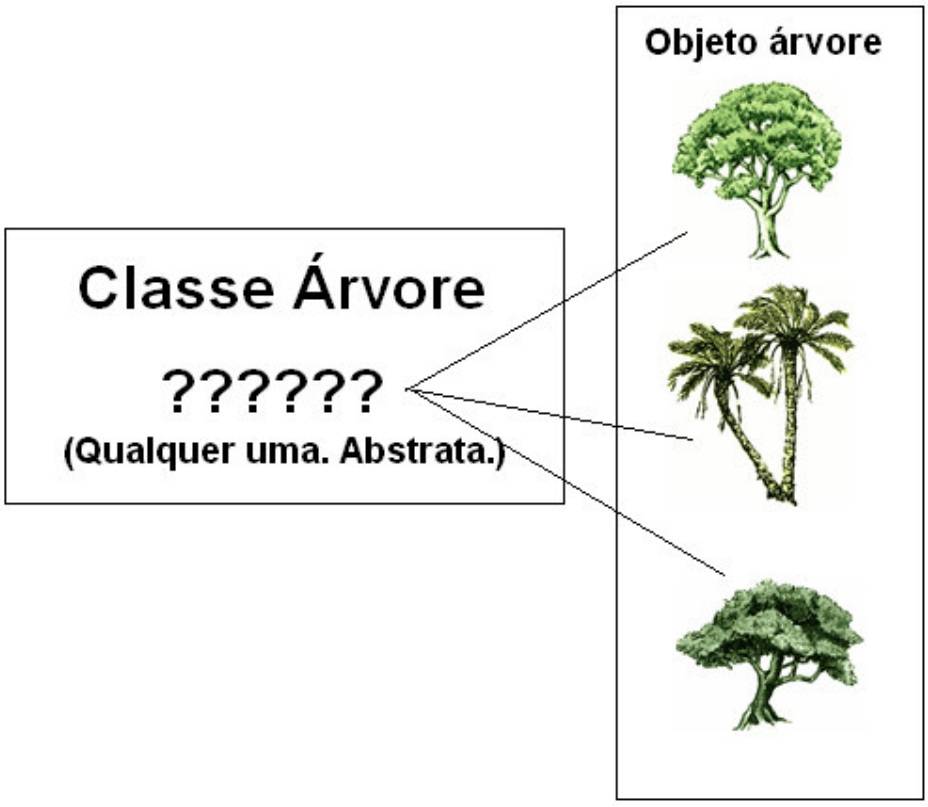

Figura 4.2 – Conceito de Classe

Figura 4.2

Fonte: http://www.linhadecodigo.com.br/artigos/img_artigos/robert_martim/image007.png

Legenda: Conceito de Classe

Veja que a quarta pergunta já foi respondida também. Ambas as espaçonaves possuem três atributos em comum, que neste caso são: vida, velocidade e potência do tiro. Os métodos mover, atirar e levantar escudo também são comuns a ambas. Quando isto acontece, podemos dizer que ambas as espaçonaves pertencem a uma mesma classe de objetos.

Caixa de Ênfase

Classe é a generalização de um ou mais objetos. Os objetos pertencentes à uma mesma classe possuem todos os atributos e métodos da sua classe.

Fim da Caixa de Ênfase

Podemos então criar uma classe de objetos chamada Espaçonave, que possui os atributos vida, velocidade e potência do tiro e os métodos atirar, mover e levantar escudo. A nave do jogador bem como as naves inimigas dos soldados do espaço são objetos que podem pertencer à classe de objetos Espaçonave.

Curiosidade

O conceito de orientação a objeto é bem antigo, e teve suas origens na década de 60, onde começou a se discutir o problema do desenvolvimento de aplicativos grandes mantendo um bom padrão de qualidade. A partir de então, começaram a surgir algumas linguagens que continham em si os conceitos de orientação a objeto. A linguagem C é uma delas.

Fim da Curiosidade

Atividade 1 – Atende ao Objetivo 2

Com base no seu plano de jogo que foi elaborado na aula 2, escolha um dos personagens e defina uma classe para ele, composta de pelo menos 5 atributos e 3 métodos. Escreva abaixo o nome da classe, os atributos e os métodos.

_____

_____

_____

_____

_____

_____

_____

_____

Fim da Atividade 1

Codificando a classe espaçonave em C#

Pois bem, uma vez que já definimos os atributos e métodos da nossa classe espaçonave, vamos escrevê-la utilizando a linguagem C#. Veja como podemos escrever o código da nossa classe espaçonave em C#:

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

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

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

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

Repare que a primeira palavra que aparece no código é **public**. O termo **public**, em português, significa público. Para o Visual C#, isto também indica que a classe Espaçonave é de domínio público, ou seja, estará disponível e poderá ser acessada em qualquer parte do código do jogo, ou ainda por outro programa.

Repare que na linha abaixo, após o comando “public class Espaconave”, existe a chave “{“. As chaves “{“ e “}” indicam o início e o fim de um trecho de código. Neste caso, estão indicando para o Visual C# onde começa e onde termina a nossa classe Espaçonave.

Preste atenção neste trecho agora:

```
private string _nome;
private int _energia;
private int _velocidade;
private int _potencia_tiro;
private int _velocidade_tiro;
```

```
private int _escudo;
```

Está reconhecendo alguns nomes de atributos que criamos? Exatamente! Este trecho define todos os atributos da classe Espaçonave. O termo **private**, ou privado, sinaliza para o Visual C# que todos os atributos da classe Espaçonave serão privados, ou seja, apenas os métodos criados dentro da própria classe Espaçonave poderão acessar estes atributos.

Perceba que as palavras **string** e **int** aparecem antes do nome dos atributos. Elas servem para indicar de que tipo são os atributos. O atributo **_nome**, por exemplo é do tipo **string**, ou seja, um conjunto de caracteres, para guardarmos o nome da espaçonave. Já o atributo **_energia**, é do tipo **int**, ou seja, só pode armazenar números inteiros. Este atributo vai nos dizer quantos pontos de vida a espaçonave ainda tem.

Agora observe este outro trecho:

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

A princípio, parece um tanto estranho colocar o nome Espaçonave novamente. Eis a explicação.

Caixa de Ênfase

Toda classe que criamos possui um método que tem o mesmo nome da classe. Este método é chamado de método de criação da classe.

Fim da Caixa de Ênfase

Dentro do método de criação da classe, colocamos valores iniciais para os nossos atributos mais importantes. Veja que os atributos energia, velocidade e velocidade do tiro receberam valores iniciais. Isto mostra que toda espaçonave que for criada dentro do jogo inicialmente possuirá 50 pontos de vida, 5 pontos de velocidade e 5 pontos na velocidade do tiro. É claro que estes valores poderão ser alterados mais tarde.

Por fim, veja o último trecho do código da nossa classe Espaçonave:

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

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

Está reconhecendo estes nomes? Sim! São os métodos **Mover e Atirar**, que criamos para a nossa classe Espaçonave. Logo após o nome **Mover**, existe um texto dentro dos parênteses, que é "**int direcao**". Para o Visual C#, isto quer dizer que para movermos a espaçonave, precisamos fornecer uma direção de destino, ou seja, um número inteiro (**int**) que indica em qual direção a espaçonave irá se mover quando este método for chamado. Chamamos estes valores de **parâmetros de entrada** do método Mover.

Existe também uma palavra estranha que vem antes do nome do método, que é o **void**. Em português, **void** significa vácuo! Isto quer dizer que o método Mover, ao ser chamado, não retornará um vácuo, ou seja, nenhum valor para quem o chamou. Apenas fará a movimentação da espaçonave.

Dentro dos métodos **Mover** e **Atirar**, existem textos na cor verde e com duas barras "//" antes. Quando colocamos estas duas barras antes do texto, mostramos para o Visual C# que o que estamos escrevendo são apenas comentários dentro do código da classe. Isto indica também que os métodos **Mover** e **Atirar** ainda não foram efetivamente codificados...

Atividade 2 – Atende ao Objetivo 2

Com base no exemplo de código apresentado acima, escreva na linguagem Visual C# o código para descrever a classe que você idealizou na atividade 1 desta aula. (Colocar 30 linhas).

Fim da Atividade 2

CAIXA DE FÓRUM **Informação sobre Fórum**

Figura 4.3
Fonte: http://www.stockxpert.com/browse_image/view/28331341/?ref=sxc_hu (Jefferson- favor redesenhar)

Você agora reconhece como funcionam as classes, os atributos e os métodos? 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 4.4
Fonte: http://www.sxc.hu/photo/1000794 (Jefferson : favor redesenhar)

Agora que você idealizou a sua classe de objetos, vá à sala de aula virtual e resolva as atividades propostas pelo tutor.

FIM CAIXA DE ATIVIDADE

Resumo

- Para transferir para o Visual C# a idéia do seu jogo, é importante compreender o conceito de classe. Imaginando um jogo composto pelas espaçonaves do jogador e inimigas, você pode generalizar o conceito de espaçonave e, com base neste, definir as características e as ações que toda espaçonave no seu jogo devem possuir;
- As características são os atributos;
- As ações possíveis são os métodos;
- Atributos e métodos, como um todo, compõem uma classe;
- A fim de representar as espaçonaves do jogo, podemos criar então uma classe chamada Espaçonave, com os atributos vida, velocidade e velocidade do tiro e os métodos mover, atirar e levantar escudo;
- Toda classe possui um método principal, que possui o mesmo nome da classe. Este método é o método de criação da classe e nele são definidos os valores iniciais para os atributos principais da classe;
- As palavras **public** e **private** indicam se os atributos ou métodos são públicos ou privados, ou seja, se podem ser acessados livremente (**public**) ou apenas dentro do código da própria classe onde foram criados (**private**).

Fim do resumo

Informações sobre a próxima aula

Na próxima aula, veremos como iniciar um projeto de jogo dentro da ferramenta que você acabou de instalar no seu computador, o Visual C#!