****

****

**Engenharia em Desenvolvimento de Jogos Digitais**

**Técnicas de Desenvolvimento de Jogos**

Trabalho realizado por:

**Eco Games**

Bruno Ferreira, 13227

João Martins, 13228

**INTRODUÇÃO:**

Este trabalho foi realizado no âmbito da introdução ao *monogame* e às suas diversas funcionalidades.

Ao longo deste trabalho, utilizamos o livro “Learn 2D Game Development with C#” como auxiliar, o que proporcionou uma melhor orientação.

**CÓDIGO:**

Funções inseridas nas classes *Game1* e *InputWrapper*:

Estas duas classes contêm funções que permitem:

-Alteração da janela do jogo;

-Alterar o jogo para *full screen*;

-Trabalhar com o sistema de coordenadas;

-Inserir imagens no projeto;

-Definir um comportamento para os objetos;

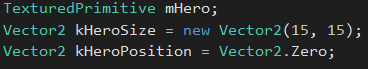
-Carregar conteúdo e fazer *update*;

-Permissão de utilização de diferentes mecanismos de jogabilidade (comandos de consola, teclado de computador, etc);

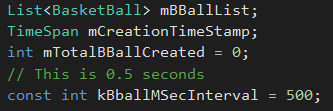
-Definir os controles do jogo.

Numa primeira fase, vamos apresentar o código inserido que está relacionado com o **3º capítulo (“***2D Graphics, Coordinates, and Game State)* do livro utilizado:

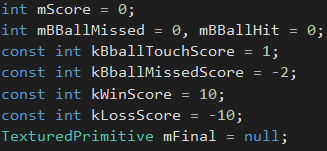
Funções inseridas na classe *MyGame*:



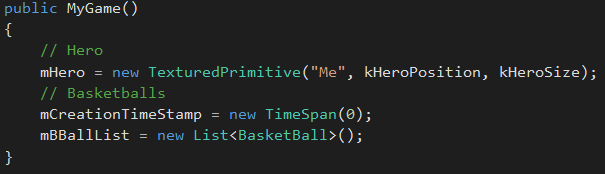
Coloca a personagem do jogo numa certa posição e com um certo tamanho.



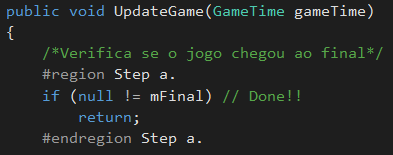
Criação de uma lista de bolas de *basket* e de seguida é indicado o intervalo de tempo (*timespan*) em que elas surgem. Ao mesmo tempo, criamos um valor inteiro para usar como apontador e adicionar à lista. Por último, adicionamos um número inteiro que é o total de bolas.



Declara variáveis que indicam o estado do jogo.

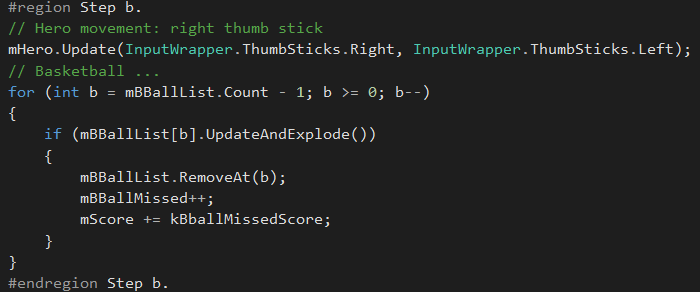


Inicializa a personagem com uma certa posição e tamanho e cria um tempo para surgirem as bolas. Para além disto, ainda contém uma lista para as bolas de *basket.*

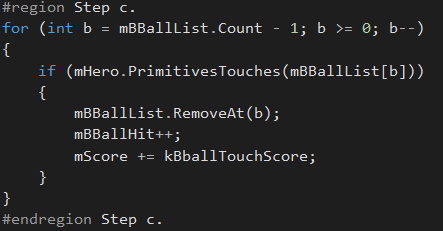


Criação de uma função que realiza *updates* no jogo, em que vão estar inseridos diversos métodos com diferentes objetivos.

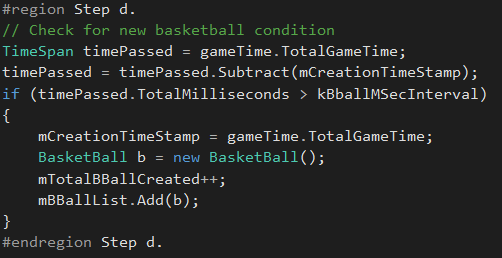
Primeiramente, encontramos esta função que vai verificando se o jogo chegou ao final.



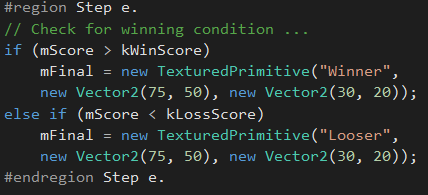
De seguida, foi criado um método que elimina as bolas que acabaram por explodir e vai atualizando o score.



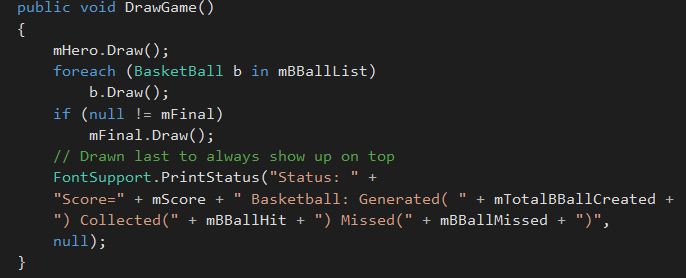
Método para remover as bolas que a personagem conseguiu apanhar e para atualizar o *score* do jogador.



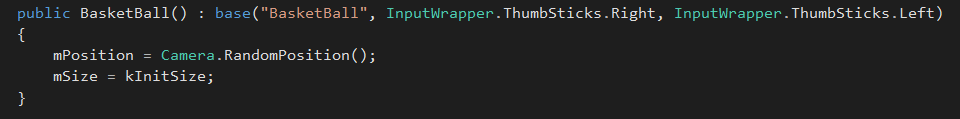
Faz com que apareçam novas bolas.



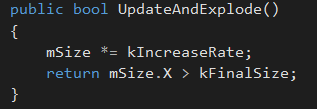
Verifica se o jogador ganhou ou perdeu.

Função que desenha a personagem e as bolas de *basket.*

Ao mesmo tempo, permite ao jogador a visualização do estado do seu jogo.

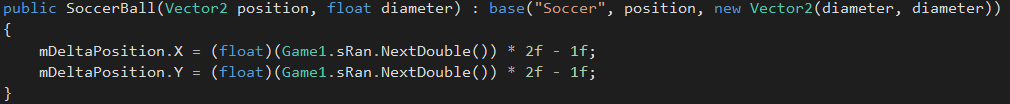
Funções inseridas na classe *Basketball: TexturedPrimitive*:

Permite que se gerem bolas com um tamanho mínimo e numa posição *random*.

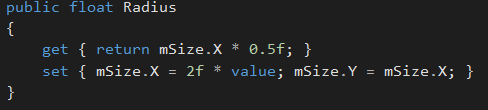


Faz com que a bola cresça e acabe por explodir quando chegar a um certo tamanho.

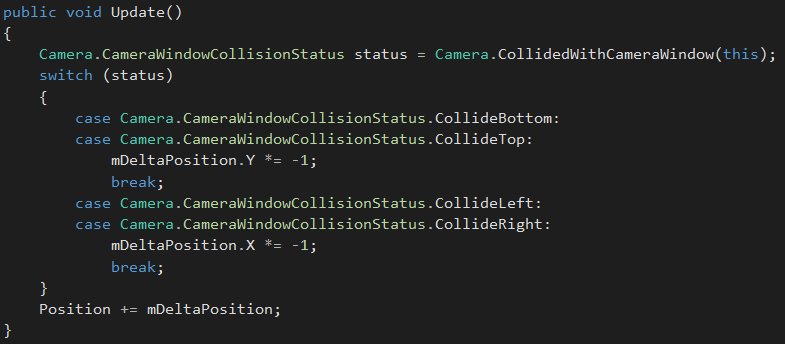
Funções inseridas na classe *SoccerBall: TexturedPrimitive*:



Passa a informação do tamanho e da posição da bola para a classe *TexturedPrimitive.*



Aumenta e diminui o tamanho da bola.

Define a localização da bola após colidir.

Nesta 2ª fase, apresentamos o código inserido no **4º capítulo** (“*Getting Things Moving”)* do livro.

Primeiramente, alteramos a classe *TexturedPrimitive*, para que o objeto se mova à volta da nossa janela e rode quer no sentido do ponteiro do relógio quer no sentido contrário.

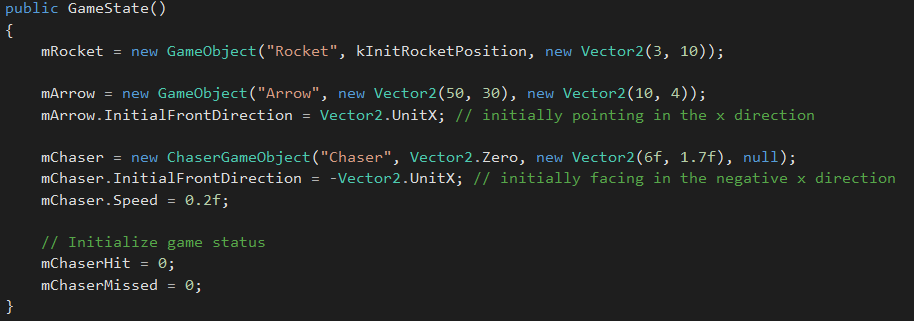
Após alterar a classe anterior, criamos uma classe *ShowVector* para representar vetores e manipular os mesmos dentro da janela.

Em suma, nesta classe podemos alterar qualquer atributo dos nossos vetores (direcção e tamanho por exemplo).

De seguida criamos a classe *FontDirection* que controla o “foguete”, nomeadamente a sua direção e a dos seus projeteis.

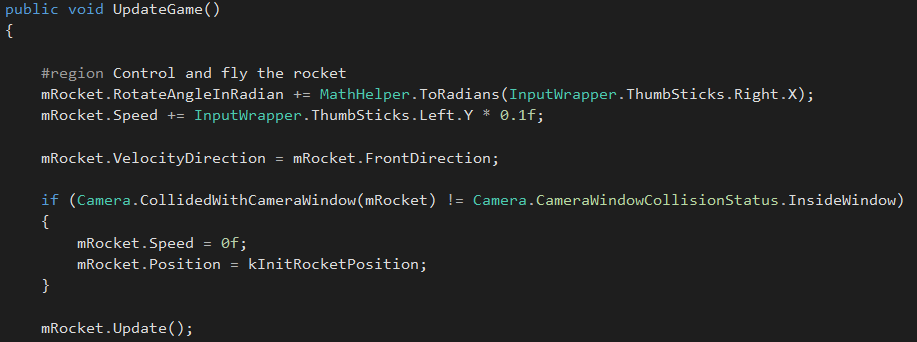
Após o sucedido, criamos a classe *GameObject* que nos permite controlar o objeto em questão, neste caso, o foguete. Ao mesmo tempo conseguimos controlar a velocidade e a sua direção.

Para finalizar, criamos a classe *ChaserGameObject* que faz com que os projeteis criados pelo projeto anterior sigam o nosso foguete, com a sua velocidade e direção.

****Funções inseridas na classe *GameState*:

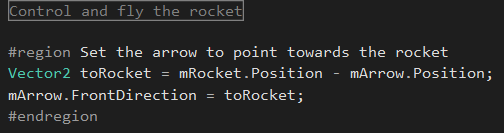
Criação de 3 objetos (*rocket, arrow e chaser),* que são inicializados com um certo tamanho. Ao mesmo tempo, a seta está direcionada para um certo sítio e o objeto *chaser* tem uma certa velocidade.

Por fim, inicializa-se o estado do jogo, em termos de colisões e não colisões.

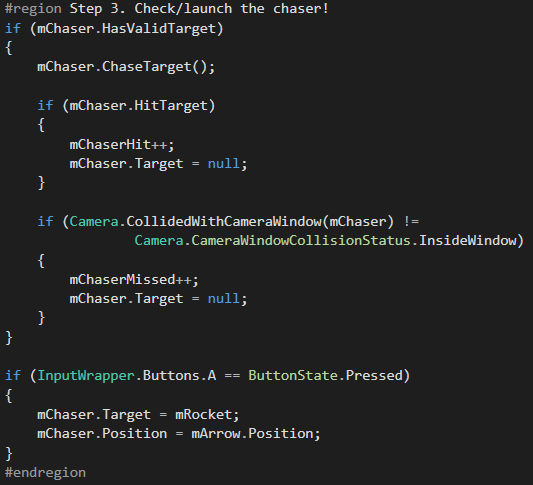
****Esta função vai atualizando o jogo.

Nesta primeira fase podemos controlar o *rocket.*

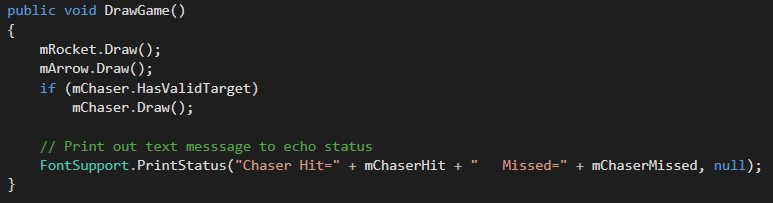
Podemos ainda observar que após uma colisão, o *rocket* para e volta à posição inicial.



Numa segunda fase, encontramos a função que faz com que a seta esteja sempre a apontar para o *rocket.*

**

Por fim, esta função permite que, após o jogador premir o botão, apareça um objeto que persegue o *rocket* desde a seta, e caso acerte ou não neste, atualiza o *score.*

****Em último lugar, dentro desta classe, temos a função que desenha o *rocket,* a seta e o objeto que vai perseguir o rocket.

Ao mesmo tempo, permite ao jogador a visualização do estado do seu jogo.

**ALTERAÇÕES FEITAS:**

Pelo livro, inicializando a personagem nas coordenadas (0, 0), esta não aparecia na janela que aparece inicialmente.

Portanto, alteramos para: “Vector 2 kHeroPosition = new Vector2(22, 30).” Assim, a personagem aparece quando o jogo inicializa.

**VANTAGENS:**

Devido aos diversos problemas que relatamos anteriormente, conseguimos ficar mais desembaraçados, ou seja, com uma maior eficácia na resolução de problemas.

Ao mesmo tempo, permitiu-nos desenvolver as qualidades cooperativas devido à constante conversação em relação ao projeto que fomos desenvolvendo.

**DIFICULDADES:**

De facto, as dificuldades foram algo muito presente ao longo do trabalho, isto porque o nosso conhecimento em C# é relativamente pequeno e perante erros temos alguma dificuldade em detetar o que pode estar na causa dos mesmos.

**CONCLUSÃO:**

Apesar de parecer um projeto muito simples, acabou por ser uma mais valia em termos de aprendizagem. Através da realização do mesmo aprendemos o básico para a criação de um jogo (controlo de câmara, colisão entre objetos e rotação dos mesmos, inserção de imagens e várias interações).

**BIBLIOGRAFIA:**

“Learn 2D Game Development with C”.