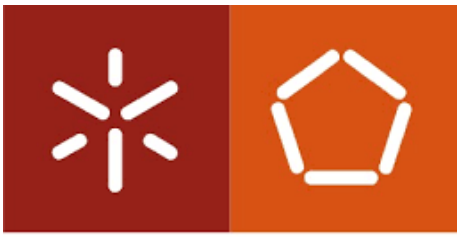
Relatório Projeto Poo

Grupo 87:
António Luís de Macedo Fernandes (a93312)
José Diogo Martins Vieira (a93251)
João Silva Torres (a93231)

June 12, 2021



Universidade do Minho

Contents

1 Introdução

No âmbito do desenvolvimento do projeto da unidade curricular Programação Orientada aos Objetos (POO) foi-nos proposto desenvolver, na linguagem Java, um programa semelhante ao Football Manager.

Ora, para que tal seja possível tivemos de introduzir e por em prática novos princípios de programação que nos foram ensinados nas aulas ao longo deste semestre como: o encapsulamento; a criação de classes abstratas, isto é, a classe Jogador. Foi como classe abstrata de modo a que o utilizador possa escolher o jogador numa dada posição.

Um dos principais objetivos foi o contínuo desenvolvimento das classes onde se encontram os Atributos dos jogadores. Numa fase introdutória optamos pela criação de classes com o tipo do Atributo, isto é, atacante, defensivo, entre outras. Classes estas que, após uma reflexão sobre a otimização, eficácia e simplicidade do código foram refeitas com a criação de classes mais simples e de fácil compreensão como o remate, jogo de cabeça, capacidade defensiva, resistencia, etc.

Em seguida faremos, de forma detalhada, a descrição da arquitetura de classes utilizada, isto é, classes, atributos, exceções, entre outras e também as decisões que foram sendo tomadas ao longo do desenvolvimento deste projeto. Juntamente iremos anexar um Diagrama de Classes com a arquitectura de classes que suporta o programa desenvolvido.

2 Classes do Trabalho

2.1 Atributos

Para se definir os diferentes atributos, utilizamos uma interface chamada Atributo que contém dois métodos: o valor e o clone. Para cada um dos seguintes atributos criamos uma diferente classe na qual definimos uma variável "valor" que contém o valor de 0-100 desse atributo.

Capacidade defensiva;

Capacidade de passe;

Cruzamento;

Destreza;

Impulsão;

Jogo de cabeça;

Motivação;

Posicionamento;

Remate;

Resistência;

Velocidade;

Ao associarmos uma interface Atributo a cada atributo, tornamos mais fácil a criação de possivéis novos atributos.

2.2 Jogador

Para definir os diferentes tipos de jogadores, criámos a classe abstrata Jogador na qual definimos as variaveis comuns dos vários tipos de jogadores. Nomeadamente, nome, número da camisola , os diferentes atributos e o histórico de equipas do respetivo jogador. Para os atributos definimos um mapa em que fazemos associar a cada key (percentagem para o calculo da habilidade) uma lista com os vários atributos com essa percentagem. Desta forma conseguimos calcular a habilidade geral de um jogador ficando responsável a cada subclasse a atribuição das diferentes percentagens para os atributos (definida na criação do tipo de jogador).

As subclasse que definimos foram:

2.2.1 Guarda-Redes

Nesta subclasse definimos as variaveis elasticidade e reflexos especificas para este tipo de jogador.

2.2.2 Defesa

Nesta subclasse definimos a variavel marcação e a variável lateral que se trata de um boolean que indica se se trata de um defesa central ou lateral.

2.2.3 Médio

Nesta subclasse definimos a variavel recuperação de bola e a variável lateral que se trata de um boolean que indica se se trata de um médio central ou lateral.

2.2.4 Avançado

Nesta subclasse definimos a variavel finalização e a variável lateral que se trata de um boolean que indica se se trata de um avançado central ou lateral.

2.3 Equipa

Nesta classe está definida uma equipa e para tal utilizámos uma variavel para o nome e um mapa em que faz associar a cada número da camisola o respetivo jogador.

2.4 Catalogo de Equipas

Esta classe é responśavel por armazenar todas as equipas e para tal definimos um mapa que faz associar a cada nome da equipa a respetiva equipa.

2.5 Jogo

Esta classe define um jogo entre duas equipas e para tal utilizamos como variáveis de instância, duas EquipaJogo, dois int para os golos de cada equipa, a data do jogo e um boolean que indica se já foi realizado ou não o respetivo jogo.

2.6 EquipaJogo

Nesta classe definimos a informação de uma equipa para um determinado jogo. Para tal fim, definimos uma variavel para o nome da equipa, o esquema tático, a lista com os número da camisola dos jogadores titulares que se encontram conforme as posições que vão jogar, um set com os números dos jogadores suplentes e uma lista com as substituições.

2.7 EsquemaTatico

Nesta classe está definido o esquema tático e para tal definimos três variáveis para o número de defesas, médios e avançados.

2.8 Substituição

Nesta classe definimos uma substuição, utilzando como variáveis de instância o número do jogador que sai, o que entra e se já foi realizada a substuição.

2.9 MomentoJogo

Esta classe é responsável para o cálculo de um momento do jogo. Para tal guardamos informação para a zona onde se encontra a bola, os golos das duas equipas (casa e fora) , a diferença do overall médio para as diferentes zonas e a equipa que possui a posse de bola.

Para tal definimos os seguintes enums:

PosseDeBolaCASA,FORA

Zonas ZONA1 (1), ZONA2 (2), ZONA3(3);

Acontecimentos PASSE_ZONA1, PASSE_ZONA2, PASSE_ZONA3, OPORTUNIDADE_GOLO

2.10 Parser

Nesta classe encontram-se definidos os metódos que realizam a leitura de ficheiros de texto, invocando o metodo parser dos respetivos jogadores, equipas e jogos.

3 Controller

Esta classe é responsável pela interação do utilizador com o nosso programa. Para tal pede ao Model sempre que for necessário aceder á estrutura de informação e ao View para imprimir no ecrã.

Os comandos que acrescentamos foram, simulação de jogo (escolha de titulares e substituições) , visualização das equipas disponivéis e respetivos jogadores (possibilidade da transeferência destes), a criação de jogadores ou equipas novas , possibilidade de guardar ficheiros e carregar ficheiros de texto(como o exemplo do logs.txt) e ficheiros anteriormente guardados pelo utilizador.

3.1 Model

Nesta classe encontram-se a lista dos jogos efetuados e o catalogo das equipas disponiveis.

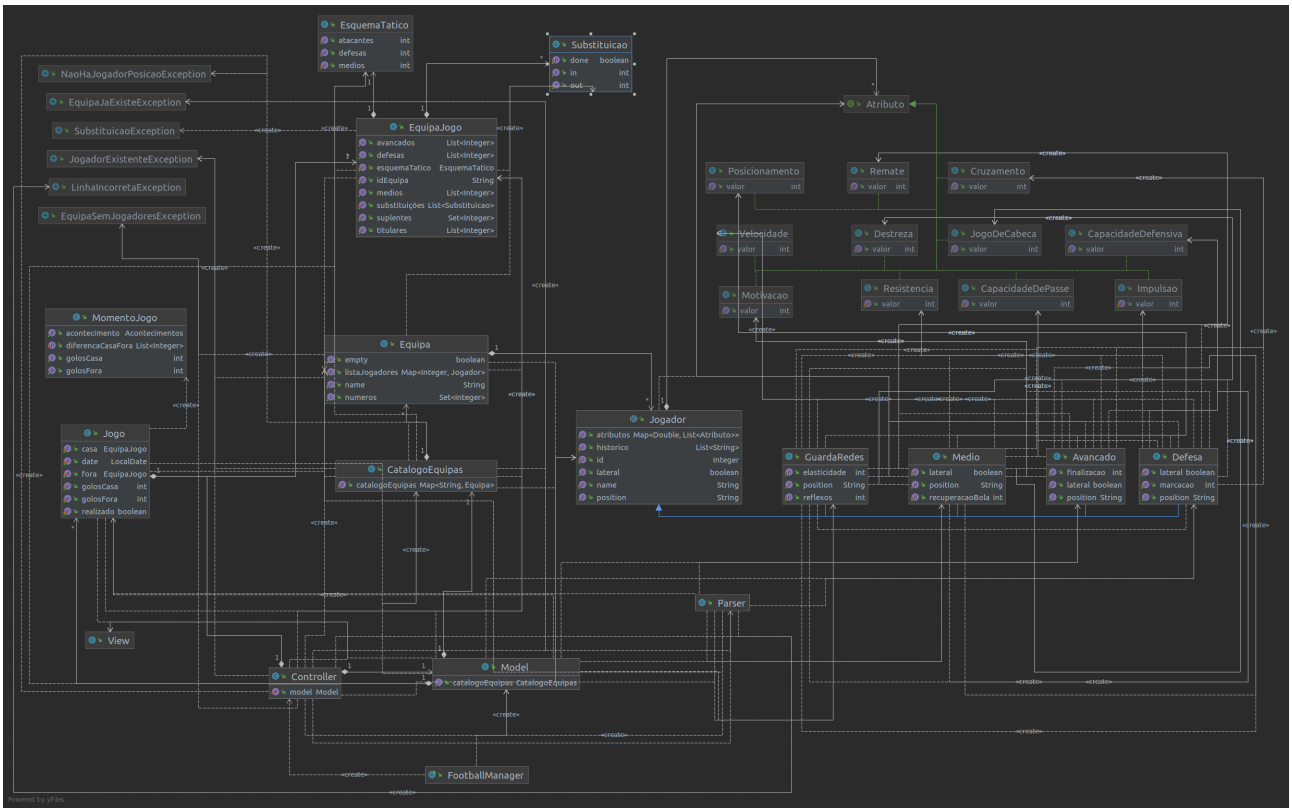
3.2 View

Aqui encontram-se os metódos de impressão no ecrã.

4 FootballManager

Aqui encontra-se a main do nosso programa que faz correr o nosso controlador.

5 Diagrama de classe



6 Conclusão

Nesta fase final do trabalho constatamos que, de uma forma geral, cumprimos todos os tópicos pedidos com sucesso: base de gestão das entidades; calcular o resultado do jogo; e efetuar a simulação do mesmo.

Dito isto, o nosso projeto foi pensado e executado de forma síncrona. Com efeito, o código é de simples compreensão, leitura e otimização e respeita todas as regras pedidas. Reconhecemos que poderíamos ter feito um melhor trabalho na parte da abstração das classes de maneira a uma possível corporação de novos tipos de desportos, equipas.

Assim, apesar disto cremos que o nosso trabalho está bem desenvolvido e otimizado e que, portanto, tivemos um bom aproveitamento global.

Concluindo, este projeto ajudou-nos a consolidar e desenvolver novas aptidões em programação orientada aos objetos (Java).