Instituto Politécnico de Beja
Escola Superior de Tecnologia e Gestão

Guia Prático 3 – Jogo do Galo – completo

João Paulo Barros

16 de março de 2021

1 Um Jogo do Galo

Neste guia é proposto concluir o programa, iniciado no GP2, de forma a permitir que dois jogadores utilizem o computador como um tabuleiro para jogar ao jogo do galo.

Deve fazer um novo projeto, eventualmente aproveitando partes do projeto do GP2.

Deve estruturar o código em duas partes principais. Estas irão corresponder a duas *packages*: (1) uma que trata da interface com o jogador e (2) outra que modela de facto o jogo. Esta é independente da interface com o utilizador. À primeira iremos chamar `pt.ipbeja.po2.tictactoe.gui` e à segunda `pt.ipbeja.po2.tictactoe.model`. Teremos assim a interface (*view+controller*) e o *model*. Toda a lógica (regras) do jogo e o seu estado (dados) deve estar nas classes que farão parte da *package* pt.ipbeja.po2.tictactoe.model. Tal significa que a interface com o utilizador não sabe nada das regras do jogo e apenas terá duas responsabilidades:

1. Comunicar ao model o que o utilizador fez (cliques,texto,etc.);
2. Executar os pedidos de atualização da interface que o model lhe faz.

2 Construção do programa com interface gráfica

Para obter uma primeira versão do programa com interface gráfica, deve seguir os seguintes passos:

1. Defina as novas packages com os nomes `pt.ipbeja.po2.tictactoe.gui` e `pt.ipbeja.po2.tictactoe.model`. Note que ambas devem ficar dentro da *package* `pt.ipbeja.po2.tictactoe`. Note também que todo o código (classes) deve agora ficar numa destas duas *packages*.
2. Crie uma classe `TicTacToeGame` dentro da *package* `pt.ipbeja.po2.tictactoe.model`.

3. A classe `TicTacToeGame` tem de saber tudo sobre o estado do jogo e sobre cada jogada que acontece. Para tal defina um `array` de `arrays` de `Mark` com o nome `board`. Cada `Mark` irá corresponder a um local no tabuleiro onde é possível colocar uma marca. Essa marca é um "X"ou um "O".

4. O tipo `Mark` deve ser um tipo enumerado (enum) com os valores `EMPTY, X_MARK, O_MARK`:

```
public enum Mark {
    EMPTY, X_MARK, O_MARK
}
```

   Nos seguintes links, encontra mais informação sobre tipos enumerados:

   - Tutorial oficial – `https://docs.oracle.com/javase/tutorial/java/javaOO/enum.html`;
   - Documentação – `https://docs.oracle.com/javase/8/docs/api/java/lang/Enum.html`.

5. Considere também um enumerado para os jogadores. Note que estes só podem ser dois: X e O e faz sentido que cada um devolva a sua marca. Para tal defina o seguinte enumerado, mais sofisticado, para o tipo `Player`:

```
public enum Player {
    X(Mark.X_MARK), O(Mark.O_MARK);

    private final Mark mark;

    Player(Mark mark) {
        this.mark = mark;
    }

    public Mark getMark() {
        return mark;
    }
}
```

   Note que neste tipo enumerado os dois valores possíveis (`X` e `O`) são definidos contendo um parâmetro do tipo `Mark`. Depois, uma variável do tipo `Player` pode devolver a marca correspondente. Por exemplo:

```
Player onePlayer = Player.O;
Mark playerMark = onePlayer.getMark();
```

   A variável `playerMark` irá ficar com o valor `Mark.O_MARK` porque é o que está guardado no objecto `onePlayer`.

6. Pode ser útil o `TicTacToeGame` ter uma função que informa qual o jogador que está a jogar (de quem é o "turno"). Tal pode ser feito com base na paridade ou imparidade do contador de jogadas (`turnCounter`):

```
public Player getCurrentPlayer() {
    return this.turnCounter % 2 == 0 ? Player.X : Player.O;
}
```

a = b ? c : d;
é o mesmo de
if (b)
  a = c
else
  a = d;

7. No *package pt.ipbeja.po2.tictactoe.gui* deve ser criada uma classe `TicTacToeBoard`. Cada objeto desta classe dever ter um atributo privado (`private`) do tipo `TicTacToeGame`.

```
private TicTacToeGame gameModel;.
```

8. Quando o jogador clica um botão na interface, o objecto da classe `TicTacToeBoard` deve informar o objecto `TicTacToeGame`. Para tal deve chamar um método positionSelected com a seguinte assinatura:

```
public void positionSelected(Position positionSelected)
```

Por exemplo, quando o botão é clicado deve ser executada a seguinte chamada:

```
gameModel.positionSelected(position);
```

9. O tipo `Position` é uma classe em que cada objecto guarda o valor da linha e da coluna. Deve pertencer ao *model* pois é independente da interface com o utilizador:

```
public class Position {

 private final int row;
 private final int col;

 public Position(int row, int col) {
     this.row = row;
     this.col = col;
 }
 // ...
}
```

10. Na classe `TicTacToeGame`, o método `positionSelected`, acima referido, deve actualizar o conteúdo do `board`. Para tal, deve mudar o valor nessa posição para `Mark.X_MARK` ou `Mark.O_MARK`.

11. Após actualizar o conteúdo do `board`, o método `positionSelected` (do `TicTacToeGame`) chama o método `onBoardMarkChanged` na classe `TicTacToeBoard`. Para tal terá de ter uma referência para o objecto dessa classe `TicTacToeBoard`. Este objecto deve ser do tipo `View`. O tipo `View` é uma interface que tem de ser implementada por todas as classes que recebem mensagens do `TicTacToeGame`. Assim, a classe `TicTacToeBoard` implementa a interface `View` o que significa definir todos os métodos nessa interface:

```
interface View {
   // ... methods to be called by TicTacToeGame object
   void onBoardMarkChanged(Mark place, Position position);
   // ...
}

class TicTacToeBoard implements View
{
    // ...
    void onBoardMarkChanged(Mark place, Position position) {
        // TODO
    }
    // ...
}

public class TicTacToeGame {
    //...
    private int turnCounter;
    private Mark[][] board;
    private View view;
    //...
}
```

12. Após actualizar o conteúdo do `board`, o método `positionSelected` deve pedir a outro método para verificar se o jogador ganhou OU se o jogo terminou com um empate. Em qualquer dos casos, deve também informar o objecto da classe `TicTacToeBoard` para este informar os jogadores sobre a vitória ou o empate. Para tal irá utilizar a mesma referência, do tipo `View`, para o objecto `TicTacToeBoard`. Para comunicar a vitória ou o empate utiliza os métodos `onGameWon` e `onGameWin`, respetivamente:

```
interface View {
    // ... methods to be called by TicTacToeGame object
    void onBoardMarkChanged(Mark place, Position position);

    void onGameWon(Player player);

    void onGameDraw();
}

class TicTacToeBoard implements View
{
    void onBoardMarkChanged(Mark place, Position position) {
        // TODO
    }

    void onGameWon(Player player) {
        // TODO
    }

    void onGameDraw() {
        // TODO
    }
}
```

3 Interface em modo texto

1. Agora adicione uma interface com o utilizador que funciona em modo de texto utilizando apenas o terminal. Para tal deve definir uma classe `pt.ipbeja.po2.tictactoe.tui` e lá colocar uma classe com o nome `TicTacToeTUI`.
2. A classe`TicTacToeTUI` deve utilizar a classe `java.util.Scanner` para ler jogadas do utilizador, e `System.out.print` e/ou `System.out.println` para mostrar o tabuleiro.
3. Note que a classe `TicTacToeTUI` irá também utilizar um objecto da classe `TicTacToeGame`. Para tal, deverá também conter um objecto dessa classe a quem pede para executar o método `positionSelected`, atrás referido.
4. Note que a classe `TicTacToeTUI` irá também receber informação do objecto da classe `TicTacToeGame` pelo que deve também implementar a interface `View`, definindo os métodos especificados na mesma.

4 Um jogo mais sofisticado

Nesta secção pretende-se criar uma nova versão do jogo. O objectivo continua a ser colocar três marcas em linha, mas agora essas marcas são como peças que se

podem colocar e depois mover para uma posição adjacente.

1. Considere que cada jogador pode optar por colocar uma peça nova (como no jogo tradicional) mas também pode optar por mover uma peça sua para uma posição adjacente.
2. Para mover uma marca terão de ser dados dois cliques: um na marca e outro na posição para onde pretende mover.
3. As interfaces com o utilizador, quer em modo de texto quer em modo gráfico, têm de permitir estes dois tipos de jogadas e impedir jogadas inválidas.

Deve terminar a resolução deste guia fora das aulas. Traga as dúvidas para a próxima aulas ou coloque-as no fórum de dúvidas da disciplina. As sugestões para melhorar este texto também são bem-vindas.