Departamento de Engenharia Informática e de Sistemas Instituto Superior de Engenharia de Coimbra *Engenharia Informática*

***Programação Avançada***

**2017/2018**



**9 Cards Siege**

Meta 2

## Grupo 53:

21230519 – [a21230519@isec.pt](mailto:a21230519@isec.pt) – Tiago Pimentel

# – Classes

## Package files

**FileUtility -** Classe que contém os métodos responsáveis por gravar/carregar o jogo de/para um ficheiro.

## Package logic

**Constants -** Classe que contém todas as constantes necessárias para o funcionamento do jogo.

**Day –** Classe responsável por encapsular todos os dados relativos a um dia no jogo.

**Die –** Classe responsável por simular o funcionamento de um dado.

**EnemyTracks -** Classe que contém as informações relativas ao inimigo. Simula a carta

*Enemy Tracks*.

**GameData -** Classe que contém todos os dados relativos ao jogo.

**GameModel -** Classe que contém o modelo de jogo, isto é, o estado atual e dados do jogo.

**PlayerTracks -** Classe que contém as informações relativas ao defensor (jogador). Simula a carta *Player Tracks*.

## Package logic.cards

**Card –** Classe "pai" abstracta que contém todos os dados e respectivos métodos comuns a todas as cartas.

**Card1 –** Classe que representa a carta #1.

**Card2 –** Classe que representa a carta #2.

**Card3 –** Classe que representa a carta #3.

**Card4 –** Classe que representa a carta #4.

**Card5 –** Classe que representa a carta #5.

**Card6 –** Classe que representa a carta #6.

**Card7 –** Classe que representa a carta #7.

## Package logic.states

**IStates –** Classe que contém o protótipo de todos os métodos inerentes à máquina de estados.

**AwaitAdditionalActionPointsSelection –** Classe responsável por gerir as transições que partem do estado *Await Additional Action Points Selection*.

**AwaitBeginning –** Classe responsável por gerir as transições que partem do estado

Await Beginning.

**AwaitGameFinish –** Classe responsável por gerir as transições que partem do estado

*Await Game Finish*.

**AwaitMovementTypeSelection –** Classe responsável por gerir as transições que partem do estado *Await Movement Type Selection*.

**AwaitOptionSelection –** Classe responsável por gerir as transições que partem do estado *Await Option Selection*.

**AwaitPlayerActionSelection –** Classe responsável por gerir as transições que partem do estado *Await Player Action Selection*.

**AwaitTopCardToBeDrawn –** Classe responsável por gerir as transições que partem do estado *Await Top Card To Be Drawn*.

**AwaitTrackSelection –** Classe responsável por gerir as transições que partem do estado *Await Track Selection*.

**StateAdapter –** Classe que contém a redefinição dos métodos que devolvem o próprio estado, de forma a facilitar a implementação da máquina de estados.

## Package ui.txt

**TextUI –** Classe responsável pela interface que faz a ligação entre a lógica e o utilizador.

## Package ui.gui

**ActionsPanel** – Classe responsável por conter todos os JPanels que se encontram na *East Border*.

**AwaitAdditionalActionPointsSelectionPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Additional Action Points Selection Panel*.

**AwaitBeginningPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Beginning*.

**AwaitGameFinishPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Game Finish*.

**AwaitMovementTypeSelectionPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Movement Type Selection*.

**AwaitOptionSelectionPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Option Selection*.

**AwaitPlayerActionSelectionPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Player Action Selection*.

**AwaitTopCardToBeDrawnPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Top Card To Be Drawn*.

**AwaitTrackSelectionPanel –** Classe responsável por criar a interface gráfica que desplota as transições que partem do estado*Await Track Selection*.

**DayPanel** – Classe responsável por conter todos os JPanels que se encontram na *North Border*.

**DeckPanel** – Classe responsável por conter todos os JPanels que se encontram na *West Border*.

**Images ­–** Classe responsável por carregar as imagens a serem utilizadas pelo método *paintComponent*.

**LogPanel** – Classe responsável por criar e mostrar um log de ações do decorrer do jogo.

**NineCardSiegeFrame** – Classe que representa a janela da aplicação.

**Resources –** Classe que providencia os métodos para carregar as imagens a serem utilizadas pela aplicação.

**SouthPanel** – Classe responsável por conter todos os JPanels que se encontram na *South Border*.

**TracksPanel** – Classe responsável por conter todos os JPanels que se encontram na *Center Border*.

## Package ui.gui.images

Esta package contém os ficheiros das imagens a ser utilizadas na aplicação.

# – Funcionalidades Não

Implementadas

Não foram implementados os eventos das cartas, exceptuando os “*Trebuchet Attacks”.*

Todas as restantes funcionalidades foram implementadas com sucesso.

# – Máquina de Estados

O modelo de máquina de estados utilizado para auxiliar a implementação deste projecto encontra-se num ficheiro em anexo.