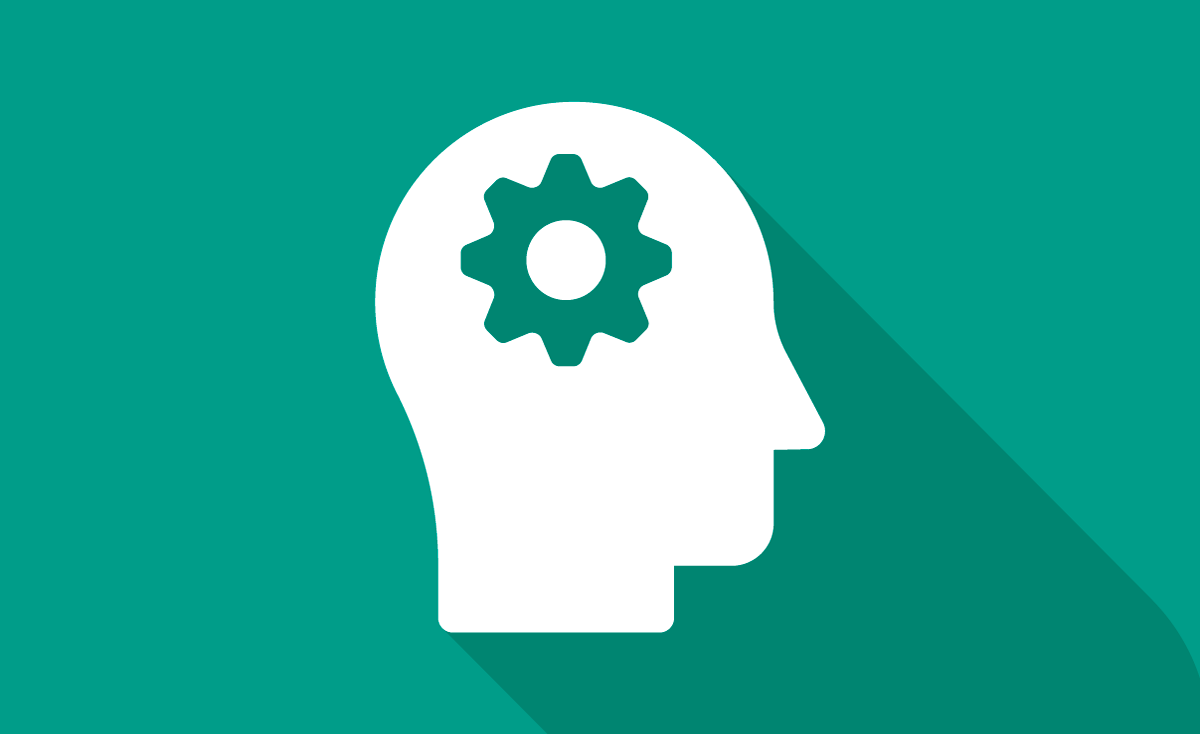


**Programação**

**Orientada a Objetos**

**POKOBAN**

****

**logotipo do projeto**

**Índice**

1. Grupo de Trabalho 3

1.1 Declaração: 4

2. Descrição do projeto 5

2.1 Definição do SOKOBAN 5

3. Arquitetura da aplicação 6

3.1 Constituição e descrição do projeto 6

3.2 Diagrama de Classes 9

4. Manual do Utilizador 10

5. Limitações e Desenvolvimentos Futuros 11

6. Conclusão 11

6.1 Conclusão do Alexandre 11

6.2 Conclusão do Bruno 11

7. Referências 12

# Grupo de Trabalho

|  |  |
| --- | --- |
| FOTO | Número: 80662  Nome: Alexandre Santos Coelho  Curso: Licenciatura em Engenharia Informática  Turma: C  Email: [alexfratzsantos@outlook.com](mailto:alexfratzsantos@outlook.com) |
| FOTO | Número: 13370  Nome: Bruno Lemos  Curso: Licenciatura em Engenharia Informática  Turma: C  Email: [brunomina1997@hotmail.com](mailto:brunomina1997@hotmail.com) |

## Declaração:

Os alunos declaram sob compromisso de honra que o projeto final é original e da sua autoria com exceção dos seguintes elementos:

|  |  |
| --- | --- |
| **Descrição** | **Fonte** |
| Biblioteca de parsing para JSON | Diretório de Maven |
| Imagens usadas para boneco, fundos, e objectos | Pesquisa no Google |

# Descrição do projeto

## Definição do SOKOBAN

O trabalho pratico desta unidade curricular consistiu no desenvolvimento de uma aplicação em linguagem Java, esta aplicação tem como objetivo permitir que o seu utilizador possa jogar um clone do jogo SOKOBAN.

O SOKOBAN é jogado num tabuleiro de quadrados, onde cada quadrado pode ser zona livre (referida como chão, onde o jogador pode navegar) ou limites (zona obstruída, geralmente são obstáculos ou paredes). O jogador irá estar dentro dos limites (que tem como principal função manter o jogador dentro do tabuleiro), apenas se pode movimentar horizontalmente ou verticalmente, existem alguns obstáculos que são movíveis, é através da interação com estes obstáculos que o jogador poderá completar o puzzle (objetivo principal do SOKOBAN). Para completar estar puzzle existem dois “marcadores” distintos, os obstáculos movíveis e os quadrados de marcação, estes são quadrados do tipo zona livre, que se diferenciam pelo seu aspeto e também pela sua função.

Para completar o jogo o jogador deve mover todos os obstáculos movíveis para cima dos quadrados de marcação, apenas quando todos os quadrados de marcação forem sobrepostos por obstáculos movíveis é que o puzzle se considera completo.

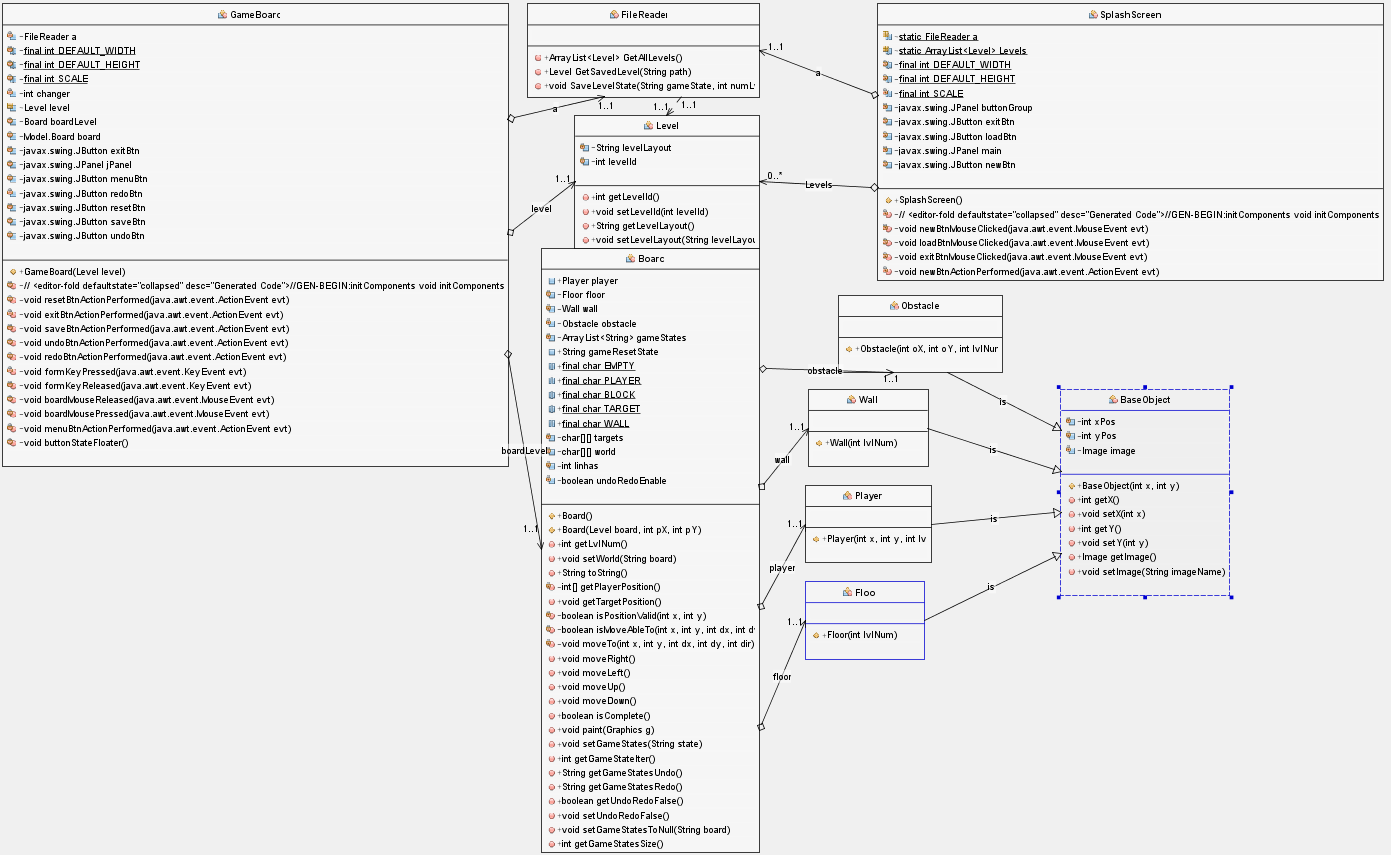
# Arquitetura da aplicação

## Constituição e descrição do projeto

Na solução do projeto resultou uma aplicação com o nome de “POKOBAN”, decidimos este nome porque o tabuleiro tem uma arte de imagens parecida com a do jogo POKEMON que se jogava num gameboy na altura da nossa infância. Esta aplicação está organiza em cinco *packages* principais, o vou passar a explicar o conteúdo e função de cada um em baixo:

* *Package* “Controlers” – neste package deverão ser contidas todas as classes que controlem as comunicações/interações com o exterior do ambiente (neste caso, a aplicação).
  + Classe “FileReader.java” – esta classe tem como função criar, escrever, e ler ficheiros, neste caso, o escopro de ficheiro com que esta classe interage é reduzido a dois ficheiros, ambos com o formato de JSON, um dos ficheiros contem a formatação dos nossos tabuleiros e alguma informação própria do tabuleiro, cada tabuleiro corresponde a um nível. O segundo ficheiro é o que se costuma chamar de save file, a sua função é conter o estado do jogo a quando o utilizador decido guardar o seu progresso, e também o estado inicial do nível em questão para que se possam resolver alguns problemas que sejam suscetíveis de aparecer.
* *Package* “JARS” – neste *package* estão vários ficheiros do tipo JAR, que servem como suporte, para a leitura, escrita e parsing de JSON para objetos em memória em vice-versa. Não vou descrever o objetivo de cada JAR mais abaixo pois, não somos os seus autores, e como referido noutra altura neste relatório são ficheiros obtidos no diretório de Maven que decidimos utilizar para facilitar a introdução de ficheiros do tipo JSON no nosso projeto.
* *Package* “Resources” - neste *package* estão inseridas todas as imagens que são utilizadas para desenhar os jogadores, os estados dos jogadores (num sub *package*), chão, obstáculos e objetivos, é aqui também que mantemos o ficheiro que contem a representação de cada tabuleiro do POKOBAN.
  + Imagem “*FateX.png*” - este ficheiro corresponde à representação gráfica dos objetivos do jogo;
  + Imagem “*FloorX.png*” - este ficheiro corresponde à representação gráfica do chão dos nossos tabuleiros;
  + Imagem “*ObstacleX.png*” - este ficheiro corresponde à representação gráfica dos obstáculos dos nossos tabuleiros;
  + Imagem “*WallX.png*” - este ficheiro corresponde à representação gráfica dos limites dos nossos tabuleiros;
  + X – corresponde ao nível em questão, ex: nível 1 usará as imagens “Fate1.png”, “Floor1.png”, “Obstacle1.png” e “Wall1.png”.
  + *Package* “PlayerSprites”;
    - *Imagem “Y\_UpZ.png” -* este ficheiro corresponde à representação gráfica do jogador nos nossos tabuleiros.
    - *Imagem “Y\_UpZ.png” -* este ficheiro corresponde à representação gráfica do jogador nos nossos tabuleiros.
    - *Imagem “Y\_UpZ.png” -* este ficheiro corresponde à representação gráfica do jogador nos nossos tabuleiros.
    - *Imagem “Y\_DownZ.png” -* este ficheiro corresponde à representação gráfica do jogador nos nossos tabuleiros.
    - Y – Corresponde ao jogador 1 ou jogador 2;
    - Z – Corresponde ao iterador que diferencia os estados do movimento do jogar. Ex: 1\_UP1.png será o jogador 1 com a perna direita a descrever o movimento.
* *Package* “Model” – neste package estão todos o os modelos que iram definir cada objeto que será utilizado para a construção do nosso jogo.
  + Classe “BaseObject” – Esta classe irá ter a função de pai, para várias outras classes, como a de Floor, Obstacle, Player, e Wall;
  + Classe “Board.java” – nesta classe que vamos instanciar os nossos objetos, desenha-los no ecrã, e controlar alguma parte dos movimentos do jogador e obstáculos através de alguma logica.
  + Classe “Floor.java” – serve como um simples modelo para o chão do jogo, tem um método para definir qual a sua imagem.
  + Classe “Obstacle.java” - serve como um simples modelo para os obstáculos do jogo, tem um método para definir qual a sua imagem.
  + Classe “Wall” - serve como um simples modelo para as paredes que definem o limite do jogo, tem um método para definir qual a sua imagem.
  + Classe “Level.java” – esta classe serve como modelo para guardar a informação de um determinado nível, guarda o seu layout e também o seu numerador que o distingue dos outros níveis.
  + Classe “Player.java” - serve como um simples modelo para os obstáculos do jogo, tem um método para definir qual a sua imagem, tem também uma propriedade boolean que define se este é o segundo jogador ou não.
* *Package* “View” – Neste *package* estão todas as classes que constroem a GUI do nosso jogo.
  + Classe “AboutScreen.java” – nesta classe desenhamos um ecrã com algumas informações sobre os criadores do jogo.
  + Classe “GameBoard.java” – é esta classe que irá usar uma instância da classe Board.java e desenhar o jogo para que o utilizador consiga visualizar o seu progresso.
  + Classe “SlashScreen.java” – Este é o ecrã principal, é nele que o utilizador poderá aceder ao “Abou”, criar um novo jogo a solo ou com um amigo, ou retomar um jogo previamente gravado.

## Diagrama de Classes



# Manual do Utilizador

O Sokoban é um jogo que pode ser jogado tanto solo como multijogador, senque que cada jogador controla um “boneco” que poderá andar livremente pelo espaço do tabuleiro, e com o objetivo de levar o obstaculo movivel para um lugar alvo.

O funcionamento do jogo é idêntico ao oficial, sendo que mudou-se apenas os “sprites” do mesmo para “sprites” de outro franchise (Pókemon, maioritáriamente) visto este ter alguns puzzles que se assemelham ao Sokoban.

Após a abertura do jogo o menu inicial é aberto no centro do seu ecrã.

Neste menu é possivel começar-se um novo jogo (tanto “single” como “multiplayer”), dar load de um jogo já guardado previamente, abrir o menu de “Sobre” e sair do jogo.

Supondo que se o utilizador queira iniciar um jogo novo, clica em qualquer um dos butões “New Single/Multiplayer Game”, que irão ambos abrir uma janela similar para que seja escolhido o nível que quer jogar.

Após o jogador escolher o nível pretendido, é aberta uma janela com o Jogo no seu estado inicial. Nesta janela é possivel verificar uma barra no topo da mesma que contem quatro botões que após clicados aplicam ações variadas.

O primeiro botão, “Reset” coloca o jogo na fase inicial do jogo mesmo após já se ter feito movimentos no mesmo (O botão “Esc” fará o mesmo efeito), o botão “Undo” e “Redo” farão com que seja possivel andar para trás e para a frente nas ações (a menos que sejam interrompidas com outro movimento pelo meio), e por fim temos o botão de “Save” que guarda o jogo na posição do jogo naquele exato momento.

Supondo que se o utilizador queira iniciar um jogo já guardado à posteriori, clica em “Load Game”, que irá dar load do jogo que foi guardado usando o botão de “Save” do tabuleiro de jogo.

Na janela de jogo o jogador pode ser movido pelo tabuleiro com as setas direcionais, ou W,A,S,D, tanto como com o rato clicando no zona central da janela e na direção que quiser que o jogador se mexa(centro-cima, mexe o jogador para cima, centro-direita mexe o jogador para a direita.)

No caso de ser um jogo com dois jogadores (“multiplayer”), o Jogador 1 pode ser movido apenas pelas setas direcionas ou com o uso do rato na janela, e o Jogador 2 apenas com o W,A,S,D.

O jogo acaba e passa ao próximo nivel após todos os locais alvos estarem preenchidos com um obstaculo movivél.

O menu de “About” tem apenas simples informação acerca dos desenvolvedores desta vairante do jogo “Sokoban” e da instituição para o qual foi desenvolvido.

E o botão “Exit Game” fará com que se saia do jogo por completo.

# Limitações e Desenvolvimentos Futuros

Ficaram por adicionar habilidades especiais a obtaculos, ou outros extras do género, ficou por se dar a oportunidade do utilizador criar os seus de niveis, e ficou por se adicionar alguma “inteligência” ao jogo em si.

Se abrirmos um jogo guardado com um dos objetivos tapados por um obstáculo e movermos esse obstáculo o jogo não vai redesenhar o objetivo

# Conclusão

## Conclusão do Alexandre

No desenvolvimento deste trabalho aprendi principalmente a trabalhar com a linguagem de java, a minha experiencia ainda que pouca já tinha alguma solidez, mas em .net, logo a familiarização não foi difícil, mas foi certamente importante para solidificar conhecimentos específicos e criar confiança para futuramente fazer mais desenvolvimento em java.

Aprendi a importar bibliotecas externas para o meu projeto através de JARs, para que fosse possível usar a formatação JSON, nos ficheiros de gravação e storage do tipo de cada nível.

Finalmente gostaria de agradecer ao professor Paulo Santos que foi paciente com as minhas diversas perguntas e me ajudou a resolver problemas com JFrame no desenho de objetos, com a utilização da GUI do netbeans para controlar os comportamentos do JFrame e também a encontrar a solução para que o plugin easyUML fizesse a sua função.

## Conclusão do Bruno

No desenvolvimento deste trabalho a maior dificuldade na minha otica foi sem duvida a transição do C# (com que trabalho diariamente) para o Java, que não tinha qualquer experiencia, e também a minha falta de experiencia com a programação de jogos, embora saber algumas coisas, não ter a noção de boas praticas nesta área bastante interessante.

Neste projeto aprendi bastante sobre colisões e como usá-las, e como integrar uma componente de backend com uma GUI.

Aplicamos alguns conhecimento sobre manipulação de JSON com a ajuda dos JARs que utilizamos, em vez do Serializable que os professores falaram nas aulas, para dar um outro toque nosso ao projeto.

Foi um trabalho que gostei bastante de desenvolver, porque ver como se faz algo na área do gaming (embora em pequena escala) era algo que já queria ter feito anteriormente e achei bastante interessante.

# Referências

Stackoverflow - <https://stackoverflow.com>;

Google Images - https://www.google.pt/;