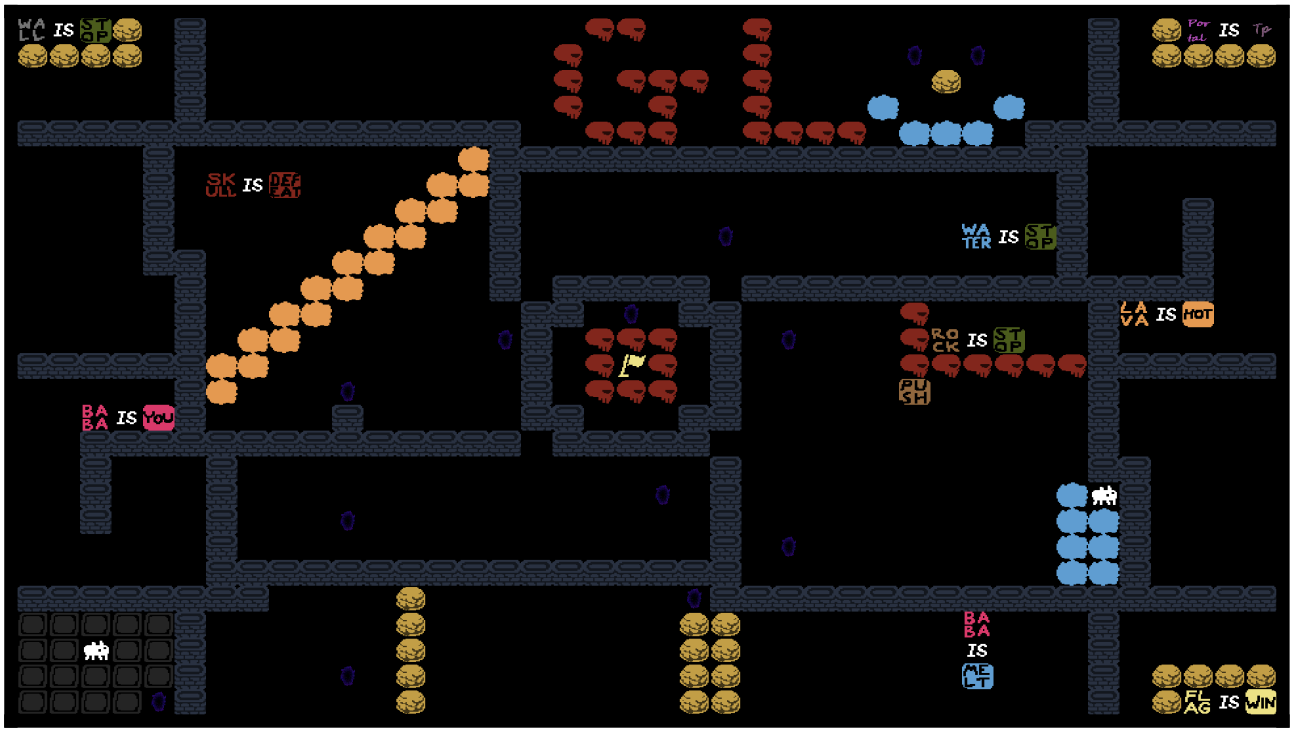
Project – Dev


WALL IS STOP
Portal IS Tp
SKULL IS DEFEAT
WATER IS STOP
LAVA IS HOT
ROCK IS STOP
PUSH
BABA IS YOU
BABA IS MELT
FLAG IS WIN

Summary

Introduction

The objective of this Projet is to create a game that corresponds to the game «Baba is you» for this we will use the zen5 library to make the graphic, player keyboard action…

◆ Objective of the game

The objective of the game is to move the element with property «you» to the element with property «Win».

Structure

◆ Folder «src»

“src” contains the following folders :

1. fr/uge/BabaIsYou → contains source files.
2. img → contains every images to draw the board for our levels.
3. Levels → contains all level from 1 to 8 to necessary for the game to function.

◆ Package «fr/uge/BabaIsYou»

The package contains the 4 following folders which correspond to our distribution of our classes in this project :

1. Draw
2. Element
3. Main
4. Read

Folder «Draw»

This folder contains everything that will be necessary for the graphic display and content of the game :

- Area → This class will allow the graphic display of the board.
- Board → This class will allow managed everything that will be related to the board (element, rule, ...).
- Rule → This class will allow managed everything that will be related to the rules of the level or rules added by the player.

Folder «Element»

This folder contains everything you need to create a game element (player included) :

- Element → This class allows the creation of a game element. To create one we will use the Type and Status enum.
- Player → This class allows you to manage and create a player. It also manages the movements of it.
- Status → This enum is used to manage all the authorized status of the game.
- Type → This enum is used to manage all the authorized type of the game.

Folder «Main»

This folder contains the main :

- Main → Calls the methods needed to play the game.

Folder «Read»

This folder contains all the commands, file or key reading and application management :

- Commands → This class contains the necessary methods for the commands the user can do (see User).
- File → This class is used to manage file reading in order to be able to create the levels of the game.
- Input → This class allows to detect and manage the keys that the user uses.
- Position → This class allows to create the position of an element of the game.
- Timer → This class is used to manage the time between each display.

Choice of implementation

Implementation choice

To make easier our program we made the choice to implement the board as a LinkedHashMap where the key will be the Position and the value an ArrayList of Element (because we can have more than 1 element in a same position). This implementation allows us to directly find one or several element(s) by their position.

Creation noun and property

We made a portal as noun and Tp (Teleport) as property.

When tu player overlaps over a object with property as Tp he will need to chose his output if it’s valid (no Stop, Defeat, Sink, Hot).

The player will be teleport to the closest object as Tp property we used for this a method distance, which return the distance between 2 objects thank’s to their Position (x, y). And in an other method we keep the more closest object which will be the output.

Modifications after beta defence

During the defense we had to modify the type of the constructor of Element from «new Element (string name, string type, string status)» by replacing the type of «type» and «status» by enum.

We had also trying to reduce more than possible reapiting code and removed useless code.

In particulary we did good modification on the following methods :

In the class Rule : Push.
In the class Board : NumberElem, PushElem.
In the class Player : Move.