CaveMan

PROG 2370 – Final Project Report

Group members: Cleison Hister, Devit Souza Bremer

1. **About**

CaveMan is a simple 2D platform game. In the game, the character named CaveMan has the objective to collect all food available in the level and then go to his home. That must be done under the 1 min time limit, or the level is restarted, and the player must start over again. If the player collects all available food in the level and goes to his home, the game advances to the next level, where the platform difficulty is increased.

1. **Implementation**

The game implementation was done by using object-oriented programming concepts. The main structures in the game are defined by classes. Those classes are hierarchically organized as follows.

* 1. **CaveMan**

This is the main class for the game. Here we create instances of levels and some background elements. We also control and draw the game status screens, timer and player points.

* 1. **Level**

Although the previous class is the game's main class, most of this game's logic is implemented in the Level class.

Here we dynamically build the background of the level, which is composed of 4 images stacked to give the impression of depth.

We also read a text file containing the level design regarding tiles. We create an animated instance of our caveman as well.

Besides that, we create a list of Food tiles that will be placed in the level on the positions indicated in the text file.

* 1. **Player**

Here we create our character from an instance of Animation. We also control the character’s movement and sound effects.

Regarding animation, the character has 3 statuses and 3 corresponding animations: idle, walking and jumping.

* 1. **Animation / AnimationPlayer**

Those two classes work together to create an animation: The class Animation reads a sprite sheet containing the animation frames. It also defines how long each frame will be shown.

The AnimationPlayer class gets an Animation object and, using its properties, moves the sprite sheet frames to create an actual animation.

* 1. **Tiles**

Tiles are small texture blocks that compose the level. They are very useful for creating beautiful and challenging game levels. Basically, any texture can be used as a tile and incorporated into the game.

In this game, we have implemented 3 types of tile collision in an enumerable object:

Passable: Tiles that don`t collide with the player. Helpful in creating decoration elements.

Impassable: Those tiles collide with the player in all directions. They are used to create the ground.

Platform: Player can walk over the tiles, jump under the tile and land over the tile. Those were used to create floating platforms.

* 1. **Food**

Food is a type of tile with a different way of being positioned and a simple animation.

* 1. **RectangleTools / FoodLimit**

Those classes are static classes that verify the intersection between tiles and the player in the game.

1. **Screenshots**

Text

Description automatically generated with medium confidence

Figure - Welcome screen

A screenshot of a video game

Description automatically generated

Figure - First level

A screenshot of a video game

Description automatically generated

Figure 3 – Second level

A screenshot of a video game

Description automatically generated

Figure 4 – Third level

A picture containing text

Description automatically generated

Figure 5 - Failure to complete level

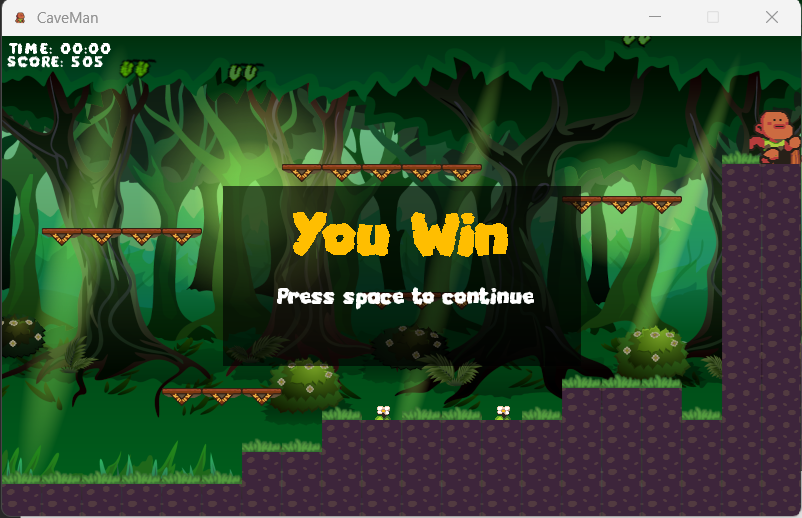


Figure 7 - Success in completing level