Maxence Desrousseaux Arthur Massanes Hugo Lachkar

Arcade Documentation

Summary

Table des matières

Graphical Libraries

The different libraries all use a class that inherit from the IGraphics interface. This interface possesses different attributes and uses different functions to manage and print all the different elements.

The Shape structure:

This structure is composed of:

- two integers x and y that are the position of an element in the window,
- a string named filePath used to store the path to the image you want to print. It can be empty if you use for libraries that don't use image, or if you want to display a message,
- A string named message used to store the message you want to print. If your using a library that don't use image like ncurse, it is what is printed,
- An integer name id that is unique among all other instance of this structure. It is used to access this shape among all other in a easy way.
- A size_t named textSize used to change the size of the text in case you're displaying text in a image-using library
- A Color named textColor used to change the color of the text you're displaying

Color is an Enumeration that contain the different color you can print your text in. The possibilities are:

- WHITE
- BLACK
- YELLOW
- RED
- GREEN
- BLUE

```
typedef struct Shape_s {
    int x;
    int y;
    std::string filePath; // can be empty for ncurses
    std::string message; // check if empty
    size_t textSize = 24;
    Color textColor = WHITE;
    int id;
} Shape;
```

The member functions:

The display functions:

The different functions are:

- addShape(Shape &shape): This function is used to translate a shape structure in its equivalent in the graphical library you're using and store it in the class of your library. In case you want to display several times the same image but in different places, it is more efficient to just use it once and then modifying the position in the shape structure and call the drawShape function several times.

- drawShap(int key): It draw the shape you added that have an id equal to the key, you can draw it several times with different position before using the drawScreen function.

- drawScreen(void): It display the window with all the shapes you've drawn after the last call to clearScreen

- clearScreen(void): It is used to remove all the shape you've drawn from the window.

The event functions

The IGraphics interface uses an Enumeration to manage the different input. This enumeration is named Input. The different Input possible are:

NO_INPUT, UP, DOWN, RIGHT, LEFT, QUIT, PAUSE, ACTION, PREV_GAME, NEXT_GAME, PREV_LIB, NEXT_LIB, RESTART_GAME, MENU

- getInput(void): it returns an Input and in case of a QUIT event, it set the next call to isRunning to false.

- isRunning(void): it returns a Boolean that is true if no QUIT event happened before, or false if a QUIT event happened.

Game Libraries

The different game libraries all use a class that inherit from the IGames interface. This interface possesses different attributes and uses different functions to manage the different part of an arcade game.

The Public functions:

- void runGame(IGraphics *, loadGraph &, std::string) : It is the function that will be used to run your games, everything needed for your game will be done using this method, including the game loop.

- int isGameChanged(void): It is the method that is used to determinate whether the user want to change the game or not after the end of your game. The return value is set to true after a PREV_GAME or a NEXT_GAME event.

- IGraphics *getLibGraphic(void): This function will be used by the core of the program to get the graphical library currently used in the program.

The Private functions:

The initialization methods:

- void initGameShapes(bool): It is the function that create all the shape that will be used.

- void getWallFoodsPositions(std::vector<std::string>): It is the function that you can use at the beginning of the game to get the position of the walls, food and power-up.

- bool getMap(std::string): It is the function you can use to get the map of your game from the filepath passed in parameter. If the map was loaded successfully it return true, otherwise it's false.

The Utile methods:

- void setShape(Shape &, int, int, int, std::string, std::string, Color, int) : It is the function that your game can use to add a shape in the graphical library used. It is also useful for the initGameShapes function.

- void switchHeadDirection(std::string, int): This function is used to change the sprite of your character and its direction. In case your character is composed of different sprite like in a snake game, it only changes the sprite of the head.

- bool startPause(void): It is used to pause your game.

The printing methods:

- void printAtPosition(Shape &, int, int): it is used to print a shape at the coordinate in parameter.

- void printMap(std::vector<Position>, Shape &): It is used to print the map of the game.

- void printGame(void): it is called in your main game loop to print all of the elements of your game.

The saving methods:

- std::vector<std::pair<std::string, int>> getTopScores(void): it is used to get all the saved scores from the save file.

- void setTopScore(std::vector<std::pair<std::string, int>>): It is used to save your score if it is among the best.

- bool isTopScore(void): It is used to check if the current score is among the best.

The Ending methods:

- void switchSelectionEndMenu(Shape *toSelect, Shape *toUnselect, indexToUnselect): it is used to manage the selection of the selected part of the ending menu.

- void endMenu(void): It is used to create and manage your ending menu.

- void handleEventEndGame(int): It is used the events of your ending menu.

- void initEndMenu(void): It is used to initiate the ending menu.

The event method:

- void handleEventGame(int, loadGraph &): It is used to manage the event of your game.

The initialization method:

- void refresh(void): It is used to refresh your screen to display the current state of things.