Arcade documentation

This document presents an extensible way of adding game libraries to our Arcade project and the used functions.

Every graphical library uses the following set of functions:

void init();

Initialize everything related to the graphical library.

void deinit();

Free memory used by the library.

void set_pixel(unsigned long x, unsigned long y, unsigned char r, unsigned char g, unsigned char b, char ncurse_char);

In its most basic form, this function adds a pixel of the desired color in the desired position.

Position is set using the x and y parameters and color is set using the r, g, and b parameter.

The char ncurses_char parameter tells which character should be print. This character is useful because we use the Ncurses library.

When we use SFML and SDL 2 this function can display rectangles or sprites. This depends on the ncurses_char parameter.

For example, in order to display a pacman coin, you must send '.' (dot character) as the ncurses_char parameter.

In order to display border blue rectangle for pacman, you must send '#'.

In order to display grey border rectangle, you must send 'X'.

In order to display snake body, you must send 'S'.

In order to display snake head, you must send 'O' with r set to 254.

In order to display snake apple, you must send 'A'.

In order to display pacman super ball, you must send 'O' with r not set to 255.

In order to display pacman body, you must send 'P'.

In order to display pacman ghost, you must send 'g'.

In order to display pacman ghost in super moden, you must send 'G'.

void clear(unsigned char r, unsigned char g, unsigned char b);

Clear the screen with the selected color. Color is set using r, g, and b parameters.

```
int get_input_pressed();
```

Returns the key pressed by the user as an int.

Keys are define as a C enumeration :

```
enum Input {

NONE,

A, B, C, D, E, F, G, H, I, J, K, L, M,

N, O, P, Q, R, S, T, U, V, W, X, Y, Z,

LEFT_ARROW, RIGHT_ARROW, UP_ARROW, DOWN_ARROW,

ENTER, SPACE, DELETE, BACKSPACE, TAB, ESC, RETURN, ESCAPE

};
```

None value is 0, A value is 1, B value is 2 etc...

```
 bool update();
```

This function update everything drawn by the **set_pixel** function. It must be called in order to draw.

It returns a bool false if the window does not exist anymore and true if the window still exists.

A basic game loop would be:

```
int key = -1;

graphicalLib.init();

while (graphicalLib.update()) {

            graphicalLib.clear(0, 0, 0);

            graphicalLib.set_pixel (5, 5,  100,  255, 15, 'X');

            key = graphicalLib.get_input_pressed();

            //Do somethings with the key value.

            graphicalLib.update();

}

graphicalLib.deinit();
```

You are free to implement new games the way you want but **be sure to respect the graphics libraries implementation as shown before.**