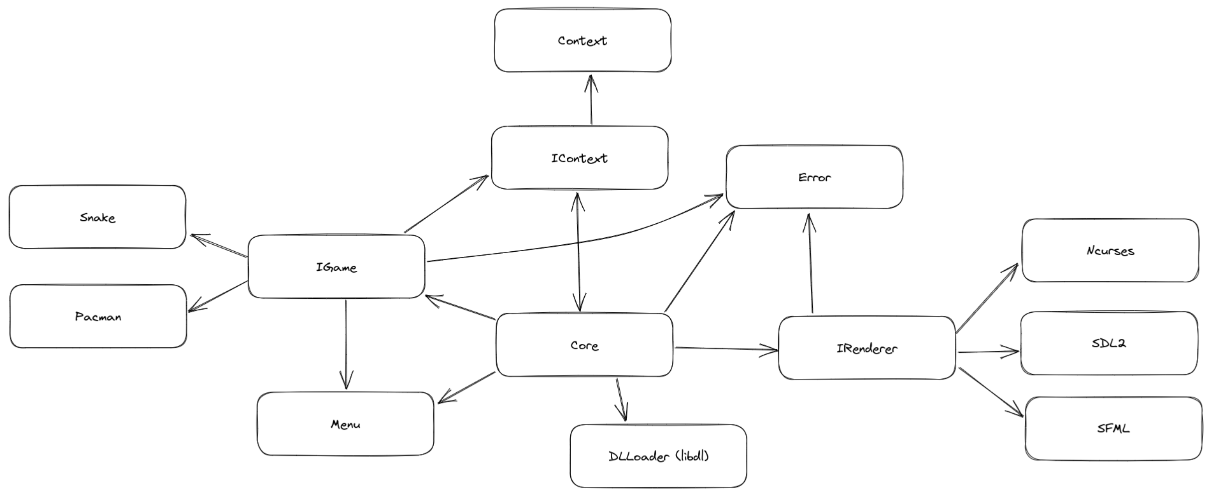
Arcade Documentation

1. Architecture:



- **Core**: This is the main part of the project. It is this part which manages the main loop, the change of game and renderer library. It also manages the management of these libraries and calls them at the right time. It also has access to the Context.
- **Context**: Allows to make an intermediate link between Game - Core - Renderer. Contains access to the core to retrieve the renderer. This allows the Game to call the Renderer without having it directly. It also makes it possible to make the link between the Game and the Core.
- **Game**: Runs the game. Has access to the Context in some cases to clear the window and refresh it according to the render and compute of the Game.
- **Renderer**: Clear, refresh, displays text or sprites depending on what the Core or Context passes to it as a parameter.

2. How to add graphical or game library

a. *Graphical library*
To implement another graphical library, you must implement at least the interface **IRenderer** functions to work with our Core.
    - **loadSpritesheets**: The game give you a list of all spritesheets used by itself and you must create the spritesheets that you keep in memory to use these spritesheets
    - **pollEvents**: You must return a vector of all key triggered by you graphical library. A converter of key in your graphical and arcade key is mandatory.
    - **clearWin**: Just clear the window.
    - **refreshWin**: Just refresh the updated window
    - **drawSprite**: You must draw on the window the sprite (using the correct spritesheet) passed by parameter at the coords (x and y) also passed in paramter.
    - **drawText**: Same as **drawSprite** but for a text.

b. *Game library*

To implement another game library, you must implement at least the infreface **IGame** functions to work with our Core.

- **getSpritesheets**: You must return a list of Spritesheets (id, graphic path, ncurses path, size of one sprite) that will use in **IRenderer** to create the corrects spritesheets in the graphical library.
- **compute**: Set the new internal state of the game. You must manage the framerate using the **dtime** parameter.
- **handleInput**: You must handle all inputs passed by the core to modify the state of your game according to the input.
- **render**: You must use the **drawSprite** and **drawText** functions with the Context and call the Context **render** function.