Guillaume Wilmot
Nyrandone Noboud
Florian Saurs

How to contribute to the Arcade

Hi, first of all, we'd like to thanks you for purchasing our Arcade. Throughout this small tutorial, you'll learn how to add new games and new graphic libraries to the Arcade to enhance your experience.

---

1) The architecture

The Arcade was designed with modularity in head. What that means is, given that it's an open source software, you are gonna be able to add almost anything to the Arcade with little to no problem. Indeed, we use dynamic shared libraries (.so) for our games and graphic libraries which means if a lib follows our interface, it will be compatible with the Arcade.

Basically, we have three main components :

- The core : It can run by itself. Its main purpose is to load the dynamic libraries, display the menu and launch the games as well as managing the "At runtime game switching" functionality.
- The Games : They include the logic of the game such as the rules. They interact with a graphic library to tell it to display the current state of the game. They also implement an "At runtime graphic library switching" functionality.
- The Graphic Libraries : Their sole purpose is to display things and to manage the events such as keyboard typing. They're called by the Arcade menu and by the games. They all need to follow the same interface.

The core doesn't need to be meddled with as it's already fully modular.

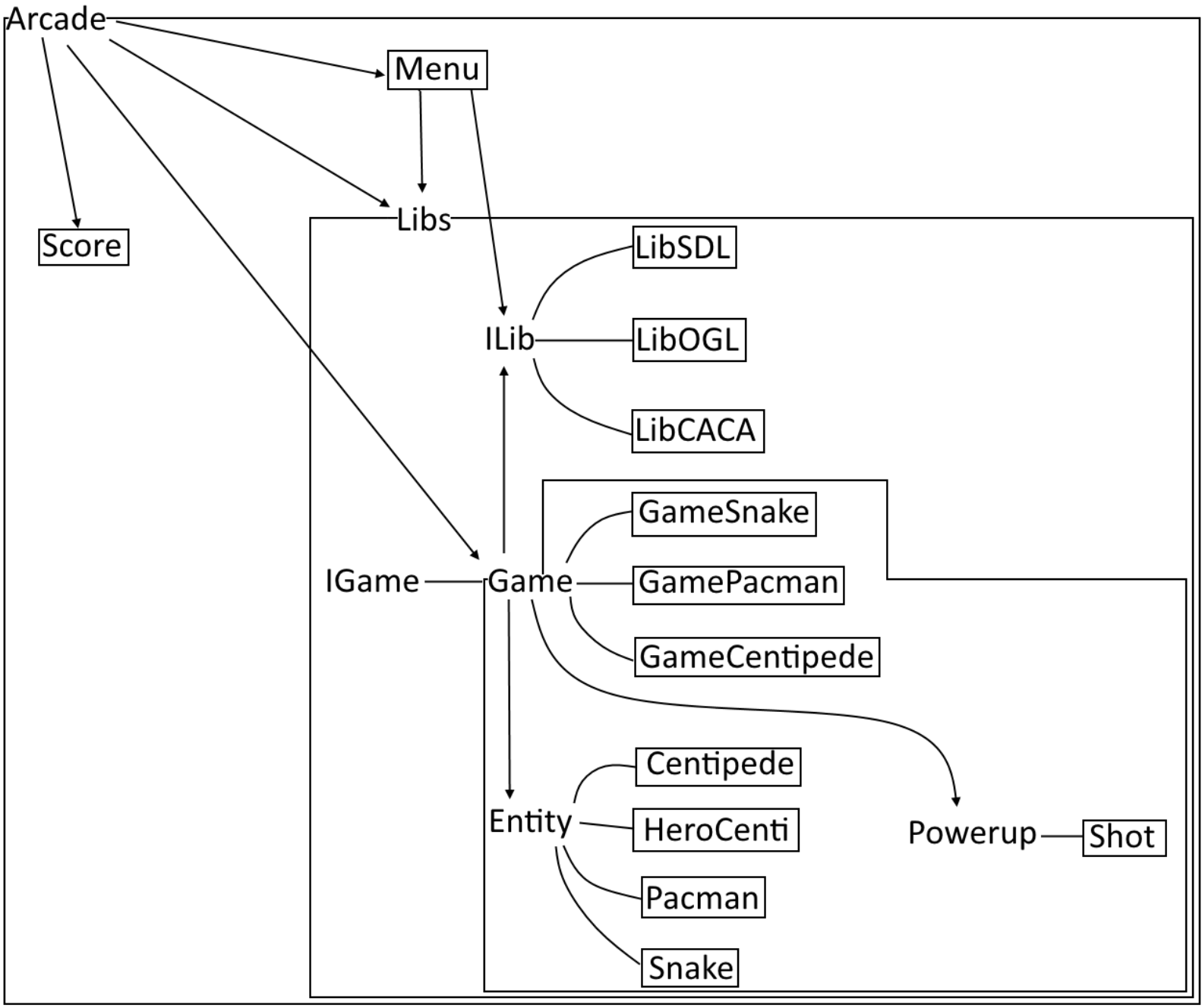

Diagram of the links between the classes

2) The Games

The Games implement some mandatory methods :

1. The game() method launches the Game.
2. The changeDirection() method ask the graphic library for an event and performs the associated action.
3. The timer method() simply waits for a determinable amount of time before displaying the next frame.
4. The generateMap() generates a map of determinable size and content.

In our games, we use a map which is represented by an array of std::string to store our decor. Then, the entities and powerups are stored in vectors of entity * and powerup * respectively.

3) The graphic Libraries

The graphic libraries implement some mandatory methods :

1. The init() method initiates everything the lib needs.
2. The end() method cleans up the resources before exiting the lib.
3. The print() method displays the current state of the game.
4. The getEvent() method returns an event if one is received and is implemented.
5. The playSound() method plays the sound received as parameter.
6. The printMenu() method displays the menu.
7. The getChar() method returns a character for the login.
8. The ChooseLib() method allows the user to choose a graphic library in the menu.
9. The ChooseGame() method allows the user to choose a game.

The lib iterates through the map to display the walls and the obstacles. Then, it iterates through the Entities and Powerups vectors to display them. You can also get the type of the game if needed for special effects. The events all need to be implemented otherwise, the user won't be able to performs all the actions. The Entities have a type that is needed to know which texture must be used. Same for the Powerups.