Documentation Technique Arcade

Quentin Sonnefrraud

April 8, 2018

1 IGraphical

```
std::string getGame() const; // get current Game from the menu
std::string getNickName() const; // get NickName from the menu
std::string getLib() const; // get current Library from the menu
bool isOpen() const; //check if the window is open
```

Listing 1: Information a donné au core

```
using entities_t = std::vector<std::pair<char, std::pair<float,
    float>>>;
```

Listing 2: Vecteur d'objets floatants (type, coordonnées)

```
int getKey();
```

Listing 3: Return la touche appuyé au format ncurses

```
int displayMap(std::vector<std::vector<uint16_t>> map, entities_t
    entities);
int menu(const std::vector<std::string> &games, const std::vector<
    std::string> &libs);
virtual bool isOpen() const;
```

Listing 4: prend en paramètre la map et les objets floatant

```
int menu(const std::vector<std::string> &games, const std::vector<
    std::string> &libs);
```

Listing 5: Appeller par le core qui envois les informations à afficher dans le menu

```
#define SNAKE_HEAD 'X'
#define SNAKE_BODY 'O'
#define SNAKE_TAIL 'o'
#define WALL '#'
#define APPLE '@'
#define EMPTY ' '
#define FOOD '.'
#define PACMAN "P"
#define GHOST "G"
#define EYES ":"
#define WEAK_GHOST "g"
```

Listing 6: Macro à respecter

```
#define RELOAD_GAME 'r'
#define PREV_LIB KEY_PDOWN
#define NEXT_LIB KEY_PUP
#define NEXT_GAME KEY_END
#define NEXT_GAME PAGE_HOME
```

Listing 7: Key Handle par le core

2 IGame

```
void sendKey(int key);
```

Listing 8: Récupère la touche de getKey

```
const std::vector<std::vector<uint16_t>> &getMap() const;
const entities_t &getEntities() const;
```

Listing 9: Return les informations a IGraphical

```
enum status_e { PLAYING, WIN, LOSE };
int getScore() const;
status_e getStatus() const;
```

Listing 10: Informations pour le Core

```
void play();
```

Listing 11: Appeler par le core 1 tick de jeu