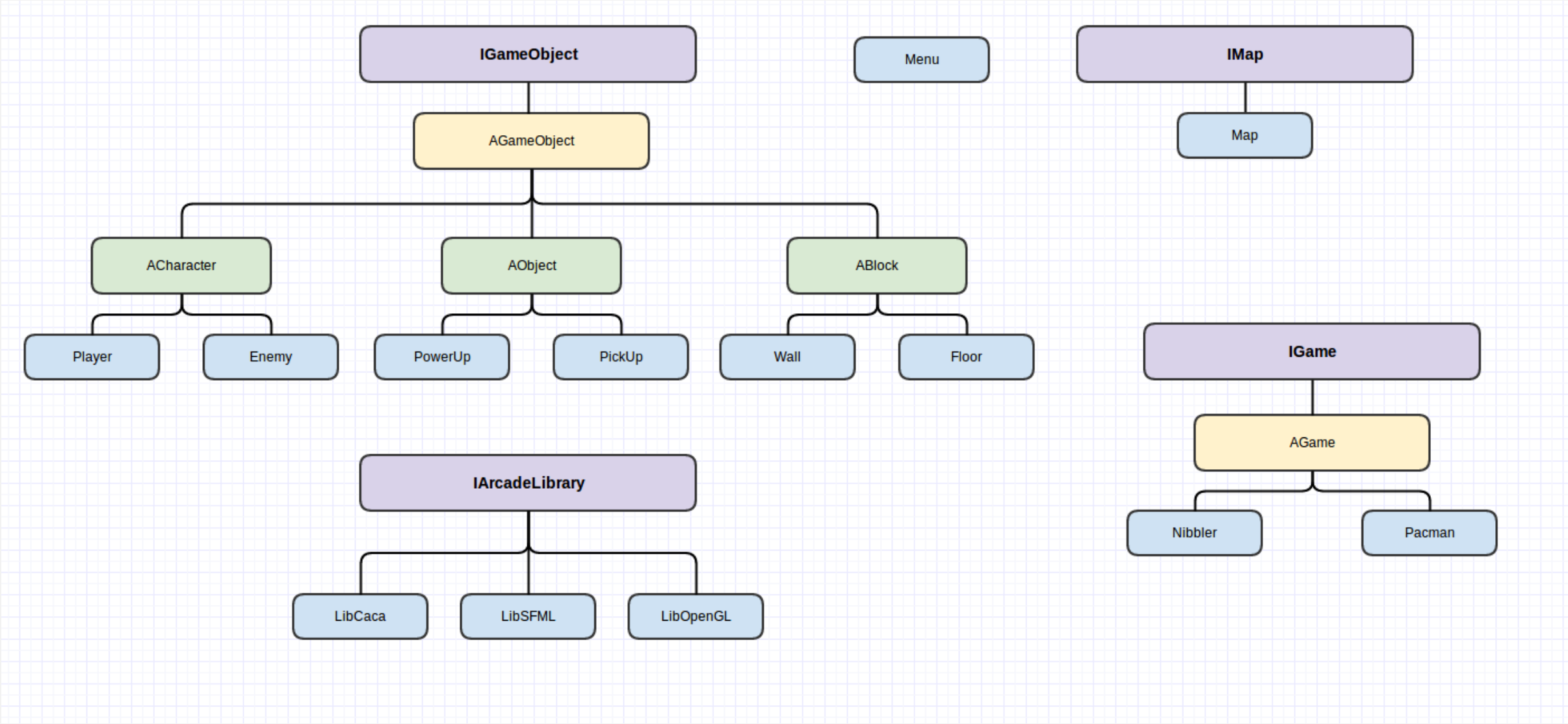
DOCUMENTATION ARCADE

Lucas Villeneuve – Samuel Osborne – Thomas Escorne

ARCHITECTURE



NAMESPACES

- Les interfaces sont encapsulées dans des namespaces

  Toutes les interfaces sont encapsulées dans le namespace ***arcade***.

- L'interface IGame est encapsulée dans le namespace ***games***.

- L'interface IArcadeLibrary est encapsulée dans le namespace ***library***.

INTERFACE IGAME

- L'interface IGame est l'interface qui contient le jeu, elle doit donc implémenter les interfaces IGameObject et Imap.
- La classe contient les méthodes suivantes :
  - const ***arcade::IMap*** *getMap() const
  - const ***arcade::IGameObject*** *getPlayer() const
  - const ***std::vector<arcade::IGameObject *>*** &getEnemies() const
  - const ***std::vector<IGameObject *>*** &getStrings() const
  - ***bool*** playRound(const ***arcade::CommandType*** &cmd)

    => cette méthode permet d'implémenter la boucle de jeu en fonction de la commande passée en paramètre.

INTERACE IMAP

- Cette interface est dédiée à la map du jeu, elle doit être implémentée dans les jeux et contient des tiles.
- La classe contient les méthodes suivantes :
  - **uint16_t** getWidth() const / ***uint16_t*** getHeight() const
  - ***arcade::IGameObject*** *getTile(const ***arcade::Position*** & pos) const
  - ***void*** setTile(const ***arcade::Position*** &pos, ***arcade::IGameObject*** *tile)
  - ***void*** setTile(***uint16_t*** x, ***uint16_t*** y, ***arcade::IGameObject*** *tile)

Note : Les classes qui héritent de IMap devraient implémenter les attributs suivants :

- const ***uint16_t*** width => la largeur de la Map
- const ***uint16_t*** height => la longueur de la Map
- ***std::vector<std::vector<arcade::IGameObject *>>*** tiles => le contenu de la Map

INTERFACE IGAMEOBJECT

- Définie l'ensemble des objets présents dans le jeu
  - Les objets héritants de AGameObject possèdent :
    - Un ***asset*** de type ***std::string*** : path vers le fichier à charger pour afficher l'objet.

      (Attention : Le path ne doit **pas** contenir l'extension du fichier, ex: .png, .txt, etc...)
    - Une ***pos*** de type ***arcade::Position*** : classe contenant la postion x et y du l'objet (cf. Protocol.hpp).
    - Un ***type*** de type ***arcade::TileType*** : type de l'objet (cf. Protocol.hpp)
- Chacun de ces attributs possèdent un "getter" et un "setter"
  - ***arcade::Position*** getPos() const / ***void*** setPos(const ***arcade::Position*** & pos) / ***void*** setPos(***uint16_t*** x, ***uint16_t*** y)
  - ***std::string*** getSprite() const / ***void*** setSprite(const ***std::string*** & asset)
  - ***arcade::TileType*** getTileType() const / ***void*** setTileType(const ***arcade::TileType*** & type)

ABSTRACT ACHARACTER

- La classe abstraite ACharacter hérite de AGameObject.
- La classe possède une méthode pure

  void move(const ***arcade::Position*** & pos)

  qui doit être implémentée dans les classes qui en hériteront.

Exemples de classes qui héritent de ACharacter : **Player**, **Ennemy**, ...

ABSTRACT AOBJECT

- La classe abstraite AObject hérite de AGameObject
- La classe contient les attributs suivants :
  - Un ***taken*** de type ***bool*** : Variable d'état permettant de savoir si l'objet à été ramassé.
  - Un ***secondAsset*** de type ***std::string*** : path du fichier à charger en cas de pick up de l'objet.

    (Attention : Le path ne doit **pas** contenir l'extension du fichier, ex: .png, .txt, etc...)

- Les méthodes implémentée sont les suivantes :
  - ***bool*** getTaken() const / ***std::string*** getSecondAsset() const
  - ***void*** take() / ***void*** setSecondAsset(const **std::string** & asset)

Exemples de classes qui héritent de AObject : ***PowerUp***, ***PickUp***, ...

ABSTRACT ABLOCK

- La classe abstraite ABlock hérite de AGameObject.
- La classe ne possède que des fonctions de "get" et de "set" pour l'attribut ***pos.***

  Cette classe est faite pour concevoir des objets du décor

Exemples de classes qui héritent de ABlock : ***Wall***, ***Floor***, ...

INTERFACE IARCADELIBRARY

- Cette interface permet de créer des librairies graphiques compatibles avec le core program.
- La classe doit contenir les méthodes suivantes :
  - ***void*** openWindow()
  - ***void*** closeWindow()
  - ***bool*** isKeyPressed(const ***arcade::Input*** & input)
  - ***bool*** isEventQuit()
  - ***void*** winClear()
  - **void** display()
  - ***void*** playMusic(const ***std::string*** & music)
  - ***void*** stopMusic()
  - ***void*** drawText(const ***std::string*** &str, const ***arcade::Position*** &pos)
  - ***void*** drawGameObject(const ***arcade::IGameObject*** * obj)
  - ***arcade::CommandType*** processInput()