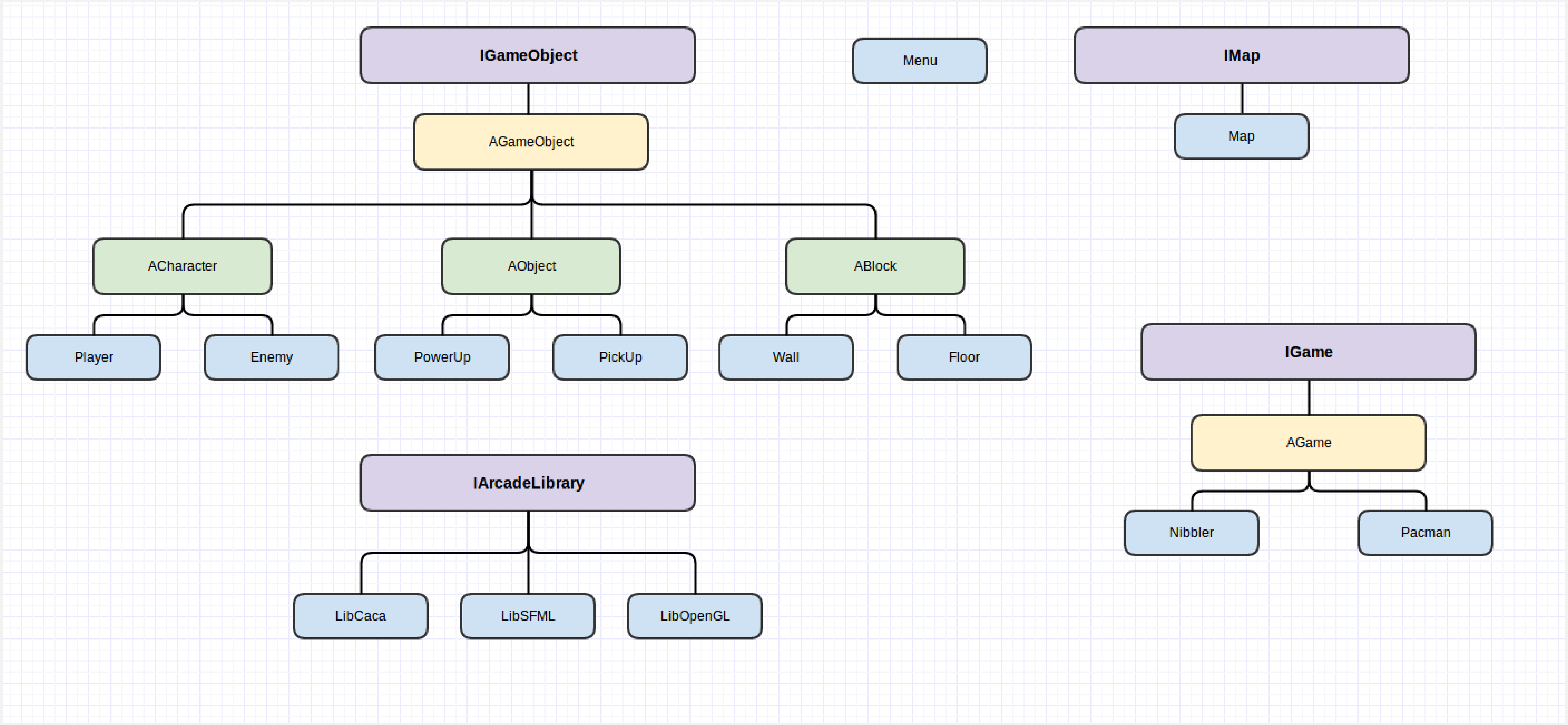
DOCUMENTATION ARCADE

Lucas Villeneuve – Samuel Osborne – Thomas Escorne

ARCHITECTURE



NAMESPACES

- Interfaces are encapsulated in namespaces

  Every interface is encapsulated in the ***arcade*** namespace.

- The interface IGame is encapsulated in the ***games*** namespace.

- The interface IArcadeLibrary is encapsulated in the ***library*** namespace.

INTERFACE IGAME

- The interface IGame is the interface which contains the game, it must implement IGameObject and Imap interfaces.
- The class contains the following methods :
  - 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)

  => you are supposed to build your game loop in this method depending on the command passed as parameter.

INTERACE IMAP

- This interface is dedicated to the map of the game, it must be implemented in the games. It contains tiles.
- The class contains the following methods :
  - 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 : Classes that inheritate from IMap should implement the following attributs :

- const uint16_t width => the Map's width
- const uint16_t height => the Map's height
- std::vector<std::vector<arcade::IGameObject *>> tiles => the content of the Map

INTERFACE IGAMEOBJECT

- This interface define all of the objects that are in the game
  - Objets that inherit from AGameObject have the following attributes :
    - ***asset*** of type *std::string* : path to the file that will be loaded to display an object.

      (Caution : The path must **not** contain the extension of the file, ex: .png, .txt, etc...)
    - ***pos*** of type *arcade::Position* : a class that contains the postion x and y of the object (cf. Protocol.hpp).
    - ***type*** of type *arcade::TileType* : type of the objet (cf. Protocol.hpp)

- Every attributes has a "getter" and a "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

- The abstract class ACharacter inheritate from AGameObject.
- The class has a pure method

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

  That must be implemented in the child classes.

Examples of classes that inheritate from ACharacter : **Player**, **Ennemy**, ...

ABSTRACT AOBJECT

- The abstract class AObject inheritate from AGameObject
- The class has the following attributes :
  - ***taken*** of type ***bool*** : State variable which tells if the object has been piked up or not.
  - ***secondAsset*** of type ***std::string*** : path of the file that will be loaded to display the object if it has been picked up.

    (Caution : The path must **not** contain the extension of the file, ex: .png, .txt, etc...)

- The class has the following methods :
  - ***bool*** getTaken() const / ***std::string*** getSecondAsset() const
  - ***void*** take() / ***void*** setSecondAsset(const **std::string** & asset)

Examples of classes that inheritate from AObject : ***PowerUp***, ***PickUp***, …

ABSTRACT ABLOCK

- The abstract class ABlock inheritate from AGameObject.
- The class only has possède "getters" and "setters" methods for the ***pos*** attribute**.**

  This class is made to build objects for landscape blocks.

Examples of classes that inheritate from ABlock : ***Wall***, ***Floor***, ...

INTERFACE IARCADELIBRARY

- This interface allows you to create graphic libraries that are compatible with the core program.
- The class must contain the following methods :
    - *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()