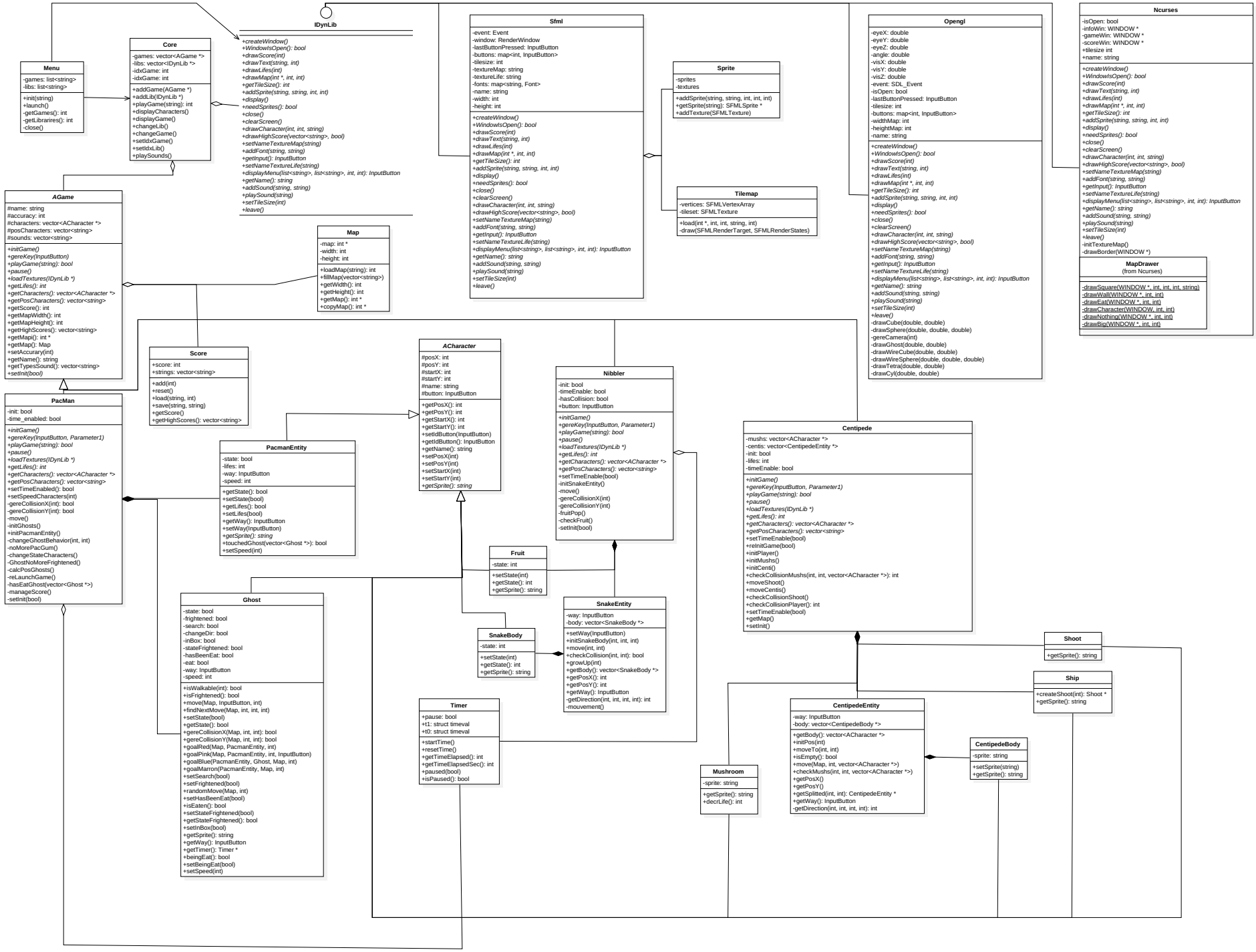
IDynLib
+createWindow()
+WindowIsOpen(): bool
+drawScore(int)
+drawText(string, int)
+drawLifes(int)
+drawMap(int *, int, int)
+getTileSize(): int
+addSprite(string, string, int, int)
+display()
+needSprites(): bool
+close()
+clearScreen()
+drawCharacter(int, int, string)
+drawHighScore(vector<string>, bool)
+setNameTextureMap(string)
+addFont(string, string)
+getInput(): InputButton
+setNameTextureLife(string)
+displayMenu(list<string>, list<string>, int, int): InputButton
+getName(): string
+addSound(string, string)
+playSound(string)
+setTileSize(int)
+leave()
Menu
-games: list<string>
-libs: list<string>
+init(string)
+launch()
-getGames(): int
-getLibrarires(): int
-close()
Core
-games: vector<AGame *>
-libs: vector<IDynLib *>
-idxGame: int
-idxGame: int
+addGame(AGame *)
+addLib(IDynLib *)
+playGame(string): int
+displayCharacters()
+displayGame()
+changeLib()
+changeGame()
+setIdxGame()
+setIdxLib()
+playSounds()
Sfml
-event: Event
-window: RenderWindow
-lastButtonPressed: InputButton
-buttons: map<int, InputButton>
-tilesize: int
-textureMap: string
-textureLife: string
-fonts: map<string, Font>
-name: string
-width: int
-height: int
+createWindow()
+WindowIsOpen(): bool
+drawScore(int)
+drawText(string, int)
+drawLifes(int)
+drawMap(int *, int, int)
+getTileSize(): int
+addSprite(string, string, int, int)
+display()
+needSprites(): bool
+close()
+clearScreen()
+drawCharacter(int, int, string)
+drawHighScore(vector<string>, bool)
+setNameTextureMap(string)
+addFont(string, string)
+getInput(): InputButton
+setNameTextureLife(string)
+displayMenu(list<string>, list<string>, int, int): InputButton
+getName(): string
+addSound(string, string)
+playSound(string)
+setTileSize(int)
+leave()
Sprite
-sprites
-textures
+addSprite(string, string, int, int, int)
+getSprite(string): SFMLSprite *
+addTexture(SFMLTexture)
Tilemap
-vertices: SFMLVertexArray
-tileset: SFMLTexture
+load(int *, int, int, string, int)
-draw(SFMLRenderTarget, SFMLRenderStates)
Opengl
-eyeX: double
-eyeY: double
-eyeZ: double
-angle: double
-visX: double
-visY: double
-visZ: double
-event: SDL_Event
-isOpen: bool
-lastButtonPressed: InputButton
-tilesize: int
-buttons: map<int, InputButton>
-widthMap: int
-heightMap: int
-name: string
+createWindow()
+WindowIsOpen(): bool
+drawScore(int)
+drawText(string, int)
+drawLifes(int)
+drawMap(int *, int, int)
+getTileSize(): int
+addSprite(string, string, int, int)
+display()
+needSprites(): bool
+close()
+clearScreen()
+drawCharacter(int, int, string)
+drawHighScore(vector<string>, bool)
+setNameTextureMap(string)
+addFont(string, string)
+getInput(): InputButton
+setNameTextureLife(string)
+displayMenu(list<string>, list<string>, int, int): InputButton
+getName(): string
+addSound(string, string)
+playSound(string)
+setTileSize(int)
+leave()
-drawCube(double, double)
-drawSphere(double, double, double)
-gereCamera(int)
-drawGhost(double, double)
-drawWireCube(double, double)
-drawWireSphere(double, double, double)
-drawTetra(double, double)
-drawCyl(double, double)
Ncurses
-isOpen: bool
-infoWin: WINDOW *
-gameWin: WINDOW *
-scoreWin: WINDOW *
+tilesize int
+name: string
+createWindow()
+WindowIsOpen(): bool
+drawScore(int)
+drawText(string, int)
+drawLifes(int)
+drawMap(int *, int, int)
+getTileSize(): int
+addSprite(string, string, int, int)
+display()
+needSprites(): bool
+close()
+clearScreen()
+drawCharacter(int, int, string)
+drawHighScore(vector<string>, bool)
+setNameTextureMap(string)
+addFont(string, string)
+getInput(): InputButton
+setNameTextureLife(string)
+displayMenu(list<string>, list<string>, int, int): InputButton
+getName(): string
+addSound(string, string)
+playSound(string)
+setTileSize(int)
+leave()
-initTextureMap()
-drawBorder(WINDOW *)
MapDrawer
(from Ncurses)
-drawSquare(WINDOW *, int, int, int, string)
-drawWall(WINDOW *, int, int)
-drawEat(WINDOW *, int, int)
-drawCharacter(WINDOW, int, int)
-drawNothing(WINDOW *, int, int)
-drawBig(WINDOW *, int, int)
AGame
#name: string
#accuracy: int
#characters: vector<ACharacter *>
#posCharacters: vector<string>
#sounds: vector<string>
+initGame()
+gereKey(InputButton)
+playGame(string): bool
+pause()
+loadTextures(IDynLib *)
+getLifes(): int
+getCharacters(): vector<ACharacter *>
+getPosCharacters(): vector<string>
+getScore(): int
+getMapWidth(): int
+getMapHeight(): int
+getHighScores(): vector<string>
+getMapi(): int *
+getMap(): Map
+setAccurary(int)
+getName(): string
+getTypesSound(): vector<string>
+setInit(bool)
Map
-map: int *
-width: int
-height: int
+loadMap(string): int
+fillMap(vector<string>)
+getWidth(): int
+getHeight(): int
+getMap(): int *
+copyMap(): int *
Score
+score: int
+strings: vector<string>
+add(int)
+reset()
+load(string, int)
+save(string, string)
+getScore()
+getHighScores(): vector<string>
ACharacter
#posX: int
#posY: int
#startX: int
#startY: int
#name: string
#button: InputButton
+getPosX(): int
+getPosY(): int
+getStartX(): int
+getStartY(): int
+setIdButton(InputButton)
+getIdButton(): InputButton
+getName(): string
+setPosX(int)
+setPosY(int)
+setStartX(int)
+setStartY(int)
+getSprite(): string
Nibbler
-init: bool
-timeEnable: bool
-hasCollision: bool
+button: InputButton
+initGame()
+gereKey(InputButton, Parameter1)
+playGame(string): bool
+pause()
+loadTextures(IDynLib *)
+getLifes(): int
+getCharacters(): vector<ACharacter *>
+getPosCharacters(): vector<string>
+setTimeEnable(bool)
-initSnakeEntity()
-move()
-gereCollisionX(int)
-gereCollisionY(int)
-fruitPop()
-checkFruit()
-setInit(bool)
Centipede
-mushs: vector<ACharacter *>
-centis: vector<CentipedeEntity *>
-init: bool
-lifes: int
-timeEnable: bool
+initGame()
+gereKey(InputButton, Parameter1)
+playGame(string): bool
+pause()
+loadTextures(IDynLib *)
+getLifes(): int
+getCharacters(): vector<ACharacter *>
+getPosCharacters(): vector<string>
+setTimeEnable(bool)
+reInitGame(bool)
+initPlayer()
+initMushs()
+initCenti()
+checkCollisionMushs(int, int, vector<ACharacter *>): int
+moveShoot()
+moveCentis()
+checkCollisionShoot()
+checkCollisionPlayer(): int
+setTimeEnable(bool)
+getMap()
+setInit()
PacMan
-init: bool
-time_enabled: bool
+initGame()
+gereKey(InputButton, Parameter1)
+playGame(string): bool
+pause()
+loadTextures(IDynLib *)
+getLifes(): int
+getCharacters(): vector<ACharacter *>
+getPosCharacters(): vector<string>
+setTimeEnabled(): bool
+setSpeedCharacters(int)
-gereCollisionX(int): bool
-gereCollisionY(int): bool
-move()
-initGhosts()
+initPacmanEntity()
-changeGhostBehavior(int, int)
-noMorePacGum()
-changeStateCharacters()
-GhostNoMoreFrightened()
-calcPosGhosts()
-reLaunchGame()
-hasEatGhost(vector<Ghost *>)
-manageScore()
-setInit(bool)
PacmanEntity
-state: bool
-lifes: int
-way: InputButton
-speed: int
+getState(): bool
+setState(bool)
+getLifes(): bool
+setLifes(bool)
+getWay(): InputButton
+setWay(InputButton)
+getSprite(): string
+touchedGhost(vector<Ghost *>): bool
+setSpeed(int)
Fruit
-state: int
+setState(int)
+getState(): int
+getSprite(): string
Ghost
-state: bool
-frightened: bool
-search: bool
-changeDir: bool
-inBox: bool
-stateFrightened: bool
-hasBeenEat: bool
-eat: bool
-way: InputButton
-speed: int
+isWalkable(int): bool
+isFrightened(): bool
+move(Map, InputButton, int)
+findNextMove(Map, int, int, int)
+setState(bool)
+getState(): bool
+gereCollisionX(Map, int, int): bool
+gereCollisionY(Map, int, int): bool
+goalRed(Map, PacmanEntity, int)
+goalPink(Map, PacmanEntity, int, InputButton)
+goalBlue(PacmanEntity, Ghost, Map, int)
+goalMarron(PacmanEntity, Map, int)
+setSearch(bool)
+setFrightened(bool)
+randomMove(Map, int)
+setHasBeenEat(bool)
+isEaten(): bool
+setStateFrightened(bool)
+getStateFrightened(): bool
+setInBox(bool)
+getSprite(): string
+getWay(): InputButton
+getTimer(): Timer *
+beingEat(): bool
+setBeingEat(bool)
+setSpeed(int)
SnakeBody
-state: int
+setState(int)
+getState(): int
+getSprite(): string
SnakeEntity
-way: InputButton
-body: vector<SnakeBody *>
+setWay(InputButton)
+initSnakeBody(int, int, int)
+move(int, int)
+checkCollision(int, int): bool
+growUp(int)
+getBody(): vector<SnakeBody *>
+getPosX(): int
+getPosY(): int
+getWay(): InputButton
-getDirection(int, int, int, int): int
-mouvement()
Timer
+pause: bool
+t1: struct timeval
+t0: struct timeval
+startTime()
+resetTime()
+getTimeElapsed(): int
+getTimeElapsedSec(): int
+paused(bool)
+isPaused(): bool
Mushroom
-sprite: string
+getSprite(): string
+decrLife(): int
CentipedeEntity
-way: InputButton
-body: vector<CentipedeBody *>
+getBody(): vector<ACharacter *>
+initPos(int)
+moveTo(int, int)
+isEmpty(): bool
+move(Map, int, vector<ACharacter *>)
+checkMushs(int, int, vector<ACharacter *>)
+getPosX()
+getPosY()
+getSplitted(int, int): CentipedeEntity *
+getWay(): InputButton
-getDirection(int, int, int, int): int
CentipedeBody
-sprite: string
+setSprite(string)
+getSprite(): string
Shoot
+getSprite(): string
Ship
+createShoot(int): Shoot *
+getSprite(): string