ParseXML

-xmlData: map<string, string,

+<<const>> getXmlData(): const map<string, string>

Class Gamedata has a map that contains constants and numeric values used by most of the classes in the animation.

Gamedata

-gameData: map<string, string>
-parser: ParseXML

+displayData(): void
+getXmlInt(): int
+getXmlFloat(): float
+getXmlStr(): string

<>
Drawable

-position: Vector2f
-velocity: Vector2f
-maxVelocity: Vector2f
+name: string

+<<const>> draw(): void
+update(): void
+X(): float
+X(x:float): void
+Y(): float
+Y(y:float): void
+getPosition(): Vector2f
+getVelocity(): Vector2f
+setPosition(pos:const Vector2f&): void
+setVelocity(vel:const Vector2f&): void

main()

Manager

-screen: SDL_Surface* const
-orbFrame: Frame*
-orb: Sprite
-backFrame: Frame*
-back: Sprite

+play(): void
-drawBackground(): void

Sprite

#frame: Frame* const

+<<constructor>> Sprite(pos:Vector2f,vel:Vector2f,
                       n:string,f:const Frame*)
+<<constructor>> Sprite(s:const Sprite&)
+<<const>> draw(): void
+update(): void
+<<const>> getFrame(): void
+deleteFrame(): void
#getDistance(const Sprite*): int

MultiSprite

+vector<Frame*> frames

+<<constructor>> Sprite(pos:Vector2f,vel:Vector2f,
                       n:string,f:const Frame*)
+<<constructor>> Sprite(s:const Sprite&)
+<<const>> draw(): void
+update(): void
+<<const>> getFrame(): void
+deleteFrame(): void
#getDistance(const Sprite*): int

<<singleton>>
Clock

-started: bool
-paused: bool
-frames: unsigned
-timeAtStart: unsigned
-timeAtPause: unsigned
-ticks: unsigned
-currTicks: unsigned
-prevTicks: unsigned

+getInstance(): Clock&
+<<const>> getTicks(): unsigned

<<getFrame>>

World - -> Viewport

<<getFrames>>

FrameFactory

+getFrame()
+getFrames()

<<singleton>>
IOManager

-screen: SDL_Surface
-font: TTF_Font

+getInstance(): static IOManager&
+getScreen(): SDL_Surface*
+loadAndSet(filename:const string&): SDL_Surface*
+printMessageAt(msg:const string&,x:Uint32,
               y:Uint32): void

The IOManager class is a singleton that's used by most of the classes in the animation. Most importantly, it contains the screen, on which everything is drawn. There should, normally, be one screen in the animation. IOManager is also useful for loading sprite surfaces, and printing text on the screen

Frame

-screen: const SDL_Surface
-spriteSurface: const SDL_Surface
-spriteWidth: const Sint16
-spriteHeight: const Sint16
-spriteSourceX: Uint16
-spriteSourceY: Uint16

+<<constructor>> Frame(fn:const string&)
+<<const>> draw(x:Sint16,y:Sint16): void
+<<const>> getWidth(): Sint16
+<<const>> getHeight(): Sint16
+Frame(spr:SDL_Surface,sprite_width:Uint16,
       sprite_height:Uint16,src_x:Uint16,
       src_y:Uint16): constructor

ExtractSurface

+<<const>> get(): SDL_Surface
-getPixel(surf:SDL_Surface*): Uint32
-putPixel(surf:SDL_Surface*,x:int,y:int,
          pixel:Uint32): void

SDL_Surface* spriteSurface

Figure 1: **Class Diagram of a framework for animations**.

Data-driven 2D Game Development
Computer Science Division, Clemson University
Game Framework
Brian Malloy, PhD
September 19, 2016