02 SDL Runtime

Functional and Implementation Guidelines

Functional Guidelines

1. Support run-time loading/unloading of all SDL-subsystems needed for the project
   1. SDL\_ttf library
   2. SDL\_image library
   3. SDL\_mixer library
   4. Video, Audio, Timer sub-system
2. Implement support for basic shapes - Point and Rectangle
3. Implement a helper set of functionalities for loading/destroying SDL\_Surface’s (CPU image primitives)
4. Implement a SDL\_window wrapper class
   1. It should hide “internal” SDL specific logic
5. Refactor your application to use all existing functionalities

Implementation Guidelines

1. Create the sdl\_utils (folder, soon to be static library)
   1. Implement sdl\_utils CMakeLists.txt file
   2. move the find\_package from the root CMakeLists.txt file in the sdl\_utils one
   3. move the direct dependencies for the utils library from the CMakeLists.txt file in the sdl\_utils one
   4. target\_include\_directories(${CMAKE\_CURRENT\_SOURCE\_DIR}/include utils PUBLIC)
   5. SDL\_INCLUDE\_DIR and other SDL\_Related should be PRIVATE but put them as PUBLIC with a TODO to revise them later on
   6. target\_link\_libraries(...same as the include)
2. Implement SDLLoader set of functionalities - init() and deinit() all SDL\_Related sub-systems
3. init()
   1. TTF\_Init()
   2. SDL\_Init(SDL\_INIT\_VIDEO)
   3. IMG\_Init

| const int32\_t imgFlags = IMG\_INIT\_PNG;  if (! (IMG\_Init(imgFlags) & imgFlags)) {  LOGERR("SDL\_image could not be initialised! SDL\_image Error: %s",  IMG\_GetError());  return FAILURE;  } |
| --- |

* 1. SDL\_Init(SDL\_INIT\_TIMER)
  2. SDL\_Init(SDL\_INIT\_AUDIO)

| if (0 > Mix\_OpenAudio(44100, //sound frequency  MIX\_DEFAULT\_FORMAT, //sample format  2, //stereo hardware channels  2048)) { //chunk size  LOGERR("SDL\_mixer could not be initialised! SDL Error: %s", Mix\_GetError());  return FAILURE;  } |
| --- |

1. deinit()

| IMG\_Quit();  TTF\_Quit();  Mix\_Quit();  SDL\_Quit(); |
| --- |

1. **SDL\_Coordinate system Lecture**
2. Create a ‘drawing’ folder in the utils library
   1. Implement a Point functionality. Create POINT\_ZERO, and POINT\_UNDEFINED
   2. Implement a Rectangle functionality. Create RECTANGLE\_ZERO, and RECTANGLE\_UNDEFINED
      1. bool isPointInRect(const struct Point \*point) //include two of the sides
3. Implement MonitorWindow struct

| struct MonitorWindow {  SDL\_Window \*sdlWindow;  struct Rectangle windowRect;  }; |
| --- |

1. Implement config for the MonitorWindow

| struct MonitorWindowCfg {  //Window modes:  //SDL\_WINDOW\_SHOWN - for windowed version  //SDL\_WINDOW\_FULLSCREEN\_DESKTOP - for fullscreen  int32\_t displayMode;  int32\_t windowWidth;  int32\_t windowHeight;  struct Point windowPos;  char \*windowName;  }; |
| --- |

1. Refactor the main.c file to use the MonitorWindow
   1. all methods should be static
   2. Add hello.png image
2. Start the implementation of sdl\_utils/Texture set of functions.
   1. //Forward declarations

typedef struct SDL\_Surface SDL\_Surface;

* 1. loadSurfaceFromFile()
  2. freeSurface()

1. Refactor the main.c file again