# 1. Các hàm:

|  |
| --- |
| //Khởi động SDL và tạo window  bool init();  //Tải media  bool loadMedia();  //Giải phóng media và tắt SDL  void close(); |

# 2. Kiểu dữ liệu mới:

* *Render (kết xuất đồ họa):* là 1 hành động, quá trình kiến tạo các mô hình thành 1 hình ảnh hay 1 cảnh phim.
* *SDL\_Surface* là 1 loại dữ liệu ảnh, chứa các pixels của 1 ảnh cùng những dữ liệu cần thiết để render.
* *SDL surface* có thể dùng CPU để render, cũng có thể dùng hình ảnh từ hardware để render (học sau vì khó).
* Chúng ta sử dụng **pointers** đến SDL\_surfaces vì:
* Có bộ nhớ động để tải hình ảnh
* Pointers tốt hơn trong việc xử lý địa chỉ
* **Chú ý phải khởi tạo NULL.**

|  |
| --- |
| // The window we'll be rendering to  SDL\_Window \*gWindow = NULL;  // The surface contained by the window  SDL\_Surface \*gScreenSurface = NULL;  // The image we will load and show on the screen  SDL\_Surface \*gHelloWorld = NULL; |

# 3. Hàm khởi tạo

* int SDL\_Init(flag): Khởi tạo thư viện SDL. Trả về 0 nếu thành công. Trả về < 0 là lỗi.
* Flag cho SDL\_Init(): <https://wiki.libsdl.org/SDL2/SDL_Init>
* const char\* SDL\_GetError(): Truy xuất 1 thông báo về lỗi cuối cùng trên luồng hiện tại.
* SDL\_Window\* SDL\_CreateWindow(): Tạo 1 window. Trả về window được tạo hoặc NULL nếu lỗi.
* SDL\_Surface\* SDL\_GetWindowSurface(): Nhận SDL\_surface được liên kết với window. Trả về NULL nếu lỗi.

Các bước khởi tạo:

* Khởi tạo SDL
* Tạo window
* Lấy bề mặt window (window surface)

|  |
| --- |
| bool init()  {      // Initialization flag      bool success = true;      // Initialize SDL      if (SDL\_Init(SDL\_INIT\_VIDEO) < 0)      {          printf("SDL could not initialize! SDL\_Error: %s\n", SDL\_GetError());          success = false;      }      else      {          // Create window          gWindow = SDL\_CreateWindow("SDL Tutorial", SDL\_WINDOWPOS\_UNDEFINED, SDL\_WINDOWPOS\_UNDEFINED, SCREEN\_WIDTH, SCREEN\_HEIGHT, SDL\_WINDOW\_SHOWN);          if (gWindow == NULL)          {              printf("Window could not be created! SDL\_Error: %s\n", SDL\_GetError());              success = false;          }          else          {              // Get window surface              gScreenSurface = SDL\_GetWindowSurface(gWindow);          }      }      return success;  } |

* Chúng ta vừa khởi tạo SDL và tạo window, và cho vào 1 hàm.

# 4. Tải media

* SDL\_Surface\* SDL\_LoadBMP(): Tải ảnh BMP từ đường dẫn.

|  |
| --- |
| bool loadMedia()  {      // Loading success flag      bool success = true;      // Load splash image      gHelloWorld = SDL\_LoadBMP("src/hello\_world.bmp");      if (gHelloWorld == NULL)      {          printf("Unable to load image %s! SDL Error: %s\n", "src/hello\_world.bmp", SDL\_GetError());          success = false;      }      return success;  } |

* Chú ý: đường dẫn là đường dẫn đến ảnh từ thư mục hiện tại mà ứng dụng đang hoạt động.

# 5. Đóng ứng dụng

Sau khi kết thúc, chúng ta phải hủy window và đóng SDL nhưng chúng ta phải “chăm sóc” cái surface ta đã tải.

* Ta sẽ giải phóng nó với void SDL\_FreeSurface().
* Còn surface màn hình, ta sẽ sử dụng void SDL\_DestroyWindow() để hủy cửa sổ.

Hãy tạo thói quen gán con trỏ bằng NULL khi nó không trỏ đến thứ gì.

Tại sao phải giải phóng tài nguyên sau khi kết thúc chương trình ư? Tôi không biết. Nhưng chúng ta hãy tránh nhưng hành vi không xác định (undefined behavior), và làm quen với nó để tiện cho việc code sau này.

|  |
| --- |
| void close()  {      // Deallocate surface      SDL\_FreeSurface(gHelloWorld);      gHelloWorld = NULL;      // Destroy window      SDL\_DestroyWindow(gWindow);      gWindow = NULL;      // Quit SDL subsystems      SDL\_Quit();  } |

# 6. Hàm main

|  |
| --- |
| int main(int argc, char \*args[])  {      // Start up SDL and create window      if (!init())      {          printf("Failed to initialize!\n");      }      else      {          // Load media          if (!loadMedia())          {              printf("Failed to load media!\n");          }          else          {              // Apply the image              SDL\_BlitSurface(gHelloWorld, NULL, gScreenSurface, NULL); |

* Đầu tiên ta sẽ khởi tạo SDL và load ảnh.
* int SDL\_BlitSurface() **dán bản sao (blit)** của surface này sang surface khác. Ở đây ta sẽ dán surface được load sang surface màn hình. Trả 0 nếu thành cồng, < 0 nếu thất bại
* Tiếp theo, ta cần hiện ảnh này lên màn hình:

|  |
| --- |
| // Update the surface              SDL\_UpdateWindowSurface(gWindow); |

* Sau khi vẽ mọi thứ cần hiện trong khung hình này, chúng ta cần phải update màn hình
* int SDL\_UpdateWindowSurface(): Copy từ window surface sang màn hình. Trả về 0 nếu thành công, < 0 nếu thất bại.
* Khi chúng ta vẽ lên màn hình, ta cơ bản không phải vẽ lên cái ảnh trên màn hình ta thấy. Mặc định, hầu hết hệ thống render có 2 bufer: 1 front buffervà 1 back buffer. Và SDL\_BlitSurface() render buffer sau. Còn những gĩ ta thấy trên màn hình là buffer trước.
* Lý do ta phải làm như vậy là vì nhiều khung hình yêu cầu vẽ nhiều objects lên màn hình. Nếu ta chỉ dùng front buffer, ta sẽ thấy khung hình với những thứ được vẽ lên, tức là ko hoàn chỉnh. Vì thế, ta cần vẽ everything vào back bufer trước đã, sau đó mới đưa back buffer lên front buffer, ta đc khung hình hoàn chỉnh.
* Điều này cũng không có nghĩa là ta cần gọi SDL\_UpdateWindowSurface() sau mỗi lần blit, chỉ cần sau khi tất cả blits cho khung hình đấy đã xong mà thôi.

|  |
| --- |
| // Hack to get window to stay up              SDL\_Event e;              bool quit = false;              while (quit == false)              {                  while (SDL\_PollEvent(&e))                  {                      if (e.type == SDL\_QUIT)                          quit = true;                  }              }          }      }      // Free resources and close SDL      close(); |

* Vì chúng ta đã render mọi thứ lên window, nên chúng ta phải hack delay để window không biến mất.
* SDL\_Event là cấu trúc dữ liệu mới cho các sự kiện <https://wiki.libsdl.org/SDL2/SDL_Event>
* int SDL\_PollEvent(): xem có sự kiện đang chờ xử lý không. Trả về 1/0 nếu có/không