Một phần quan trọng của bất kỳ loại API game nào là khả năng xử lý thời gian. Trong bài viết này chúng ta sẽ làm một bộ đếm thời gian (timer) mà chúng ta có thể khởi động lại.

|  |
| --- |
| //Using SDL, SDL\_image, SDL\_ttf, standard IO, strings, and string streams  #include <SDL.h>  #include <SDL\_image.h>  #include <SDL\_ttf.h>  #include <stdio.h>  #include <string>  #include <sstream> |

* Trong bài viết này ta sẽ dùng stringstream.

|  |
| --- |
| bool loadMedia()  {      //Loading success flag      bool success = true;      //Open the font      gFont = TTF\_OpenFont( "22\_timing/lazy.ttf", 28 );      if( gFont == NULL )      {          printf( "Failed to load lazy font! SDL\_ttf Error: %s\n", TTF\_GetError() );          success = false;      }      else      {          //Set text color as black          SDL\_Color textColor = { 0, 0, 0, 255 };            //Load prompt texture          if( !gPromptTextTexture.loadFromRenderedText( "Press Enter to Reset Start Time.", textColor ) )          {              printf( "Unable to render prompt texture!\n" );              success = false;          }      }      return success;  } |

* Như đã đề cập trong bài viết về font rendering, bạn sẽ phần giảm thiểu số lần render văn bản. Chúng ta sẽ có một texture để nhắc nhở input và một texture để display thời gian hiện tại trong milliseconds. Texture thời gian thay đổi mỗi khung hình, vì thế chúng ta phải render mỗi khung hình đó. Nhưng texture nhắc nhở không đổi, vì thế chúng ta có thể render nó một lần trong hàm load file.

|  |
| --- |
| //Main loop flag              bool quit = false;              //Event handler              SDL\_Event e;              //Set text color as black              SDL\_Color textColor = { 0, 0, 0, 255 };              //Current time start time              Uint32 startTime = 0;              //In memory text stream              std::stringstream timeText; |

* Trước khi chúng ta vào main loop, chúng ta sẽ khai báo vài biến. Hai cái mà chúng ta cần chú ý là biến startTime (uint32\_t) và biến timeText (stringstream).

|  |
| --- |
| //While application is running  while( !quit )  {      //Handle events on queue      while( SDL\_PollEvent( &e ) != 0 )      {          //User requests quit          if( e.type == SDL\_QUIT )          {              quit = true;          }          //Reset start time on return keypress          else if( e.type == SDL\_KEYDOWN && e.key.keysym.sym == SDLK\_RETURN )          {              startTime = SDL\_GetTicks();          }      } |

* Có hàm SDL\_GetTicks() trả lại thời gian từ lúc chương trình chạy trong milliseconds. Trong bản demo này, chúng ta sẽ có một bộ đếm thời gian mà bắt đầu mỗi khi chúng ta nhấn phím quay lại.
* Nhớ cách chúng ta khởi tạo thời gian bắt đầu là 0 lúc bắt đầu chương trình? Điều này nghĩa là thời gian của bộ đếm chỉ là thời gian hiện tại kể từ lúc chương trình bắt đầu được trả về bởi SDL\_GetTicks(). Nếu chúng ta khởi động lại bộ đếm giờ khi SDL\_GetTicks() ở 5000 ms, thì tại 10000 ms: (current time – start time) = 5000 ms. Vì vậy mặc dù bộ đếm trong SDL\_GetTicks() chưa được khởi động lại, nhưng chúng tôi có thể đặt bộ đếm thời gian theo dõi thời gian bắ đầu tương đối và đặt lại thời gian bắt đầu của nó.

|  |
| --- |
| //Set text to be rendered  timeText.str( "" );  timeText << "Milliseconds since start time " << SDL\_GetTicks() - startTime; |

Ở đây chúng ta sử dụng string stream. Đầu tiên, chúng ta dùng lệnh .str() với xâu rỗng "" để khởi tạo nó rỗng. Sau đó chúng ta dùng nó như cout và in vào nó "Milliseconds since start time " và thời gian hiện tại trừ thời gian tương đối. Vì vậy, nó sẽ in thời gian kể từ khi chúng ta bắt đầu bộ đếm giờ.

|  |
| --- |
| //Render text  if( !gTimeTextTexture.loadFromRenderedText( timeText.str().c\_str(), textColor ) )  {      printf( "Unable to render time texture!\n" );  } |

Vì bây giờ chúng ta có thời gian trong string stream, chúng ta có thể lấy một string từ stringstream và dùng nó để render thời gian hiện tại tới texture.

|  |
| --- |
| //Clear screen  SDL\_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );  SDL\_RenderClear( gRenderer );  //Render textures  gPromptTextTexture.render( ( SCREEN\_WIDTH - gPromptTextTexture.getWidth() ) / 2, 0 );  gTimeTextTexture.render( ( SCREEN\_WIDTH - gPromptTextTexture.getWidth() ) / 2, ( SCREEN\_HEIGHT - gPromptTextTexture.getHeight() ) / 2 );  //Update screen  SDL\_RenderPresent( gRenderer ); |

Cuối cùng, chúng ta sẽ render cái prompt texture và time texture tới màn hình.