Để có thể lưu và load dữ liệu cần thiết để giữ data giữa các phiên chơi. SDL\_RWops xử lý file sẽ giúp chúng ta thực hiện in/out tệp đa nền tảng để lưu data.

|  |
| --- |
| //Data points  Sint32 gData[ TOTAL\_DATA ]; |

* Ở đây chúng ta khai báo một mảng số nguyên có dấu 32 bit. Cái này sẽ là dữ liệu mà chúng ta sẽ load và lưu lại. Trong bài demo này, mảng sẽ có độ dài 10.

|  |
| --- |
| //Open file for reading in binary      SDL\_RWops\* file = SDL\_RWFromFile( "nums.bin", "r+b" ); |

* Trong hàm load đa phương tiện, chúng ta sẽ mở file lưu để đọc sử dụng. Đối số đầu tiên là đường dẫn đến file và đối số thứ hai định nghĩ cách chúng ta sẽ mở nó. "r+b" nghĩa là nó đang được mở để đọc nhị phân.

|  |
| --- |
| //File does not exist      if( file == NULL )      {          printf( "Warning: Unable to open file! SDL Error: %s\n", SDL\_GetError() );            //Create file for writing          file = SDL\_RWFromFile( "nums.bin", "w+b" ); |

Bây giờ, nếu file không tồn tại, điều này không chính xác là lỗi. Nó có thể là đây là lần đầu tiên chương trình chạy và file chưa được tạo. Nếu file không tồn tại, chúng ta prompt một cảnh báo và tạo một file bằng cách mở một file với "w+b". Điều này sẽ mở một file để ghi nhị phân.

|  |
| --- |
| if( file != NULL )          {              printf( "New file created!\n" );              //Initialize data              for( int i = 0; i < TOTAL\_DATA; ++i )              {                  gData[ i ] = 0;                  SDL\_RWwrite( file, &gData[ i ], sizeof(Sint32), 1 );              }                //Close file handler              SDL\_RWclose( file );          }         else          {              printf( "Error: Unable to create file! SDL Error: %s\n", SDL\_GetError() );              success = false;          } |

Nếu một file được tạo thành công, chúng ta bắt đầu ghi dữ liệu khởi tạo lên nó sử dụng SDL\_RWwrite. Đối số thứ nhất là file chúng ta đang ghi, đối số thứ hai là địa chỉ của đối tượng trong bộ nhớ chúng ta cần ghi, đối số thứ ba là số bytes mỗi đối tượng chúng ta đang ghi, và cuối cùng là số đối tượng chúng ta đang ghi. Sau khi chúng ta ghi xong tất cả các đối tượng, chúng ta đóng file đã ghi bằng SDL\_RWclose.

Nếu file không được tạo, chungs ta báo lỗi tới console và đặt success flag sang false.

|  |
| --- |
| //File exists      else      {          //Load data          printf( "Reading file...!\n" );          for( int i = 0; i < TOTAL\_DATA; ++i )          {              SDL\_RWread( file, &gData[ i ], sizeof(Sint32), 1 );          }          //Close file handler          SDL\_RWclose( file );      } |

* Nếu chúng ta load thành công trong lần đầu, tất cả những gì chúng ta cần làm là đọc dữ liệu sử dụng SDL\_RWread. Hàm này cơ bản có tác dụng như SDL\_RWwrite nhưng ngược lại.

|  |
| --- |
| //Initialize data textures      gDataTextures[ 0 ].loadFromRenderedText( std::to\_string( gData[ 0 ] ), highlightColor );      for( int i = 1; i < TOTAL\_DATA; ++i )      {          gDataTextures[ i ].loadFromRenderedText( std::to\_string( gData[ i ] ), textColor );      } |

* Sau khi file được load, chúng ta render text texture mà liên quan đến từng con số của chúng ta. Hàm loadFromRenderedText chỉ chấp nhận string nên chúng ta file chuyển số nguyên thành string.

|  |
| --- |
| void close()  {      //Open data for writing      SDL\_RWops\* file = SDL\_RWFromFile( "nums.bin", "w+b" );      if( file != NULL )      {          //Save data          for( int i = 0; i < TOTAL\_DATA; ++i )          {              SDL\_RWwrite( file, &gData[ i ], sizeof(Sint32), 1 );          }          //Close file handler          SDL\_RWclose( file );      }      else      {          printf( "Error: Unable to save file! %s\n", SDL\_GetError() );      } |

Khi chúng ta đóng màn hình, chúng ta mở file lên lại để ghi và đọc tất cả dữ liệu.

|  |
| --- |
| //Main loop flag              bool quit = false;              //Event handler              SDL\_Event e;              //Text rendering color              SDL\_Color textColor = { 0, 0, 0, 0xFF };              SDL\_Color highlightColor = { 0xFF, 0, 0, 0xFF };              //Current input point              int currentData = 0; |

* Trước khi vào main loop, chúng ta khai báo currentData để theo dõi những số nguyên ta nhập vào thay thế. Chúng ta cũng khai báo cả màu của văn bản thuần túy và màu của văn bản đang hightlight.

|  |
| --- |
| else if( e.type == SDL\_KEYDOWN )                      {                          switch( e.key.keysym.sym )                          {                              //Previous data entry                              case SDLK\_UP:                              //Rerender previous entry input point                              gDataTextures[ currentData ].loadFromRenderedText( std::to\_string( gData[ currentData ] ), textColor );                              --currentData;                              if( currentData < 0 )                              {                                  currentData = TOTAL\_DATA - 1;                              }                                //Rerender current entry input point                              gDataTextures[ currentData ].loadFromRenderedText( std::to\_string( gData[ currentData ] ), highlightColor );                              break;                                //Next data entry                              case SDLK\_DOWN:                              //Rerender previous entry input point                              gDataTextures[ currentData ].loadFromRenderedText( std::to\_string( gData[ currentData ] ), textColor );                              ++currentData;                              if( currentData == TOTAL\_DATA )                              {                                  currentData = 0;                              }                                //Rerender current entry input point                              gDataTextures[ currentData ].loadFromRenderedText( std::to\_string( gData[ currentData ] ), highlightColor );                              break; |

* Khi chúng ta nhấn up hay down, chúng ta sẽ render data cũ ở màu thuần túy, di chuyển đến data tiếp theo (với vài kiểm tra giới hạn), và render dât hiện tại với màu highlight.

|  |
| --- |
| //Decrement input point                              case SDLK\_LEFT:                              --gData[ currentData ];                              gDataTextures[ currentData ].loadFromRenderedText( std::to\_string( gData[ currentData ] ), highlightColor );                              break;                                //Increment input point                              case SDLK\_RIGHT:                              ++gData[ currentData ];                              gDataTextures[ currentData ].loadFromRenderedText( std::to\_string( gData[ currentData ] ), highlightColor );                              break;                          } |

* Khi chúng ta nhấn left hoặc right, chúng ta tang hoặc giảm data hiện tại và render texture liên kết với nó.

|  |
| --- |
| //Clear screen  SDL\_SetRenderDrawColor( gRenderer, 0xFF, 0xFF, 0xFF, 0xFF );  SDL\_RenderClear( gRenderer );  //Render text textures  gPromptTextTexture.render( ( SCREEN\_WIDTH - gPromptTextTexture.getWidth() ) / 2, 0 );  for( int i = 0; i < TOTAL\_DATA; ++i )  {      gDataTextures[ i ].render( ( SCREEN\_WIDTH - gDataTextures[ i ].getWidth() ) / 2, gPromptTextTexture.getHeight() + gDataTextures[ 0 ].getHeight() \* i );  }  //Update screen  SDL\_RenderPresent( gRenderer ); |

* Ở cuối main loop, chúng ta render tất cả texture lên màn hình.