

Giống với nhấn phím, SDL có CTDL sự kiện để xử lú mouse event như di chuột, nhấn chuột và nhả chuột. Trong bài viết này, chúng ta sẽ tạo ra các nút mà ta có thể tương tác.

|  |
| --- |
| // Button constants  const int BUTTON\_WIDTH = 300;  const int BUTTON\_HEIGHT = 200;  const int TOTAL\_BUTTONS = 4;  enum LButtonSprite  {      BUTTON\_SPRITE\_MOUSE\_OUT = 0,      BUTTON\_SPRITE\_MOUSE\_OVER\_MOTION = 1,      BUTTON\_SPRITE\_MOUSE\_DOWN = 2,      BUTTON\_SPRITE\_MOUSE\_UP = 3,      BUTTON\_SPRITE\_TOTAL = 4  }; |

* Trong bài viết này, chúng ta có 4 nút trên màn hình. Dựa vào chuột đang ở đâu, nhấn chuột, nhả chuột, chúng ta sẽ display các sprite khác nhau. Những hằng số này được định nghĩa để làm điều đó.

|  |
| --- |
| // Texture wrapper class  class LTexture  {  public:      …  #if defined(SDL\_TTF\_MAJOR\_VERSION)      // Creates image from font string      bool loadFromRenderedText(std::string textureText, SDL\_Color textColor);  #endif      …  private:      …  }; |

* Chúng ta có một vài sửa đổi nhỏ cho texture class. Trong bài này, chúng ta sẽ không dùng SDL\_ttf, nghĩa là không dùng hàm loadFromRenderedText(). Nhưng thay vì xóa nó đi nhỡ mà sau này cần, chúng ta đặt nó trong if define để compiler bỏ qua nó nếu chúng ta ko include SDL\_ttf. Nó sẽ kiểm tra xem macro SDL\_TTF\_MAJOR\_VERSION có được định nghĩa không.

|  |
| --- |
| // The mouse button  class LButton  {  public:      // Initializes internal variables      LButton();      // Sets top left position      void setPosition(int x, int y);      // Handles mouse event      void handleEvent(SDL\_Event \*e);      // Shows button sprite      void render();  private:      // Top left position      SDL\_Point mPosition;      // Currently used global sprite      LButtonSprite mCurrentSprite;  }; |

* Đây là class đại diện cho một button. Nó có một **constructor** để khởi tạo, một **setter** vị trí, một **event handler** cho event loop, và **hàm render**. Nó cũng có vị trí và sprite enumeration để chúng ta biết được sprite nào sẽ được render cho button.

|  |
| --- |
| LButton::LButton()  {      mPosition.x = 0;      mPosition.y = 0;      mCurrentSprite = BUTTON\_SPRITE\_MOUSE\_OUT;  }  void LButton::setPosition(int x, int y)  {      mPosition.x = x;      mPosition.y = y;  } |

* Đây là constructor cho button và thiết lập vị trí. Như bạn có thể thấy, chúng khởi tạo là sprite mặc định và đặt vị trí.

|  |
| --- |
| void LButton::handleEvent(SDL\_Event \*e)  {      // If mouse event happened      if (e->type == SDL\_MOUSEMOTION || e->type == SDL\_MOUSEBUTTONDOWN || e->type == SDL\_MOUSEBUTTONUP)      {          // Get mouse position          int x, y;          SDL\_GetMouseState(&x, &y); |

* Đây là trọng tâm của bài này, nơi mà chúng tôi xử lý mouse events. Hàm này sẽ được gọi trong event lôp và sẽ xử lý sự kiện lấy từ hàng đợi sự kiện cho từng button.
* Đầu tiên, chúng ta sẽ kiểm tra xem liệu event tiếp theo trong mouse event, đặc biệt là event di chuột, nhấn chuột, nhả chuột.
* Nếu một trong những event diễn ra, chúng ta kiểm tra vị trí chuột sử dụng SDL\_GetMouseState(). Phụ thuộc vào cuột có ở trên nut hay không, chúng ta sẽ display các sprite khác nhau.

|  |
| --- |
| // Check if mouse is in button  bool inside = true;  // Mouse is left of the button  if (x < mPosition.x)  {      inside = false;  }  // Mouse is right of the button  else if (x > mPosition.x + BUTTON\_WIDTH)  {      inside = false;  }  // Mouse above the button  else if (y < mPosition.y)  {      inside = false;  }  // Mouse below the button  else if (y > mPosition.y + BUTTON\_HEIGHT)  {      inside = false;  } |

* Ở đây, chúng ta sẽ check xem chuột ở trong hay ở ngoài nút. Vì chúng ta sử dụng hệ tọa độ khác với SDL nên bắt đầu của một button là ở trái trên. Điều này nghĩa là mỗi x nhỏ hơn x position thì nằm ngoài button và mỗi y nhỏ hơn y position thì nằm ngoài button….
* Đó là cái mà đoạn code này làm. Nếu vị trí chuột ở ngoài nút, thì inside sẽ là false. Ngược lại sẽ là true.

|  |
| --- |
| // Mouse is outside button  if (!inside)  {      mCurrentSprite = BUTTON\_SPRITE\_MOUSE\_OUT;  }  // Mouse is inside button  else  {      // Set mouse over sprite      switch (e->type)      {      case SDL\_MOUSEMOTION:          mCurrentSprite = BUTTON\_SPRITE\_MOUSE\_OVER\_MOTION;          break;      case SDL\_MOUSEBUTTONDOWN:          mCurrentSprite = BUTTON\_SPRITE\_MOUSE\_DOWN;          break;      case SDL\_MOUSEBUTTONUP:          mCurrentSprite = BUTTON\_SPRITE\_MOUSE\_UP;          break;      }  } |

* Cuối cùng, ta thiết lập button sprite phụ thuộc vào chuột có ở trong button hay không và mouse event.

|  |
| --- |
| void LButton::render()  {      // Show current button sprite      gButtonSpriteSheetTexture.render(mPosition.x, mPosition.y, &gSpriteClips[mCurrentSprite]);  } |

* Trong hàm render, chúng ta chỉ render sprite hiện tại cho button.

|  |
| --- |
| // 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;          }          // Handle button events          for (int i = 0; i < TOTAL\_BUTTONS; ++i)          {              gButtons[i].handleEvent(&e);          }      }      // Clear screen      SDL\_SetRenderDrawColor(gRenderer, 0xFF, 0xFF, 0xFF, 0xFF);      SDL\_RenderClear(gRenderer);      // Render buttons      for (int i = 0; i < TOTAL\_BUTTONS; ++i)      {          gButtons[i].render();      }      // Update screen      SDL\_RenderPresent(gRenderer);  } |

* Đây là main loop của chúng ta. Trong event loop, chúng ta xử lý quit event và event cho tất cả button. Trong phần render, tất cả button đều được render tới màn hình.
* Cũng có cả [sự kiện cuộn chuột](http://wiki.libsdl.org/SDL_MouseWheelEvent) nhưng không được đề cập ở đây, nhưng bạn có thể tìm trong tài liệu và chơi với nó. Nó không quá khó để sử dụng.