**LÝ THUYẾT GAME CƠ BẢN**

1. **GAME 2D, CÁC THỂ LOẠI GAME**

* **Game 2D**

Game 2D là game không thể xoay góc quay và không có ấn tượng ba chiều rõ rệt. Game này cuộn bản đồ theo hai chiều là ngang và thẳng. Bên cạnh đó, từ nhân vật, tiền cảnh, hậu cảnh, … của game 2D giống phim hoạt hình cắt giấy (mọi thứ đều từ hình vẽ trên giấy).

Thiết kế game 2D là việc xây dựng game trên hệ thống đồ họa máy tính hai chiều (Dimensional). Đồ họa hai chiều có ít hoặc không có sự tham gia của các hiệu ứng ba chiều đặc trưng như ánh sáng, đổ bóng hay phản chiếu.

* **Các thể loại game:** 7 thể loại game phổ biến: RPG, RTS, FPS, SPORT, MOBA, SLG, CASUAL

1. *Game nhập vai – RPG (Role Playing Games)*

Được hiểu đơn giản là trò chơi nhập vai. Tại đây, người chơi được nhập vai trong một nhân vật cụ thể nào đó để chiến đấu, thực hiện nhiệm vụ theo thời gian thực hoặc theo lượt. RPG còn có hai biến thể:

* + MMORPG (Multi Massive Online Roleplaying Games). Trong đó, nhiều người chơi có thể tham gia kết hợp cùng một lúc.
  + ARPG (Action Role Playing Game) là tựa game nhập vai hành động.

1. *Game chiến thuật – RTS (Real Time Strategy)*

Được hiểu là chiến lược theo thời gian thực hay nói một cách dễ hiểu là “điều binh khiển tướng”. Những tựa game thuộc thể loại RTS thường có nội dung là xây dựng căn cứ, phát triển công nghệ, thu thập tài nguyên và phát triển quân đội.

1. *Game bắn súng – FPS (First Person Shooter)*

Được hiểu đơn giản là ngắm bắn súng. Trong game, người chơi sẽ được sở hữu những khẩu súng hoặc những vũ khí bất kỳ và việc của họ là ngắm bắn những đối thủ của mình. Do đó, trên màn hình, bạn chỉ thấy tay cầm vũ khí và đối phương mà không thể nhìn thấy toàn thân nhân vật mà người chơi điều khiển.

1. *Game thể thao – Sports*

Các tựa game này thường mô phỏng cách thức chơi, thi đấu của các môn thể thao hay các giải trên thực tế. Người chơi có thể hóa thân thành vận động viên hay người điều khiển cuộc thi đấu.

1. *Game Moba (Multiplayer online battle arena)*

Là thể loại game đấu trường trực tuyến nhiều người chơi. Tại đây, game thủ sẽ tham gia chiến đấu tại một khu vực và cùng đồng đội chiến đấu với một đội khác.

1. *Game mô phỏng – SLG (SimuLation Game)*

Một thể loại game mô phỏng – sao chép các hoạt động khác nhau trong đời sống thực dưới hình thức trò chơi, phục vụ mục đích khác nhau như đào tạo, phân tích hoặc dự đoán.

1. *Casual*

Đây là những trò chơi có những màn chơi ngắn, cách chơi đơn giản. Có nhiều tình tiết về giải đố, sắp xếp hay né chướng ngại vật, …

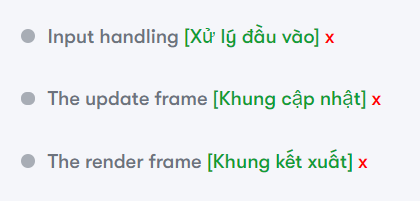
1. **GAME LOOP (vòng lặp game)**

Phần cốt lõi của hầu hết các game chính là vòng lặp được dùng để cập nhật và hiển thị trạng thái của game.

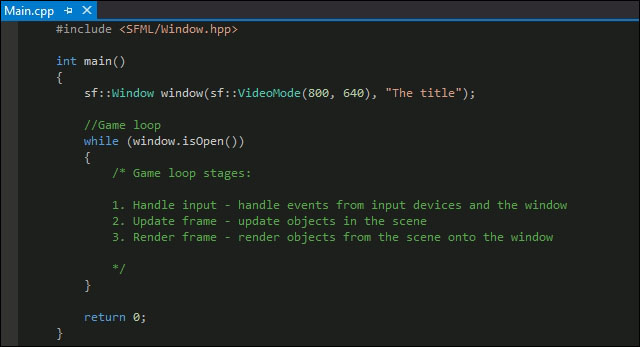
Vòng lặp trò chơi là điều khiển luồng tổng thể cho toàn bộ chương trình trò chơi. Đó là một vòng lặp vì trò chơi tiếp tục thực hiện một loạt các hành động lặp đi lặp lại cho đến khi người dùng thoát.

Một vòng lặp trò chơi chạy liên tục trong suốt quá trình chơi. Mỗi lượt của vòng lặp, nó xử lý đầu vào của người dùng, cập nhật trạng thái trò chơi và kết xuất trò chơi. Nó theo dõi thời gian trôi qua để kiểm soát tốc độ của trò chơi.

Một vòng lặp trò chơi điển hình có ba giai đoạn chính



Minh hoạ:



Việc xử lý đầu vào trong SFML có thể được thực hiện thông qua việc nắm bắt các sự kiện đã được gửi qua cửa sổ hoặc bằng cách truy vấn trực tiếp các thiết bị đầu vào cho trạng thái hiện tại của chúng. Cả hai phương pháp đều có những cách sử dụng khác nhau. Ví dụ: chúng ta có thể muốn đóng cửa sổ bằng cách nhấn 1 nút hoặc muốn di chuyển đặc trưng chính sang bên phải miễn là một phím nhất định được nhấn (truy vấn phím trực tiếp).

Chúng ta đạt đến giai đoạn cập nhật khung sau khi chúng ta nắm bắt và sử dụng các sự kiện của mình. Đây là giai đoạn mà chúng ta muốn nâng cao logic trò chơi của mình và cập nhật trạng thái cho trò chơi.

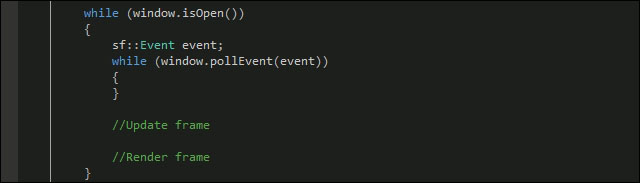
Giai đoạn cuối cùng của vòng lặp diễn ra ngay sau khi chúng ta cập nhật xong các đối tượng của mình. Ở đây, chúng ta xóa mọi thứ đã được vẽ từ lần trước và hiển thị lại mọi đối tượng trên màn hình.

Quay trở lại ví dụ về vòng lặp trò chơi của chúng tôi, nó hiện không thực hiện những thứ mà nó phải thực hiện và nếu chúng ta cố gắng chạy mã, rõ ràng là cửa sổ không phản hồi với các đầu vào. Điều này là do chúng tôi không thực hiện bước đầu tiên trong ba bước quan trọng trong vòng lặp — xử lý đầu vào.

1. **Event handling - Xử lý sự kiện**

Sự kiện có thể được thăm dò từ cửa sổ bằng bool Window::pollEvent(sf::Event& event).Nếu có một sự kiện đang chờ xử lý, hàm sẽ trả về true và biến sự kiện sẽ được lấp đầy bởi dữ liệu sự kiện. Nếu không, hàm trả về false. Cũng cần lưu ý rằng có thể có nhiều sự kiện cùng một lúc; vì vậy chúng ta phải đảm bảo nắm bắt mọi sự kiện có thể.

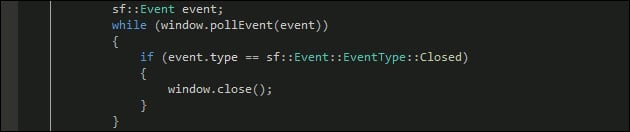
* ***Vòng lặp sự kiện điển hình:***



Event::Closed: Sự kiện này được kích hoạt khi hệ điều hành phát hiện người dùng muốn đóng cửa sổ — nút đóng, tổ hợp phím, v.v.

* ***Sử dụng events:***

Sau khi chúng ta nhận được sự kiện bằng cách gọi Window::pollEvent(), chúng ta có thể kiểm tra kiểu của nó bằng cách xem Event::type. Sự kiện có kiểu Event::EventType, là một enum bên trong lớp Event. Dưới đây là cách một sự kiện kết thúc điển hình có thể được triển khai:



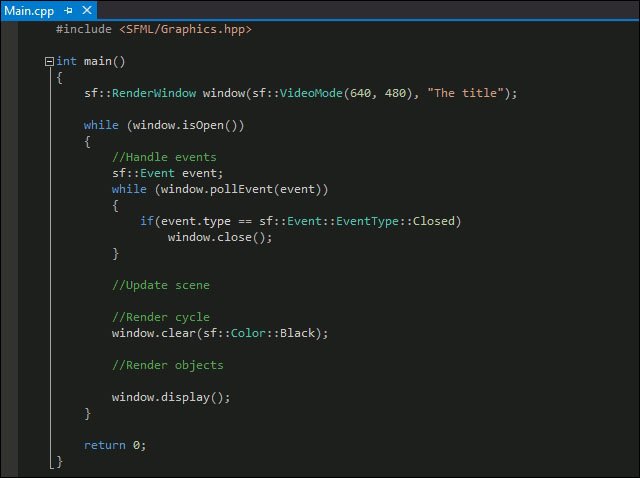
Tại đây, hàm Window::close() sẽ đảm nhận việc đóng cửa sổ. Nếu một biến window nằm ngoài phạm vi, trình hủy được gọi và window bị đóng.

1. **Shape rendering - Kết xuất hình dạng**

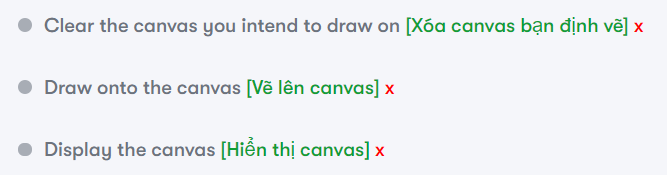
Rõ ràng, chúng ta sẽ không cần cửa sổ nếu không có bất kỳ đối tượng nào được hiển thị và chúng ta sẽ không cần các sự kiện nếu chúng ta không muốn sử dụng đầu vào để tạo hoạt ảnh cho các đối tượng đó. SFML cung cấp khá nhiều cách để chúng ta hiển thị các đối tượng trên màn hình. Tuy nhiên, trước khi tiến hành kết xuất, chúng ta cần đảm bảo rằng chu trình kết xuất của chúng ta đang diễn ra đúng vị trí và đúng thứ tự.

* ***The render frame – Khung kết xuất***

Vòng lặp trò chơi với chu kỳ kết xuất:



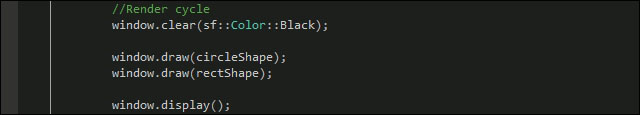
Về cơ bản, chu kỳ kết xuất được phân chia thành:



* ***Shape drawing***

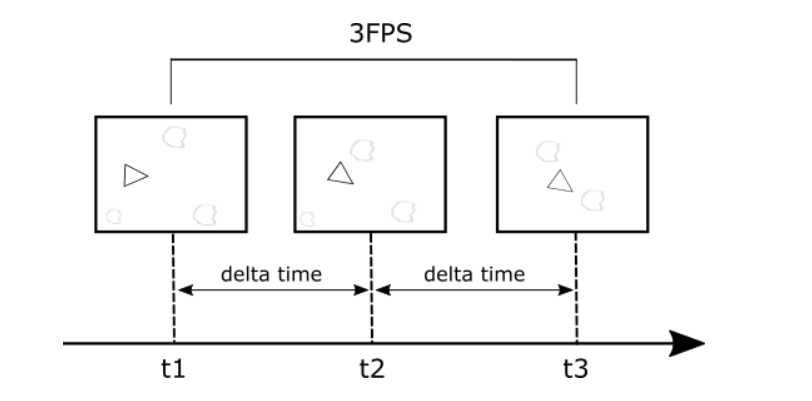
Khi chúng ta muốn vẽ một hình, chúng ta phải tạo đối tượng trước. Để hiển thị các hình trước đó, chúng ta có thể sử dụng hàm RenderWindow::draw()vào khung kết xuất

Ví dụ:



1. **DELTATIME**

Delta time là thời gian cần để hiển thị khung hình. Nói cách khác: delta time là khoảng thời gian giữa khung hình cuối cùng và khung hình hiện tại.



1. **SPRITE, TEXTURE**



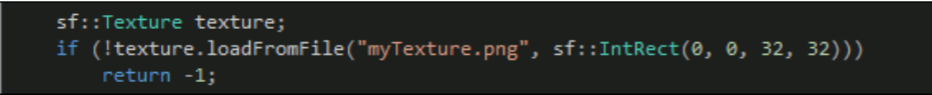
1. **Texture**

Một kết cấu là một hình ảnh. Nhưng chúng ta gọi nó là texture vì nó có một vai trò rất cụ thể: được ánh xạ tới một thực thể 2D

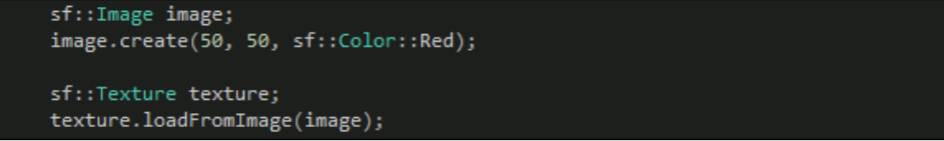
* Tải texture từ file



* Texture :: loadFromFile() cung cấp nhiều chức năng hơn một chút. Khi tải một kết cấu từ file, chúng ta có thể chọn chỉ tải một phần nhỏ của hình ảnh.
  + Ví dụ: Chúng ta sẽ chỉ tải hình vuông 32 x 32 pixel từ hình ảnh gốc, bắt đầu ở góc trên bên trái:



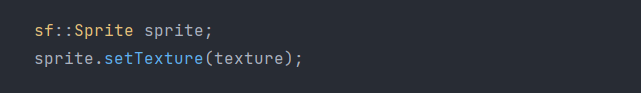
* Ví dụ trên tải toàn bộ hình ảnh và chỉ sau đó tạo kết cấu từ hình chữ nhật cụ thể. Phương pháp này trở nên không thành công nếu chúng ta muốn sử dụng hình ảnh nhiều lần. Cách thay thế là tải hình ảnh một lần và sử dụng nó để tạo ra các kết cấu.



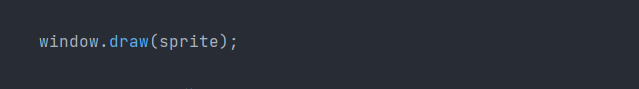
1. **Sprite**

Sprite không có gì khác hơn là một hình chữ nhật có kết cấu

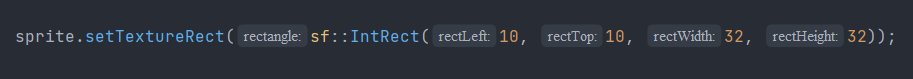
* Tạo sprite



* Vẽ sprite



* Nếu không muốn sprite dùng toàn bộ texture, chúng ta có thể thiết lập hình chữ nhật kết cấu



1. **ANIMATION**

Là tập một hình ảnh động dựa trên sự thay đổi liên tục của nhiều sprite khác nhau.



1. **CÁC KHÁI NIỆM CƠ BẢN**

***1. GameObject***

- Một đối tượng cụ thể trong game gọi là một game object, có thể là nhân vật, đồ vật nào đó.

Ví dụ: cây cối, xe cộ, nhà cửa, người...

***2. Component***

- Một GameObject sẽ có nhiều thành phần cấu tạo nên nó như là hình ảnh (sprite render), tập hợp các hành động (animator), thành phần xử lý va chạm (collision), tính toán vật lý (physical), mã điều khiển (script), các thành phần khác... mỗi thứ như vậy gọi là một component của GameObject.

***3. Sprite***

- Là một hình ảnh 2D của một game object có thể là hình ảnh đầy đủ, hoặc có thể là một bộ phận nào đó.

***5. Mã Frame***

- Mã Frame hay Frame là một trạng thái của một animation. Có thể được tạo nên từ 1 sprite hay nhiều sprite khác nhau.