|  |  |
| --- | --- |
| public class Game implements Runnable {  private GameWindow gameWindow;    private GamePanel gamePanel;    private Thread gameThread;      /\*- Update cho trò chơi của mình - \*/  private final int UPS\_S = 200;    /\* - Tạo một thực thể người chơi - \*/  private Player player;      public Game() {  initClasses();  gamePanel = new GamePanel(this);  gameWindow = new GameWindow(gamePanel);      /\* -Nhận được sự kiện từ bàn phím -\*/  gamePanel.requestFocus();  /\*- Khởi chạy luồng - \*/  StartGameLoop();    }  @Override  public void run() {    /\* - Biến lưu tru thoi luong moi khung hinh sẽ kéo dài - \*/  /\* - Hai biến này được dùng để theo dõi tính toán thời gian đã trôi qua bằng cách lấy ( thời gian hiện tại – thời gian trước đó )  double timePerUpdate = 1000000000.0/ UPS\_S; // Thời gian giữa 1 lần cập nhật trạng thái.  long previousTime = System.nanoTime();  // Biến này được dùng để check xem thời gian trôi qua đã đủ 1 giây hay chưa  long lastCheck = System.currentTimeMillis();  int updates =0;    double deltaU =0;    while (true){  long currentTime = System.nanoTime();  deltaU += (currentTime - previousTime) / timePerUpdate;  previousTime = currentTime;    if (deltaU >= 1) {  update();  updates++;  deltaU--;  }    /-Nếu thời gian hiện tại mà trừ đi thời gian ban đầu thì chúng ta sẽ in ra số lần cập nhật khung hình trong 1 giây và cho biến cập nhật trở về ban đầu.  if(System.currentTimeMillis() - lastCheck >= 1000) {  lastCheck = System.currentTimeMillis();  System.out.println("UPS: " + updates);    updates =0;  }  }  } | int UPS\_S số lần cập nhật trạng thái nhân vật(Trạng thái game ) trong mỗi giây. Cụ thể ở đây là số lần cập nhật trạng thái là 200 lần trong 1 giây.  Ở trong hàm run này :  + double timePerUpdate là khoảng thời gian tối đa mà mỗi lần cập nhật 1 trạng thái được phép mất , được tính bằng nanotime. Nó đảm bảo rằng trong 1 giây thì chỉ được phép thực hiện bao nhiêu lần cập nhật trạng thái  ( Khi ta cộng tổng tất cả thời gian timePerUpdate thì chúng ta sẽ có được 1 giây)  Tóm lại biến này giúp chúng ta xác định được khoảng thời gian giữa các lần cập nhật logic , đặc biệt là để kiểm soát tần suất cập nhật số lần cập nhật mong muốn mỗi giây.  + long previousTime = System.nanoTime() : Biến này lưu trữ thời điểm cập nhật logic trước đó trong đơn vị nanoSecond (Chung quy lại thì cái hàm System.nanoTime() nó sẽ lấy thời gian của hiện tại và giá trị của biến này nó sẽ chạy theo thời gian.  Tóm lại thì biến này nó sẽ cung cấp thời điểm thời gian của lần cập nhật trước đó để tính toán thời gian trôi qua giữa các lần cập nhật, giúp xác định khi nào càn thực hiện lần thực hiện tiếp theo.  + update để số lần game cập nhật trạng thái hoặc nhân vật cập nhật trạng thái .    + lastCheck này biến này lưu trữ thời điểm của lần kiểm tra cuối cùng về thông tin , thời gian cập nhật để một khi đủ thời gian thì chúng ta sẽ in ra số lần cập nhật  + deltaU = 0 : Biến này đánh dấu mức độ thời gian đã trôi qua kể từ lần cập nhật logic trước đó, dựa trên timePerUpdate.  Tóm lại thì biến này được sử dụng để kiểm tra xem đã đến lúc thực hiện cập nhật logic hay chưa, và đồng thời đảm bảo thời gian giữa các lần cập nhật logic là đều đặn.  Tạo một vòng lặp vô tận  Lấy ra thời gian hiện tại  Double timePerUpdate = 1000000000.0 / UPS\_S;  Công thức này tính thời gian tối đa được phép giữa mỗi lần cập nhật logic (tính bằng nanoseconds). timePerUpdate là thời gian mà mỗi lần cập nhật logic của trò chơi được thực hiện dựa trên số lần cập nhật mong muốn trong mỗi giây.  deltaU += (currentTime - previousTime) / timePerUpdate;  deltaU là một biến để theo dõi xem đã đến lúc cập nhật logic hay chưa. Mỗi lần lặp trong game loop, deltaU được cập nhật bằng cách thêm vào thời gian đã trôi qua kể từ lần lặp trước chia cho timePerUpdate. Điều này giúp đảm bảo rằng cập nhật logic sẽ xảy ra khi deltaU đạt hoặc vượt quá giá trị 1.  if (deltaU >= 1) {  update();  updates++;  deltaU--;  }  Trong vòng lặp game loop, kiểm tra xem đã đến lúc cập nhật logic chưa. Nếu deltaU lớn hơn hoặc bằng 1, thì thực hiện cập nhật logic (update()), tăng biến đếm updates và giảm giá trị của deltaU.  if (System.currentTimeMillis() - lastCheck >= 1000) { ... }  Kiểm tra xem đã đến lúc in ra thông tin về UPS chưa. Nếu thời gian kể từ lần kiểm tra cuối cùng (System.currentTimeMillis() - lastCheck) vượt qua 1000 milliseconds (1 giây), thì in ra thông tin về số lần cập nhật logic (UPS) trong 1 giây, sau đó đặt lại biến đếm updates về 0. |
|  |  |