```
#include <SFML/Graphics.hpp>
#include <iostream>
/*
 Global variable for directions, to be used with multiple points of code,
 better to make it easily accessible.
*/
enum Direction{Down, Left, Right, Up};

int main(){

  sf::Vector2i source(1, Down);//Starting position.

  sf::RenderWindow window(sf::VideoMode(800, 600),
                          "Mouse Input");

  window.setKeyRepeatEnabled(false);
  sf::Clock clock;//To be used for the frameRate reproduction.
  float frameCounter = 0, frameSpeed = 500, switchFrame = 100;
  bool updateFrame = true;

  sf::Texture pTexture;
  sf::Sprite playerImage;

  //Can be renamed to whatever texture being brought in, IntRect Param useable
  if(!pTexture.loadFromFile("Player.png"))
    std::cout << "Could not load player Image" <<std::endl;

  playerImage.setTexture(pTexture);

  while(window.isOpen()){
    sf::Event Event;

    while(window.pollEvent(Event)){
      switch(Event.type){
      case sf::Event::Closed:
        window.close();
        break;
      default:
        break;
      }
    }

    //Follows the mouse as long as the left mouse button is pressed down.
    if(sf::Mouse::isButtonPressed(sf::Mouse::Left)){
      sf::Vector2i Position = sf::Mouse::getPosition(window);
      if(playerImage.getPosition().x > Position.x){
        source.y = Left;
        playerImage.move(-1, 0);
      }else if(playerImage.getPosition().x < Position.x){
        source.y = Right;
        playerImage.move(1, 0);
      }else if(playerImage.getPosition().y >Position.y){
        source.y = Up;
        playerImage.move(0, -1);
      }else if(playerImage.getPosition().y < Position.y){
        source.y = Down;
        playerImage.move(0, 1);
      }
    }

    // Keypress dominated movement as well.
    if(sf::Keyboard::isKeyPressed(sf::Keyboard::Up)){
      source.y = Up;
```

```
   64:       playerImage.move(0, -1);
   65:     }else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Down)){
   66:       source.y = Down;
   67:       playerImage.move(0, 1);
   68:     }else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Right)){
   69:       source.y = Right;
   70:       playerImage.move(1, 0);
   71:     }else if(sf::Keyboard::isKeyPressed(sf::Keyboard::Left)){
   72:       source.y = Left;
   73:       playerImage.move(-1, 0);
   74:     }
   75:
   76:     /*
   77:       If window is not set as the the param, the global window scope's relayed
   78:       This causes the mouse positon vector to capture the location of the mouse
   79:       relateable to the MouseMoved Event, but is static.
   80:
   81:       sf::Vector2i mousePosition = sf::Mouse::getPosition(window);
   82:       std::cout << "X: " << mousePosition.x
   83:       << " Y: "<< mousePosition.y <<std::endl;
   84:     */
   85:
   86:     /*
   87:       Moves the move to the given position. If window is not set as a parameter
   88:       then moves to 100, 100 on the global scope, as in the full window.
   89:       sf::Mouse::setPosition(sf::Vector2i(100,100), window);
   90:     */
   91:
   92:     //Simple Boolean if-Else statement with ?, need to practice that
   93:     frameCounter = (updateFrame) ?
   94:       frameCounter + frameSpeed* clock.restart().asSeconds() :
   95:       0;
   96:
   97:     if(frameCounter>= switchFrame){
   98:       source.x++;
   99:       if(source.x *32 >= (signed int) pTexture.getSize().x)
  100:         source.x = 0;
  101:       frameCounter = 0;
  102:     }
  103:
  104:     //Refresh the image.
  105:     playerImage.setTextureRect(sf::IntRect(source.x*32, source.y*32, 32, 32));
  106:
  107:     window.draw(playerImage);
  108:     window.display();
  109:     window.clear();
  110:   }
  111:
  112:   return 0;//Done implicitly, but I like to do it anyways.
  113: }
```