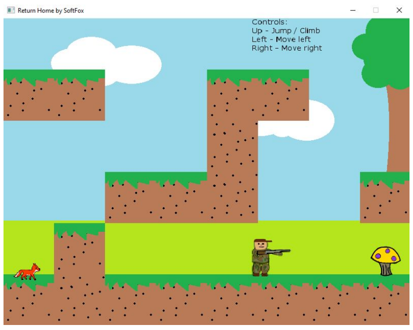
Physics Component for SoftFox Game

Developed by Samantha Wills

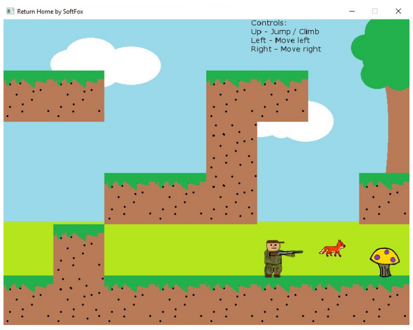


Aim of the game

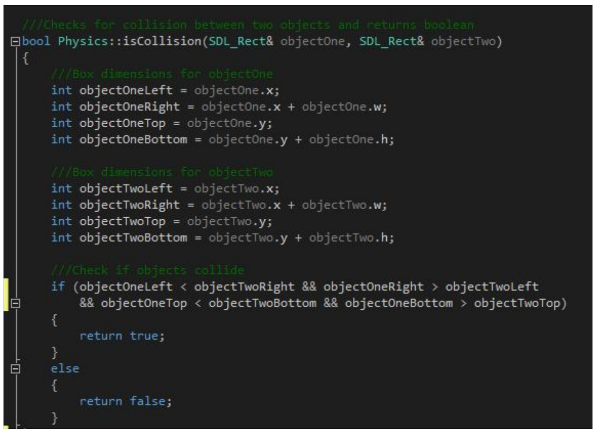
Our game is a 2D platformer where the player controls a fox sprite and navigates around the world with obstacles and a goal.

Purpose of Physics Component

For this platformer, physics simulates basic real world physics such as gravity, upforce and collisions.



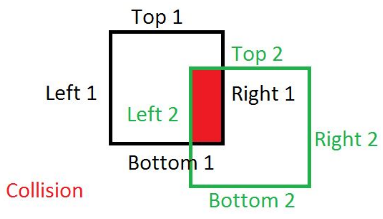
Code for Game Collisions



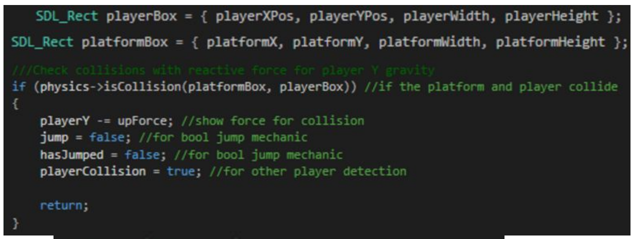
```
///Checks for collision between two objects and returns boolean
bool Physics::isCollision(SDL_Rect& objectOne, SDL_Rect& objectTwo)
{
    ///Box dimensions for objectOne
    int objectOneLeft = objectOne.x;
    int objectOneRight = objectOne.x + objectOne.w;
    int objectOneTop = objectOne.y;
    int objectOneBottom = objectOne.y + objectOne.h;

    ///Box dimensions for objectTwo
    int objectTwoLeft = objectTwo.x;
    int objectTwoRight = objectTwo.x + objectTwo.w;
    int objectTwoTop = objectTwo.y;
    int objectTwoBottom = objectTwo.y + objectTwo.h;

    ///Check if objects collide
    if (objectOneLeft < objectTwoRight && objectOneRight > objectTwoLeft
        && objectOneTop < objectTwoBottom && objectOneBottom > objectTwoTop)
    {
        return true;
    }
    else
    {
        return false;
    }
```

Box Collision Detection Method

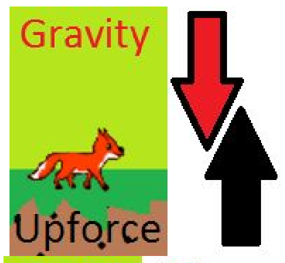
The collision detection works by drawing a rectangle around each object and then checking if certain areas overlap.



- Is Bottom 1 between Top and Bottom 2?
- Is Left 1 to the left of Right 2?
- Is Right 1 to the right of Left 2?



```
    SDL_Rect playerBox = { playerXPos, playerYPos, playerWidth, playerHeight };
SDL_Rect platformBox = { platformX, platformY, platformWidth, platformHeight };

///Check collisions with reactive force for player Y gravity
if (physics->isCollision(platformBox, playerBox)) //if the platform and player collide
{
    playerY -= upForce; //show force for collision
    jump = false; //for bool jump mechanic
    hasJumped = false; //for bool jump mechanic
    playerCollision = true; //for other player detection

    return;
}
```

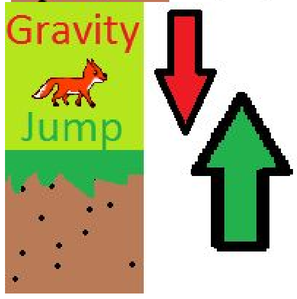


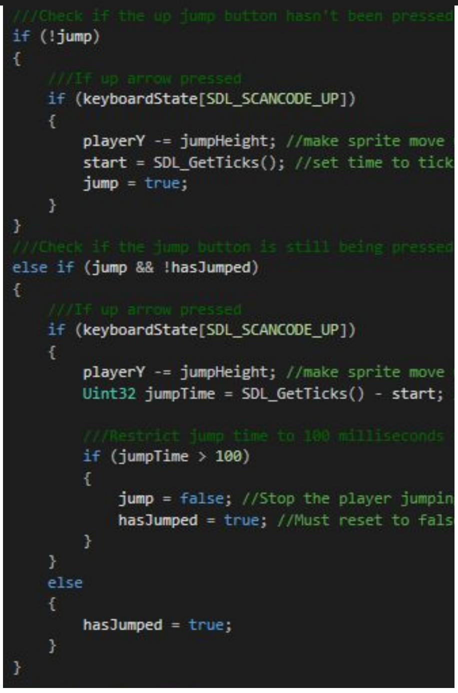
UpForce Cancels Out Gravity

When the sprite collides with a platform, an upforce cancels out the gravity by moving the player in the opposite direction with the same force.

Jump Move Sprite Upward

When the player presses the jump button, the sprite should have a greater jump force which moves the sprite up the screen. The code checks to see if the jump button has already been pressed, so that the sprite can only jump once and sets a time for how long the sprite can jump for. When the jump time limit is exceeded, or the jump button is released, the jump force is removed and the sprite falls until there is an upforce to stop gravity.

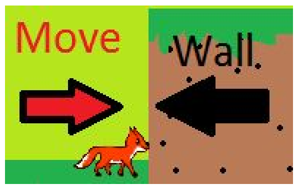


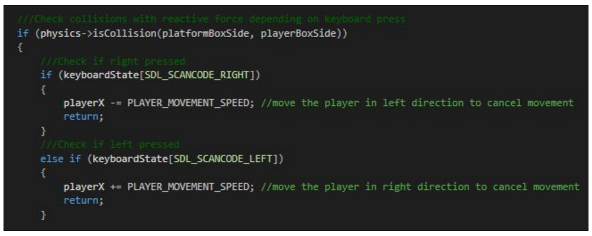


```
///Check if the up jump button hasn't been pressed
if (!jump)
{
    ///If up arrow pressed
    if (keyboardState[SDL_SCANCODE_UP])
    {
        playerY -= jumpHeight; //make sprite move
        start = SDL_GetTicks(); //set time to tick
        jump = true;
    }
}
///Check if the jump button is still being pressed
else if (jump && !hasJumped)
{
    ///If up arrow pressed
    if (keyboardState[SDL_SCANCODE_UP])
    {
        playerY -= jumpHeight; //make sprite move
        Uint32 jumpTime = SDL_GetTicks() - start;

        ///Restrict jump time to 100 milliseconds
        if (jumpTime > 100)
        {
            jump = false; //Stop the player jumpin
            hasJumped = true; //Must reset to fals
        }
    }
    else
    {
        hasJumped = true;
    }
}
```

Wall Collision Against Sprite

If the player tried to walk into a wall and it detects a collision, it pushes against the key movement.





```
///Check collisions with reactive force depending on keyboard press
if (physics->isCollision(platformBoxSide, playerBoxSide))
{
    ///Check if right pressed
    if (keyboardState[SDL_SCANCODE_RIGHT])
    {
        playerX -= PLAYER_MOVEMENT_SPEED; //move the player in left direction to cancel movement
        return;
    }
    ///Check if left pressed
    else if (keyboardState[SDL_SCANCODE_LEFT])
    {
        playerX += PLAYER_MOVEMENT_SPEED; //move the player in right direction to cancel movement
        return;
    }
```