2D GRAPHICS

Vuong Ba Thinh

Department of CS –CSE Faculty - HCMUT

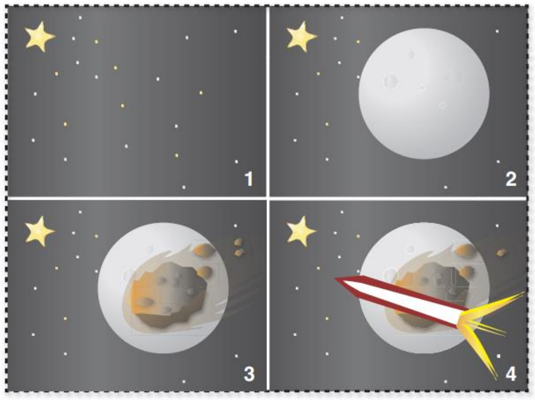
Outline

- Sprites
- Scrolling
- Tile Maps

Sprite

- Sprite: A sprite is a 2D visual object within the game world that can be drawn using a single image on any given.
- Framework: SDL, XNA, Cocos2d
- Format:
  - PNG: less space
  - TGA: native
  - PVR: iOS

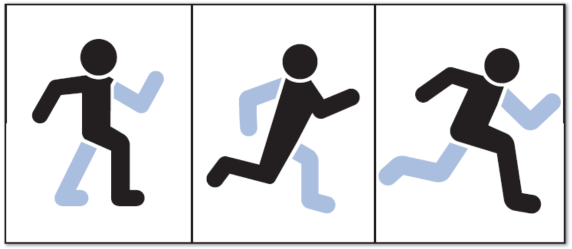
```
class Sprite
    ImageFile image
    int drawOrder
    int x, y
    function Draw()
        // Draw the image at the correct (x,y)
        ...
    end
end
```

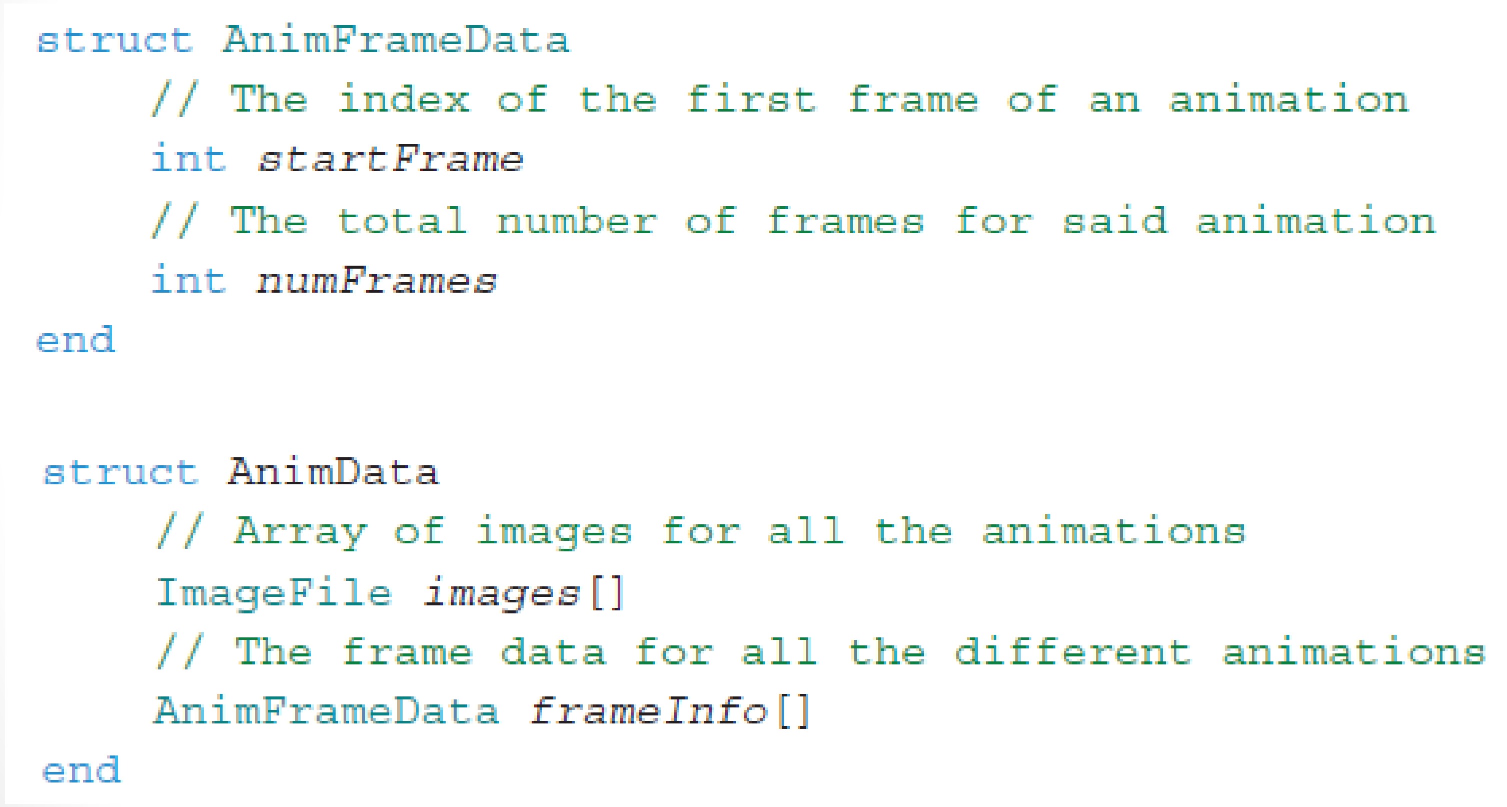


Sprite

```
SortedList spriteList

// When creating a new sprite...
Sprite newSprite = specify image and desired x/y
newSprite.drawOrder = set desired draw order value
// Add to sorted list based on draw order value
spriteList.Add(newSprite.drawOrder, newSprite)

// When it's time to draw...
foreach Sprite s in spriteList
   s.Draw()
loop
```

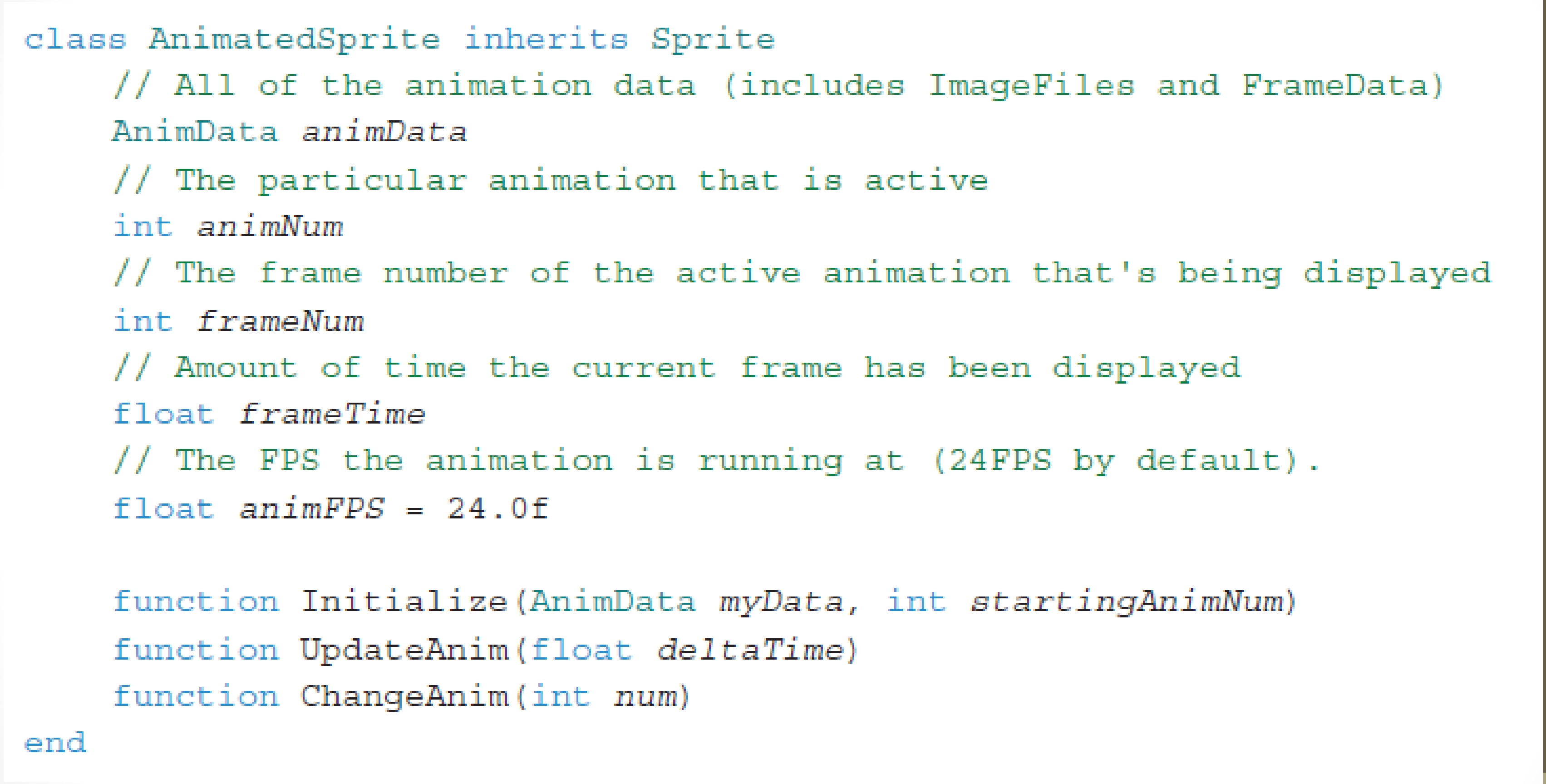
Animating Sprite





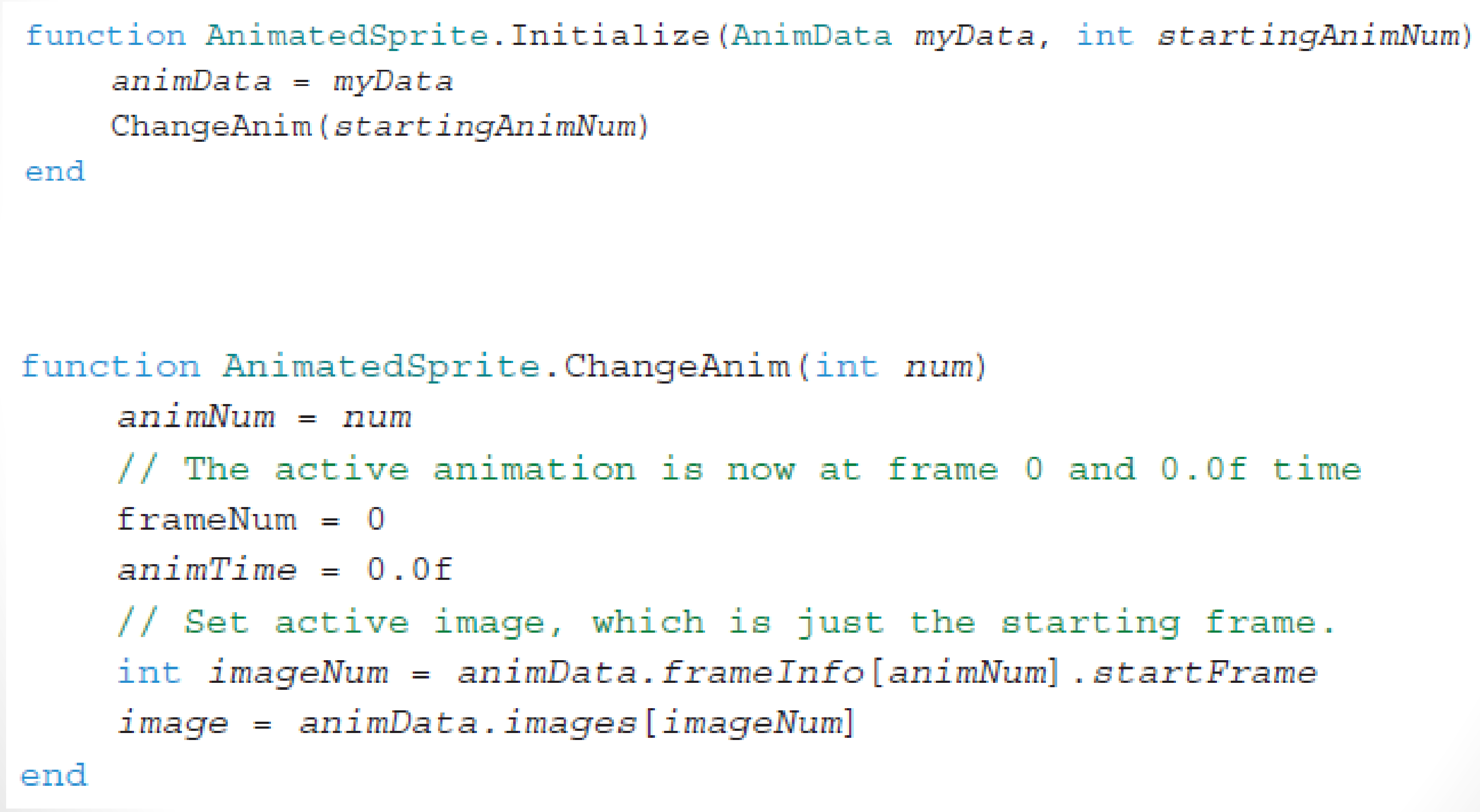
```
struct AnimFrameData
    // The index of the first frame of an animation
    int startFrame
    // The total number of frames for said animation
    int numFrames
end

struct AnimData
    // Array of images for all the animations
    ImageFile images[]
    // The frame data for all the different animations
    AnimFrameData frameInfo[]
end
```

Animating Sprite



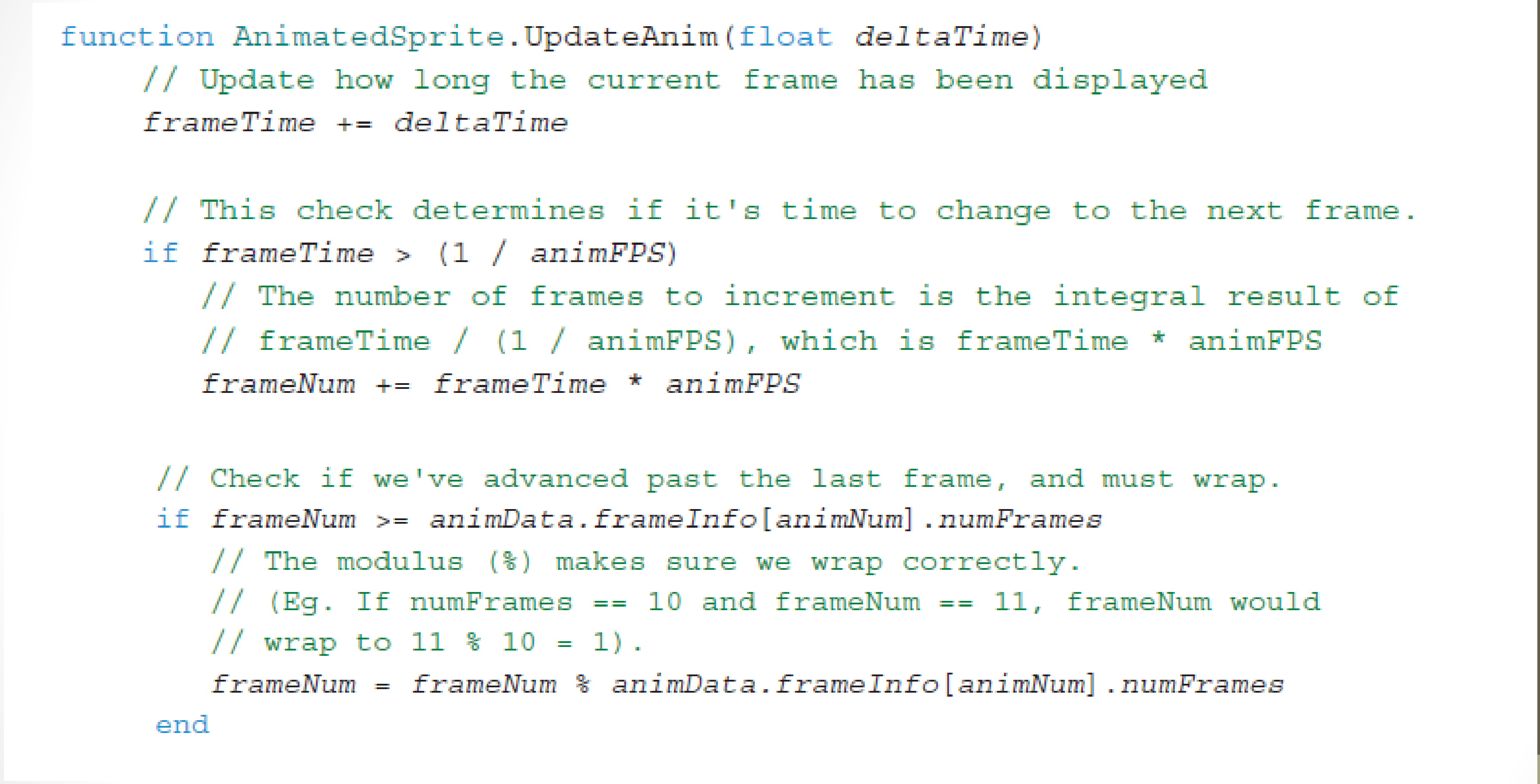
```
class AnimatedSprite inherits Sprite
    // All of the animation data (includes ImageFiles and FrameData)
    AnimData animData
    // The particular animation that is active
    int animNum
    // The frame number of the active animation that's being displayed
    int frameNum
    // Amount of time the current frame has been displayed
    float frameTime
    // The FPS the animation is running at (24FPS by default).
    float animFPS = 24.0f

    function Initialize(AnimData myData, int startingAnimNum)
    function UpdateAnim(float deltaTime)
    function ChangeAnim(int num)
end
```

Animating Sprite



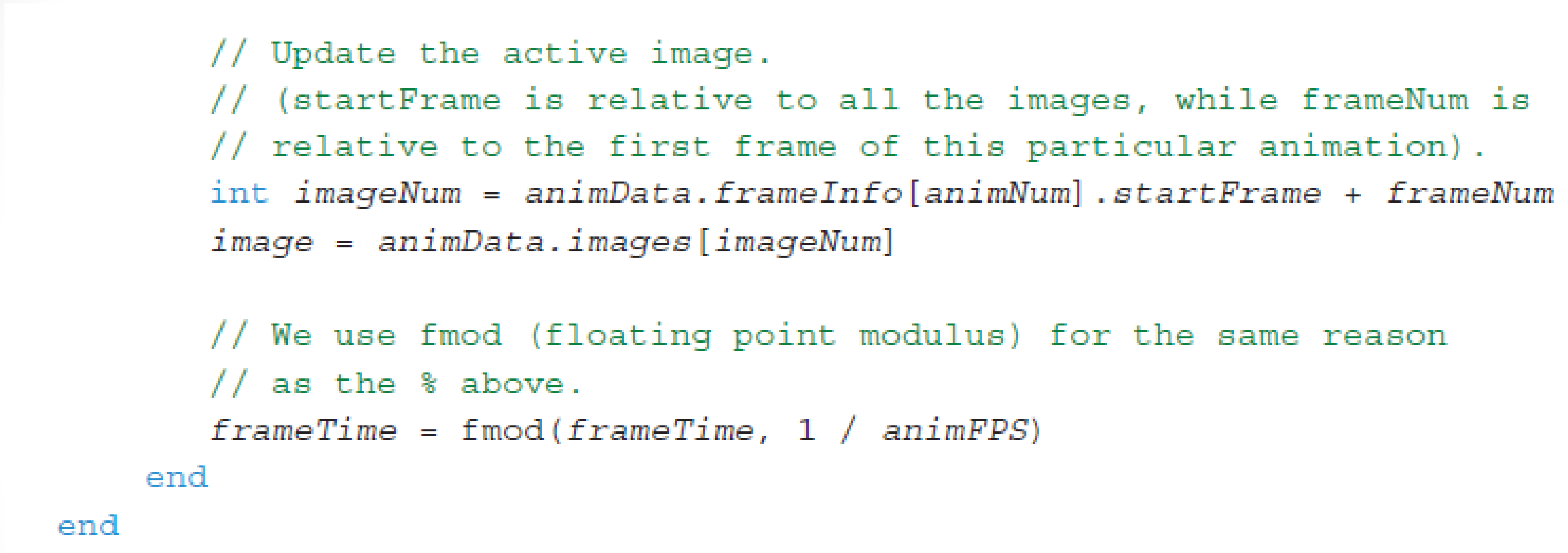
```
function AnimatedSprite.Initialize(AnimData myData, int startingAnimNum)
    animData = myData
    ChangeAnim(startingAnimNum)
end



function AnimatedSprite.ChangeAnim(int num)
    animNum = num
    // The active animation is now at frame 0 and 0.0f time
    frameNum = 0
    animTime = 0.0f
    // Set active image, which is just the starting frame.
    int imageNum = animData.frameInfo[animNum].startFrame
    image = animData.images[imageNum]
end
```

Animating Sprite



```
function AnimatedSprite.UpdateAnim(float deltaTime)
    // Update how long the current frame has been displayed
    frameTime += deltaTime

    // This check determines if it's time to change to the next frame.
    if frameTime > (1 / animFPS)
       // The number of frames to increment is the integral result of
       // frameTime / (1 / animFPS), which is frameTime * animFPS
       frameNum += frameTime * animFPS

    // Check if we've advanced past the last frame, and must wrap.
    if frameNum >= animData.frameInfo[animNum].numFrames
       // The modulus (%) makes sure we wrap correctly.
       // (Eg. If numFrames == 10 and frameNum == 11, frameNum would
       // wrap to 11 % 10 = 1).
       frameNum = frameNum % animData.frameInfo[animNum].numFrames
    end
```

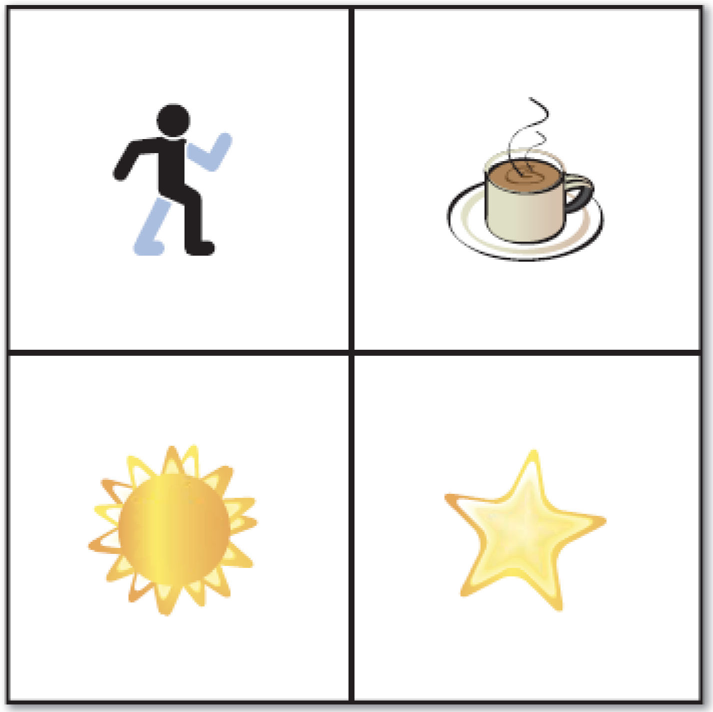
Animating Sprite

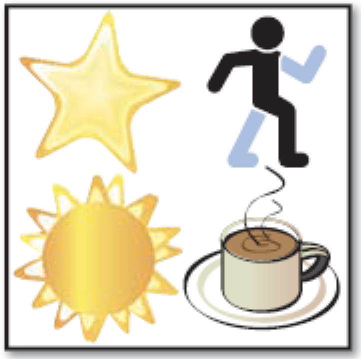


```
        // Update the active image.
        // (startFrame is relative to all the images, while frameNum is
        // relative to the first frame of this particular animation).
        int imageNum = animData.frameInfo[animNum].startFrame + frameNum
        image = animData.images[imageNum]

        // We use fmod (floating point modulus) for the same reason
        // as the % above.
        frameTime = fmod(frameTime, 1 / animFPS)
    end
end
```

Sprite Sheet

Tool: TexturePacker


(a)

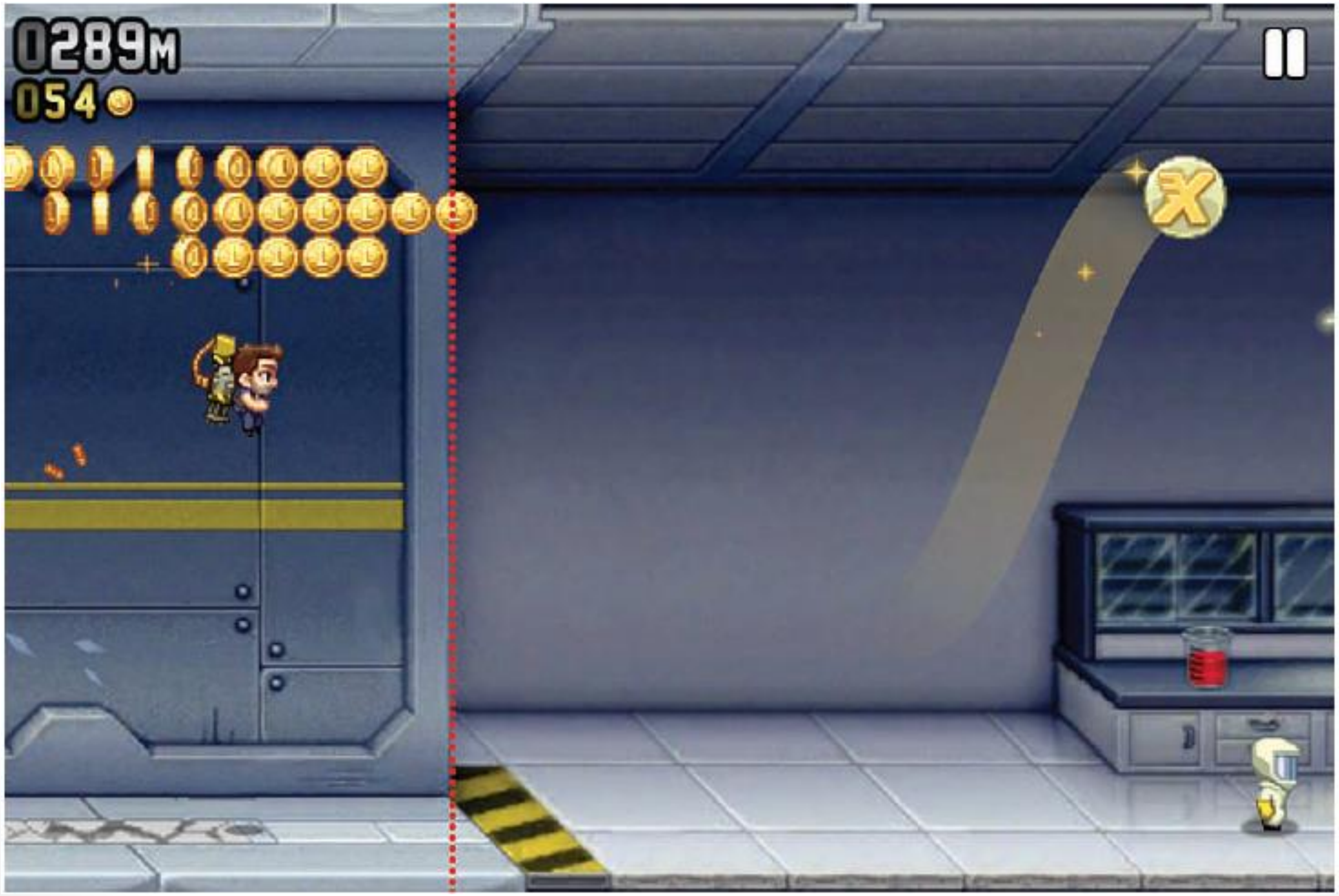

(b)

Scrolling Game

- List 3 games???

Single-Axis Scrolling

- Jetpack Joyride



Background Sprite List

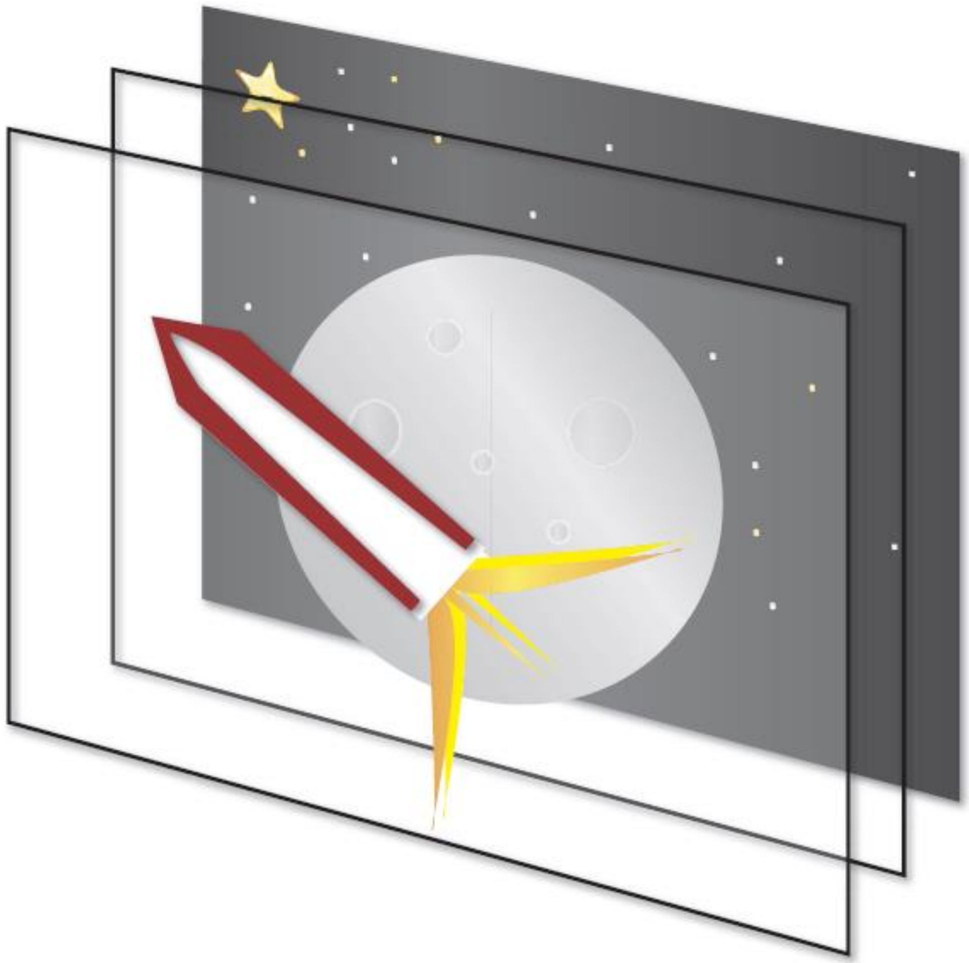
```
const int screenWidth = 960 // An iPhone 4/4S sideways is 960x640
// All the screen-sized image backgrounds
string backgrounds[] = { "bg1.png", "bg2.png", /*...*/}
// The total number of screen-sized images horizontally
int hCount = 0
foreach string s in backgrounds
    Sprite bgSprite
    bgSprite.image.Load(s)
    // 1st screen would be x=0, 2nd x=960, 3rd x=1920, ...
    bgSprite.x = hCount * screenWidth
    bgSprite.y = 0
    bgSpriteList.Add(bgSprite)
    screenCount++
loop
```

Clamp Algorithm

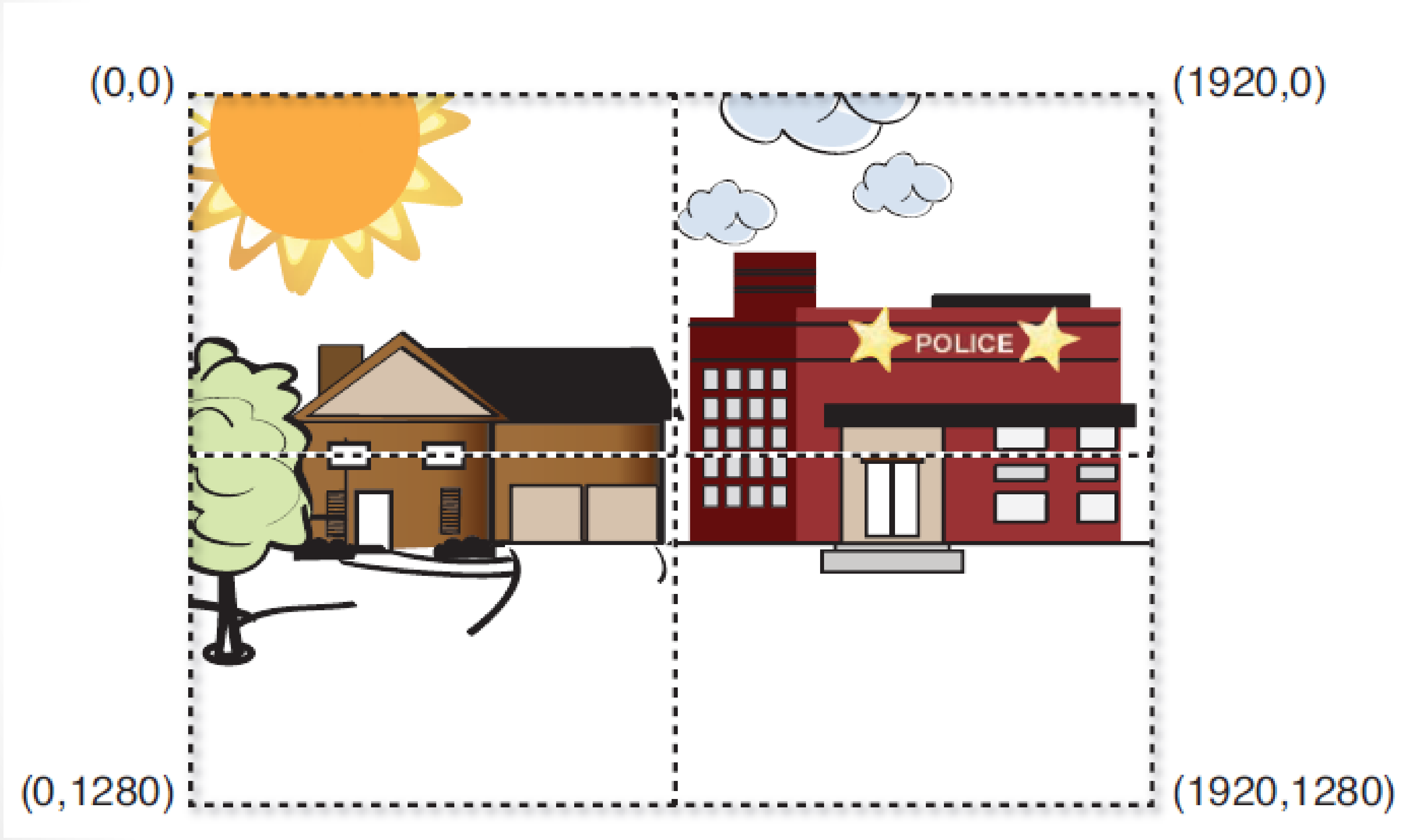
```
// camera.x is player.x as long as its clamped within the valid range
camera.x = clamp(player.x, screenWidth / 2,
                           hCount * screenWidth - screenWidth / 2)

Iterator i = bgSpriteList.begin()
while i != bgSpriteList.end()
   Sprite s = i.value()

   // find the first bg image to draw
   if (camera.x - s.x) < screenWidth
      // Image 1: s.x = 0, camera.x = 480, screenWidth/2 = 480
      // 0 - 480 + 480 = 0
      draw s at (s.x - camera.x + screenWidth/2, 0)
      // draw the bg image after this, since it might also be visible
      i++
      s = i.value()
      draw s at (s.x - camera.x + screenWidth/2, 0)
      break
   end
   i++
loop
```

Infinite Scrolling

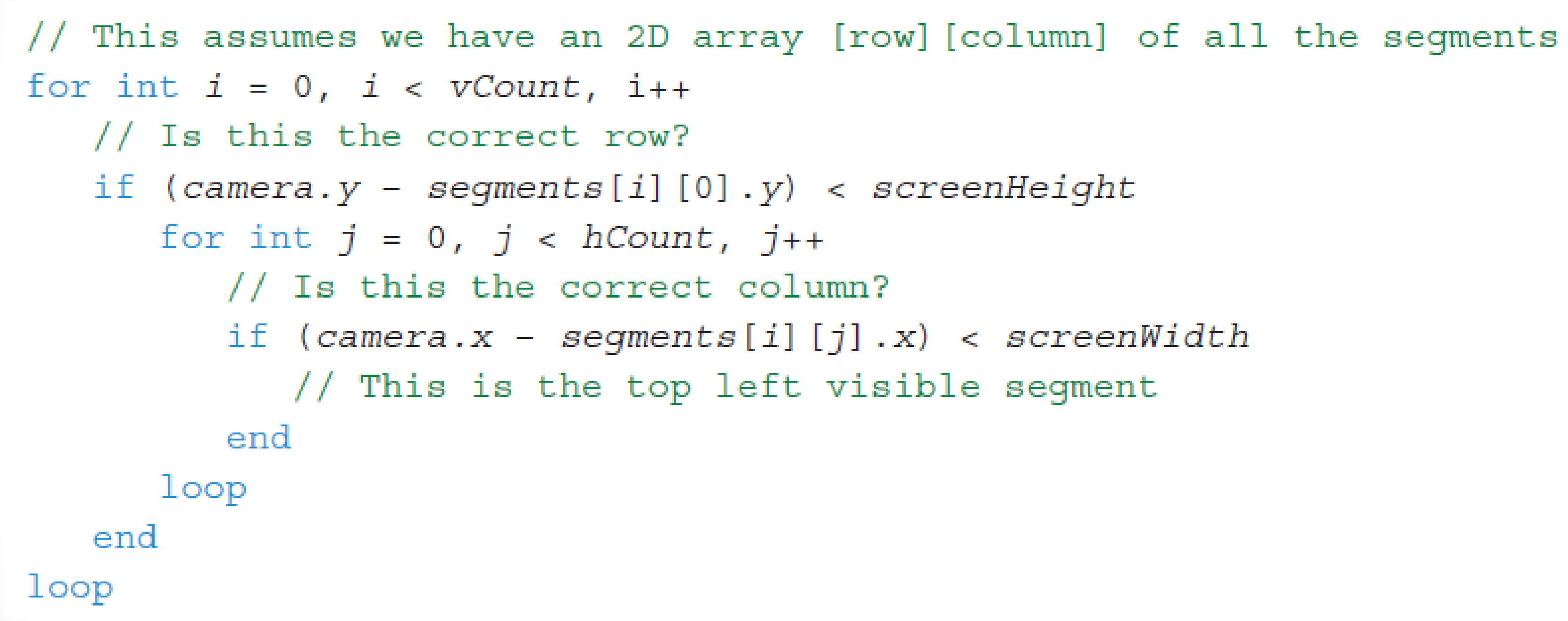
Parallax Scrolling



4-ways Scrolling

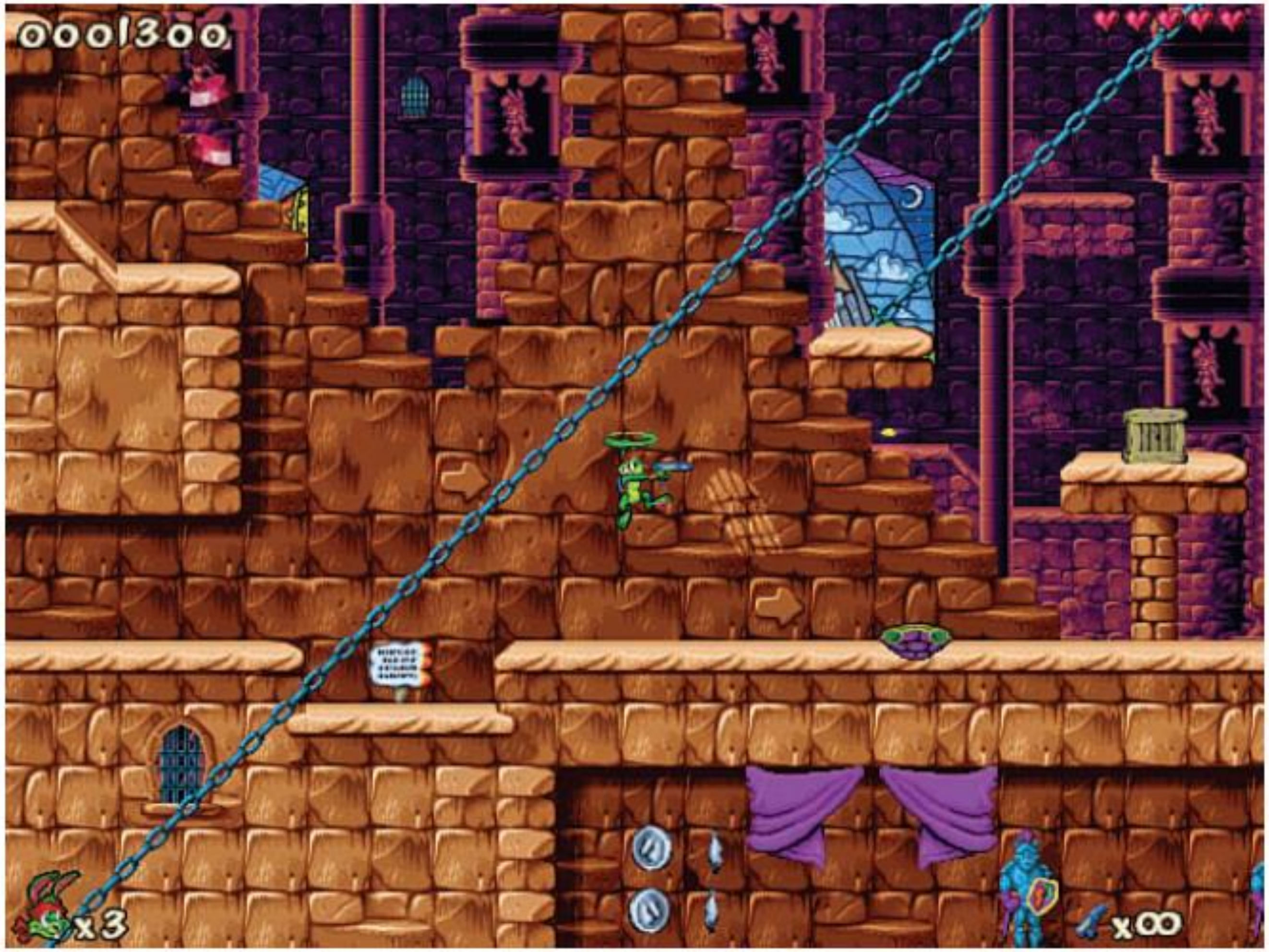


4-ways Scrolling



```
// This assumes we have an 2D array [row][column] of all the segments
for int i = 0, i < vCount, i++
   // Is this the correct row?
   if (camera.y - segments[i][0].y) < screenHeight
      for int j = 0, j < hCount, j++
         // Is this the correct column?
         if (camera.x - segments[i][j].x) < screenWidth
            // This is the top left visible segment
         end
      loop
   end
loop
```

Tile Maps



Jazz Jackrabbit 2

Tile Maps

```
// Basic level file format
5,5
0,0,1,0,0
0,1,1,1,0
1,1,2,1,1
0,1,1,1,0
0,0,1,0,0
```

```
class Level
    const int tileSize = 32
    int width, height
    int tiles[][]
    function Draw()
end
```

```
function Draw()
    for int row = 0, row < height, row++
        for int col = 0, col < width, col++
            // Draw tiles[row][col] at (col*tileSize, row*tileSize)
        loop
    loop
end
```