**Spike:** Spike No.18

**Title:** Task 18 – Sprites and Graphics

**Author:** Khang Trinh - 102118468

**Goals / Deliverables:**

The goal of this spike is to teach the developer how to load and display images and sprites.

**Technologies, Tools, and Resources used:**

* Visual Studio 2017

**Useful Links:**

1. How to load and animate sprites in a sprite sheet

<https://youtu.be/2NVgHrOFneg>

1. Intro to sprite sheet animation

<https://gamedevelopment.tutsplus.com/tutorials/an-introduction-to-spritesheet-animation--gamedev-13099>

# Tasks undertaken:

## Texture handling

## Step 1: Load files

To load an image into the game, you need to perform 2 smaller steps

* Load the image into a surface
* Create the texture from that surface and assign the rendering context to the renderer

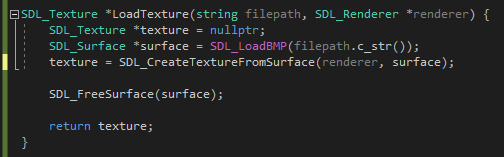




Fig 1. How loading textures in SDL looks like

## Step 2: Display images



Fig 2. Always clear the screen before displaying a new image

## Sprites handling

For this section, we’ll be using the image below. Each “frame” of the sheet is a stripe of colour. In other words, this sheet has 3 frames.

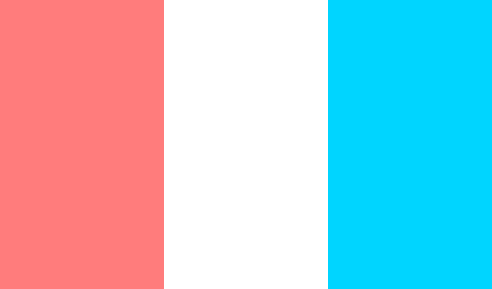


Fig 1. The sprite sheet we'll be using for this Spike

## Step 1: Define width and height of a frame

A sprite sheet will always come with multiple frames tightly packed next to each other. To “synthesize” them into frames, we need to first define how big a frame is in the sheet.





Fig 2. How to get the size of the entire sheet



Fig 3. The sheet is 3 frames wide and 1 frame high, so we’ll divide the size accordingly

## Step 2: Create a Rect

We’ve defined the dimensions of a frame, now we need to create a rect - something that can use this dimension data. Think of this like a viewport that only shows an area of a texture.

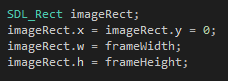


Fig 4. How to create a rect

## Step 3: Assign a frame to the rect

We can then make the rect display a particular frame by moving it using the frame unit we’ve defined earlier (ie. Move 1 frame to the right, move 5 frames downward, etc.).

**NOTE**: The first frame is frame 0, not frame 1.



Fig 5. This will display frame 3 in the sheet we’re using

## Step 4: Display the frame

Displaying a frame of a sprite sheet is handled in a similar way as displaying an image, the only difference is you need to pass in the rect along with the sprite sheet, and any transform you want to apply to the sprite (scale, position, etc.)



Fig 6. How to display a frame of a sprite sheet

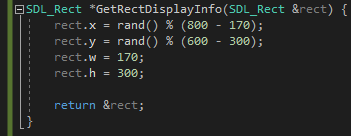


Fig 7. For this example, a frame will have a size of 170x300 and displayed at a random location on the screen

## Clean up

Last, but most important, remember to always release the resources before quitting the application.

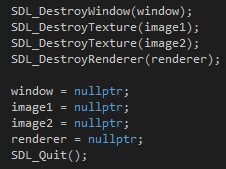


Fig 8. Releasing resources before quitting application