Part 7 – Spritesheets and Animation

In the previous part, we put an object on the screen that used a single image of a soldier. Most often though, you will want to use a sprite sheet for your objects that contain one or more animations.

Using config we can cut the image up into many graphic sections and make them available for animations.

Back over in the Orx project, in the `orx/tutorial/data/anim` asset folder is a spritesheet for our hero called soldier_full.png which looks like this:

We can change the [HeroGraphic] to use this image file instead. But first, copy this image to the `data/texture` folder in our project.

Now it is possible to change the HeroGraphic texture:

```
[HeroGraphic]
Texture        = soldier_full.png
```

Of course if you ran the game now, the hero would become the entire sprite sheet, but we only want the first image on the sheet. That's fine. We can crop it with:

```
[HeroGraphic]
Texture        = soldier_full.png
TextureOrigin  = (0,0,0)
TextureSize    = (32,32,0)
```

Now if you run the game (you don't need to recompile - only config changes have been made), you will see just a single image for the hero object which is extracted from the sheet.

In order to use all six frames on the sprite sheet to animate our hero character, we can define some animation for him.

Our first task is to give the hero object an animation set:

```
[HeroObject]
Graphic      = HeroGraphic
Position     = (-350, 100, 0)
Scale        = 2
AnimationSet = HeroAnimationSet
```

Next is to define the `AnimationSet` itself:

```
[HeroAnimationSet]
Texture   = soldier_full.png
FrameSize = (32, 32, 0)
```

Just like a graphic section, again, the `Texture` to the sprite sheet is supplied. Next is the `FrameSize` property is supplied. This means, for every sprite on the sprite sheet, they will be all 32 x 32 pixels in size. This help Orx calculate all the sprites on the sheet.

Next, add an animation name and how many sprites to use for the animation:

```
[HeroAnimationSet]
Texture   = soldier_full.png
FrameSize = (32, 32, 0)
HeroRun   = 6 ; or -1 would be fine too.
StartAnim = HeroRun
```

You can declare an animation by using a name for it like: `HeroRun` and supplying the amount of frames to use in the sheet for the animation. In our case we are using all six.

This is a very simple sheet. Because all the sprites are being used on the sheet, the value of -1 would have worked just as well.

`StartAnim` for our object can be set to `HeroRun` which is our hero's one and only animation.

We can also supply additional information about the `HeroRun` animation, like the speed of the animation. In order to do that, a `HeroRun` section needs to be supplied:

```
[HeroRun]
KeyDuration = 0.1
```

Now the time between frames is 0.1.

Run this and the little hero will start running forever.

Pretty simple animation set up isn't it?

Time to make some platforms.

Next: Part 8 – Platforms and Texture Repeating
