IN-CLASS STUDENT ACTIVITY 4.1: ANIMATE BASIC CHARACTERS

Lesson Objective 4.1:

Animate basic characters. *Topics*: movement, frames per second (FPS), apply sprite animation.

Resources, software, and additional files needed for this lesson:

1. A workstation with Microsoft Visual Studio 2010® (or Visual Studio 2010 Express®), and XNA® 4.0
2. JumpingJacks.png, located in 98-374-ENU-IC-resources-4.1.zip folder

Guiding questions:

1. How do characters (sprites) move across the screen?
2. How does the frame per second (FPS) rate affect the animation?
3. What is a sprite sheet?

Student Activity:

Directions to the student:

Read the following scenario and complete the hands-on activities. Note that the screenshots in the activity may appear differently on your system. Answer the questions as you work through the steps. Verify your answers with the instructor. Request assistance from the instructor as needed.

Scenario:

Congratulations! You have just been hired as a student intern at Tailspin Toys. Your first assignment is to create a basic 2-D animation of a stick figure doing jumping jacks. Your supervisor has provided a sprite sheet of figures in different positions that, when shown in rapid order, appear to be one figure doing jumping jacks.

Content:

1. Start a new Microsoft Visual Studio 2010, XNA Game Studio 4.0 project, called JumpingJacks.
2. Add the sprite sheet to the content pipeline.
3. If possible, open the sprite sheet in a graphics program so you can obtain the values for *x, y,* Width, and Height of each sprite on the page (you can use Expression Design, Paint.net). If not, ask the instructor for these values.
4. Declare all global variables for a basic animation, including:
    a. Texture2D for the sprite sheet
    b. Vector2 for the position to draw the sprites
    c. Six *Rectangle* objects for each part of the sprite sheet using the values for x, y, width, height
    d. Vector2 for the origin (optional Vector for scaling)
    e. Integer for tracking which image should be displayed
5. In the *LoadConten()* method, define the texture, origin, and position to place the stick figure on the screen using the variables defined above.

6. In the *Update()* method, delay the animation. Here is the code used for this example, you may have another way:

   Add this code to your *Update()* method:

```
   if (elapsed >= duration)
   {
      setupImage();
      elapsed -= duration;
   }
   elapsed += gameTime.TotalGameTime.TotalSeconds;

Include these two global variables:
   double duration = 20.0f;
   double elapsed = 0.0f;
```

7. The *Update()* method calls the *setupImage()* function, which determines which stick figure to draw using the count variable defined earlier.
8. Update the *Draw()* method to show your stick figure using the generic image object that is changed through each iteration of the program.
9. Test your program and make adjustments as necessary, and have fun!