```
1  using Microsoft.Xna.Framework;
2  using Microsoft.Xna.Framework.Graphics;
3  using System;
4  using System.Collections.Generic;
5  using System.Linq;
6  using System.Text;
7
8
9
10 namespace GameDemo.Shared
11 {
12     public abstract class Sprite
13     {
14         Texture2D texture;
15         protected Vector2 position;
16         int totalFrames;
17         protected Point frameSize;
18         Point sheetSize;
19         protected Vector2 collisionOffset;
20         protected Point currentFrame;
21         int timeSinceLastFrame;
22         protected int millisecondsPerFrame;
23         protected Vector2 speed;
24         protected float scale;
25
26         public Texture2D Texture
27         {
28             get { return texture; }
29             private set { texture = value; }
30         }
31
32         // scales the animation size
33         public float Scale{
34             get { return scale; }
35             set { scale = value; }
36         }
37
38
39         const int defaultMillisecondsPerFrame = 16;
40
41         public Sprite(Texture2D texture, Vector2 position, Point frameSize, int ↵
               totalFrames, Vector2 collisionOffset, Point currentFrame, Point     ↵
             sheetSize, Vector2 speed, int secondsperFrame,float scale)
42         {
43             this.texture = texture;
44             this.position = position;
45             this.frameSize = frameSize;
46             this.totalFrames = totalFrames;
47             this.collisionOffset = collisionOffset;
48             this.currentFrame = currentFrame;
49             this.sheetSize = sheetSize;
50             this.speed = speed;
51             this.millisecondsPerFrame = secondsperFrame;
52             this.scale = scale;
53         }
54         public Sprite(Texture2D texture, Vector2 position, Point frameSize, int ↵
```

```
            totalFrames, Vector2 collisionOffset, Point currentFrame, Point
          sheetSize, Vector2 speed,float scale) : this(texture, position,
          frameSize, totalFrames,
55          collisionOffset, currentFrame, sheetSize, speed,
              defaultMillisecondsPerFrame,scale)
56      { }
57
58      // handle the sprite frames (can work with multiple rows and column)
59      public virtual void Update(GameTime gametime)
60      {
61
62          timeSinceLastFrame += gametime.ElapsedGameTime.Milliseconds;
63          if (timeSinceLastFrame > millisecondsPerFrame)
64          {
65              timeSinceLastFrame = 0;
66
67              ++currentFrame.X;
68
69              if (currentFrame.X >= sheetSize.X ||
                  (currentFrame.X*currentFrame.Y*2>totalFrames))
70              {
71                  currentFrame.X = 0;
72                  ++currentFrame.Y;
73                  if (currentFrame.Y >= sheetSize.Y )
74                      currentFrame.Y = 0;
75              }
76          }
77
78
79      }
80      public virtual void Draw(GameTime gametime, SpriteBatch
          spritebatch,float scale, SpriteEffects spriteEffects)
81      {
82
83          spritebatch.Draw(texture, position, new Rectangle(currentFrame.X *
              frameSize.X, currentFrame.Y * frameSize.Y, frameSize.X,
              frameSize.Y), Color.White, 0, Vector2.Zero, scale,spriteEffects,
              0);
84      }
85
86      // The collision rectangle of each enemy
87      public abstract Rectangle collisionRect();
88
89      }
90  }
91
92
93
```