```
using System;
using System.Collections.Generic;
using System.Linq;
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Audio;
using Microsoft.Xna.Framework.Content;
using Microsoft.Xna.Framework.GamerServices;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using Microsoft.Xna.Framework.Media;

namespace SpaceInvaders
{
    /// <summary>
    /// This is the main type for your game
    /// </summary>
    public class Game1 : Microsoft.Xna.Framework.Game
    {
        GraphicsDeviceManager graphics;
        SpriteBatch spriteBatch;
        Texture2D invader, ship, bullet;
        Rectangle rectship, rectbullet;
        Rectangle[,] rectinvader;
        String[,] invaderalive;
        int invaderspeed = 3;
        int rows = 5;
        int cols = 10;
        String direction = "RIGHT";
        String bulletvisible = "NO";

        public Game1()
        {
            graphics = new GraphicsDeviceManager(this);
            Content.RootDirectory = "Content";
        }

        /// <summary>
        /// Allows the game to perform any initialization it needs to before starting to run.
        /// This is where it can query for any required services and load any non-graphic
        /// related content.  Calling base.Initialize will enumerate through any components
        /// and initialize them as well.
        /// </summary>
        protected override void Initialize()
        {
            // TODO: Add your initialization logic here

            base.Initialize();
        }

        /// <summary>
        /// LoadContent will be called once per game and is the place to load
        /// all of your content.
        /// </summary>
        protected override void LoadContent()
        {
            // Create a new SpriteBatch, which can be used to draw textures.
            spriteBatch = new SpriteBatch(GraphicsDevice);

            // TODO: use this.Content to load your game content here
            invader = Content.Load<Texture2D>("invader");
            rectinvader = new Rectangle[rows, cols];
            invaderalive = new String[rows, cols];
            for (int r = 0;r < rows;r++)
                for (int c = 0; c < cols; c++)
                {
                    rectinvader[r, c].Width = invader.Width;
```

```
                rectinvader[r, c].Height = invader.Height;
                rectinvader[r, c].X = 60 * c;
                rectinvader[r, c].Y = 60 * r;
                invaderalive[r, c] = "YES";
            }
        ship = Content.Load<Texture2D>("ship");
        rectship.Width = ship.Width;
        rectship.Height = ship.Height;
        rectship.X = 0;
        rectship.Y = 425;

        bullet = Content.Load<Texture2D>("bullet");
        rectbullet.Width = bullet.Width;
        rectbullet.Height = bullet.Height;
        rectbullet.X = 0;
        rectbullet.Y = 0;
    }

    /// <summary>
    /// UnloadContent will be called once per game and is the place to unload
    /// all content.
    /// </summary>
    protected override void UnloadContent()
    {
        // TODO: Unload any non ContentManager content here
    }

    /// <summary>
    /// Allows the game to run logic such as updating the world,
    /// checking for collisions, gathering input, and playing audio.
    /// </summary>
    /// <param name="gameTime">Provides a snapshot of timing values.</param>
    protected override void Update(GameTime gameTime)
    {
        // Allows the game to exit
        if (GamePad.GetState(PlayerIndex.One).Buttons.Back == ButtonState.Pressed)
            this.Exit();

        // TODO: Add your update logic here
        int rightside = graphics.GraphicsDevice.Viewport.Width;
        int leftside = 0;

        // moving all of our aliens
        for (int r = 0; r < rows; r++)
            for (int c = 0; c < cols; c++)
            {
                if (direction.Equals("RIGHT"))
                    rectinvader[r, c].X = rectinvader[r, c].X + invaderspeed;
                if (direction.Equals("LEFT"))
                    rectinvader[r, c].X = rectinvader[r, c].X - invaderspeed;
            }
        // check to see if any have gone past the rightside
        String changedirection = "N";
        for (int r = 0; r < rows; r++)
            for (int c = 0; c < cols; c++)
            {
                if (invaderalive[r,c].Equals("YES"))
                {
                   if (rectinvader[r, c].X + rectinvader[r, c].Width > rightside)
                   {
                       direction = "LEFT";
                       changedirection = "Y";
                   }
                   if (rectinvader[r, c].X < leftside)
                   {
                       direction = "RIGHT";
```

```
                    changedirection = "Y";
                }
            }
        }
    // if direction has changed, then move aliens down
    if (changedirection.Equals("Y"))
    {
        for (int r = 0; r < rows; r++)
            for (int c = 0; c < cols; c++)
                rectinvader[r, c].Y = rectinvader[r, c].Y + 3;
    }
    // move ship
    KeyboardState kb = Keyboard.GetState();
    if (kb.IsKeyDown(Keys.Left))
        rectship.X = rectship.X - 3;
    if (kb.IsKeyDown(Keys.Right))
        rectship.X = rectship.X + 3;
    if (kb.IsKeyDown(Keys.Space) && bulletvisible.Equals("NO"))
    {
        bulletvisible = "YES";
        rectbullet.X = rectship.X + (rectship.Width / 2) - (rectbullet.Width / 2);
        rectbullet.Y = rectship.Y - rectbullet.Height + 2;
    }
    // if bullet visible, move bullet
    if (bulletvisible.Equals("YES"))
        rectbullet.Y = rectbullet.Y - 5;

    // check to see if bullet has hit an alien
    if (bulletvisible.Equals("YES"))
       for (int r = 0; r < rows; r++)
           for (int c = 0; c < cols; c++)
           {
               if (invaderalive[r,c].Equals("YES"))
                  if (rectbullet.Intersects(rectinvader[r, c]))
                  {
                      bulletvisible = "NO";
                      invaderalive[r, c] = "NO";
                  }
           }

    // if bullet has gone off top of screen, make it invisible
    if (rectbullet.Y + rectbullet.Height < 0)
        bulletvisible = "NO";

    int count = 0;
    for (int r = 0; r < rows; r++)
        for (int c = 0; c < cols; c++)
            if (invaderalive[r,c].Equals("YES"))
                count = count + 1;

    // check to see if half of aliens are dead
    if (count > (rows * cols / 2))
        invaderspeed = invaderspeed;
    if (count < (rows * cols / 3))
        invaderspeed = 6;

    // check to see if game over
    for (int r = 0; r < rows; r++)
        for (int c = 0; c < cols; c++)
        {
            if (invaderalive[r, c].Equals("YES"))
                if (rectinvader[r, c].Y + rectinvader[r, c].Height > rectship.Y)
                    this.Exit();
        }

    base.Update(gameTime);
```

```
        }

        /// <summary>
        /// This is called when the game should draw itself.
        /// </summary>
        /// <param name="gameTime">Provides a snapshot of timing values.</param>
        protected override void Draw(GameTime gameTime)
        {
            GraphicsDevice.Clear(Color.CornflowerBlue);

            // TODO: Add your drawing code here
            spriteBatch.Begin();
            for (int r = 0; r < rows; r++)
                for (int c = 0; c < cols; c++)
                    if (invaderalive[r,c].Equals("YES"))
                       spriteBatch.Draw(invader, rectinvader[r, c], Color.White);
            spriteBatch.Draw(ship, rectship, Color.White);
            if (bulletvisible.Equals("YES"))
                spriteBatch.Draw(bullet, rectbullet, Color.White);
            spriteBatch.End();

            base.Draw(gameTime);
        }
    }
}
```