Bouncing Ball in MonoGame:

Add file to screen:

```
Texture2D myTexture;
Rectangle myRectangle;
```

LoadContent:

```
myTexture= Content.Load<Texture2D>("ball");
myRectangle= new Rectangle(100,100,Texture.Width,Texture.Height);
```

Draw:

```
spriteBatch.Begin();
spriteBatch.Draw(myTexture,myRectangle.Color.White);
spriteBatch.End();
```

Screen boundary parameters:

```
Int screenWidth;
Int screenHeight;
```

In LoadContent add:

```
screenWidth=GraphicsDevice.Viewport.Width;
screenHeight=GraphicsDevice.Viewport.Height;
```

In Update: (The code below stops the rectangle of the object we defined from moving outside the screen)

```
if (myRectangle.X<=0){myRecangle.X=0}
if (myRectangle.Y<=0){myRecangle.Y=0}
if (myRectangle.X +
myTexture.Width>=screenWidth){myRectangle.X=screenWidth-Texture.Width)
if (myRectangle.Y +
myTexture.Height>=screenHeight){myRectangle.Y=screenHeight-
Texture.Height)
```

In Update:

```
Vectore2D velocity;
```

In LoadContent:

```
Velocity.X=3f;
Velocity.Y=3f;
```

In Update:

```
myRectangle.X=myRectangle.X + (int) velocity.X; //cast velocity in int
myRectangle.Y=myRectangle.Y + (int) velocity.Y; //cast velocity in int
```

Bounding properties:

In Update:

```
if (myRectangle.X<=0){velocity.X=-velocity.X};
if myRectangle.Y<=0){velocity.Y=-velocity.Y};
if myRectangle.X + myTexture.Width>=screenWidth){velocity.X=-
velocity.X};
if myRectangle.Y + myTexture.Height>=screenHeight){velocity.Y=-
velocity.Y};
```

Object Bouncing in Different Directions every time it is loaded:

Define random variable:

```
Random myRandom=new Random;
```

In Update:
anytime RandomLoad is called the object goes towards one of the defined direction randomly.

```
Void RandomLoad()
{
Int random=myRandom.Next(0,4); \\ for 0,1,2,3

//Down-Right
If (random==0)
{velocity.X=3f;
velocity.Y=3f;}

//Down-Left-
```

```
If (random==1)
{velocity.X=-3f;
velocity.Y=3f;}

//Up-Right
If (random==2)
{velocity.X=3f;
velocity.Y=-3f;}

//Up-Left
If (random==3)
{velocity.X=-3f;
velocity.Y=-3f;}
}
```

In LoadContent:

```
//finally defined in load content means everytime the program is
loaded
RandomLoad();
```

In Update:

```
if (Keyboard.GetState().IsKeyDown(Keys.Escape))
this.Exit();
```

In LoadContent:

```
if (Keyboard.GetState().IsKeyDown(Keys.Space))
{
myRectangle.X=(screenWidth/2)-(myTexture.Width/2);
myRectangle.Y=(screenHeight/2)-(myTexture.Height/2);

RandomLoad();
}
```

Collision:

Two object interact:

In Update:

```
If (ballRectangle.Intersects(tileRectangle)
velocity.Y=- velocity.Y;
```

Completed Code:

```
using Microsoft.Xna.Framework;
using Microsoft.Xna.Framework.Graphics;
using Microsoft.Xna.Framework.Input;
using System;

namespace BouncingBall
{
   /// <summary>
   /// This is the main type for your game.
   /// </summary>
   public class Game1 : Game
   {
      GraphicsDeviceManager graphics;
      SpriteBatch spriteBatch;
      Texture2D BallTexture;
      Rectangle BallRectangle;
      Texture2D TileTexture;
      Rectangle TileRectangle;
      Vector2 velocity;
      Random myRandom = new Random();

      int screenWidth;
      int screenHeight;

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


      protected override void Initialize()
      {
         base.Initialize();
      }

      protected override void LoadContent()
      {
         // Create a new SpriteBatch, which can be used to draw textures.
         spriteBatch = new SpriteBatch(GraphicsDevice);
         BallTexture = Content.Load<Texture2D>("ball");
         BallRectangle = new Rectangle(100, 100, BallTexture.Width, BallTexture.Height);

         TileTexture = Content.Load<Texture2D>("tile");
         TileRectangle = new Rectangle(200, 400, TileTexture.Width, TileTexture.Height);


         screenWidth=GraphicsDevice.Viewport.Width;
         screenHeight = GraphicsDevice.Viewport.Height;
         BallRectangle.X += BallRectangle.X + (int)velocity.X;

      }
      void RandomLoad()
      {
         int random = myRandom.Next(0, 4);
         if (random == 0)
```

```
         {
            velocity.X = 5f;
            velocity.Y = 5f;
         }
         if (random == 1)
         {
            velocity.X = -5f;
            velocity.Y = 5f;
         }
         if (random == 2)
         {
            velocity.X = 5f;
            velocity.Y = -5f;
         }
         if (random == 3)
         {
            velocity.X = -5f;
            velocity.Y = -5f;
         }
      }

      protected override void UnloadContent()
      {
      }


      protected override void Update(GameTime gameTime)
      {
         if (Keyboard.GetState().IsKeyDown(Keys.Escape))
            this.Exit();
         if (BallRectangle.Intersects(TileRectangle))
         {
            velocity.Y = -velocity.Y;
         }
        //TILE
         if (Keyboard.GetState().IsKeyDown(Keys.Right)) TileRectangle.X += 3;
         if (Keyboard.GetState().IsKeyDown(Keys.Left)) TileRectangle.X -= 3;
         if (TileRectangle.X <= 0) { TileRectangle.X = 0; }
         if (TileRectangle.X + TileTexture.Width > screenWidth) { TileRectangle.X =
screenWidth- TileTexture.Width; }


         BallRectangle.X = BallRectangle.X+(int)velocity.X;
         BallRectangle.Y = BallRectangle.Y + (int)velocity.Y;


        // if (Keyboard.GetState().IsKeyDown(Keys.Down)) BallRectangle.Y += 3;
         if (BallRectangle.X <= 0) velocity.X = -velocity.X;
         if (Keyboard.GetState().IsKeyDown(Keys.Space)) {

            RandomLoad();

         }
           if (BallRectangle.Y <= 0) velocity.Y=-velocity.Y;
           if (BallRectangle.X + BallTexture.Width >= screenWidth)  {velocity.X = -
velocity.X;}
           if (BallRectangle.Y + BallTexture.Height >= screenHeight) {velocity.Y = -
velocity.Y;}
```

```
            base.Update(gameTime);
        }

        protected override void Draw(GameTime gameTime)
        {
            GraphicsDevice.Clear(Color.CornflowerBlue);

            spriteBatch.Begin();
            spriteBatch.Draw(BallTexture, BallRectangle, Color.White);
            spriteBatch.Draw(TileTexture, TileRectangle, Color.White);
            spriteBatch.End();



            // TODO: Add your drawing code here

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