UNIVERSITY OF Hull

# Mad Game Development

*Rob Miles*

*Department of Computer Science*

## Introduction

- Farseer Overview
- Bodies and Geometries
- Creating a shelf, target and ball
- A little bit of Physics
- Detecting collisions between objects

## An Apology

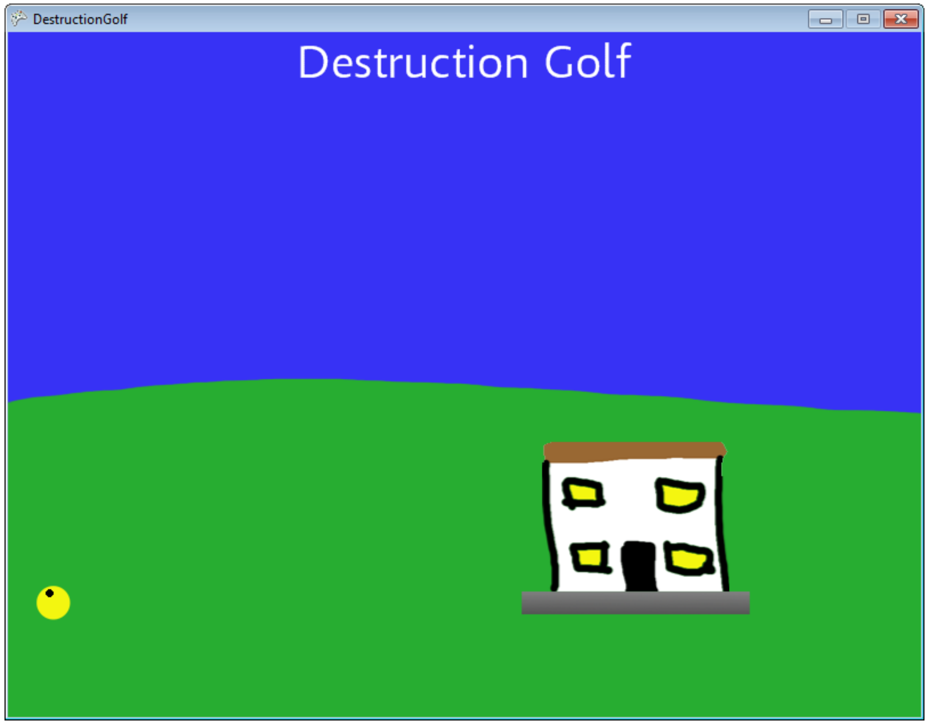
- I don’t know a lot about Farseer
- I know just enough to get it to do what I want
  - Which suits me fine
- There is a lot more you can do with this tool but for the purpose of this talk I am going to keep things really simple
- If you know better ways of doing what I’m doing, then please let me know

## Farseer Versions

- I am using Farseer version 2.n
  - There is a new version which is 3.0
  - This is somewhat different, but even better
- But I haven't learnt how to use it yet
- So we are using 2.n
  - This is good however, as it works with XNA 3.1 and on the Windows Phone with XNA 4.0

## What we are building

- We are using essentially the same scenario as before
- Only this time Farseer will be in control of the movement and behaviour of our game objects
- The house has to sit on a shelf which doesn't move
- The aim of the game is to knock the house off the shelf



## Farseer Objects

- Objects in the Farseer universe are made up of a Body and a Geometry
- We create instances of these and add them to the Farseer simulator
- When the XNA game updates we also update the simulator so that it moves the objects around
- We create the simulator at the start of the program

## The Simulator

```
//The physics simulator is always needed.
// We supply a gravity of 250 units
PhysicsSimulator simulator =
                        new PhysicsSimulator(new Vector2(0, 250));
```

- When we create the simulator we also give it a vector that indicates the direction of gravity
  - A vector is a way of expressing a direction
  - We can create it by giving it x and y direction values
- We can change this to make gravity work in any direction we like, or with any amount of strength, which might be fun

## The shelf data

```
/// <summary>
/// Texture of the shelf the target sits on
/// </summary>
Texture2D shelfTexture;
Body shelfBody;
Geom shelfGeom;
Vector2 shelfOrigin;
```

- The shelf is the simplest object on the screen
- It has a texture that is used to draw it, a Farseer body and a Farseer geometry
- It also has an origin value, which is the “centre” of the object

## The shelf body

```
shelfTexture = Content.Load<Texture2D>(@"images\shelf");
shelfOrigin = new Vector2(shelfTexture.Width / 2.0f,
                          shelfTexture.Height / 2.0f);
shelfBody = BodyFactory.Instance.CreateRectangleBody(
                     shelfTexture.Width, shelfTexture.Height, 1);
shelfBody.IsStatic = true;
shelfBody.Position = new Vector2(550, 500);
simulator.Add(shelfBody);
```

- This creates a shelf body  and places it in the world
- Note that I’ve scaled the texture to fit the display world
- I have also made the shelf static, so that it can’t move around in the world

## The shelf geometry

```
shelfGeom = GeomFactory.Instance.CreateRectangleGeom(shelfBody,
                        shelfTexture.Width, shelfTexture.Height);
shelfGeom.RestitutionCoefficient = 0.3f;
shelfGeom.FrictionCoefficient = 0.5f;
simulator.Add(shelfGeom);
```

- This creates a shelf geometry  and places it in the world
- This sets out how much bounce (RestitutionCoefficient) we get from the shelf and how slippery (FrictionCoefficient) we want it to be
- Playing with these values is fun

## Drawing the shelf

```
spriteBatch.Draw(shelfTexture, shelfGeom.Position, null,
        Color.White, shelfGeom.Rotation, shelfOrigin, 1,
        SpriteEffects.None, 1);
```

- We use a more advanced version of the Draw method that lets us specify a rotation value for the texture
  - The shelf won’t rotate, but other objects will
- Note that we get these values out of the geometry for the shape
- The draw operations for all the game objects look the same

## The Target data

- The target is the house we are aiming at
- It has the same information stored about it as the shelf
- We will place the target on the shelf when the game starts
- The only difference between the target and the shelf is that the target is allowed to move
- The aim of the game is to knock the target off the shelf

# The Target mass

```
targetBody = BodyFactory.Instance.CreateRectangleBody(
                targetTexture.Width, targetTexture.Height, 3);
```

- Bodies in Farseer are given mass
- The higher the mass the harder they are to get moving, and the more damage they do when they hit
- The mass of the shelf is not an issue, since it is fixed in position
- I've given the target a mass of 3
- This is a value you might want to fine tune

UNIVERSITY OF Hull

## Ball Position

```
ballGeom = GeomFactory.Instance.CreateCircleGeom(ballBody,
                                    ballTexture.Width/2, 50);
ballGeom.RestitutionCoefficient = 0.3f;
ballGeom.RestitutionCoefficient = 0.3f;
ballGeom.FrictionCoefficient = 0.5f;
```

- The ball geometry is based on a circle rather than a rectangle
- When create a circle you give the radius and the number of *edges*
  - This is because the circle is actually an approximation
- The more edges, the more accurate the approximation but the more work it is to calculate

## Ball Launching

```
aimingShot = false;
ballBody.IsStatic = false;
ballBody.ApplyImpulse(
           new Vector2(gamePad.ThumbSticks.Left.X * 500,
                      -gamePad.ThumbSticks.Left.Y * 500));
```

- When we are aiming the shot we make the ball static
  - Otherwise it just falls off the screen
- When the ball is launched we apply an impulse in the direction of the thumbstick
  - 500 seems to give a good ball speed

UNIVERSITY OF Hull

## Detecting Collisions

```
ballGeom.OnCollision +=
                      new CollisionEventHandler(ballHits);
```

- Farseer will detect collisions for us
- We can bind an event handler to the collision event
- When the collision occurs the handler method is called automatically
- This makes use of C# delegates which are references to methods in classes

## Collision handler

```
bool ballHits(Geom geometry1, Geom geometry2,
              ContactList contactList)
{
    if (ballSoundActive) return true;
    if (geometry1 == shelfGeom || geometry2 == shelfGeom)
        return true;
    ballSoundActive = true;
    bangSound.Play();
    return true;
}
```

- Called whenever a collision is detected
- Checks to see if the ball has hit the shelf or the target

University of Hull

## Collision handler

```
bool ballHits(Geom geometry1, Geom geometry2,
              ContactList contactList)
{
    if (ballSoundActive) return true;
    if (geometry1 == shelfGeom || geometry2 == shelfGeom)
        return true;
    ballSoundActive = true;
    bangSound.Play();
    return true;
}
```

- Only plays the sound if the ball has hit the house
- Only plays the sound once

## Updating Farseer

```
simulator.Update(gameTime.ElapsedGameTime.Milliseconds * .0005f);
```

- We need to add this method call to the Update method to keep the Farseer simulation running
- The time factor controls the rate at which the engine updates
- You can tune this if you have problems with items moving so fast they go inside or through each other
- For the purpose of my program the value above works fine

## Scrolling Background

```
if (ballGeom.Position.X + scrollMargin > width)
{
   // need to scroll the screen
   xOffset = width - (ballGeom.Position.X + scrollMargin);
}
```

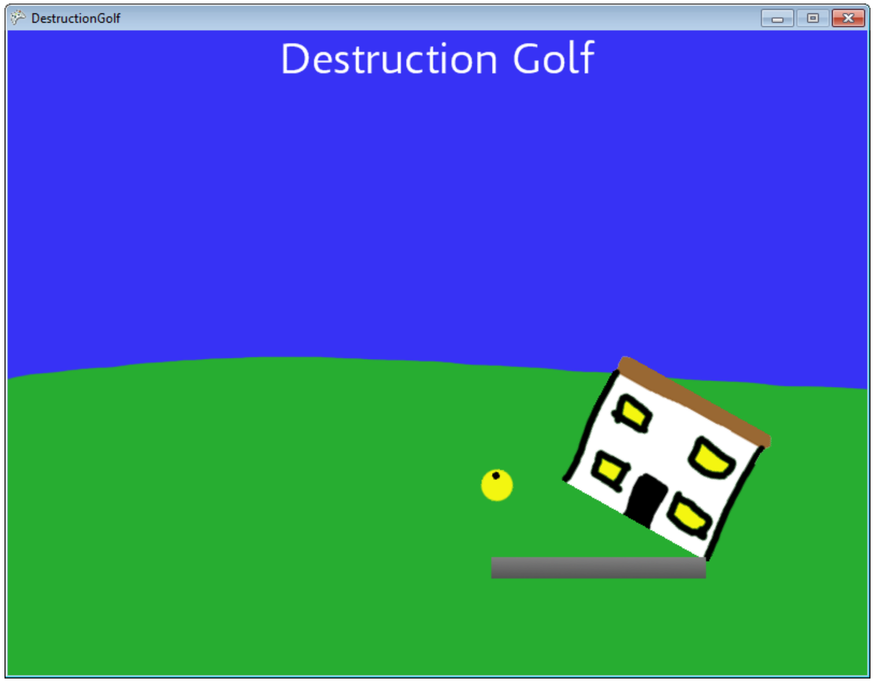
- The scrolling is done in exactly the same way as before
- Only now I draw a rectangle the right size ☺
- We can get the position out of the ball geometry so our program can tell where the ball is

## Messing with the game

- You can dramatically change the way the game plays by
  - Changing gravity
  - Changing the friction and bounciness of the objects
  - Changing the mass of the objects
  - Changing the initial velocity of the ball
- This is fun!

## Sample Code

- All the code you have seen
- The house now has a flat roof because it has turned into a rectangle
- Farseer lets you create polygons, so you could create a triangular roof to put on top of the house



## Questions?

www.destructiongolf.com