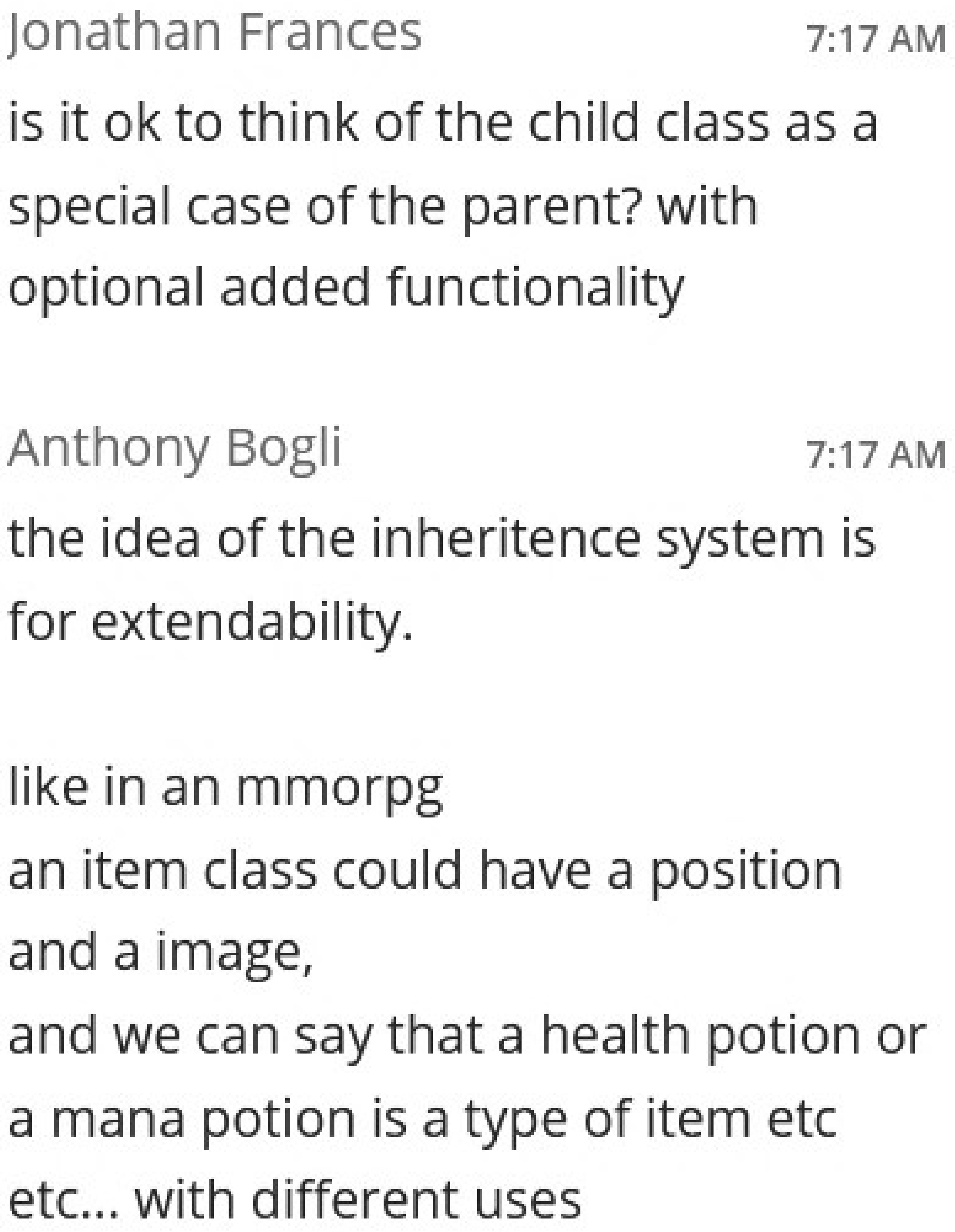


Jonathan Frances 7:17 AM

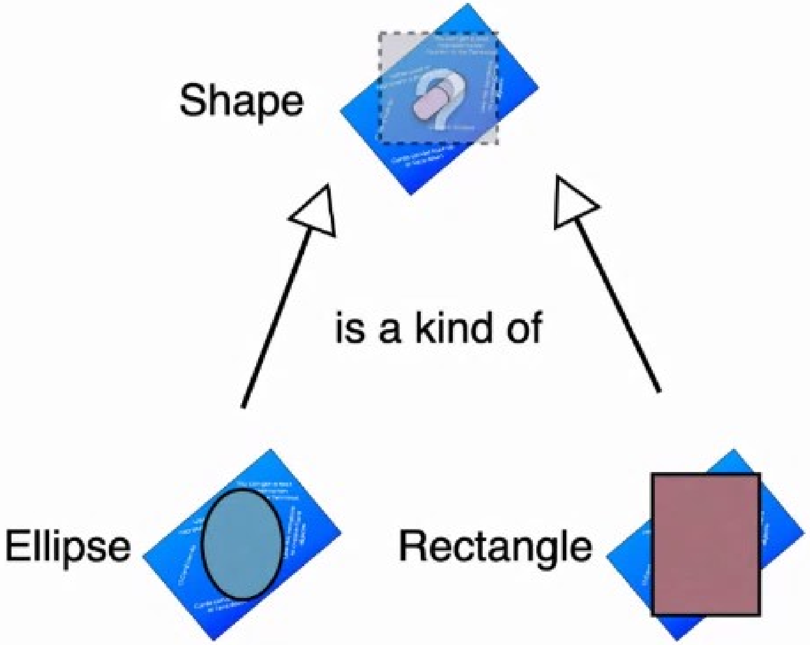
is it ok to think of the child class as a special case of the parent? with optional added functionality

Anthony Bogli 7:17 AM

the idea of the inheritence system is for extendability.

like in an mmorpg
an item class could have a position
and a image,
and we can say that a health potion or
a mana potion is a type of item etc
etc... with different uses

Parent classes can have *placeholder* methods that **must** be overridden



How does Shape Draw?
It doesn’t; Draw is a placeholder = **abstract**
abstract classes **cannot create objects**

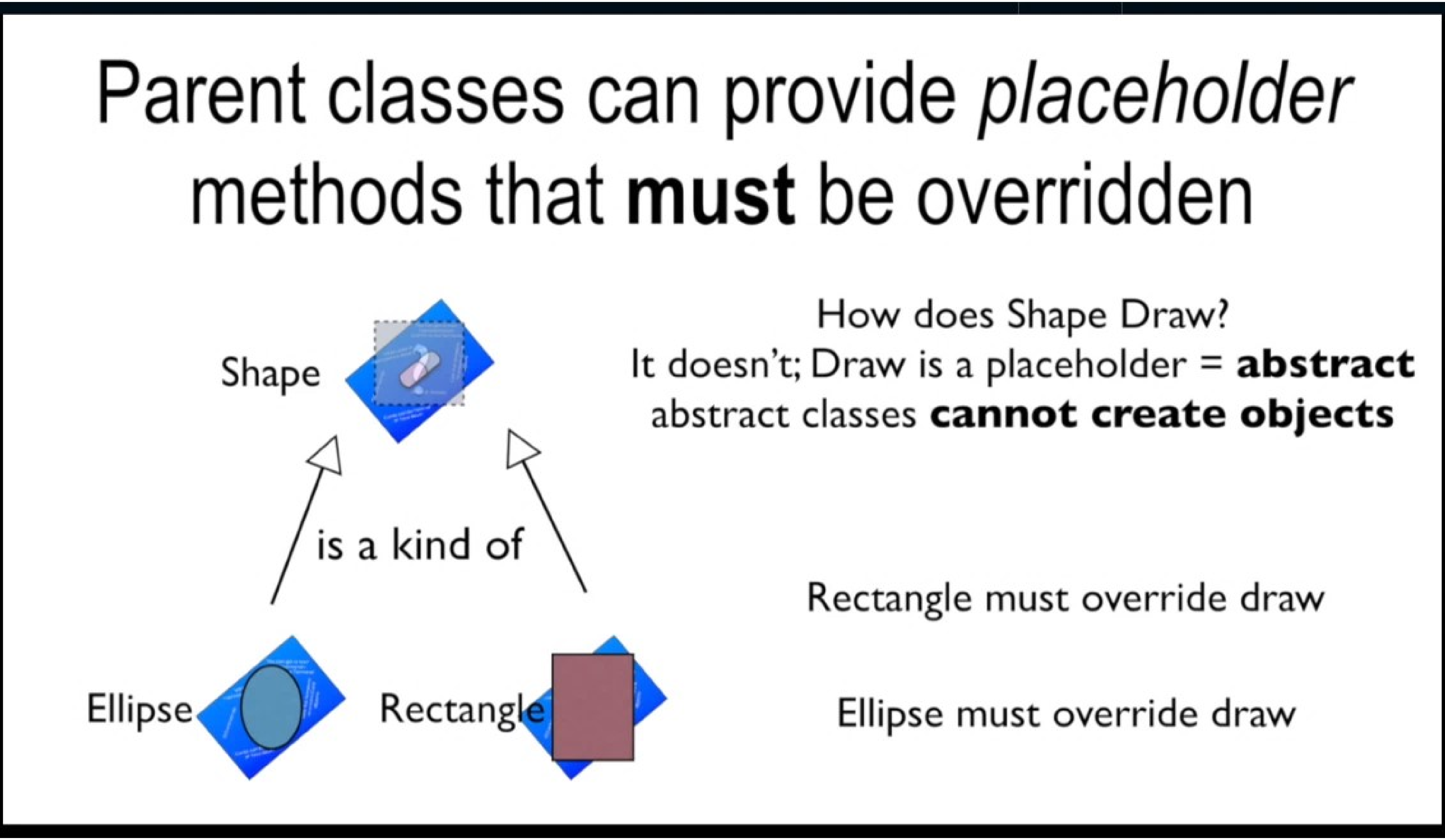
Rectangle must override draw

Ellipse must override draw

So abstract methods are a guarantee that all children must implement that method, and any class with an abstract method can't be created as objects?

If shape and rectangle both have a method with same name and we call it via object of rectangle than which method will be called?

Rect

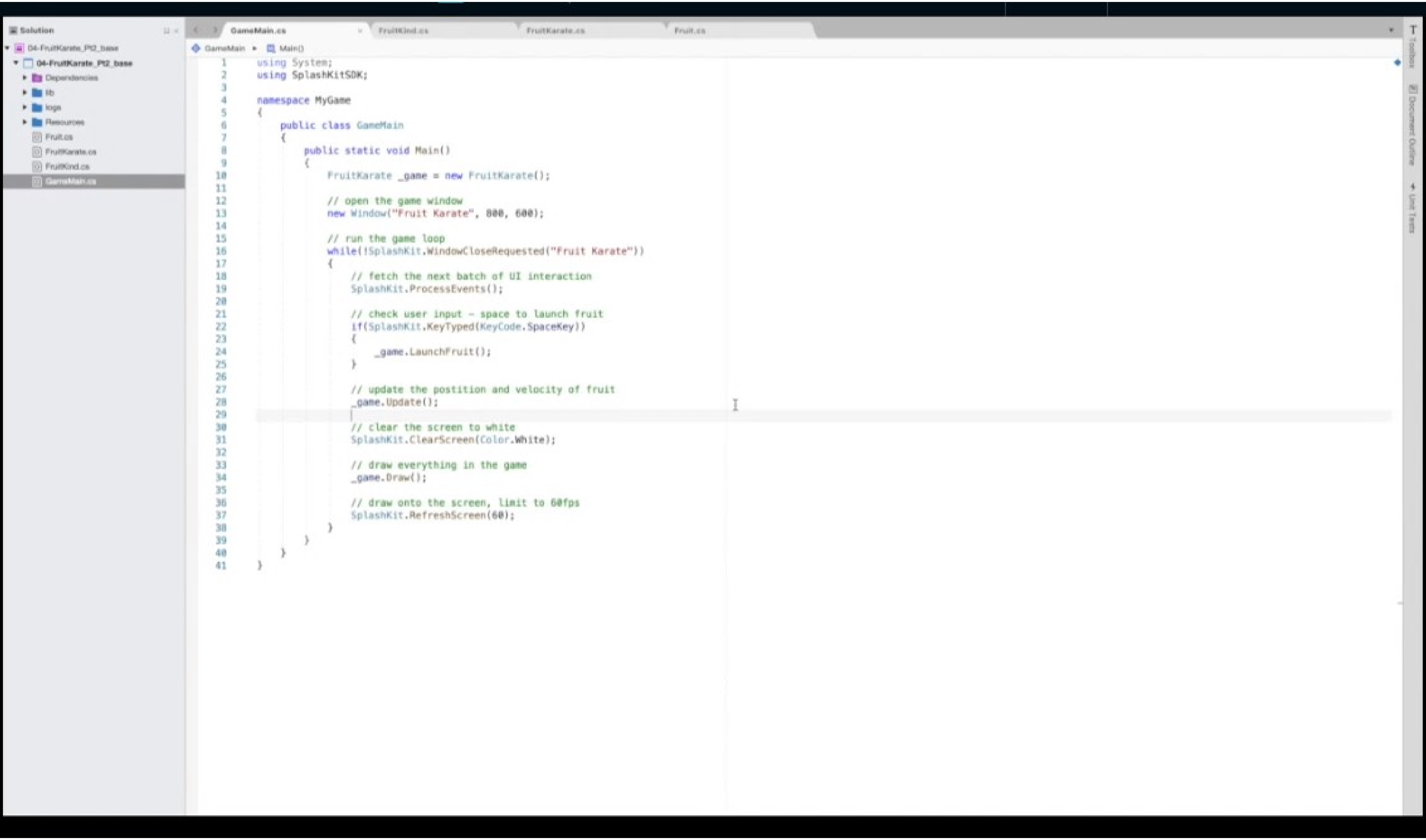


Parent classes can provide *placeholder* methods that **must** be overridden

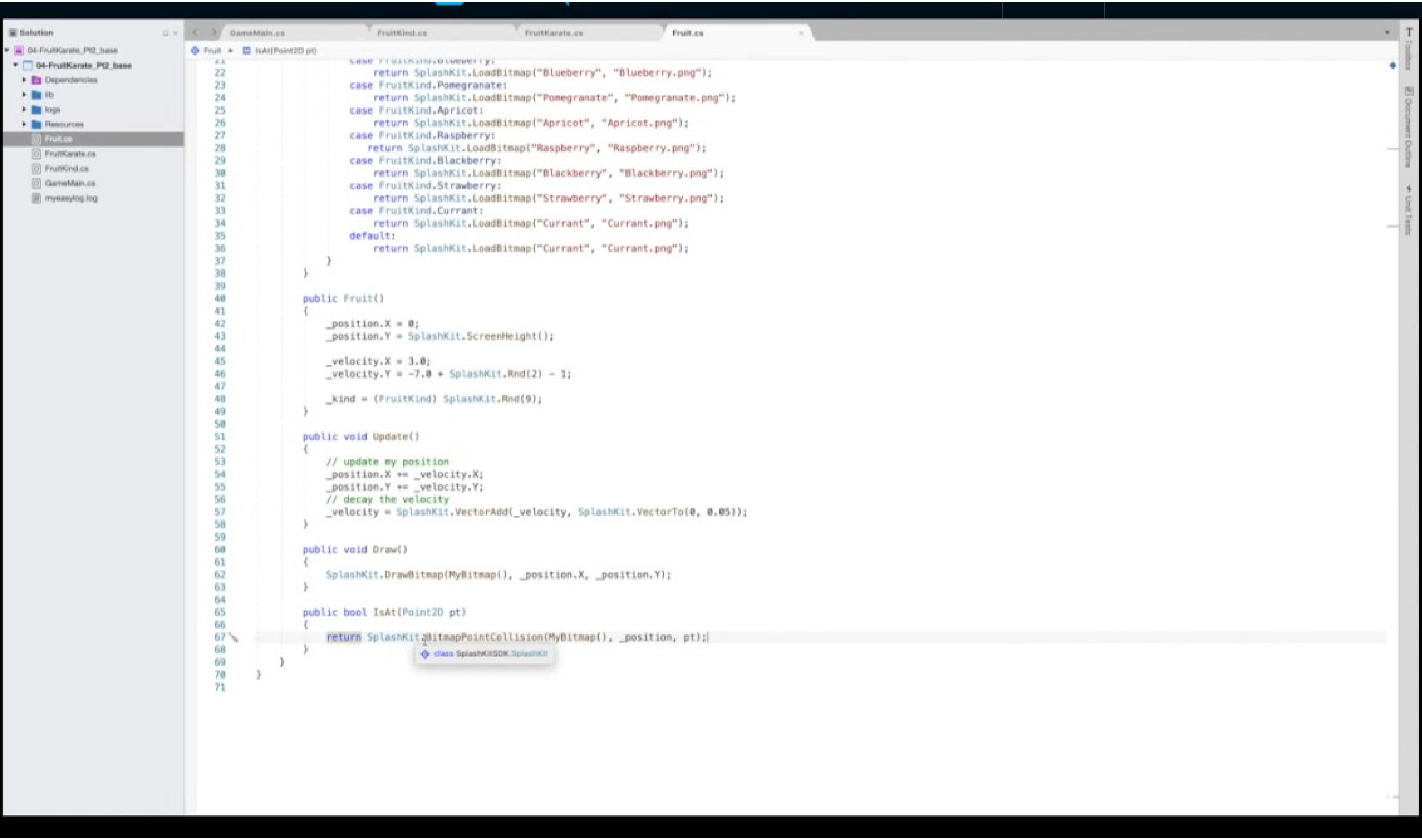
How does Shape Draw?
It doesn't; Draw is a placeholder = **abstract**
abstract classes **cannot create objects**

Rectangle must override draw

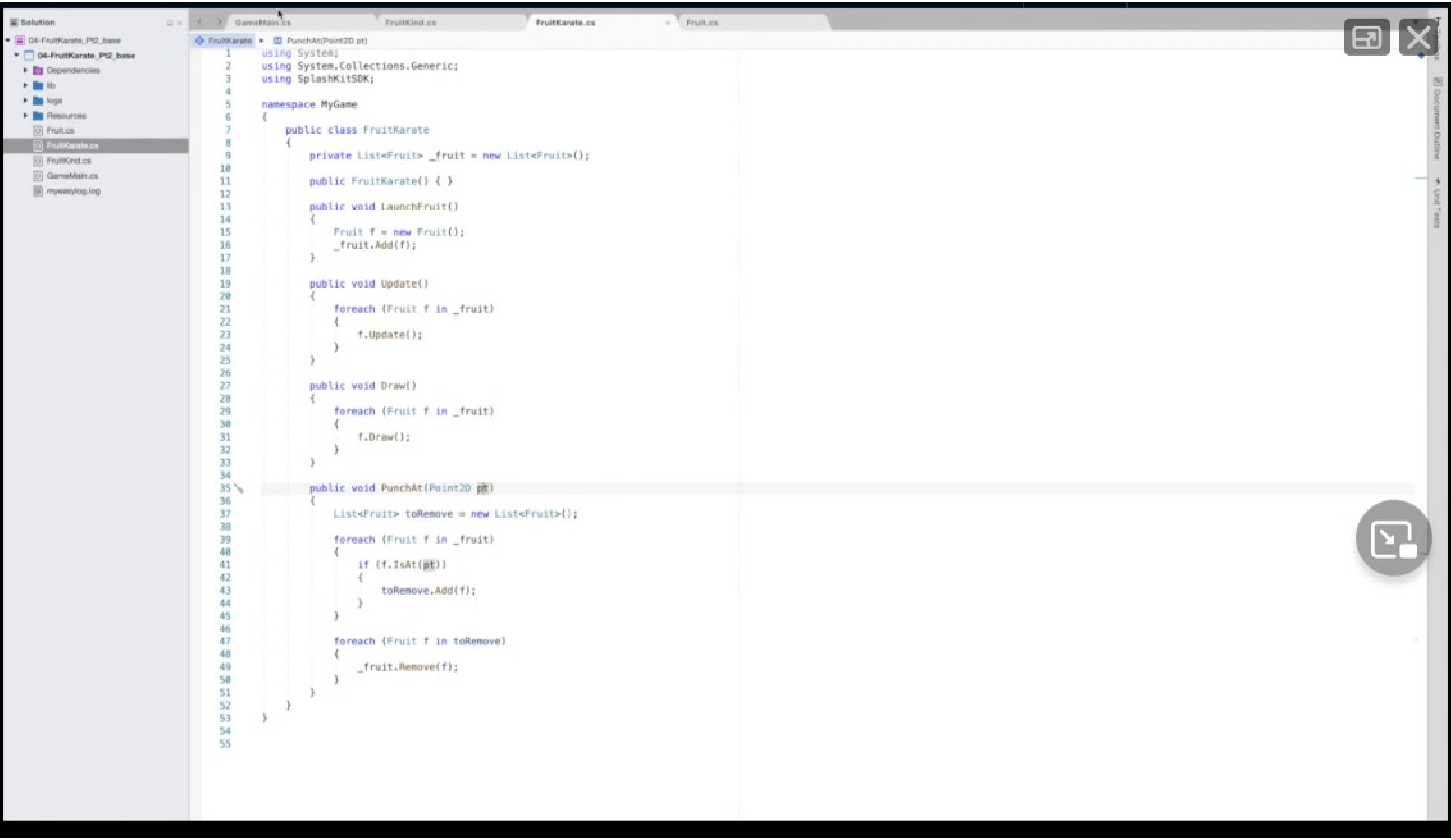
Ellipse must override draw



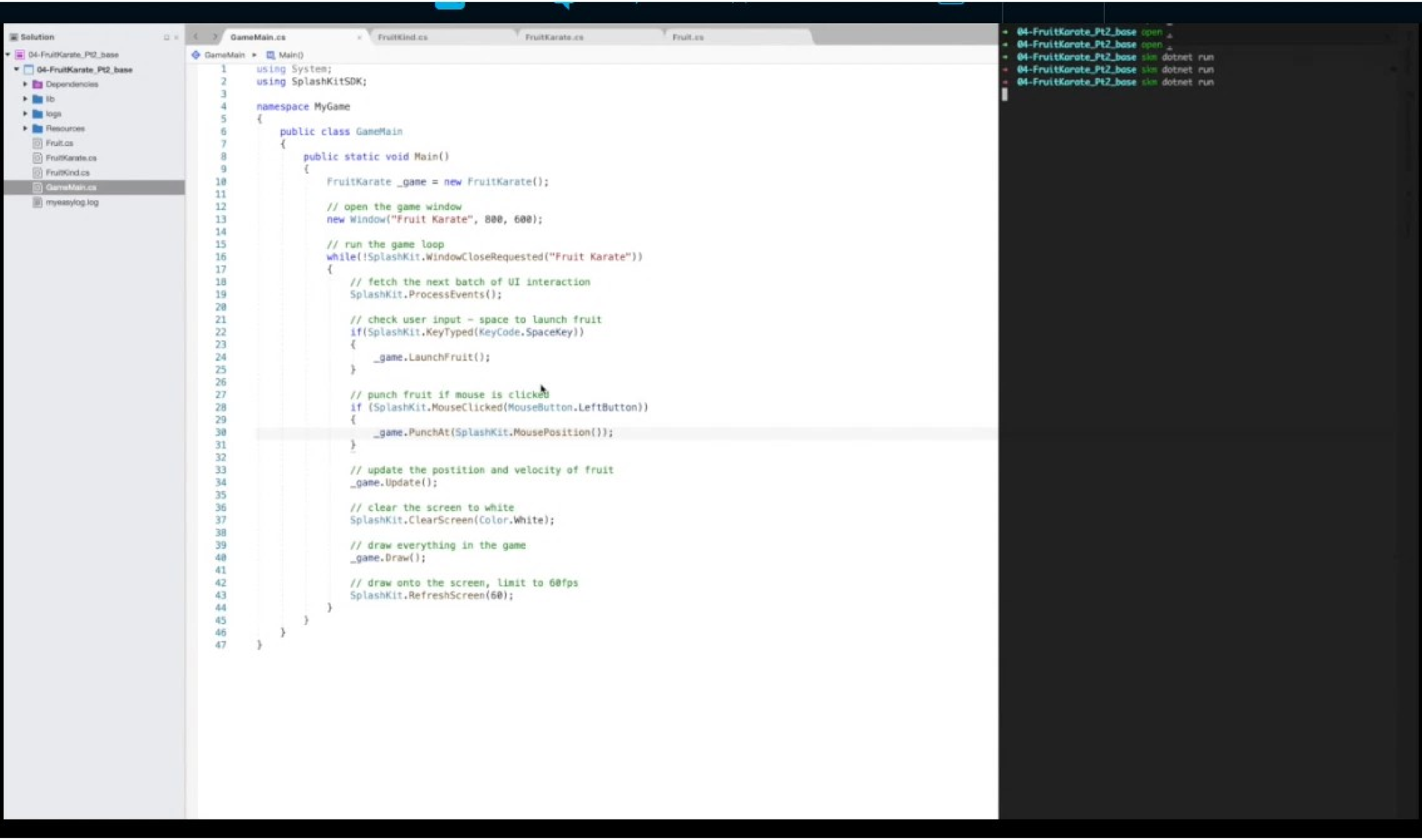
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class GameMain
    {
        public static void Main()
        {
            FruitKarate _game = new FruitKarate();

            // open the game window
            new Window("Fruit Karate", 800, 600);

            // run the game loop
            while(!SplashKit.WindowCloseRequested("Fruit Karate"))
            {
                // fetch the next batch of UI interaction
                SplashKit.ProcessEvents();

                // check user input - space to launch fruit
                if(SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    _game.LaunchFruit();
                }

                // update the postition and velocity of fruit
                _game.Update();

                // clear the screen to white
                SplashKit.ClearScreen(Color.White);

                // draw everything in the game
                _game.Draw();

                // draw onto the screen, limit to 60fps
                SplashKit.RefreshScreen(60);
            }
        }
    }
}
```



```
                case FruitKind.Blueberry:
                    return SplashKit.LoadBitmap("Blueberry", "Blueberry.png");
                case FruitKind.Pomegranate:
                    return SplashKit.LoadBitmap("Pomegranate", "Pomegranate.png");
                case FruitKind.Apricot:
                    return SplashKit.LoadBitmap("Apricot", "Apricot.png");
                case FruitKind.Raspberry:
                   return SplashKit.LoadBitmap("Raspberry", "Raspberry.png");
                case FruitKind.Blackberry:
                    return SplashKit.LoadBitmap("Blackberry", "Blackberry.png");
                case FruitKind.Strawberry:
                    return SplashKit.LoadBitmap("Strawberry", "Strawberry.png");
                case FruitKind.Currant:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
                default:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
            }
        }

        public Fruit()
        {
            _position.X = 0;
            _position.Y = SplashKit.ScreenHeight();

            _velocity.X = 3.0;
            _velocity.Y = -7.0 + SplashKit.Rnd(2) - 1;

            _kind = (FruitKind) SplashKit.Rnd(9);
        }

        public void Update()
        {
            // update my position
            _position.X += _velocity.X;
            _position.Y += _velocity.Y;
            // decay the velocity
            _velocity = SplashKit.VectorAdd(_velocity, SplashKit.VectorTo(0, 0.05));
        }

        public void Draw()
        {
            SplashKit.DrawBitmap(MyBitmap(), _position.X, _position.Y);
        }

        public bool IsAt(Point2D pt)
        {
            return SplashKit.BitmapPointCollision(MyBitmap(), _position, pt);
        }
    }
}
```



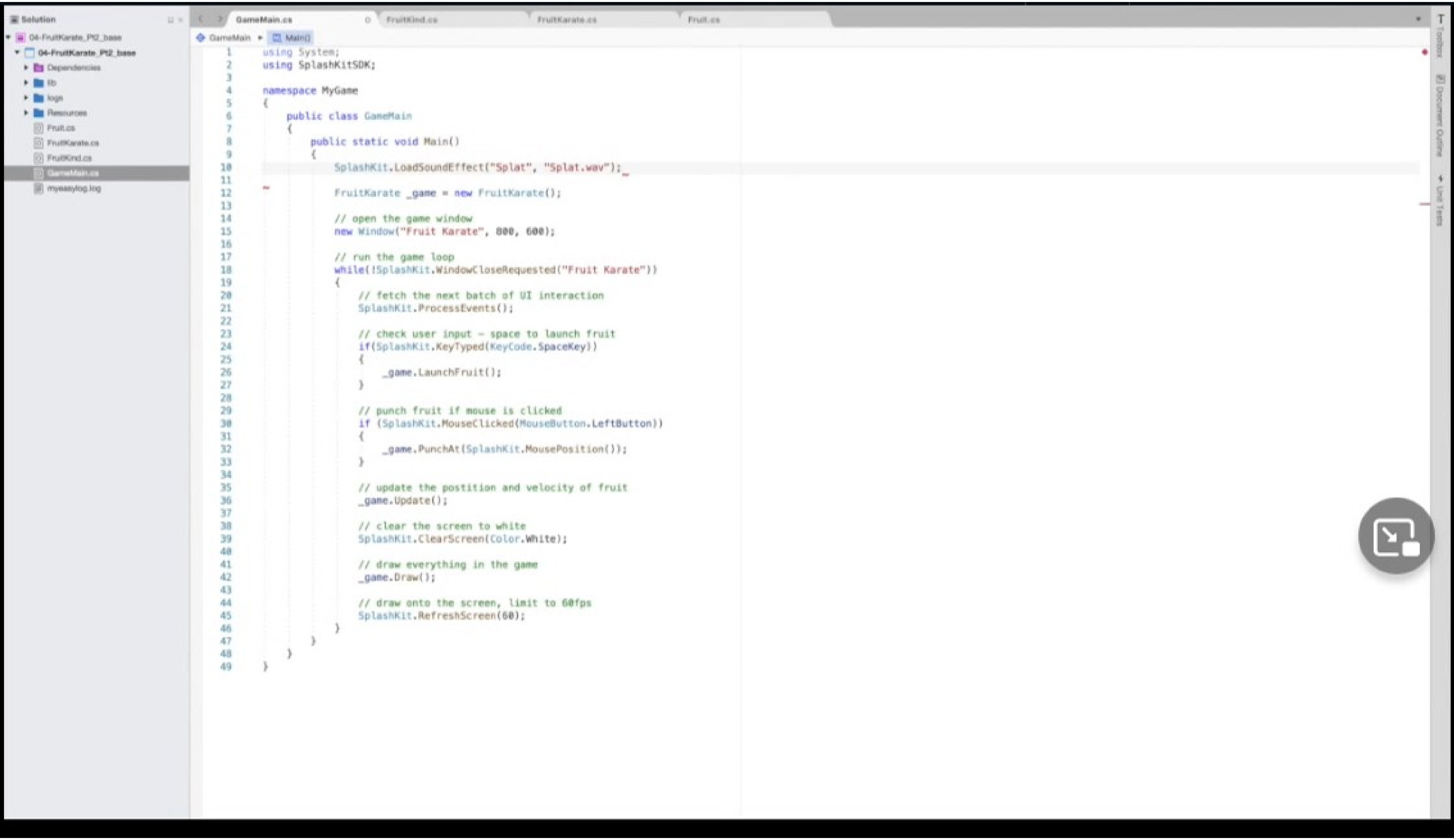
```
using System;
using System.Collections.Generic;
using SplashKitSDK;

namespace MyGame
{
    public class FruitKarate
    {
        private List<Fruit> _fruit = new List<Fruit>();

        public FruitKarate() { }

        public void LaunchFruit()
        {
            Fruit f = new Fruit();
            _fruit.Add(f);
        }

        public void Update()
        {
            foreach (Fruit f in _fruit)
            {
                f.Update();
            }
        }

        public void Draw()
        {
            foreach (Fruit f in _fruit)
            {
                f.Draw();
            }
        }

        public void PunchAt(Point2D pt)
        {
            List<Fruit> toRemove = new List<Fruit>();

            foreach (Fruit f in _fruit)
            {
                if (f.IsAt(pt))
                {
                    toRemove.Add(f);
                }
            }

            foreach (Fruit f in toRemove)
            {
                _fruit.Remove(f);
            }
        }
    }
}
```



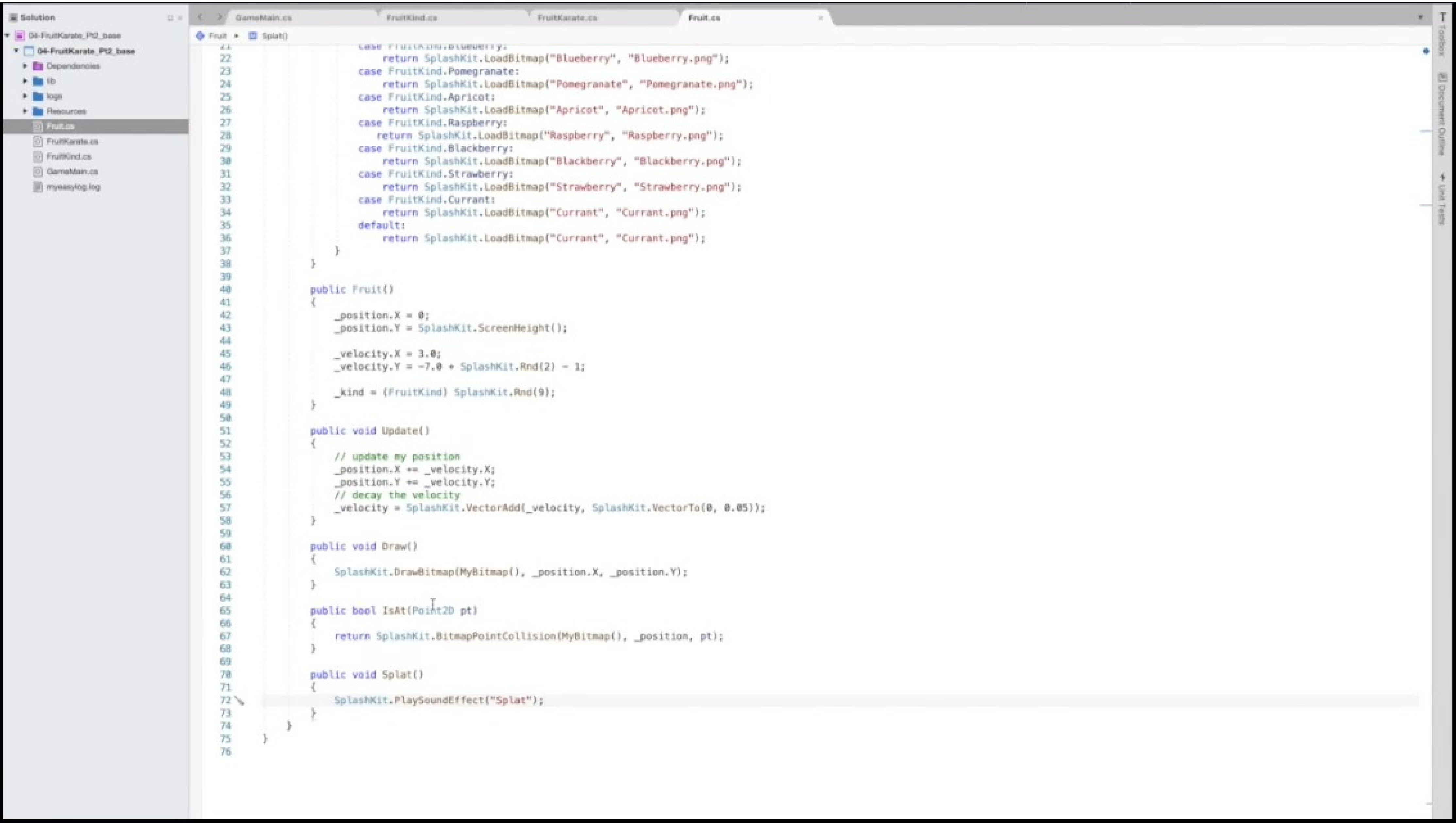
```csharp
using System;
using SplashKitSDK;

namespace MyGame
{
    public class GameMain
    {
        public static void Main()
        {
            FruitKarate _game = new FruitKarate();

            // open the game window
            new Window("Fruit Karate", 800, 600);

            // run the game loop
            while(!SplashKit.WindowCloseRequested("Fruit Karate"))
            {
                // fetch the next batch of UI interaction
                SplashKit.ProcessEvents();

                // check user input - space to launch fruit
                if(SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    _game.LaunchFruit();
                }

                // punch fruit if mouse is clicked
                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    _game.PunchAt(SplashKit.MousePosition());
                }

                // update the postition and velocity of fruit
                _game.Update();

                // clear the screen to white
                SplashKit.ClearScreen(Color.White);

                // draw everything in the game
                _game.Draw();

                // draw onto the screen, limit to 60fps
                SplashKit.RefreshScreen(60);
            }
        }
    }
}
```

```
→ 04-FruitKarate_Pt2_base open .
→ 04-FruitKarate_Pt2_base open .
→ 04-FruitKarate_Pt2_base skm dotnet run
→ 04-FruitKarate_Pt2_base skm dotnet run
→ 04-FruitKarate_Pt2_base skm dotnet run
```

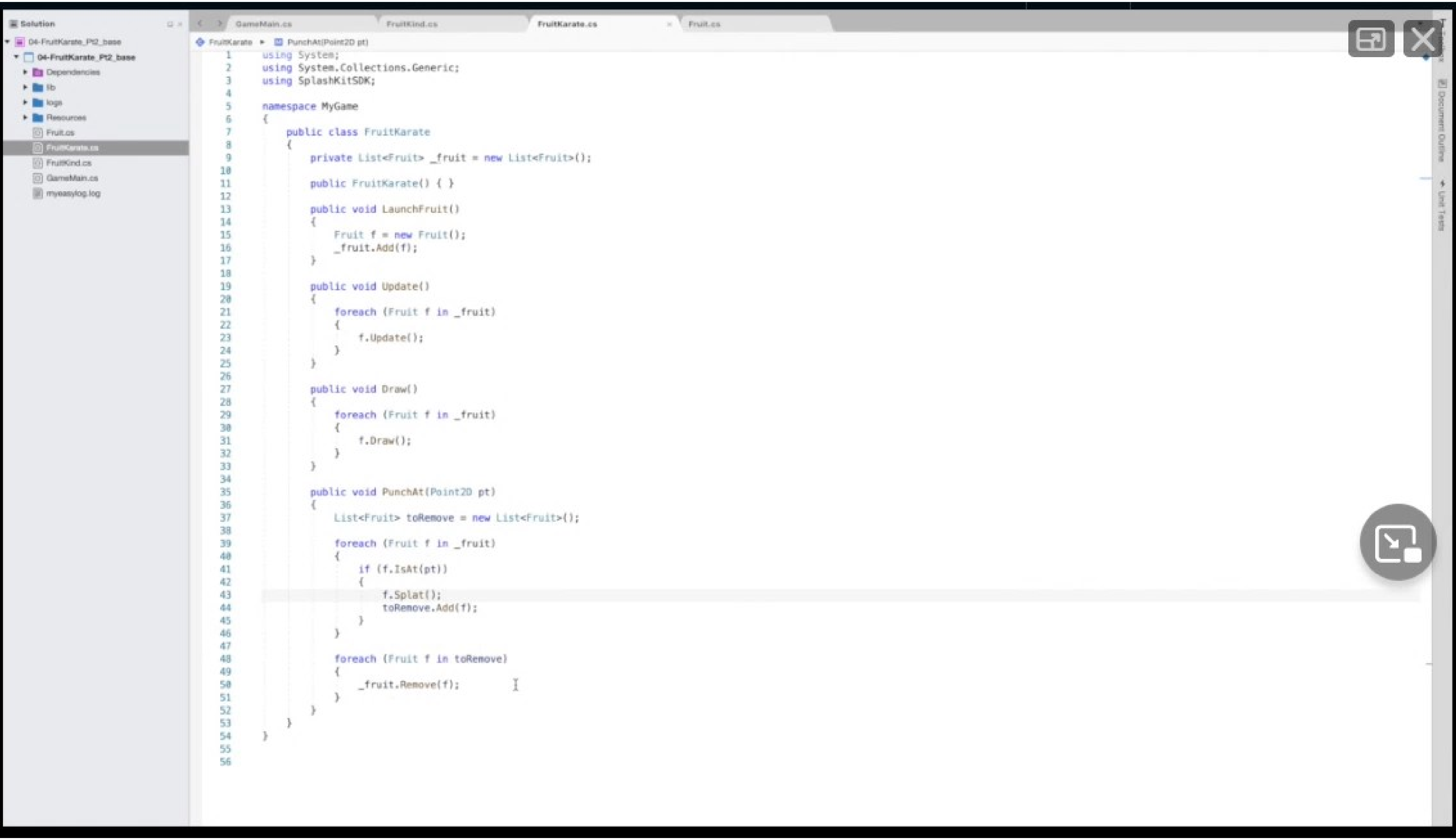
game = new FruitKarate();
Fruit Karate



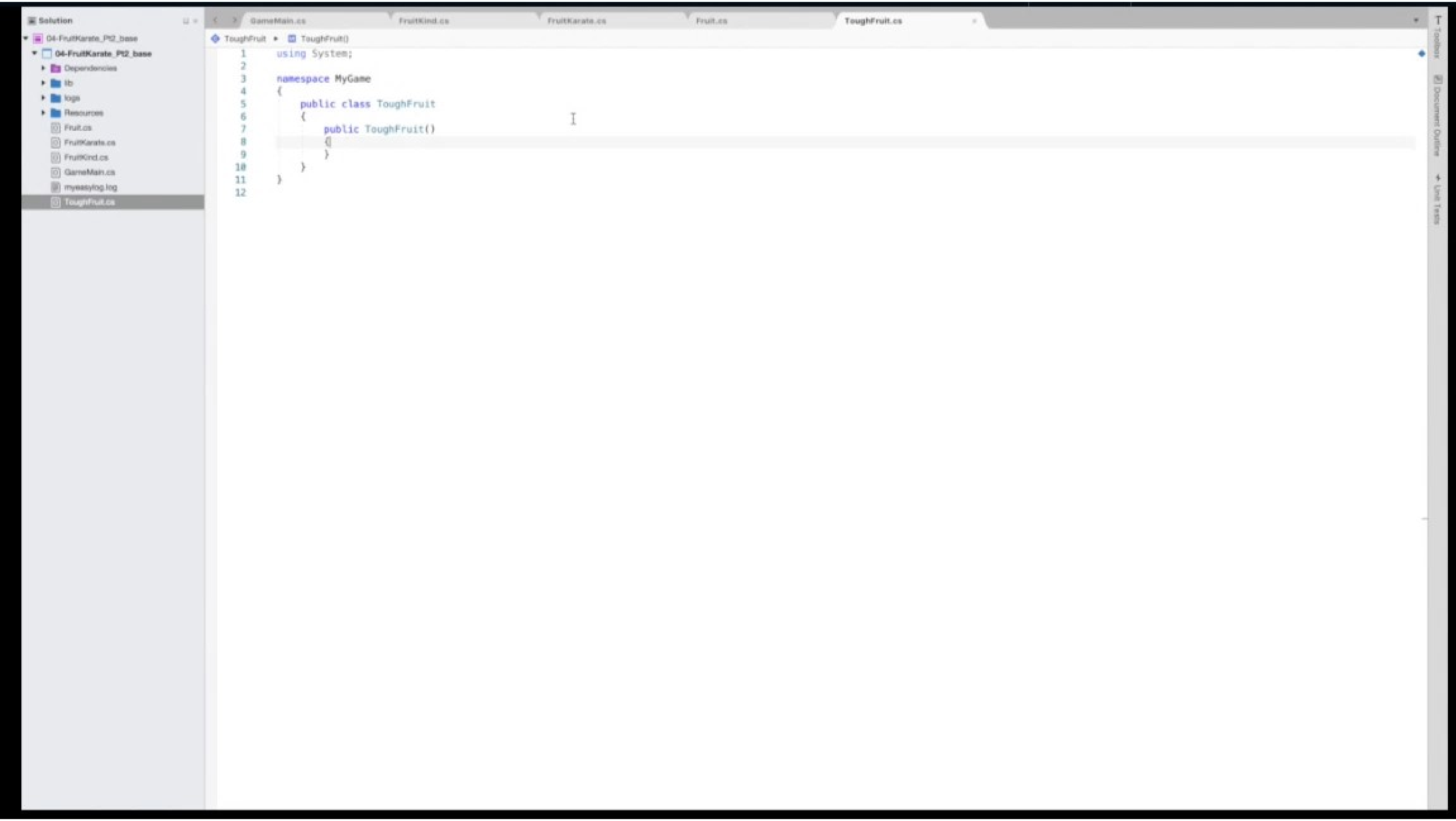
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class GameMain
    {
        public static void Main()
        {
            SplashKit.LoadSoundEffect("Splat", "Splat.wav");_

            FruitKarate _game = new FruitKarate();

            // open the game window
            new Window("Fruit Karate", 800, 600);

            // run the game loop
            while(!SplashKit.WindowCloseRequested("Fruit Karate"))
            {
                // fetch the next batch of UI interaction
                SplashKit.ProcessEvents();

                // check user input - space to launch fruit
                if(SplashKit.KeyTyped(KeyCode.SpaceKey))
                {
                    _game.LaunchFruit();
                }

                // punch fruit if mouse is clicked
                if (SplashKit.MouseClicked(MouseButton.LeftButton))
                {
                    _game.PunchAt(SplashKit.MousePosition());
                }

                // update the postition and velocity of fruit
                _game.Update();

                // clear the screen to white
                SplashKit.ClearScreen(Color.White);

                // draw everything in the game
                _game.Draw();

                // draw onto the screen, limit to 60fps
                SplashKit.RefreshScreen(60);
            }
        }
    }
}
```



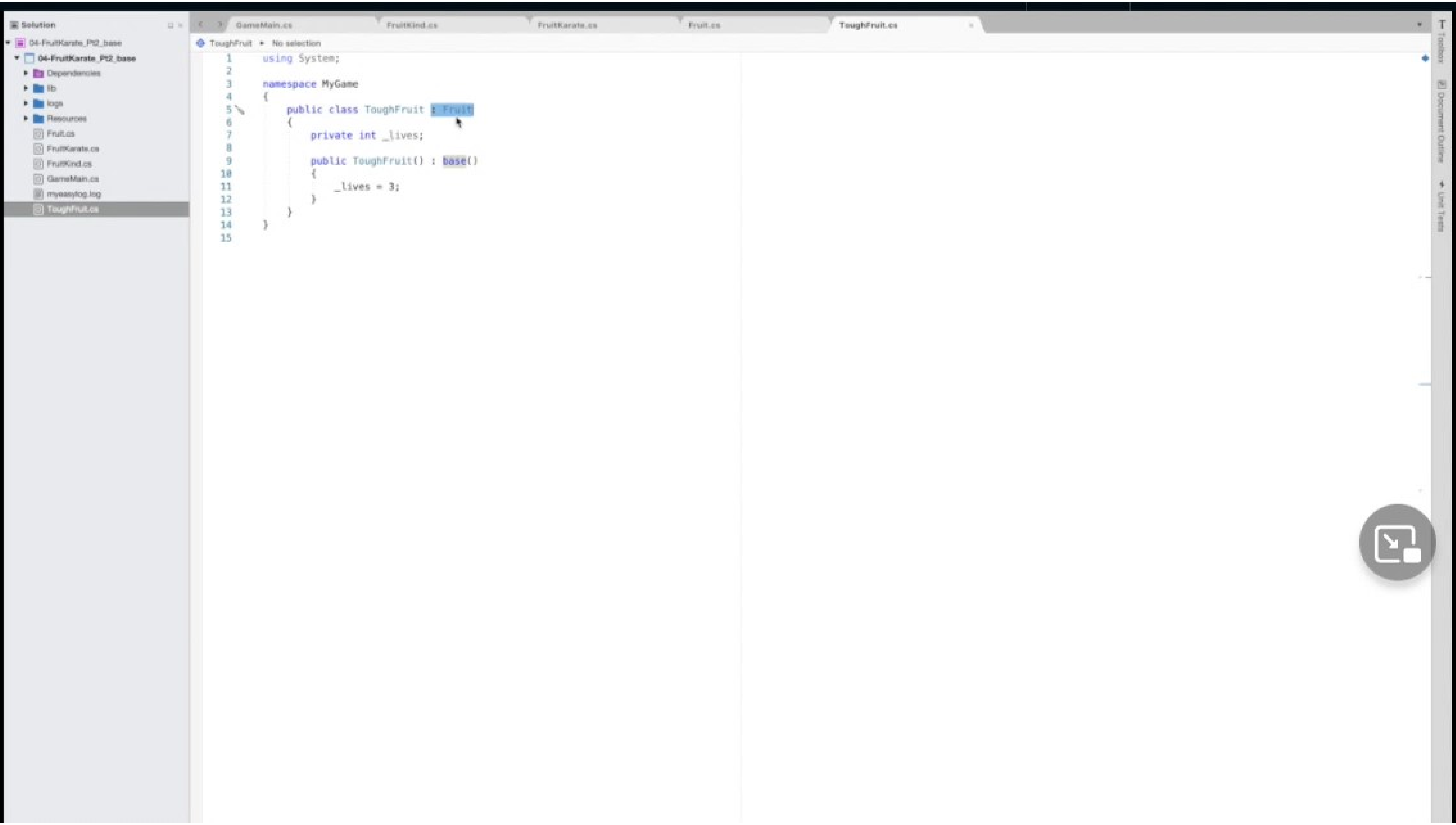
```
                case FruitKind.Blueberry:
                    return SplashKit.LoadBitmap("Blueberry", "Blueberry.png");
                case FruitKind.Pomegranate:
                    return SplashKit.LoadBitmap("Pomegranate", "Pomegranate.png");
                case FruitKind.Apricot:
                    return SplashKit.LoadBitmap("Apricot", "Apricot.png");
                case FruitKind.Raspberry:
                   return SplashKit.LoadBitmap("Raspberry", "Raspberry.png");
                case FruitKind.Blackberry:
                    return SplashKit.LoadBitmap("Blackberry", "Blackberry.png");
                case FruitKind.Strawberry:
                    return SplashKit.LoadBitmap("Strawberry", "Strawberry.png");
                case FruitKind.Currant:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
                default:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
            }
        }

        public Fruit()
        {
            _position.X = 0;
            _position.Y = SplashKit.ScreenHeight();

            _velocity.X = 3.0;
            _velocity.Y = -7.0 + SplashKit.Rnd(2) - 1;

            _kind = (FruitKind) SplashKit.Rnd(9);
        }

        public void Update()
        {
            // update my position
            _position.X += _velocity.X;
            _position.Y += _velocity.Y;
            // decay the velocity
            _velocity = SplashKit.VectorAdd(_velocity, SplashKit.VectorTo(0, 0.05));
        }

        public void Draw()
        {
            SplashKit.DrawBitmap(MyBitmap(), _position.X, _position.Y);
        }

        public bool IsAt(Point2D pt)
        {
            return SplashKit.BitmapPointCollision(MyBitmap(), _position, pt);
        }

        public void Splat()
        {
            SplashKit.PlaySoundEffect("Splat");
        }
    }
}
```



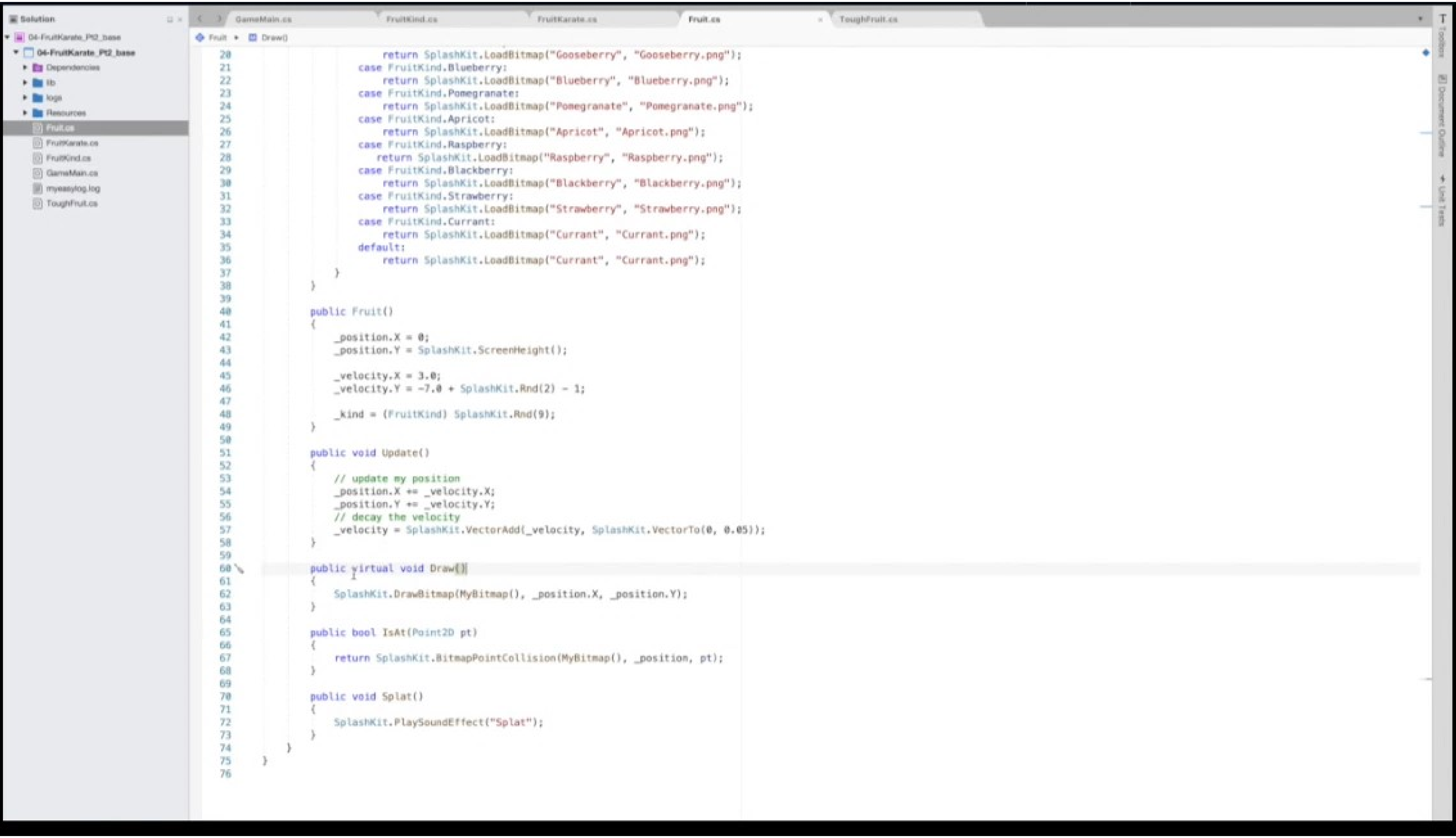
```
using System;
using System.Collections.Generic;
using SplashKitSDK;

namespace MyGame
{
    public class FruitKarate
    {
        private List<Fruit> _fruit = new List<Fruit>();

        public FruitKarate() { }

        public void LaunchFruit()
        {
            Fruit f = new Fruit();
            _fruit.Add(f);
        }

        public void Update()
        {
            foreach (Fruit f in _fruit)
            {
                f.Update();
            }
        }

        public void Draw()
        {
            foreach (Fruit f in _fruit)
            {
                f.Draw();
            }
        }

        public void PunchAt(Point2D pt)
        {
            List<Fruit> toRemove = new List<Fruit>();

            foreach (Fruit f in _fruit)
            {
                if (f.IsAt(pt))
                {
                    f.Splat();
                    toRemove.Add(f);
                }
            }

            foreach (Fruit f in toRemove)
            {
                _fruit.Remove(f);
            }
        }
    }
}
```



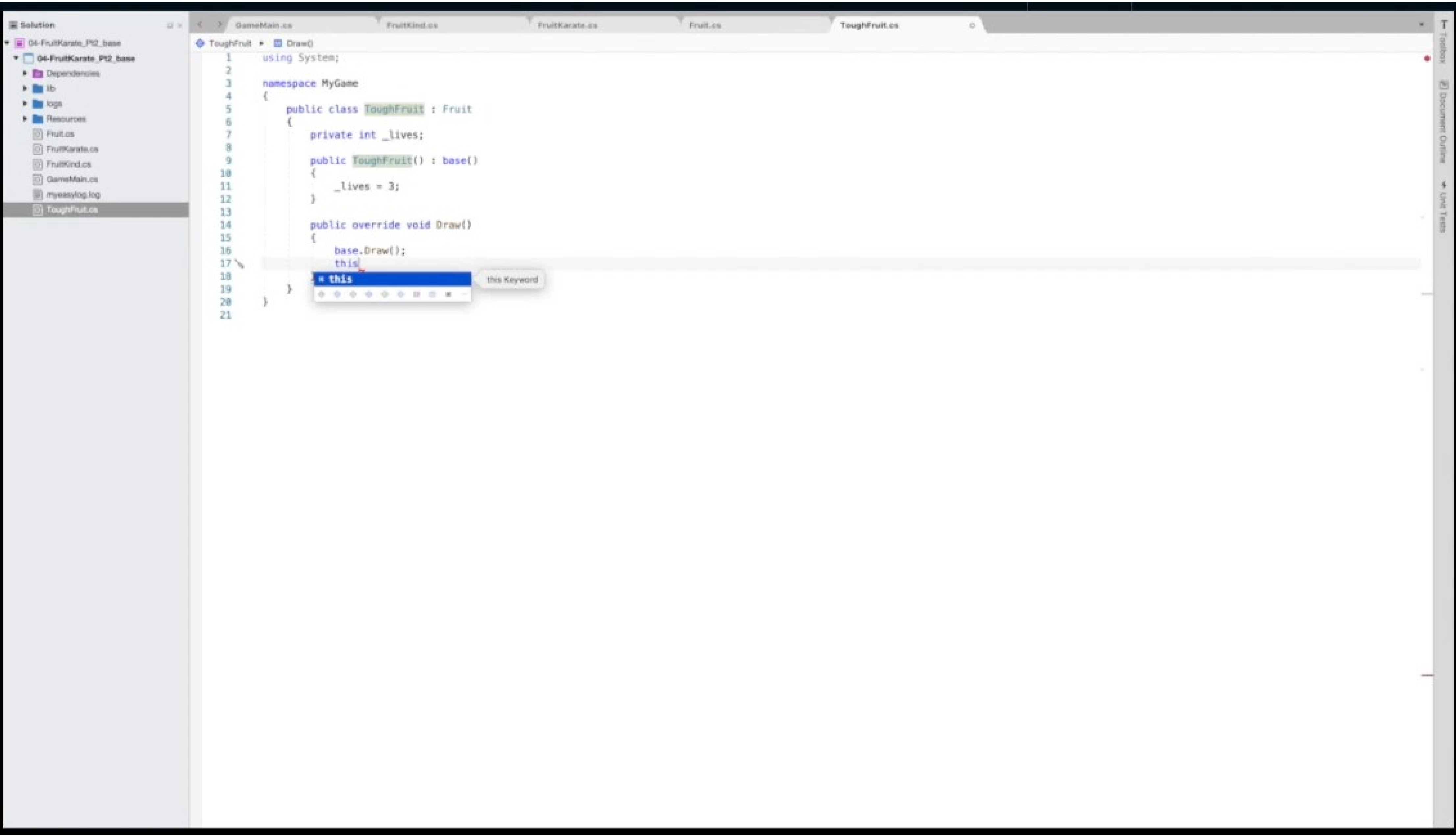
```
using System;

namespace MyGame
{
    public class ToughFruit
    {
        public ToughFruit()
        {
        }
    }
}
```



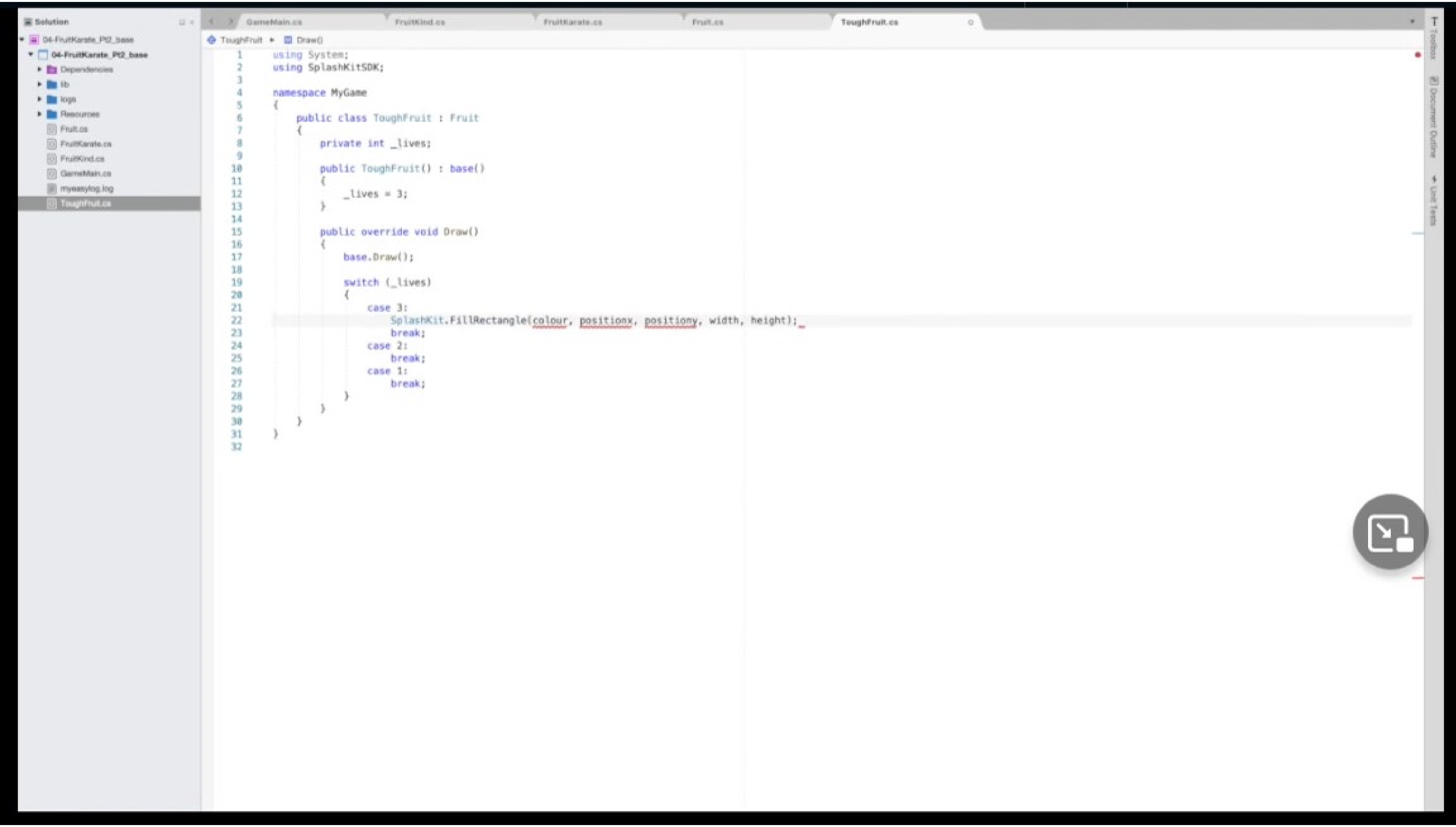
```
using System;

namespace MyGame
{
    public class ToughFruit : Fruit
    {
        private int _lives;

        public ToughFruit() : base()
        {
            _lives = 3;
        }
    }
}
```



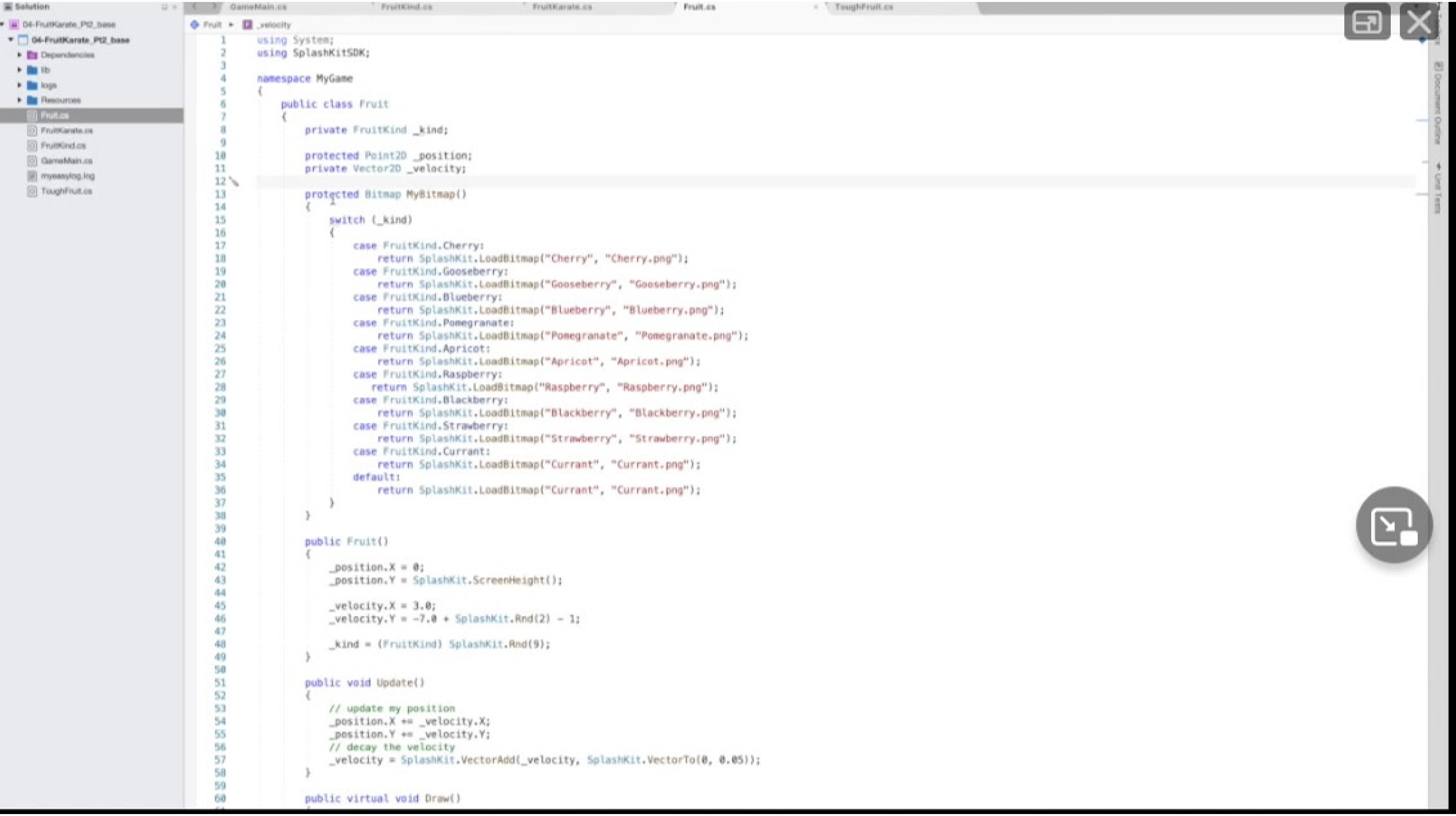
```
                    return SplashKit.LoadBitmap("Gooseberry", "Gooseberry.png");
                case FruitKind.Blueberry:
                    return SplashKit.LoadBitmap("Blueberry", "Blueberry.png");
                case FruitKind.Pomegranate:
                    return SplashKit.LoadBitmap("Pomegranate", "Pomegranate.png");
                case FruitKind.Apricot:
                    return SplashKit.LoadBitmap("Apricot", "Apricot.png");
                case FruitKind.Raspberry:
                   return SplashKit.LoadBitmap("Raspberry", "Raspberry.png");
                case FruitKind.Blackberry:
                    return SplashKit.LoadBitmap("Blackberry", "Blackberry.png");
                case FruitKind.Strawberry:
                    return SplashKit.LoadBitmap("Strawberry", "Strawberry.png");
                case FruitKind.Currant:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
                default:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
            }
        }

        public Fruit()
        {
            _position.X = 0;
            _position.Y = SplashKit.ScreenHeight();

            _velocity.X = 3.0;
            _velocity.Y = -7.0 + SplashKit.Rnd(2) - 1;

            _kind = (FruitKind) SplashKit.Rnd(9);
        }

        public void Update()
        {
            // update my position
            _position.X += _velocity.X;
            _position.Y += _velocity.Y;
            // decay the velocity
            _velocity = SplashKit.VectorAdd(_velocity, SplashKit.VectorTo(0, 0.05));
        }

        public virtual void Draw()
        {
            SplashKit.DrawBitmap(MyBitmap(), _position.X, _position.Y);
        }

        public bool IsAt(Point2D pt)
        {
            return SplashKit.BitmapPointCollision(MyBitmap(), _position, pt);
        }

        public void Splat()
        {
            SplashKit.PlaySoundEffect("Splat");
        }
    }
}
```



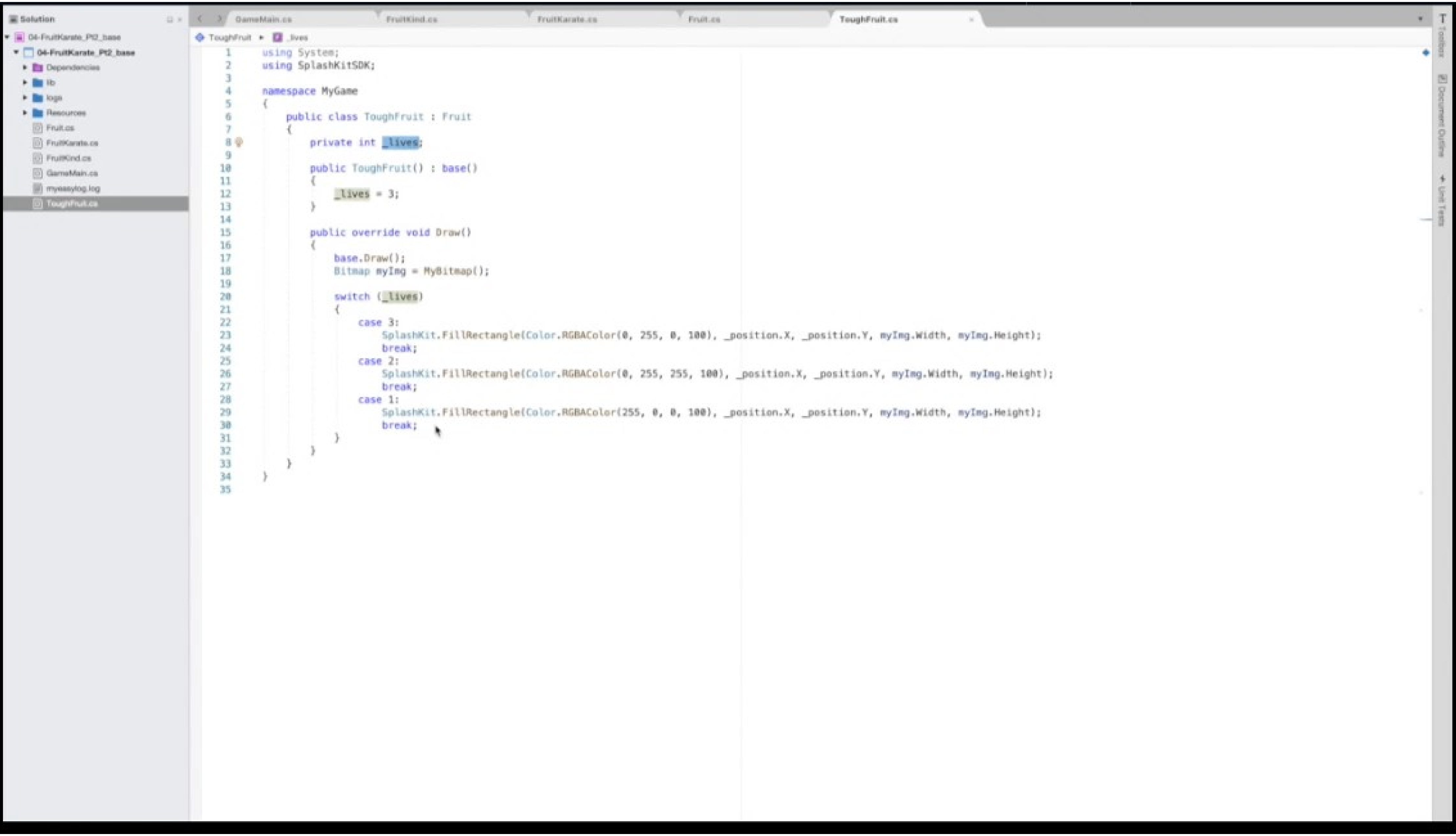
```
using System;

namespace MyGame
{
    public class ToughFruit : Fruit
    {
        private int _lives;

        public ToughFruit() : base()
        {
            _lives = 3;
        }

        public override void Draw()
        {
            base.Draw();
            this
        }
    }
}
```



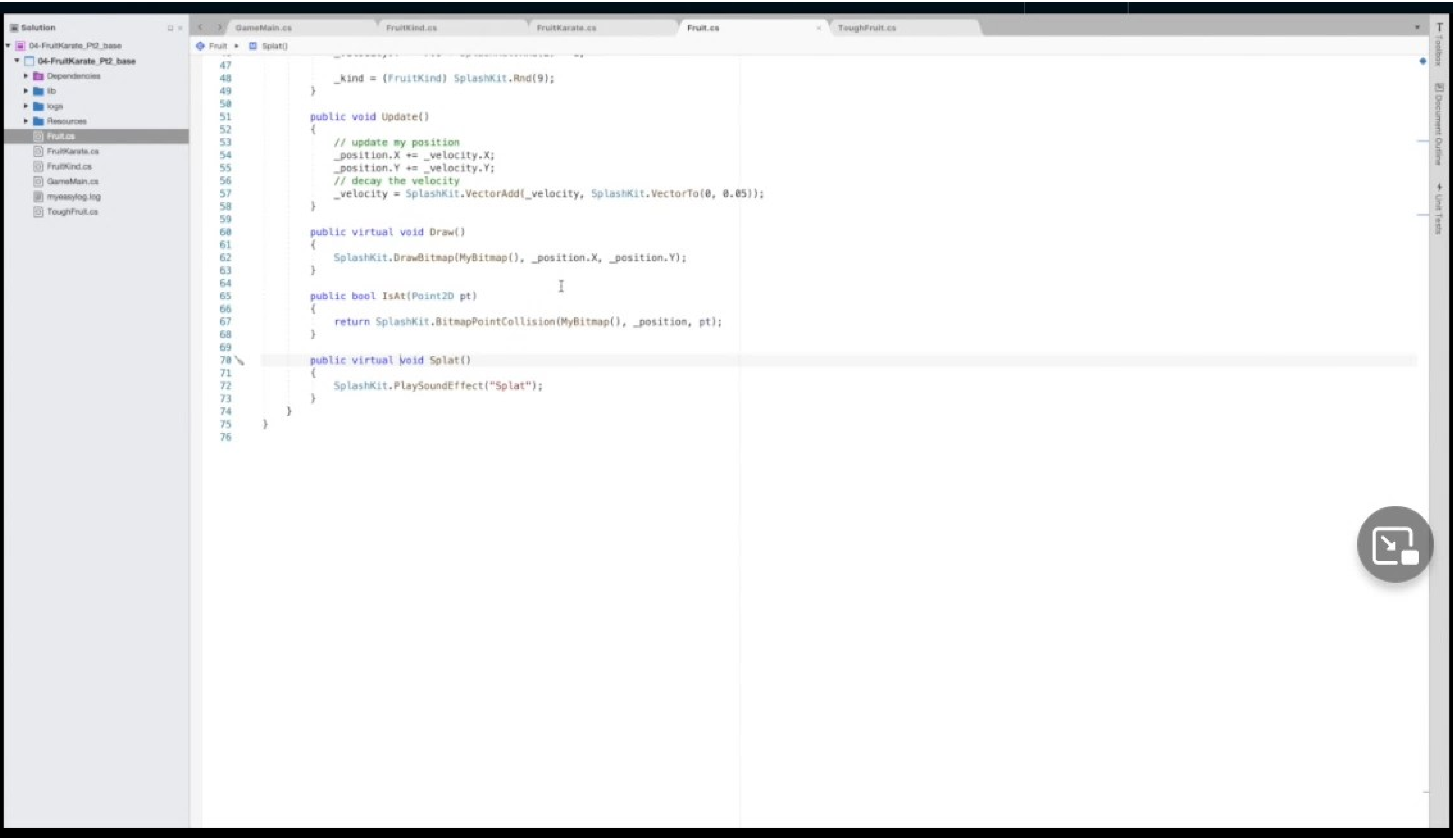
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class ToughFruit : Fruit
    {
        private int _lives;

        public ToughFruit() : base()
        {
            _lives = 3;
        }

        public override void Draw()
        {
            base.Draw();

            switch (_lives)
            {
                case 3:
                    SplashKit.FillRectangle(colour, positionx, positiony, width, height);_
                    break;
                case 2:
                    break;
                case 1:
                    break;
            }
        }
    }
}

```



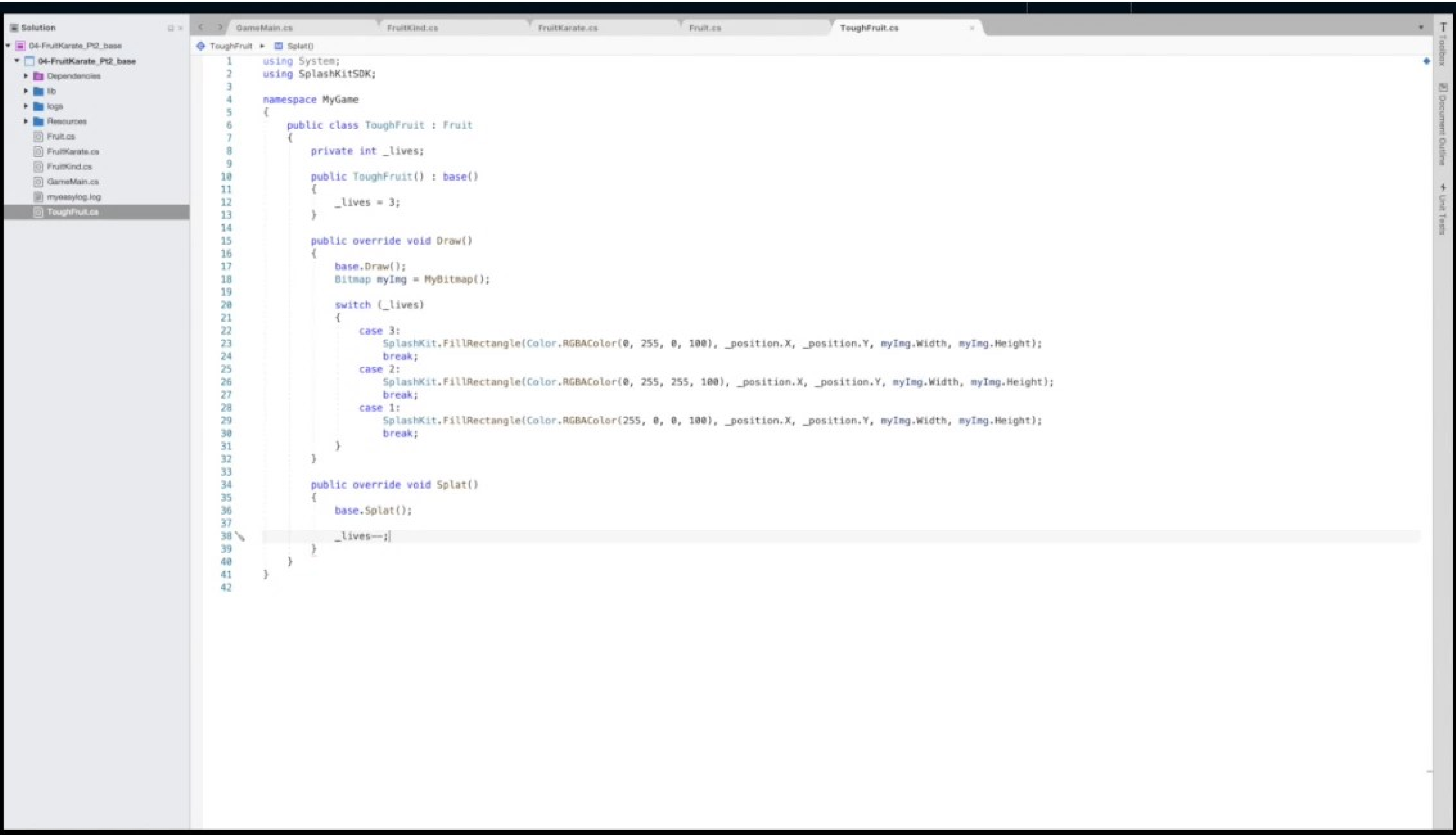
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class Fruit
    {
        private FruitKind _kind;

        protected Point2D _position;
        private Vector2D _velocity;

        protected Bitmap MyBitmap()
        {
            switch (_kind)
            {
                case FruitKind.Cherry:
                    return SplashKit.LoadBitmap("Cherry", "Cherry.png");
                case FruitKind.Gooseberry:
                    return SplashKit.LoadBitmap("Gooseberry", "Gooseberry.png");
                case FruitKind.Blueberry:
                    return SplashKit.LoadBitmap("Blueberry", "Blueberry.png");
                case FruitKind.Pomegranate:
                    return SplashKit.LoadBitmap("Pomegranate", "Pomegranate.png");
                case FruitKind.Apricot:
                    return SplashKit.LoadBitmap("Apricot", "Apricot.png");
                case FruitKind.Raspberry:
                   return SplashKit.LoadBitmap("Raspberry", "Raspberry.png");
                case FruitKind.Blackberry:
                    return SplashKit.LoadBitmap("Blackberry", "Blackberry.png");
                case FruitKind.Strawberry:
                    return SplashKit.LoadBitmap("Strawberry", "Strawberry.png");
                case FruitKind.Currant:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
                default:
                    return SplashKit.LoadBitmap("Currant", "Currant.png");
            }
        }

        public Fruit()
        {
            _position.X = 0;
            _position.Y = SplashKit.ScreenHeight();

            _velocity.X = 3.0;
            _velocity.Y = -7.0 + SplashKit.Rnd(2) - 1;

            _kind = (FruitKind) SplashKit.Rnd(9);
        }

        public void Update()
        {
            // update my position
            _position.X += _velocity.X;
            _position.Y += _velocity.Y;
            // decay the velocity
            _velocity = SplashKit.VectorAdd(_velocity, SplashKit.VectorTo(0, 0.05));
        }

        public virtual void Draw()
```



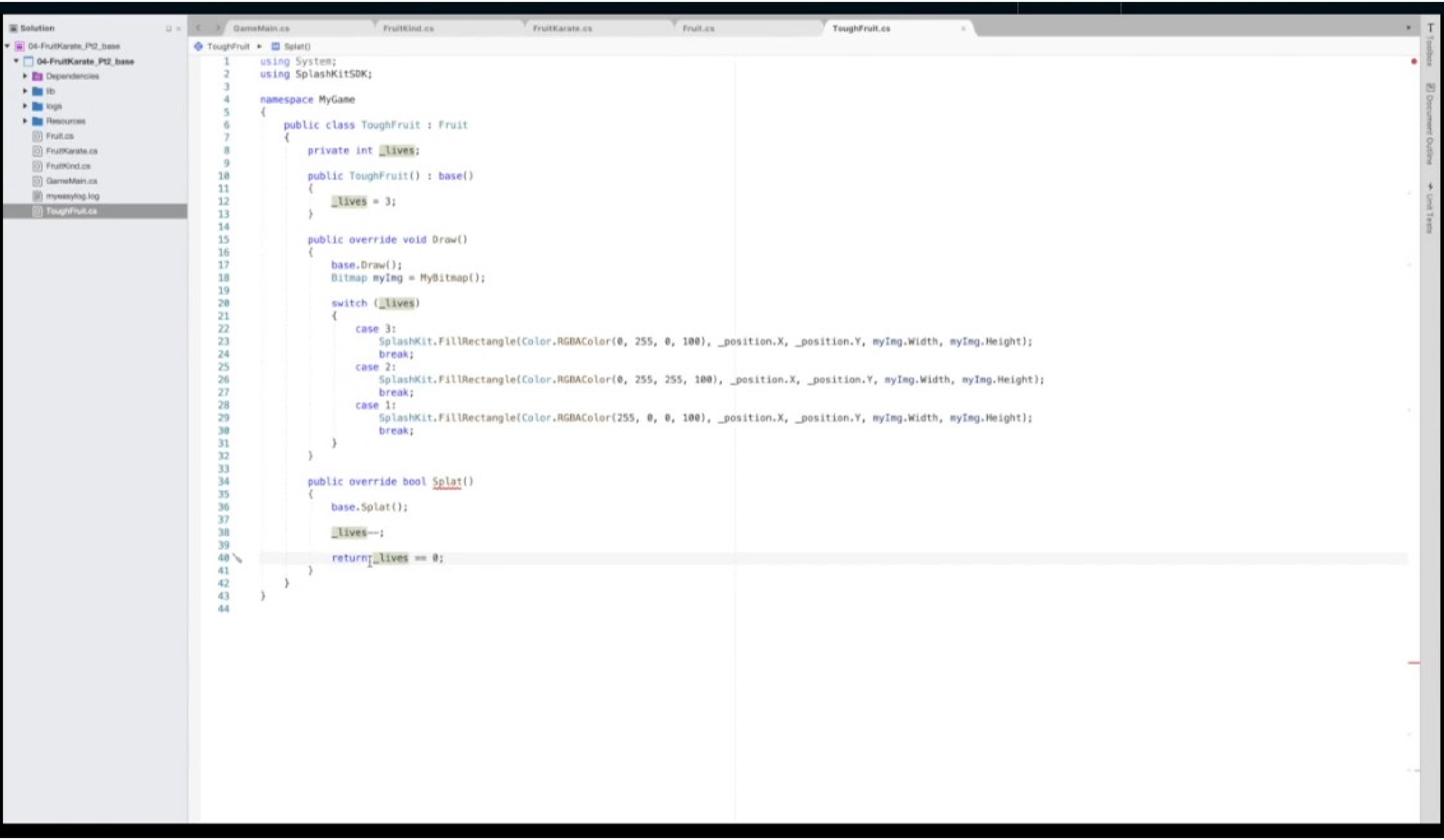
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class ToughFruit : Fruit
    {
        private int _lives;

        public ToughFruit() : base()
        {
            _lives = 3;
        }

        public override void Draw()
        {
            base.Draw();
            Bitmap myImg = MyBitmap();

            switch (_lives)
            {
                case 3:
                    SplashKit.FillRectangle(Color.RGBAColor(0, 255, 0, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
                case 2:
                    SplashKit.FillRectangle(Color.RGBAColor(0, 255, 255, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
                case 1:
                    SplashKit.FillRectangle(Color.RGBAColor(255, 0, 0, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
            }
        }
    }
}
```



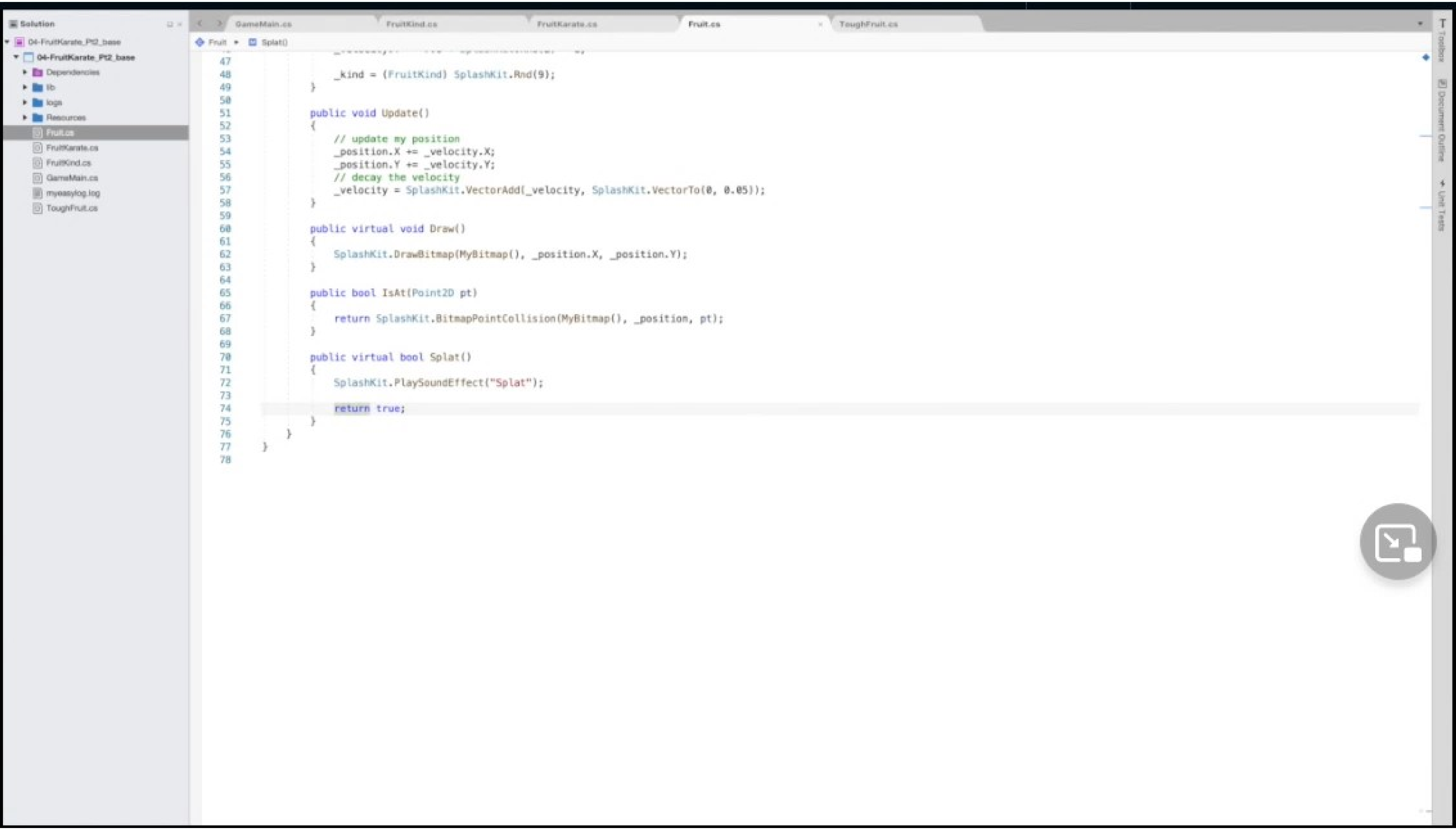
```

            _kind = (FruitKind) SplashKit.Rnd(9);
        }

        public void Update()
        {
            // update my position
            _position.X += _velocity.X;
            _position.Y += _velocity.Y;
            // decay the velocity
            _velocity = SplashKit.VectorAdd(_velocity, SplashKit.VectorTo(0, 0.05));
        }

        public virtual void Draw()
        {
            SplashKit.DrawBitmap(MyBitmap(), _position.X, _position.Y);
        }

        public bool IsAt(Point2D pt)
        {
            return SplashKit.BitmapPointCollision(MyBitmap(), _position, pt);
        }

        public virtual void Splat()
        {
            SplashKit.PlaySoundEffect("Splat");
        }
    }
}

```



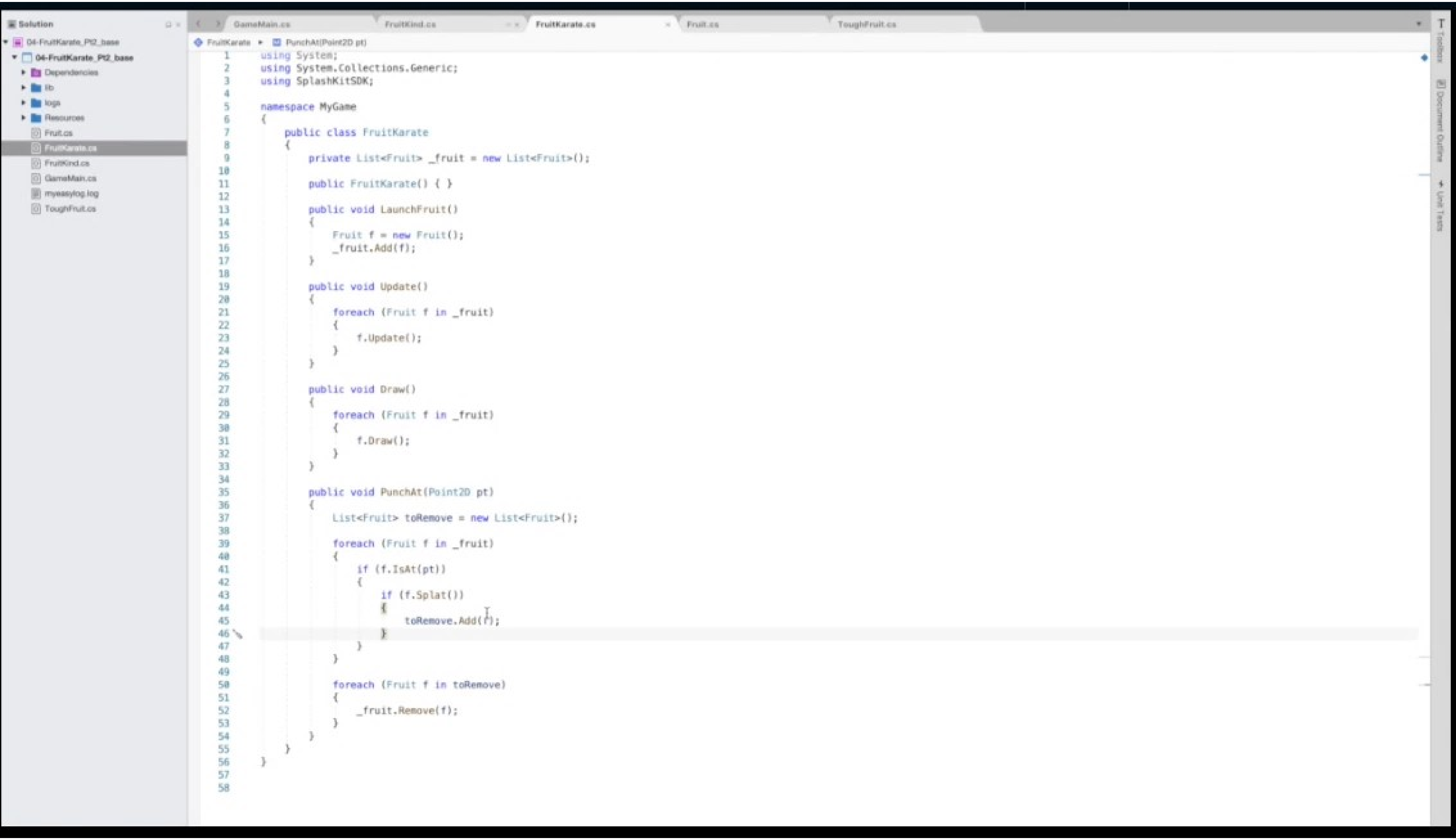
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class ToughFruit : Fruit
    {
        private int _lives;

        public ToughFruit() : base()
        {
            _lives = 3;
        }

        public override void Draw()
        {
            base.Draw();
            Bitmap myImg = MyBitmap();

            switch (_lives)
            {
                case 3:
                    SplashKit.FillRectangle(Color.RGBAColor(0, 255, 0, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
                case 2:
                    SplashKit.FillRectangle(Color.RGBAColor(0, 255, 255, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
                case 1:
                    SplashKit.FillRectangle(Color.RGBAColor(255, 0, 0, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
            }
        }

        public override void Splat()
        {
            base.Splat();

            _lives--;
        }
    }
}
```



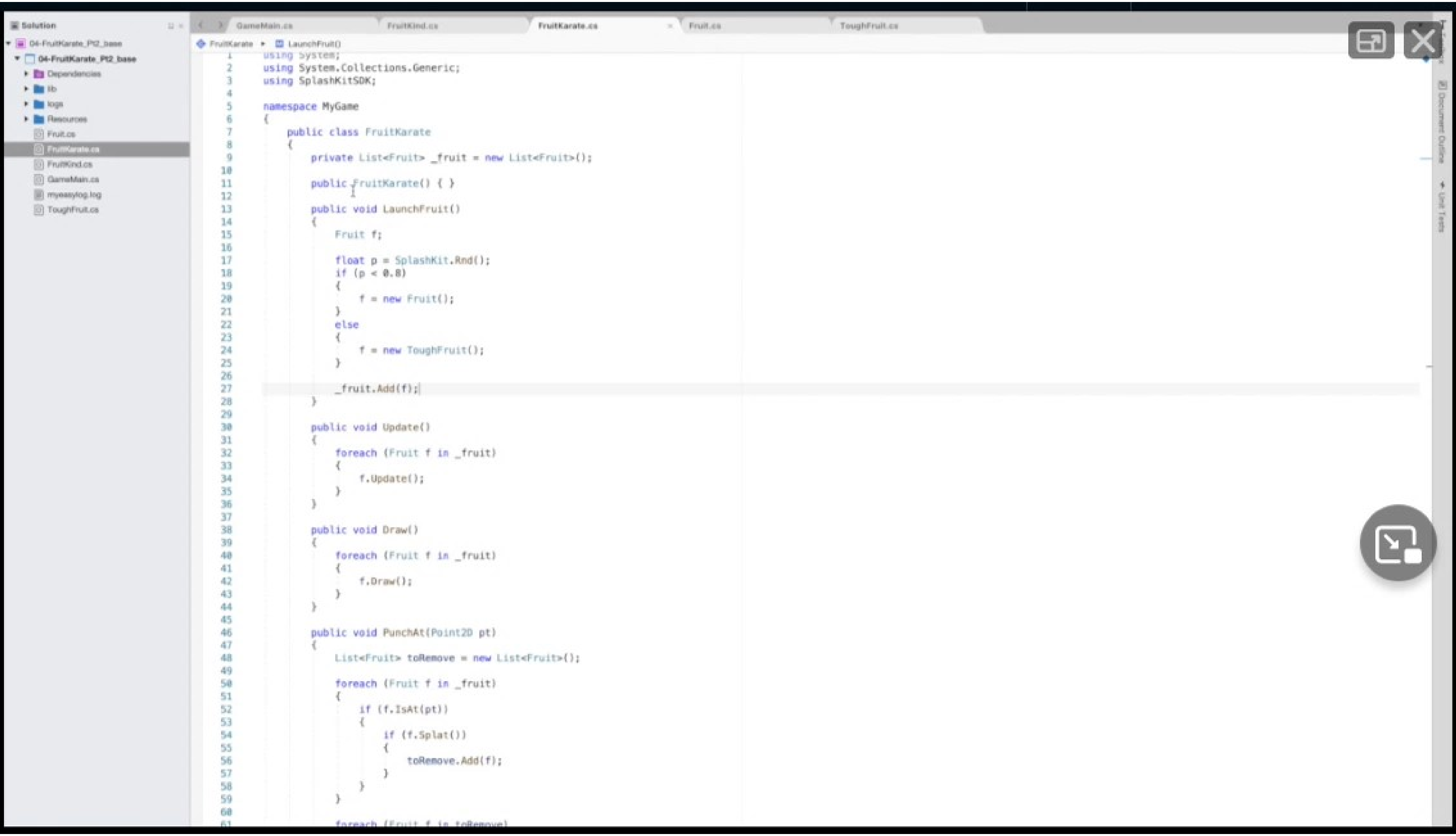
```
using System;
using SplashKitSDK;

namespace MyGame
{
    public class ToughFruit : Fruit
    {
        private int _lives;

        public ToughFruit() : base()
        {
            _lives = 3;
        }

        public override void Draw()
        {
            base.Draw();
            Bitmap myImg = MyBitmap();

            switch (_lives)
            {
                case 3:
                    SplashKit.FillRectangle(Color.RGBAColor(0, 255, 0, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
                case 2:
                    SplashKit.FillRectangle(Color.RGBAColor(0, 255, 255, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
                case 1:
                    SplashKit.FillRectangle(Color.RGBAColor(255, 0, 0, 100), _position.X, _position.Y, myImg.Width, myImg.Height);
                    break;
            }
        }

        public override bool Splat()
        {
            base.Splat();

            _lives--;

            return _lives == 0;
        }
    }
}
```



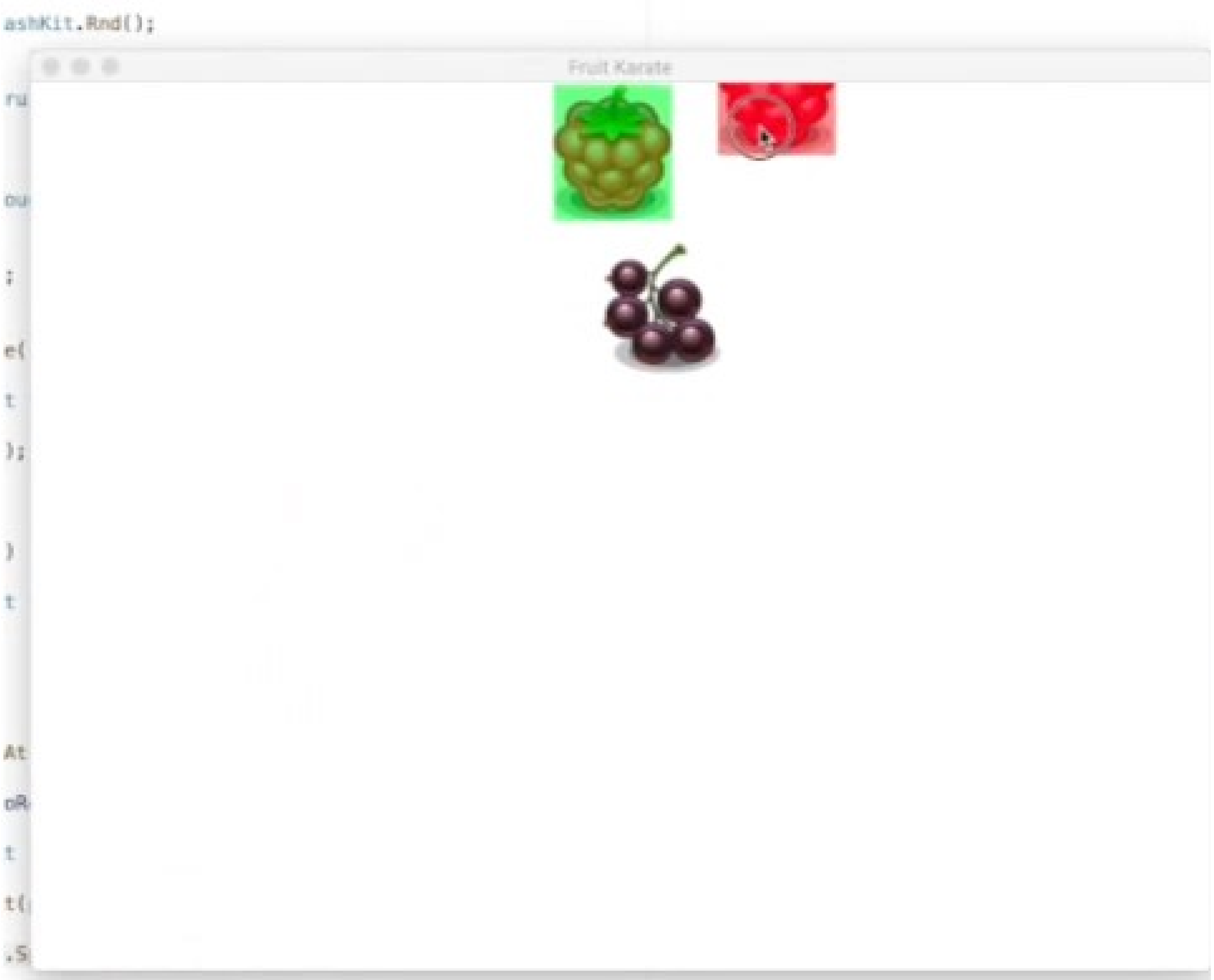
```

            _kind = (FruitKind) SplashKit.Rnd(9);
        }

        public void Update()
        {
            // update my position
            _position.X += _velocity.X;
            _position.Y += _velocity.Y;
            // decay the velocity
            _velocity = SplashKit.VectorAdd(_velocity, SplashKit.VectorTo(0, 0.05));
        }

        public virtual void Draw()
        {
            SplashKit.DrawBitmap(MyBitmap(), _position.X, _position.Y);
        }

        public bool IsAt(Point2D pt)
        {
            return SplashKit.BitmapPointCollision(MyBitmap(), _position, pt);
        }

        public virtual bool Splat()
        {
            SplashKit.PlaySoundEffect("Splat");

            return true;
        }
    }
}

```



```
using System;
using System.Collections.Generic;
using SplashKitSDK;

namespace MyGame
{
    public class FruitKarate
    {
        private List<Fruit> _fruit = new List<Fruit>();

        public FruitKarate() { }

        public void LaunchFruit()
        {
            Fruit f = new Fruit();
            _fruit.Add(f);
        }

        public void Update()
        {
            foreach (Fruit f in _fruit)
            {
                f.Update();
            }
        }

        public void Draw()
        {
            foreach (Fruit f in _fruit)
            {
                f.Draw();
            }
        }

        public void PunchAt(Point2D pt)
        {
            List<Fruit> toRemove = new List<Fruit>();

            foreach (Fruit f in _fruit)
            {
                if (f.IsAt(pt))
                {
                    if (f.Splat())
                    {
                        toRemove.Add(f);
                    }
                }
            }

            foreach (Fruit f in toRemove)
            {
                _fruit.Remove(f);
            }
        }
    }
}
```



```
using System;
using System.Collections.Generic;
using SplashKitSDK;

namespace MyGame
{
    public class FruitKarate
    {
        private List<Fruit> _fruit = new List<Fruit>();

        public FruitKarate() { }

        public void LaunchFruit()
        {
            Fruit f;

            float p = SplashKit.Rnd();
            if (p < 0.8)
            {
                f = new Fruit();
            }
            else
            {
                f = new ToughFruit();
            }

            _fruit.Add(f);
        }

        public void Update()
        {
            foreach (Fruit f in _fruit)
            {
                f.Update();
            }
        }

        public void Draw()
        {
            foreach (Fruit f in _fruit)
            {
                f.Draw();
            }
        }

        public void PunchAt(Point2D pt)
        {
            List<Fruit> toRemove = new List<Fruit>();

            foreach (Fruit f in _fruit)
            {
                if (f.IsAt(pt))
                {
                    if (f.Splat())
                    {
                        toRemove.Add(f);
                    }
                }
            }

            foreach (Fruit f in toRemove)
```



```
ashKit.Rnd();

ru

ou

;

e(
t
);

}
t

At
oR
t
t(
.S
```