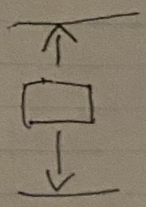
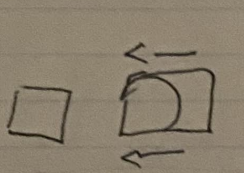
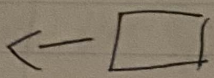


Wall-Box
if hit reduce health by 1 or
kill player
Bounce off & reverse control?



Box-object
if hit bounce object back?
& reduce health (maybe reset box)



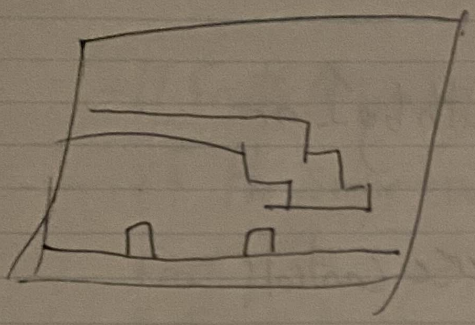
object
Move forward
Spawn random Y
Reset when edge hit

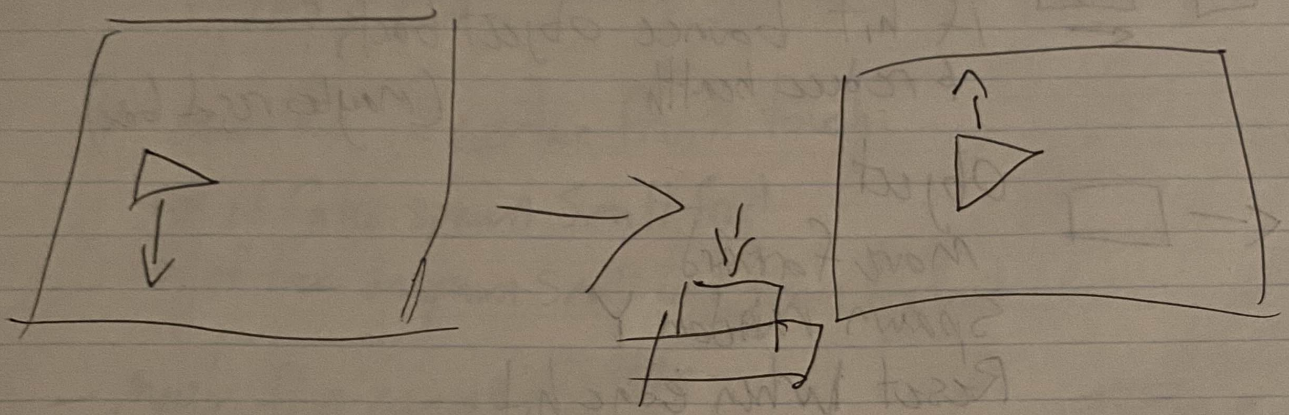
---

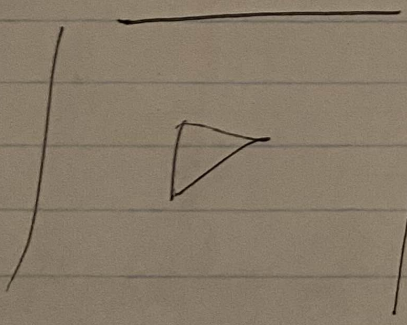
still to do:

- [x] Plug in player, Score, audio
- [ ] Add collision to main
  - [ ] game

-> player has still not been provided. Player/obstac
collision cannot be added without it. This isn't on
me. Game over screen was also not provided.







- Character ✓
- Button ✓
- Load images
- Collision (character / object / wall)
- Audio ✓
- Objects
- Movement
- Score & Game over ✓

```
// The namespace your code is in.
namespace Game10003
{
    /// <summary>
    ///     Your game code goes inside this class!
    /// </summary>
    2 references
    public class Game
    {
        // Place your variables here:
        Obstacle obstacle;
        PlayerCollision playerCollision;


        /// <summary>
        ///     Setup runs once before the game loop begins.
        /// </summary>
        1 reference
        public void Setup()
        {
            Window.SetSize(600, 600);
            Window.SetTitle("test");
            //adds new obstacle class (could turn into an array for extra difficulty?)
            obstacle = new Obstacle();
            playerCollision = new PlayerCollision();
        }

        /// <summary>
        ///     Update runs every frame.
        /// </summary>
        1 reference
        public void Update()
        {
            Window.ClearBackground(Color.White);
            //draws obstacle, adds wall collision, and updates position
            obstacle.DrawObstacle();
            obstacle.ObstacleWallCollision();
            obstacle.UpdatePosition();


        }
    }
}

```

```
2 references
public class PlayerCollision
{
    0 references
    public void Wall()
    {
        //Creates wall collision, commented out currently because i do not have a player class yet
        //Variable names may be incorrect, will plug in player class when created
        // float playerLeftEdge = playerPosition.X;
        // float playerRightEdge = playerPosition.X + playerSize.X;
        // float playerTopEdge = playerPosition.Y;
        // float playerBottomEdge = playerPosition.Y + playerSize.Y;

        //Defines window parameters, right and left can be defined if needed, but are unneccessary for right now
        // bool topOfWindow = playerTopEdge <= 0;
        // bool bottomOfWindow = playerBottomEdge >= Window.Height;

        //if (topOfWindow || bottomOfWindow)
        //{
        //Insert definitions for what happens here (most likely --health or kill screen, that's not on me though)
        //}
    }
}
```

```
{
    Vector2 obstaclePosition = new Vector2 (600, 300);
    Vector2 obstacleSize;
    float speed = 300;

    1 reference
    public Obstacle()
    {
        //obstaclePosition.Y = 300;
        obstacleSize = new Vector2(100, 50);
        obstaclePosition.Y = Random.Float(0, 600);
    }

    1 reference
    public void UpdatePosition()
    {
        //updates position
       obstaclePosition.X -= Time.DeltaTime * speed;
    }
    1 reference
    public void DrawObstacle()
    {
        //draws obstacle
        Draw.FillColor = Color.Red;
        Draw.Rectangle(obstaclePosition, obstacleSize);
    }
    //obstacle collision, add player class in there \/\/\/
    0 references
    public void ObstaclePlayerCollision()
    {
        //Import player later
       // float playerLeftEdge = playerPosition.X;
       // float playerRightEdge = playerPosition.X + playerSize.X;
       // float playerTopEdge = playerPosition.Y;
       // float playerBottomEdge = playerPosition.Y + playerSize.Y;

        float obstacleLeftEdge = obstaclePosition.X;
        float obstacleRightEdge = obstaclePosition.X + obstacleSize.X;
        float obstacleTopEdge = obstaclePosition.Y;
        float obstacleBottomEdge = obstaclePosition.Y + obstacleSize.Y;

       // bool doesOverlapLeft = playerLeftEdge < obstacleRightEdge;
       // bool doesOverlapRight = playerRightEdge < obstacleLeftEdge;
       // bool doesOverlapTop = playerTopEdge < obstacleBottomEdge;
       // bool doesOverlapBottom = playerBottomEdge > obstacleTopEdge;

       // bool doesOverlap = doesOverlapLeft && doesOverlapRight && doesOverlapTop && doesOverlapBottom;

       // if (doesOverlap = true)
        {
            //insert --score or kill screen here, not my job though so idk
            Console.WriteLine("ping");
        }
    }

    1 reference
    public void ObstacleWallCollision()
    {
        //resets block once it hits the wall (could potentially track score using this?)
        float obstacleLeftEdge = obstaclePosition.X;

        bool leftOfWindow = obstacleLeftEdge <= 0 - obstacleSize.X;

        if (leftOfWindow == true)
        {
            //resets to a random position once it passes the wall
            obstaclePosition.X = Random.Float(1000, 1500);
            obstaclePosition.Y = Random.Float(0, 600);
        }
    }
```