

Process & Decision Documentation

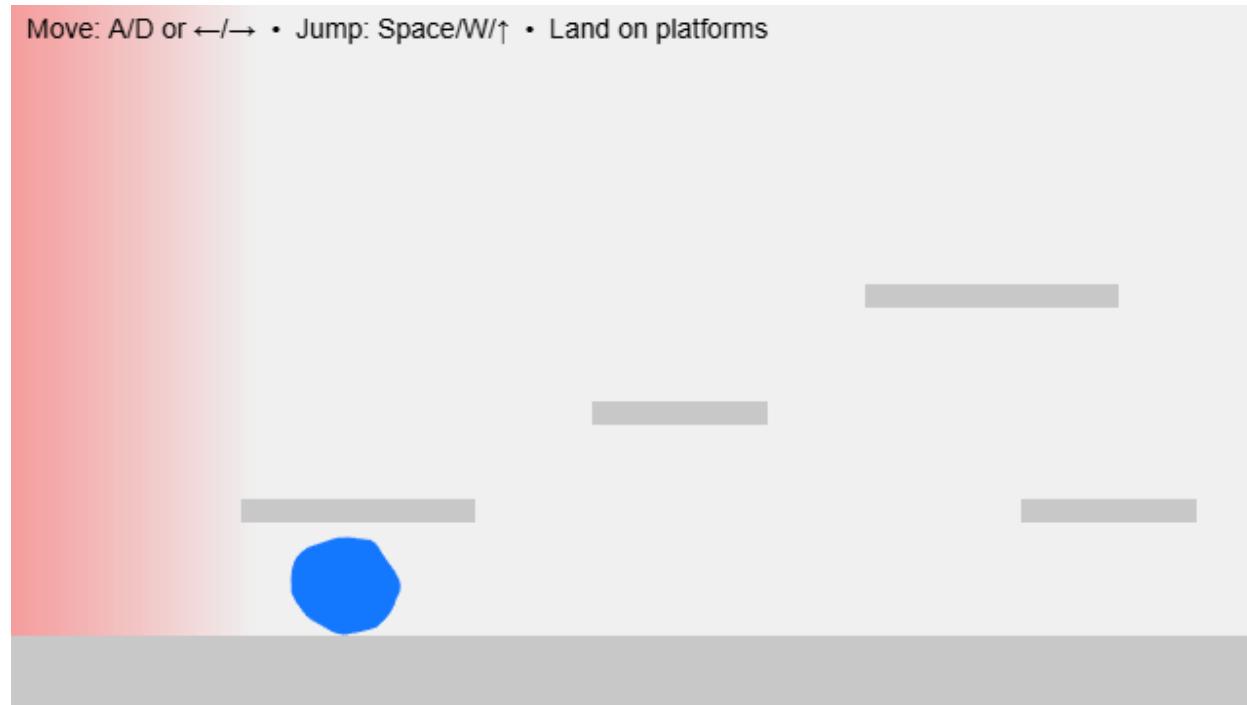
Project/Assignment Decisions

My plan is to elicit the feeling of “panic” or “stress” by redesigning the blob’s movement to have it slide in the held direction until it collides with an object to stop its path. This idea is mostly informed by the bonus of the side quest (add a “mischief” mechanic), where although the blob will not have gravity, the objects it collides with do, some breaking and others falling to the ground. The effect this will have on the work is that I will have to remove the gravity of the blob, perfectly enclose its environment, and create new child objects of the ‘platform’ object.

Due to time constraints I am deciding on limiting the new objects to only breakable glass blocks.

Role-Based Process Evidence

First draft of red “warning lights” to give a sense of danger to avoid:



```
// Draws the background gradient to the left of the screen
function backgroundGradient() {
  const endX = width / 5;

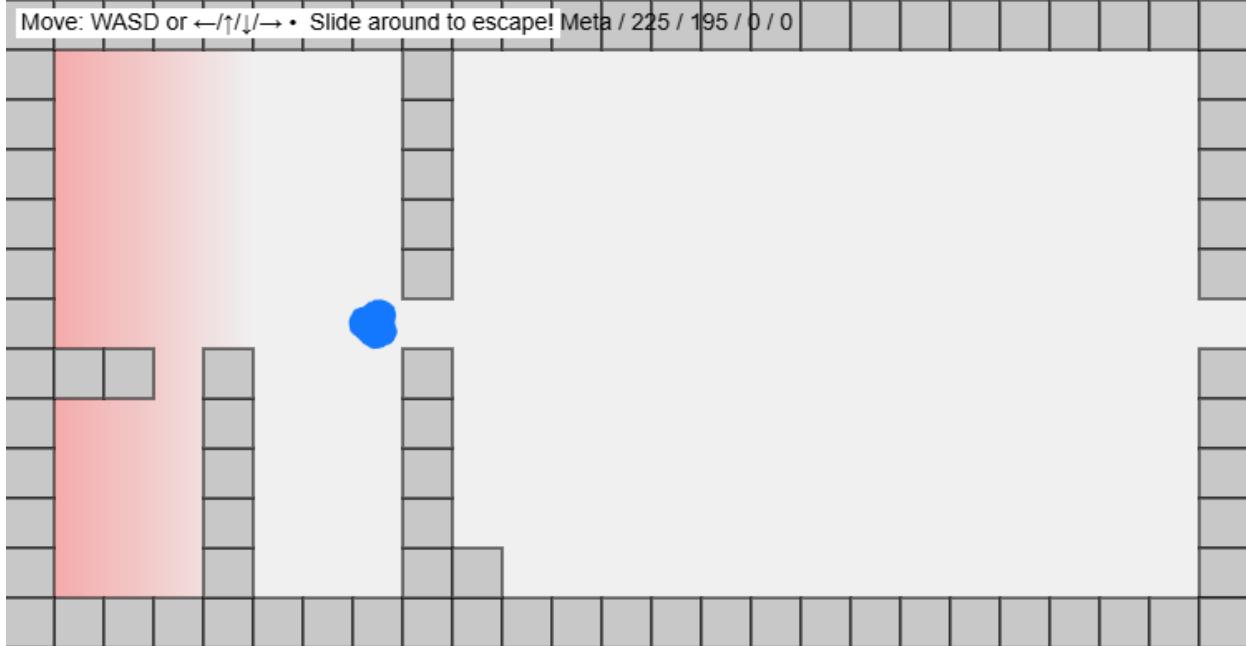
  for (let x = 0; x < endX; x++) {
```

```

        stroke(255, 0, 0, map(x, 0, endX, 100, 0));
        line(x, 0, x, height);
    }
}

```

Draft of ‘glass’ block (hole created by slamming into it) without visuals:



```

for (const s of platforms) {
    if (s.glass & !s.onGround) {
        if (s.y >= floorY3 - gridSize) {
            s.y = floorY3 - gridSize;
            s.onGround = true;
            s.broken = true;
            continue;
        }

        s.y += blob3.gravity;
    }
}

```

Additional Evidence

```

blob3.vx += (blob3.accel * moveX) ** 3;
blob3.vy += (blob3.accel * moveY) ** 3;

// There's a math reason for this I swear

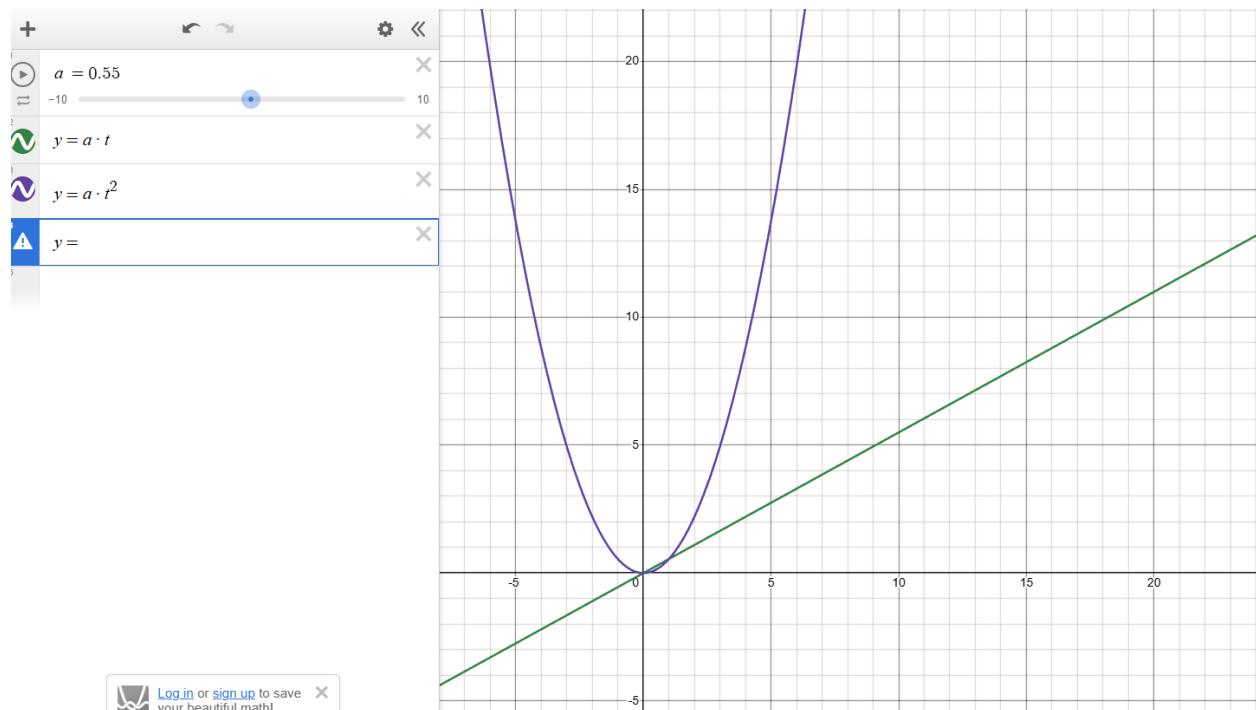
```

```
// blob3.vx **= 3;

// --- Apply friction and clamp speed ---
blob3.vx *= blob3.onGround ? blob3.frictionGround : blob3.frictionAir;
blob3.vx = constrain(blob3.vx, -blob3.maxRun, blob3.maxRun);
blob3.vy *= blob3.onGround ? blob3.frictionGround : blob3.frictionAir;
blob3.vy = constrain(blob3.vy, -blob3.maxRun, blob3.maxRun);
```

I changed the way acceleration was calculated because I realized that when I tried graphing it in Desmos, it showed that it was actually a square root function because move was constrained to 1. By removing that limit, it was able to be a linear function and actually behave normally which allowed me to be able to work with the maxRun now that it actually gets used. Otherwise, the acceleration would hit a horizontal asymptote at around 4.3333 and would limit itself on its own without maxRun.

I have tried to recreate how I have graphed it, but I genuinely am unable to. I came into this without knowing how acceleration actually worked, saw that this was not working, learned how acceleration worked through graphing and then fixed it before actually reading online what it was.



```
// --- Panic form when moving ---
if (round(blob3.vx) !== 0 || round(blob3.vy) !== 0) {
    blob3.points = blob3.panicPoints;
    blob3.wobble = blob3.panicWobble;
```

```
    blob3.t += blob3.tSpeed * 10;
} else {
    blob3.points = blob3.basePoints;
    blob3.wobble = blob3.baseWobble;
}
```

Since the blob's movement is defined by a sinusoidal function and the t variable is controlling the time in the function, I realized that I can just speed it up by 10x to make the blob's animation more erratic/faster. Couple this with the increased points, and it results in a quick and spiky animation.

```
for (const s of platforms) {
    if (s.glass & !s.onGround) {
        if (s.y >= floorY3 - gridSize) {
            s.y = floorY3 - gridSize;
            s.onGround = true;
            s.broken = true;
            continue;
        }

        s.y += blob3.gravity;
    }
}
```

To get the glass blocks to work I repurposed the now obsolete gravity variable and applied it to the glass blocks instead. To get it to remain still when needed I made it, so it only begins to apply gravity once it has collided with the blob once. A side effect of this is that had I made a glass block be able to be pushed from below or above, the glass would still move downwards regardless of impact direction. To get around this I just never made it possible and had the blob only interact with them from the sides.

GenAI Documentation

No GenAI used for this task.