Bacteria Project One

COMP-4471, Winter 2021

https://github.com/Soupy1184/COMP-4471-Project1
Christopher Campbell and Ben Puhalski

Table of Contents

Description

This is a JavaScript program that implements a bacteria game using WebGL.

Contains the following mechanics:

- Bacteria spawn around disk randomly
- Bacteria grow uniformly out from it's origin
- Bacteria consume other bacteria and grow relative to the consumed bacteria relative to the bacteria's origin
- Eliminate bacteria by clicking them to administer 'poison'
- Score functionality
- Adds score depending on how large the bacteria eliminated was
- Subtracts score depending on how large the bacteria consumed was (divided by 2)
- Restart button (refreshes page)
- Lose conditions
- One bacteria is too large (180 degrees)
- There are too many bacteria (> 5)
- Random bacteria colour

Bacteria Rendering

Logically the bacteria uses **2 circles** for the **edges** of them and uses the **TRIANGLE_STRIP** from WebGL to render in between and connect the 2 edges. Every time render is called the bacteria will grow relative to how much time has passed since the last frame using the Date.now() function. Every time this function is called 2 vertices for each side of the bacteria (4 total) are stored in the bacteria object to then later be rendered. The edge circles use TRIANGLE_FAN to display.

Detecting

```
    isWithin(angleNum) {
        if(angleNum >= this.minAngle && angleNum <= this.maxAngle) {
            return true;
        } else if(this.maxAngle < this.minAngle && (angleNum >= this.minAngle ||
 angleNum
                                                    <= this.maxAngle)) {
            return true;
        } else {
            return false;
        }
    }
```

The above function is used when detection if a mouse click will intersect, if two bacteria are colliding with each other and whether a space if available to be spawned when a bacteria is created.

The function uses the angles that represent how big and where a bacteria is on the circle.

Consuming

When a bacteria is consumed by another bacteria first it is found which is larger when colliding, then a bacteria grows in size relative to how large the consumed bacteria was relative to the origin of that bacteria. Lastly, the consumed bacteria is removed from the rendered bacteria array.
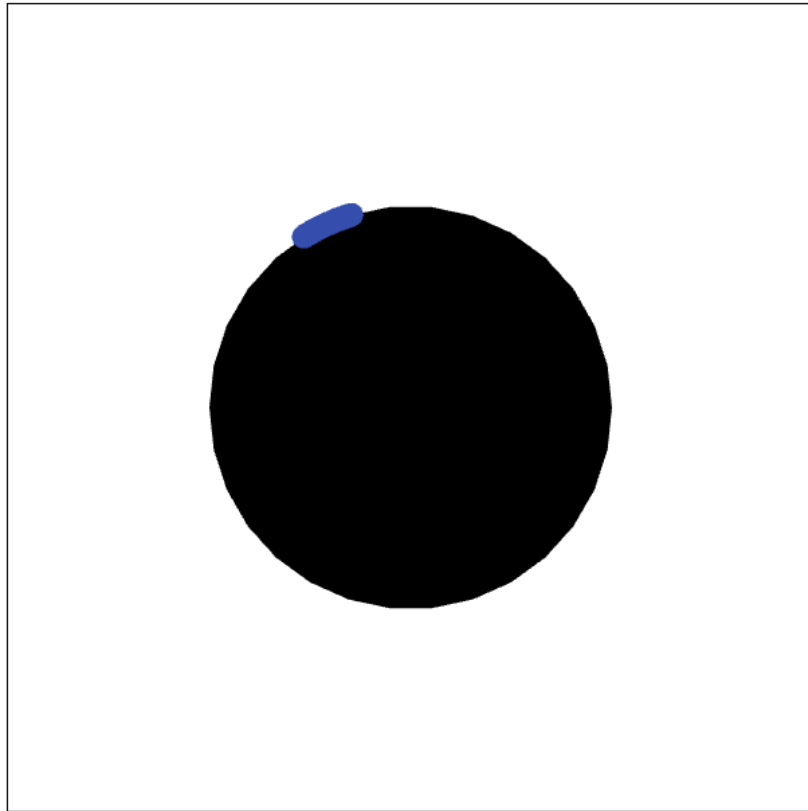
```
    growTo(size) {
        var targetSize = this.getSize() + size;
        console.log("target: " + targetSize + "\tthis.getSize() " +
 this.getSize() + "\tsize: " + size);
        while(this.getSize() < targetSize) {
            this.growthFunction(0.008);
        }
    }
```
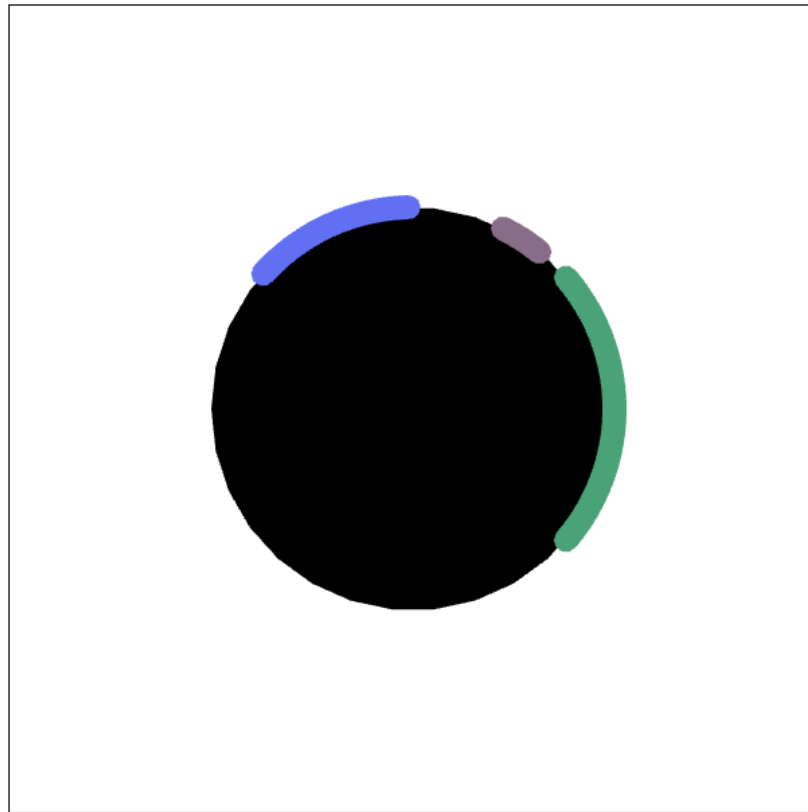
Screenshots

First Display

Score: 0

When the program is launched a score counter and canvas will be displayed. The canvas is outlined using css. The disk will be rendered using a TRIANGLE_FAN method using 32 vertices originating from 0, 0. The program waits for 4 seconds to spawn the first bacteria at a random point and spawns with a random colour and instantly begins growing.
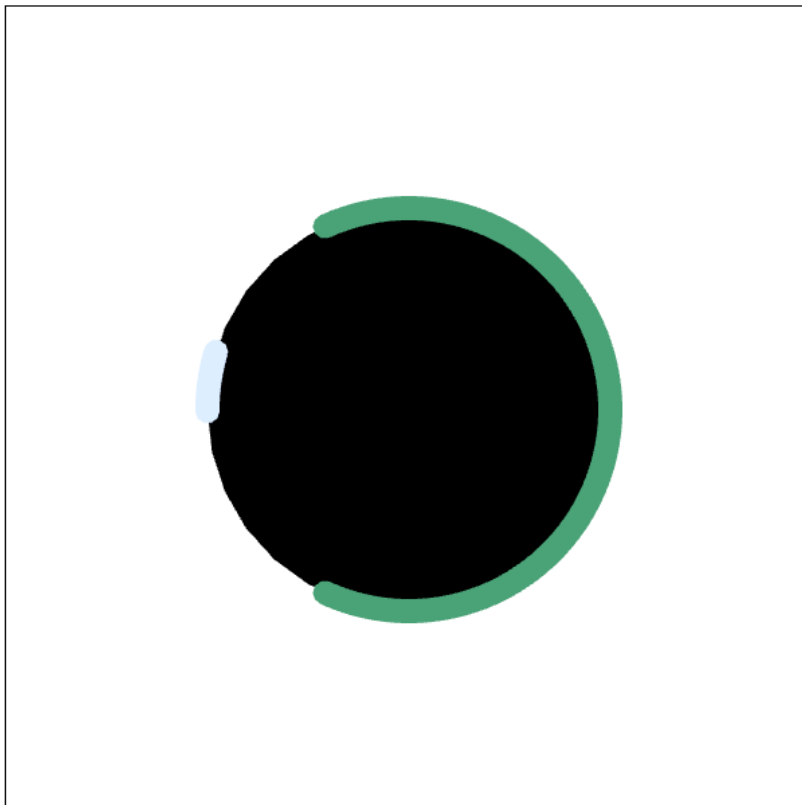
Bacteria Consuming

Score: 0

This screenshot shows after sometime 3 bacteria have spawned and the green bacteria on the right is about to consume the one above it and soon after that will consume the one above that.

**Now that's one huge bacteria! They're beyond your control Final Score: -10**

Restart

After this happens the bacteria grows in size exactly by how large the other 2 were and how much time passed while consuming them. This also shows one of the lose states which is when a bacteria is larger than 180 degrees. Also, the score is subtracted from every time a bacteria consumes another bacteria which is why it is negative. This also triggers the restart button to appear.
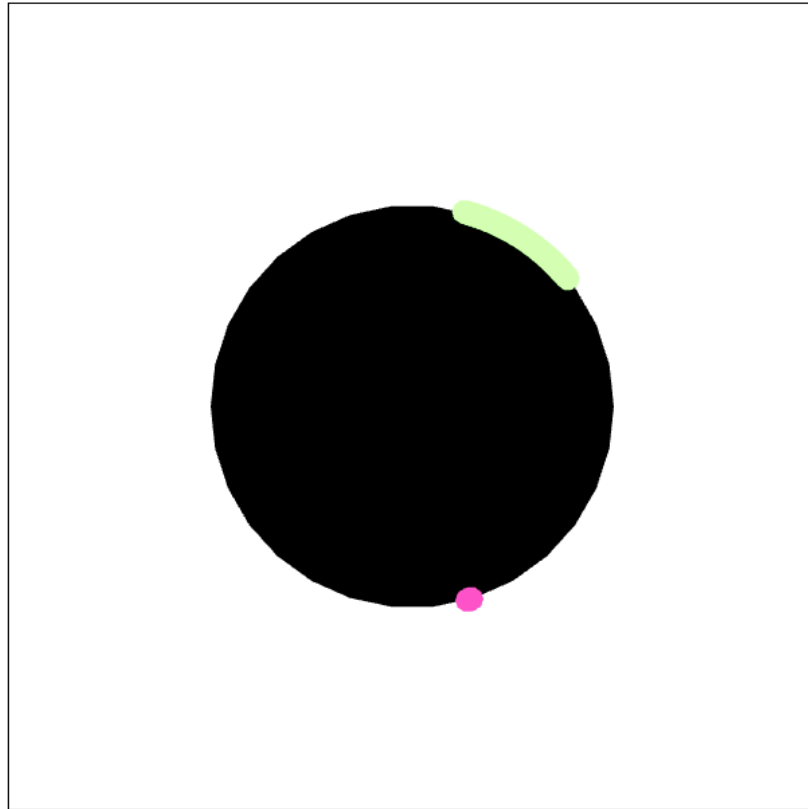
Eliminating Bacteria

Score: 0
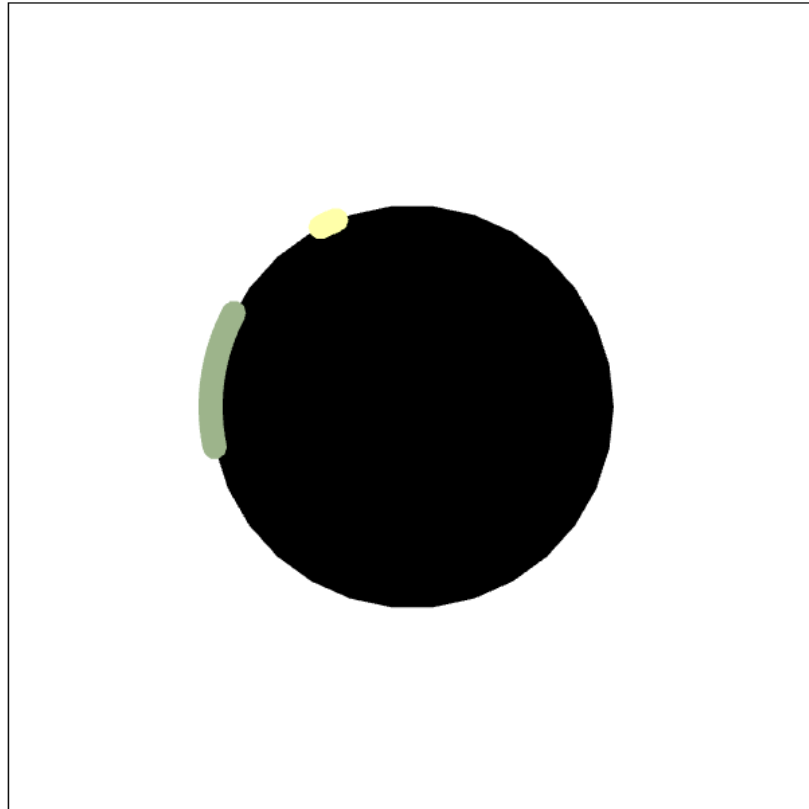
The player is about to click on the left bacteria right after it consumes the one above it.

Score: 19

After the player clicks on that bacteria and administers the poison to remove it, it disappears and a 19 score is added to the player's total score because it was so large. A smaller amount of score would be added if the player clicked on the pink bacteria at the bottom (most likely 1 or 2).