Software Engineering Immersive Course

Project 01 - GAME - The Very Hungry Caterpillar

Brief

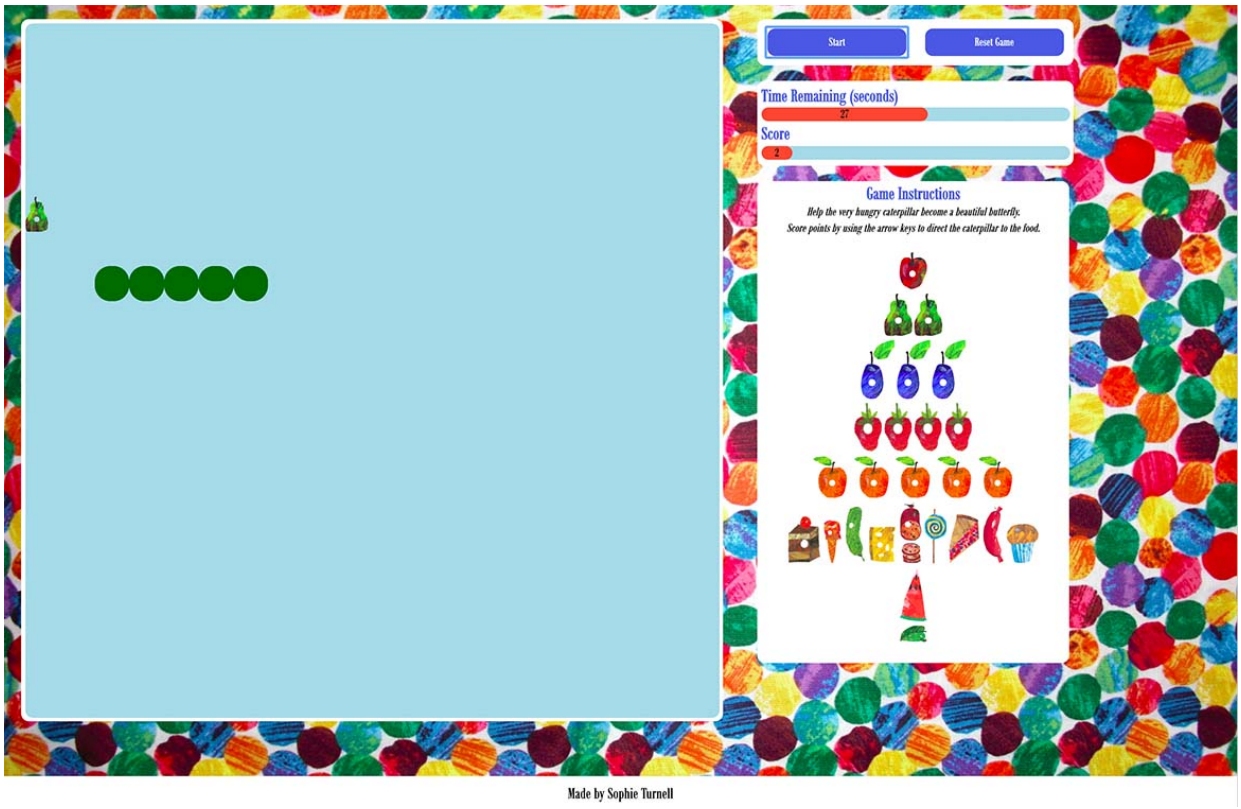
Design a grid-based game using HTML, CSS, and JavaScript technologies learnt in the first three weeks of the course. Players must be able to win and lose.

Timeframe: 1 week, independent coding

Summary

My first coding project was built with vanilla JavaScript using Flexbox and CSS3 to apply styling. Based on the classic Nokia 3210 game Snake, I substituted the snake for Eric Care's The Very Hungry Caterpillar. The Caterpillar must eat the foods on the right of the screen in order to win and turn into a butterfly.

I used DOM manipulation to enable players to guide the caterpillar to eat the array of foods that appear on the game grid at random. Each bite increases the player's score, and the length and speed of the caterpillar. The game is over if the caterpillar hits the edge of the board.



Deployment

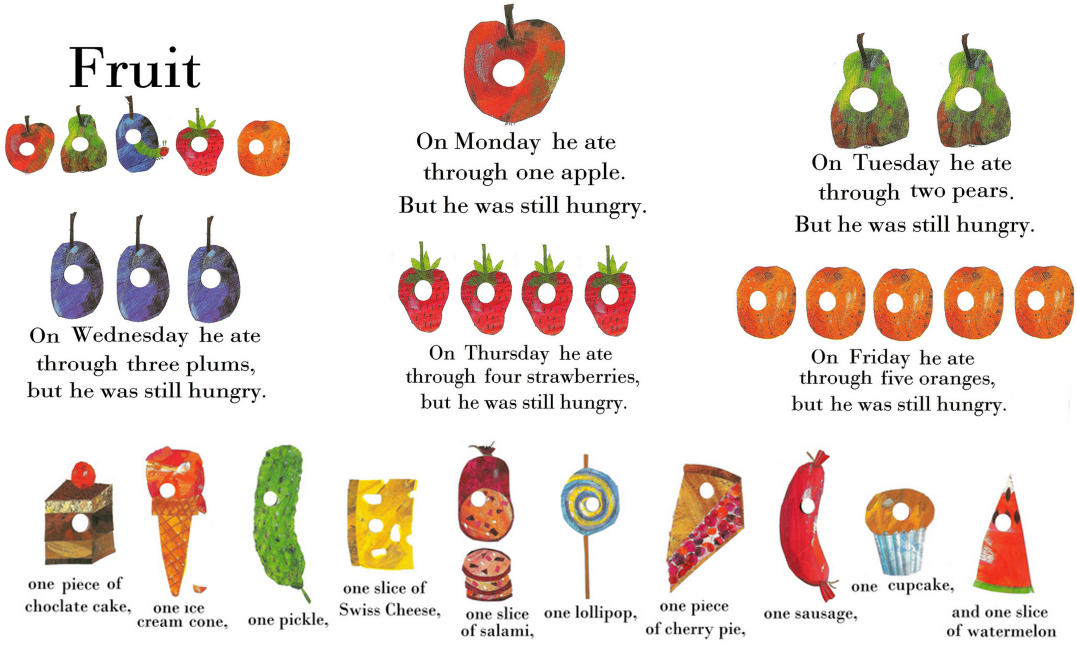
View code on GitHub here: github.com/sophieturnell/project-01.
Use the "clone or download" button and open the index.html file in your browser.

Technologies & Methodologies Used

- HTML 5
- CSS 3
- Vanila JavaScript (ES6)
- Google Fonts
- Flexbox layout module
- GitHub
- Git
- Adobe Photoshop
- Adobe Illustrator

Features

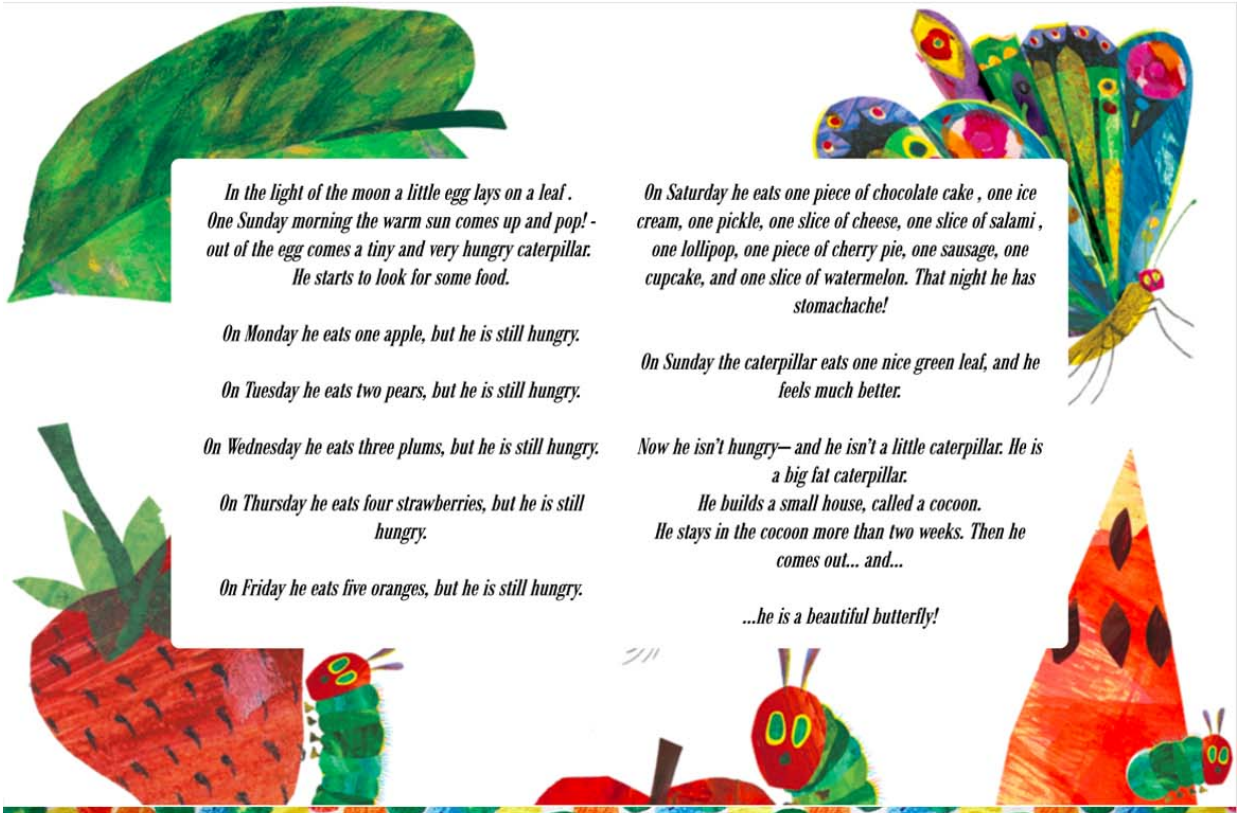
- When the player presses play the timer starts and the caterpillar starts moving
- Player can change the direction of the caterpillar using the up, down, left and right keys
- The caterpillar needs to eat the foods which appear as in the book below:

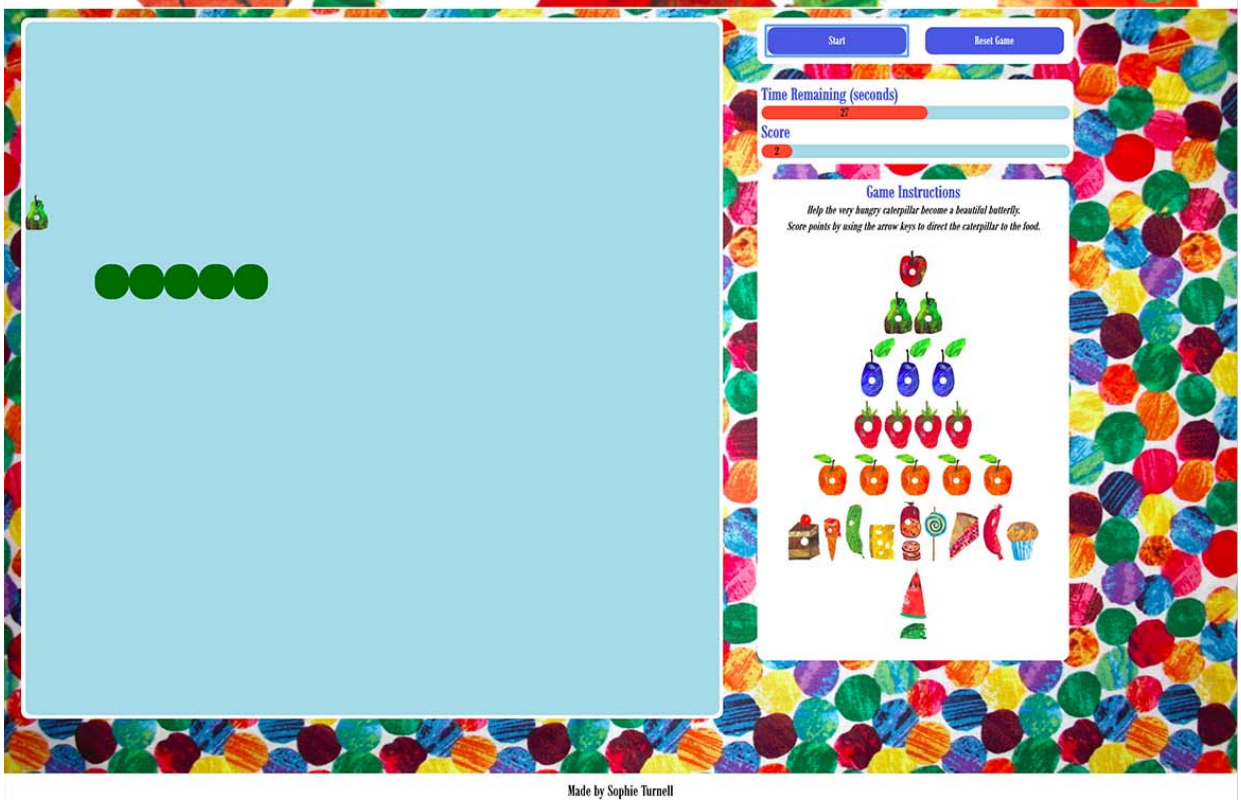


- When the caterpillar swallows the food:
    - the food is removed and the next item is generated in a random cell;
    - the caterpillar grows in length;
    - the caterpillar speeds up;
    - the score board records the number of foods eaten;
- Win condition: the caterpillar eats all of the foods on the list on the right (if I had had more time the caterpillar would turn into a butterfly)
- Lose condition: the caterpillar hits a wall
- Reset button:
    - removes the dead caterpillar;
    - removes the score;
    - resets the speed of the caterpillar;
    - resets the countdown timer and resets the food to start from the beginning with the array of food;
- Start button adds new food and the moving caterpillar

Website Architecture


by Eric Carle


In the light of the moon a little egg lays on a leaf .
One Sunday morning the warm sun comes up and pop! -
out of the egg comes a tiny and very hungry caterpillar.
He starts to look for some food.
On Monday he eats one apple, but he is still hungry.
On Tuesday he eats two pears, but he is still hungry.
On Wednesday he eats three plums, but he is still hungry.
On Thursday he eats four strawberries, but he is still hungry.
On Friday he eats five oranges, but he is still hungry.
On Saturday he eats one piece of chocolate cake , one ice cream, one pickle, one slice of cheese, one slice of salami , one lollipop, one piece of cherry pie, one sausage, one cupcake, and one slice of watermelon. That night he has stomachache!
On Sunday the caterpillar eats one nice green leaf, and he feels much better.
Now he isn't hungry— and he isn't a little caterpillar. He is a big fat caterpillar.
He builds a small house, called a cocoon.
He stays in the cocoon more than two weeks. Then he comes out... and...
...he is a beautiful butterfly!

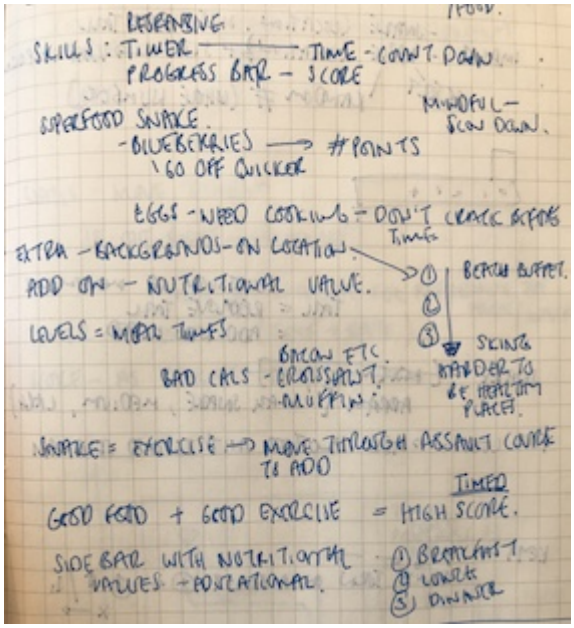

Start
Reset Game
Time Remaining (seconds)
27
Score
2
Game Instructions
Help the very hungry caterpillar become a beautiful butterfly.
Score points by using the arrow keys to direct the caterpillar to the food.
Made by Sophie Turnell

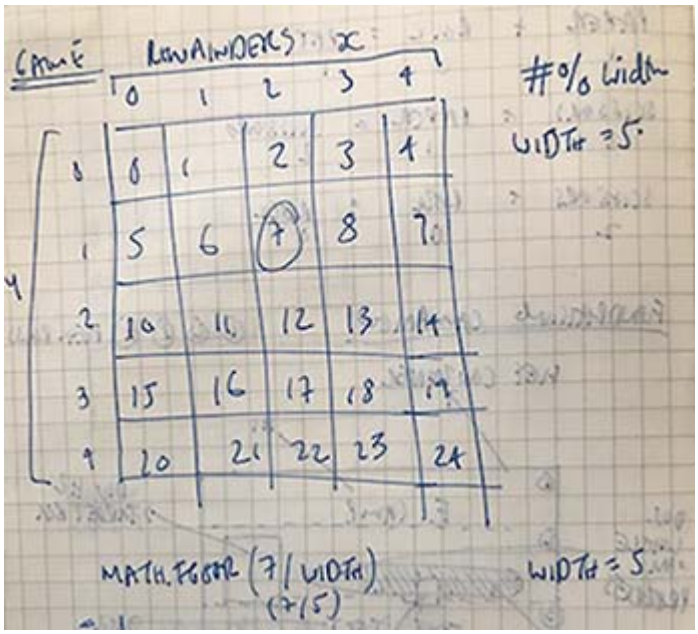
Approach

Day 1 & 2

When we were offered a choice of 10 grid-based games to build I can remember feeling daunted by the prospect. I spent the first two days working out which game I wanted to build and how that would work. Although this was not the most effective use of time for getting the project finished, it furthered my understanding of our first three weeks of class.

I decided to take the plunge and go with Snake as I had lots of ideas on how to style and enhance it and make it unique and I had the clearest idea of how the logic would work for this.





Day 3

I realised early on that reaching my MVP of having a working snake logic would take me most of the week.

Creating the classic snake game with win and lose conditions was my MVP. I broke down the game into the following elements:

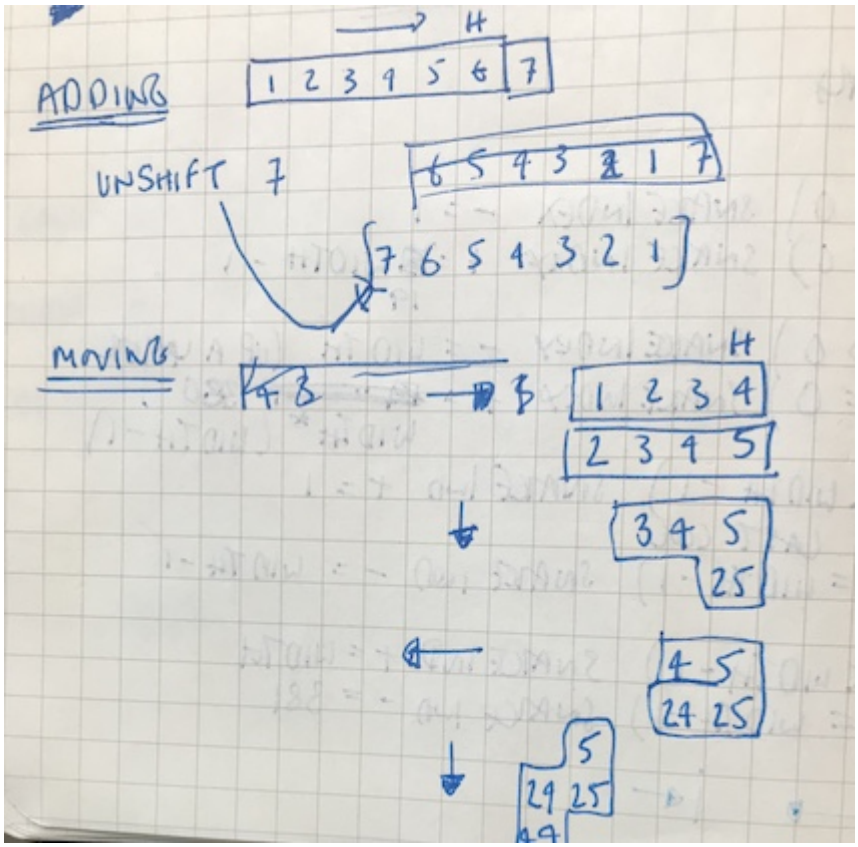
- Building a grid;
- Adding food at random;

- Adding a snake;
- Creating the illusion of the snake moving; and
- Killing the snake when it hit a wall

I wanted to add a score bar and timer so planned space for these in my wireframes. I then drafted some logic for the elements. This was really useful as it allowed me to think about how it would work in a visual way that I could look at at the same time as writing code. I was able to bring different pieces of logic together and establish an order using post-it notes. I started to build the structure in HTML5, formatting it with the Flexbox layout module in CSS3.

Day 4

I started to add the logic for the elements outlined above. Lots of trial and error here. A combination of lists and adding tasks to a calendar enabled me to effectively manage my time throughout the project. My notebook is full of amended lists as different items took priority. This was a long way from working still but my snake was moving which was a big win.




```
  // CHANGE DIRECTION OF SNAKE ARRAY
  function moveSnakeDown() {
    eraseSnake()
    snakeArray.pop()    // deletes last cell in snakeArray
    snakeArray.unshift(snakeArray[0] + width)   // adds width of grid
to the head of snakeArray cell (one row down)
    drawSnake()
  }

  function moveSnakeUp() {
    eraseSnake()
    snakeArray.pop()
    snakeArray.unshift(snakeArray[0] - width)   // subtracts width of
```

```
  grid to head cell (one row up)
      drawSnake()
    }

    function moveSnakeLeft() {
      eraseSnake()
      snakeArray.pop()
      snakeArray.unshift(snakeArray[0] - 1)    // subtracts one cell
  from head of snakeArray
      drawSnake()
    }

    function moveSnakeRight() {
      eraseSnake()
      snakeArray.pop()
      snakeArray.unshift(snakeArray[0] + 1)    // adds one cell to head
  of snake array
      drawSnake()
    }
```

Day 5

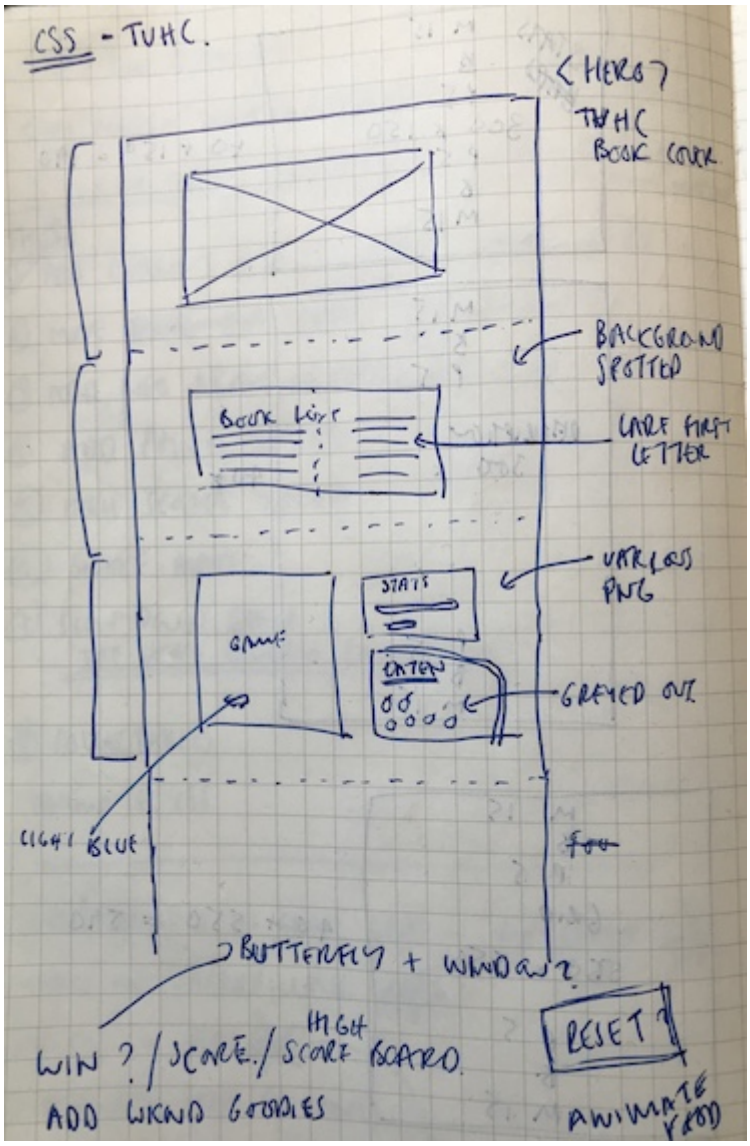
Worked on directions keys for caterpillar and removing the food when the cells matched. This took all day and only some of the food was removing. More brainstorms - lots of ideas but needed a much longer timeframe...

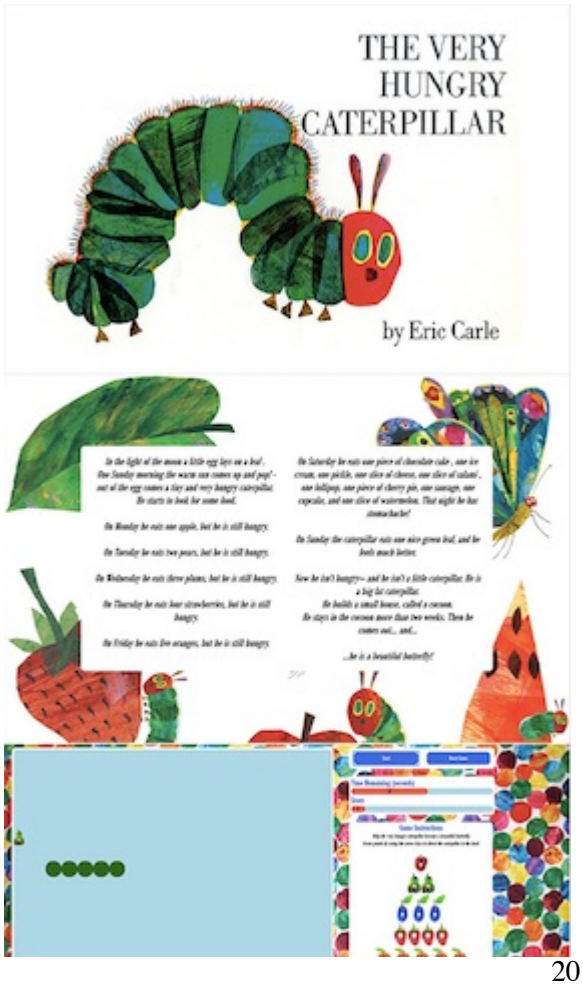
Day 6

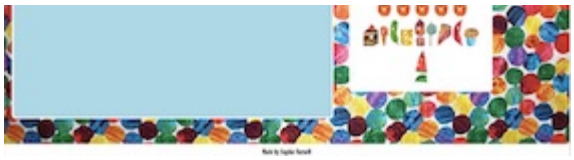
CSS and not chill. Added progress bars for score and timer and buttons. I was up very late trying to get these all to work. In future I will try not to make these sort of additions at the last minute. I realised that using a radius on the snake cells made it look like a caterpillar so decided to style it like The Very Hungry Caterpillar book.

Day 7

Came up with a new idea for the layout on the tube.


CSS - TWHC.
<HERO>
TWHC BOOK COVER
BACKGROUND SPOTTED
BOOK LIST
LARGE FIRST LETTER
STATS
GAME
ENTER
VARIOUS PNG
GREYED OUT
LIGHT BLUE
BUTTERFLY + WINDOWS
WIN ? / SCORE. / HIGH SCORE BOARD.
RESET ?
ADD WKND GOODIES
ANIMATE


THE VERY HUNGRY CATERPILLAR
by Eric Carle



CSS and panic about presenting our first project. I got distracted with the CSS thinking it would only take another 10mins but actually taking hours. I had a huge list of improvements to functionality (and styling) I wanted to make and retrospectively should have focused on the functionality. Although the presentation went well, the screen scrolled when you used the keys and start and reset buttons weren't working fully so the user had to refresh the page when they were in the right place. I have since fixed this although there are still some bugs I'd like to fix with more time.

Takeaways

In future I need to be careful not to get overexcited by the planning stage and to focus on hitting the MVP early. Styling and playing around with individual lines of code helped throughout to visualise the project and come up with ideas of how to solve problems. I found console logging particularly useful for checking exactly what was happening, (and what wasn't!)

Challenges Overcome

- Working out the right order of various functions
- Adding functionality to the buttons

```
// RESET BUTTON
 resetButton.addEventListener('click', () => {
   console.log('reset button clicked')
   gameOver()
   eraseSnake()
   timerNumber = 50
   timerId = 0
   currentCountdown.innerHTML = timerNumber
   timeRemaining.style.width = 100 + '%'
   score = 0
   scoreDisplay.innerHTML = score
   userScore.style.width = (score * 5) + '%'
   grid.classList.remove('food')
 })
```

- Stopping the screen from scrolling when using the keys to direct the caterpillar

```
// STOP DEFAULT SCROLLING
window.addEventListener('keydown', function (e) {
  if ([37, 38, 39, 40].indexOf(e.keyCode) > -1) {
    e.preventDefault()
  }
}, false)
```

- using the score as an index of which food should be shown

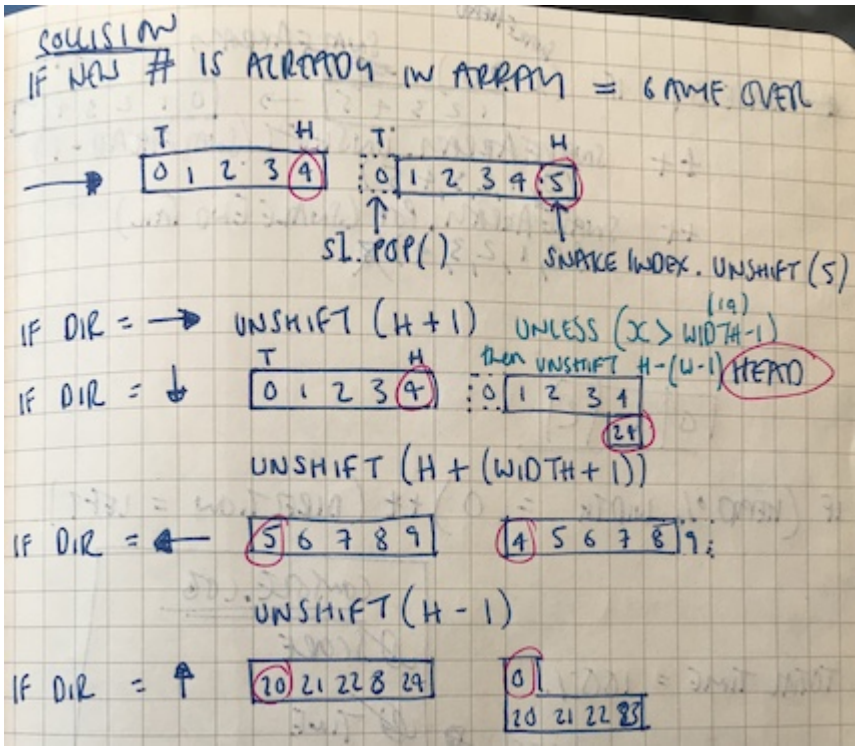
```
// CHANGE FOOD ITEM
  function changeFoodItem() {
    const foodArrayURL = [
      './assets/apple.png',
      './assets/pear.png',
      './assets/pear.png',
      './assets/plum.png',
      './assets/plum.png',
      './assets/plum.png',
      './assets/strawberry.png',
      './assets/strawberry.png',
      './assets/strawberry.png',
      './assets/strawberry.png',
      './assets/orange.png',
      './assets/orange.png',
      './assets/orange.png',
      './assets/orange.png',
      './assets/orange.png',
      './assets/chocolateCake.png',
      './assets/iceCreamCone.png',
      './assets/pickle.png',
      './assets/swissCheese.png',
      './assets/sliceOfSalami.png',
      './assets/lolly.png',
      './assets/cherryPie.png',
      './assets/sausage.png',
      './assets/cupcake.png',
      './assets/watermelon.png'
    ]
    foodArrayURL.forEach(element => element)
    document.querySelector('.food').style.backgroundImage =
`url('${foodArrayURL[score]}')`
  }
```

- removing rogue strawberries that remained after they had been eaten

---

Challenges Still To Overcome

- removing the food after the game is reset
- Timer stops play when it runs out
- killing the caterpillar when it bites itself




```
// SNAKE COLLIDES WITH ITSELF GAME OVER
 function killSnake() {
   if (snakeArray.slice(1).includes(snakeArray[0])) { // if snake
body includes snake head game over
     console.log('snake bites itself')
     // return gameOver()
   }
 }
```

Future Improvements

Many!

Small improvements:

- Add animation on the list of foods to eat on the right
- Adding a butterfly when all foods are eaten-user has won
- Add head to caterpillar

Larger improvements:

- Responsive design
- Multi-player mode
- High score table

---

sophieturnell.com
linkedin.com/in/sophieturnell/
