Canvas tag

* The canvas tag is used to draw shapes with javascript in HTML. It is a tag that lets us draw a shape without using extra tags. It has a method getContext('2d') that is used to draw shapes. [Canvas\_API](https://developer.mozilla.org/en-US/docs/Web/API/Canvas_API)

const ctx = gameCanvas.getContext('2d')

ctx.fillStyle = "white"

ctx.strokeStyle = "black"

cts.fillRect(x,y,width,height) //x is from left and y is from top of **canvas**

cts.strokeRect(x,y,width,height) //x is from left, y is from top

Positioning

* We can use the chessboard concept to know where the snake or the food is. Divide the rows into a specific number of cells and the columns, for example, 30 rows with 10px width and height and 30 columns.

Drawing the sneak

* To draw shapes, you have to know where and what height and width you want to draw. The snake has four cells; every cell has a position in the order and with the same height and width. We use a list to specify the dimensions and position.

let snake = [

{ x : 150 , y : 150 },

{ x : 140 , y : 150 },

{ x : 130 , y : 150 },

{ x : 120 , y : 150 },

{ x : 110 , y : 150 },]

snake.forEach(snakePart => {

ctx.fillStyle = 'lightgreen';

ctx.strokeStyle = 'black'

ctx.fillRect(snakePart.x , snakePart.y , 10 , 10);

ctx.strokeRect(snakePart.x , snakePart.y , 10 , 10);

})

To create the food

* An x and y, with the use of Math.random(), but the position in the canvas should be a factor of the height and width since the shape of food has a certain height and width.



* You also have to check if the position is not on the sneak.

let randomNumber = (max , min) => Math.round((Math.random() \* (max - min) + min) / 10) \* 10

let createFood = () => {

foodX = randomNumber(0,gameCanvas.width - 10);

foodY = randomNumber(0,gameCanvas.height - 10);

snake.forEach(snakePart => {

if(snakePart.x === foodX && snakePart.y === foodY) {

createFood()

}

})

}

* To make the snake move, if the array of snake cells is from head to tail, you can

To move the snake

* use list.shift( ) to add a new head to the snake cell array based on the previous head and use pop to remove the previous tail.

let advanceSnake = () => {

const head = { x : snake[0].x + 0 , y : snake[0].y - 10}

snake.unshift(head);

snake.pop()

}

* We have to create a new canvas each time we update the snake and food because if you just draw a new snake, it will add it to the previous canvas

function main() {

// game over

setTimeout(() => {

clearCanvas()

drawFood()

advanceSnake()

drawSnake()

main();

}, 100);

}