T2: REACTIVE JS Puc-Man

Grupo 3:
- Agustin Rios
- Martin Ocqueteau
- Nicolas Fraga

Juguemos!

Sobre la Animación y Dom....

Lo usual: JQuery

Sobre la Animación y Dom....

Lo usual: JQuery

Lo que usamos: D3

(Aunque está destinada mayormente a la producción de infogramas)

¿Entonces por qué D3?

```
const setDivGrid = (margin, height, width) => (id) => {
  return d3.select(id)
    .append('svg')
    .attr('id', `${id}-svg`)
    .attr('width', width + margin.left + margin.right)
    .attr('height', height + margin.top + margin.bottom);
};

const setSVGG = (margin, id) => (grid) => {
  return grid.append('g')
    .attr('id', id)
    .attr('transform', `translate(${margin.left}, ${margin.top})`);
};
```

¿Entonces por qué D3?

```
const drawCubes = (dim) => (grid, data, j) => {
    const colorByNumber = (number) => {···
    };
    /// d3 add cubes
    grid
        .selectAll('rect')
        .data(data)
        .enter().append('rect')
        .attr('x', (_, i) => (i - ((j * data.length) / (j + 1))) * dim.cubeSize)
        .attr('y', () => j * dim.cubeSize)
        .attr('width', dim.width)
        .attr('height', dim.height)
        .attr('fill', (d, _) => colorByNumber(d));
};
```

¿Entonces por qué D3?

```
grid
    .selectAll('image')
    .data(data)
    .join(
        (enter) =>
            enter
                .append('svg:image')
                .attr('x', (d) => d.x * dim.cubeSize)
                .attr('y', (d) => d.y * dim.cubeSize),
        (update) =>
            update
                .transition()
                .duration(timeTransition)
                .attr('x', (d) => d.x * dim.cubeSize)
                .attr('y', (d) => d.y * dim.cubeSize),
        (exit) =>
            exit
                .remove(),
    )
    .attr('width', dim.width)
    .attr('height', dim.height)
    .attr('xlink:href', (d) => d.image)
    .attr('preserveAspectRatio', 'none');
```

Construcción del Juego

- Uso de funciones asíncronas y promesas.

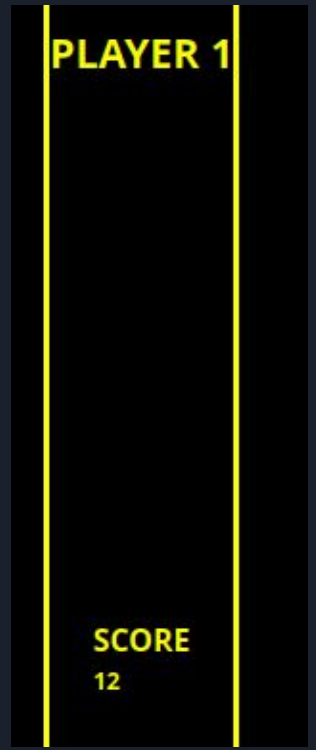
- Implementado principalmente mediante lo aprendido en programación funcional.

```
const MAP = (mapGrid) => {
  const _mapGrid = mapGrid;

  return {
    draw(array, drawFunction) {
      const mapLoads = [];
      array.forEach((row, j) => {
        mapLoads.push(...row);
        drawFunction(_mapGrid, mapLoads, j);
      });
    },
    insertElements(array, drawFunction) {
      drawFunction(_mapGrid, array);
    },
    delete(deleteFunction) {
      deleteFunction(_mapGrid);
    },
  };
};
```

```
const drawGame = (mapGrid) => {
  const mazeMap = MAP(setGameGrid(mapGrid, '#maze'));
  const _map = [];

  return {
    async setMap(map) {
      await _map.push(...map);
    },
    async drawMaze() {
      await mazeMap.draw(
        _map,
        drawCubes({
          cubeSize: CUBE_SIZE,
          width: CUBE_SIZE,
          height: CUBE_SIZE,
        }),
      );
    },
    async drawPoints() {…
    },
    async deletePoints() {…
    },
    async drawMob(mobList) {…
    },
  };
};
```

Puntajes

```
54  // Scores
55  function getPlayerObservables() {
56    const playerDivs = [...document.querySelectorAll('div.player-wrapper')];
57    return playerDivs.map((playerDiv) => fromEvent(playerDiv, 'eatDot'));
58  }
59
60  export function makePlayerSubscriptions() {
61    const player1Score = counter();
62    const player2Score = counter();
63    const [clickP1$, clickP2$] = getPlayerObservables();
64    clickP1$.subscribe(() => updateScore(1, player1Score));
65    clickP2$.subscribe(() => updateScore(2, player2Score));
66  }
67
```



Movimiento

Pacman

```
39    #keyPress = (event) => {
40      // left
41      if (event.keyCode === this.keys.left) {
42        if (this.currentDirection === move.right) this.currentDirection = move.left;
43        this.requestedDirection = move.left;
44      }
```

```
70    move = () => {
71      if (this.canMove(this.requestedDirection)) {
72        this.currentDirection = this.requestedDirection;
73      }
74      if (!this.canMove(this.currentDirection)) {
75        this.currentDirection = move.stop;
76      }
77      if (this.map[this.y][this.x] === 0) {
78        this.map[this.y][this.x] = -1;
79        d3.selectAll(
80          `circle[cx='${(this.x + 1 / 2) * this.size}'][cy='${(this.y + 1 / 2) * this.size}']`
81        ).remove();
82        signalEvent(this.div, event.eatDot)
83      }
```

Agentes: Movimiento

Fantasma

- Idéntico a Pacman en casi toda la implementación.

- Al llamar a sus métodos de movimiento se utilizan funciones random

Código principal

Loop de los movimientos en constante revisión si hay colisiones entre pacmans y fantasmas

```
export function gameLooper(mobs) {
  [pacman1, pacman2, ...ghosts] = mobs;
  const gLoop = setInterval(() => {
    // TODO: cambiar para que el juego se detenga al observar evento de pausa/fin de juego
    frame(mobs);
    ghosts.forEach((ghost) => {
      if (ghost.checkCollision(pacman1) || ghost.checkCollision(pacman2)) {
        mobs.forEach((mob) => mob.currentDirection = 0)
        clearInterval(gLoop)
        // window.alert('Han perdido!')
      }
    })
  }, timeTransition);
}
```

Ahora...

```
[Dudas,
Comentarios,
Curiosidades].size() > 0
? conversemos()
: finalizarPresentación()
```

Hasta la próxima tarea