Edyst Frontend Challenge

The challenge is themed around the ubiquitous game - 2048 https://gabrielecirulli.github.io/2048/.

It is divided into three parts:

1. Testing the waters

Given a 4x4 array representing the state of the board at any given instant, you have to place a new tile (either `2` or `4`) on it. - Generally more `2`'s are placed than `4`'s, but we leave the proportion in which they appear for you to decide. - Return the board and the value of the tile you have just assigned. - If it is not possible to place a tile, return the board and `-1`.

```
let const placenew = (board) => { ... }

let initial state = [
    [8, 0, 0, 0],
    [4, 2, 0, 0],
    [4, 0, 4, 0],
    [2, 0, 0, 0],
]

{updated_state, newtile} = placenew(initial_state)

console.log(updated_state)
[
    [8, 0, 0, 0],
    [4, 2, 0, 0],
    [4, 0, 4, 0],
    [2, 0, 4, 0],
]

console.log(newtile)
4
```

2. Swipe right, please

Given a 4x4 array representing the state of the board at any given instant, and a direction in which the board is swiped, you have to return the updated value of the board. For example:

```
let const swipe = (board, direction) => { ... }

let initial state = [
```

```
    [8, 0, 0, 0],
    [4, 2, 0, 0],
    [4, 0, 4, 0],
    [2, 0, 4, 0],
]

let direction = "UP"

updated_state = swipe(initial_state, direction)

console.log(updated_state)

[
    [8, 2, 8, 0],
    [8, 0, 0, 0],
    [2, 0, 0, 0],
    [0, 0, 0, 0],
]
```

Remember, the swipe function only takes care of updating the board by merging tiles. In the game, a new tile is placed on the board, before the tiles are merged in the direction of the swipe. Assume that the initial state of the board already contains the newly generated tile.

Refer to this for the rules of merging (wikipedia): $>$ 2048 is played on a gray 4×4 grid, with numbered tiles that slide smoothly when a player moves them using the four arrow keys. Every turn, a new tile will randomly appear in an empty spot on the board with a value of either 2 or 4.[2] Tiles slide as far as possible in the chosen direction until they are stopped by either another tile or the edge of the grid. If two tiles of the same number collide while moving, they will merge into a tile with the total value of the two tiles that collided.[16][19] The resulting tile cannot merge with another tile again in the same move.

3. A view of vue

Given that you have successfully implemented the two functions above, design and build a simple tile based version of the 2048 game using `vue.js`.

- The user should be presented with a `New Game` button that resets the score and clears the board
- In a game, the user should be able to press the up,down,left,right arrow keys to swipe the board - and the state of the board as well as the points accumulated should get updated dynamically
- The game ends with a success message when the user reaches tile 2048.
- If it is not possible to place another tile, and no merges are possible, the game should end with a message asking the user to retry.

Bonus steps

Completing these steps are not required, but will add immensely to your candidature for Edyst.

- Include a `README` and `package.json` defined with build and run steps
- Include one or more automated tests of your `Vue.js` component using any testing framework of your choice

Suggestions

- try making classes out of the tile and the board objects
- the examples make heavy use of the newer es6 features, but don't be put off by it. Keep most of the logic for the game outside of your vue files. Remember, let vue do what it does best: rendering an HTML document from a data object.

Submission details

- Upload your repo to Github and share it with us
- Also host the code somewhere and share the link with us. You could use www.netlify.com to simple drag and drop a static website.
- We will review the code and get back to you for an interview - either online or in person
- For any queries, please reply to the email thread
- All the best!