**Turn an array into a two dimensional array**

You can use the Array.prototype.reduce function to do this in one line.

**const toMatrix = (arr, width) => {**

**arr.reduce((rows, key, index) => (index % width == 0 ? rows.push([key]) : rows[rows.length -1].push(key)) && rows, []);**

**}**

This function takes two parameters:

1. **arr** – is an array of number e.g. [0, 1, 2, 3, 4, 5, 6, 7, 8]
2. **width** – the number of columns in a row e.g. 3

The **arr.reduce** function takes three parameters:

1. **rows** – current state of the array (accumulated value)
2. **key** – current value in the array
3. **index** – current index in the array

**index % width == 0 ?**

* Checks to see if the current index in the array has a remainder of zero when compared against the number of columns you want in a row.

If this is true

**rows.push([key])**

* This pushes a new array to rows with the current value inside that array

If this is false

**rows[rows.length -1].push(key)**

* Moves to the last embedded array inside the rows array then pushed the current value to it

***&& rows***

* Think this returns rows at the end

**,[]**

* This is the initial value of rows(accumulator) – in this case it’s set to an empty array