Introduction to Game of Life and p5.js

You have learned enough fundamental knowledge to attempt your first personal project. We are going to attempt the game of Conway's Game of Life with the help of a library called `p5.js`.

Conway's Game of Life

Conway's game of life is a **zero-player game** that is created by the English Computer Scientist and Mathematician John Horton Conway . It is very typical programming problem because of the simplicity, beauty and inherent complexity of this problem.

Let's look at what Game of Life looks like. Here is one famous example called Gosper's Glider Gun.

What is happening here ? It seems that we are having a form of life spitting something crawling to the corner. You may assume that the above animation is based on a very complex rule to generate such pattern. Surprisingly, the rules of game of life is very simple.

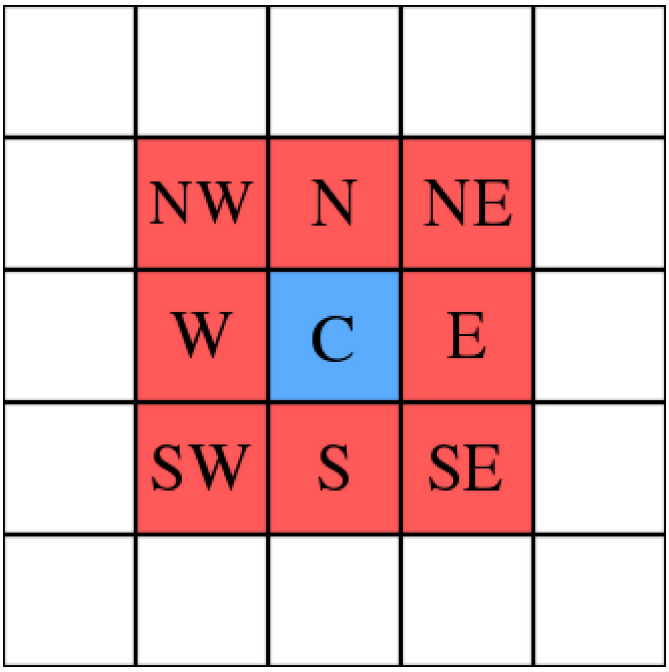
Rules of Game of Life

Game of Life runs on a 2-dimensional grid plane with square grid. On each grid, there could be with life (colored) or no life(white). So it looks like the following.

0	0	0	0	0	0	0
0	0	1	1	1	0	0
0	0	2	1	2	0	0
0	0	3	2	3	0	0
0	0	2	1	2	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

What are all those numbers? It is the neighbor of the grid. We call it the Moore neighborhood which includes all neighboring grids. We would hide the number of neighbors of all box with 0 neighbors

since it is trivial.

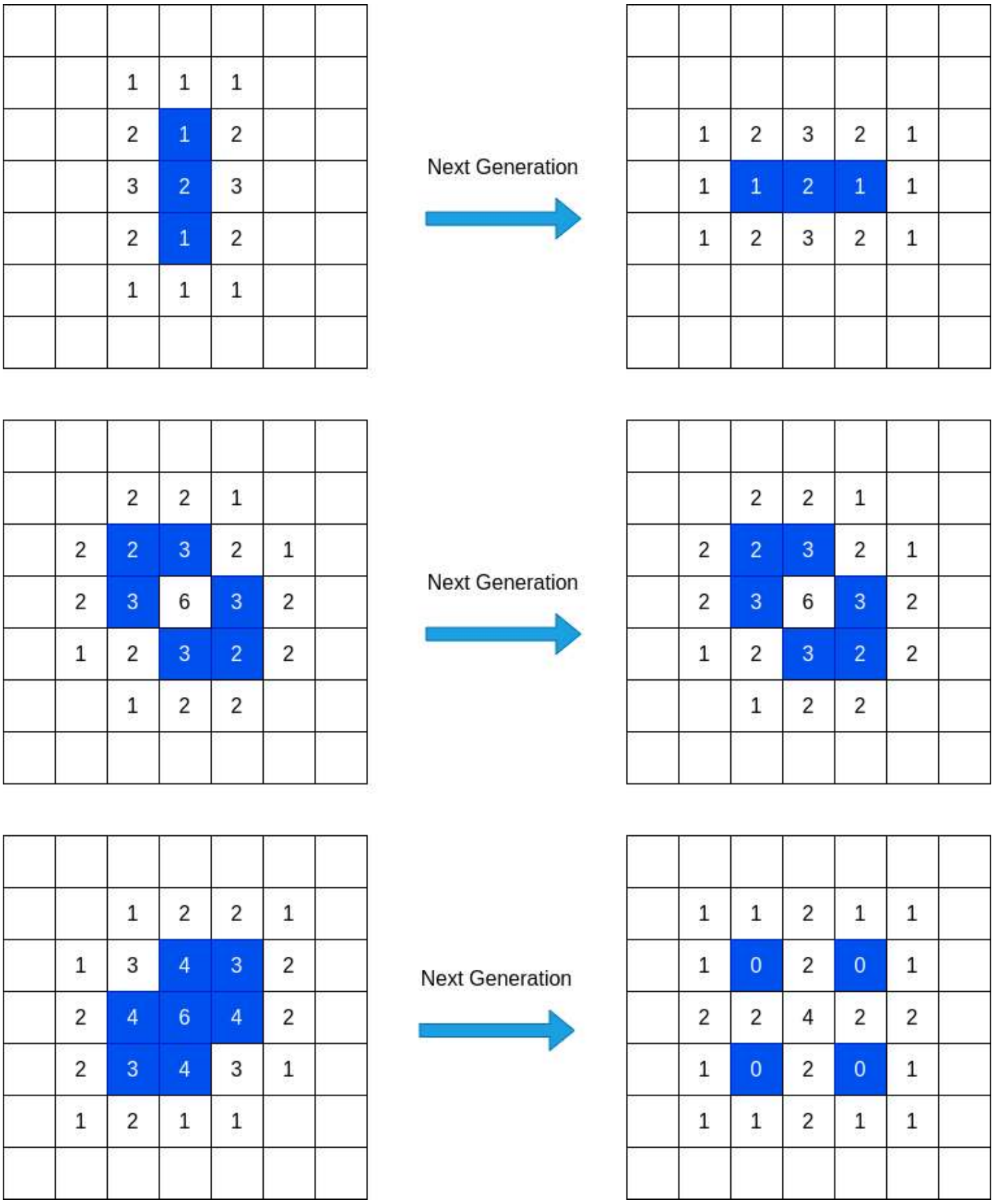


Here are the rules of Game of life:

- If a box has life and it has less than 2 neighbors. It dies of loneliness. The box becomes lifeless next generation.
- If a box has life and it has more than 3 neighbors. It dies of overpopulation. The box becomes lifeless next generation.
- If a box has life and it has 2 to 3 neighbors. Nothing changes for this box.
- If a box has no life and it has 3 neighbors, the box in next generation fills with life because of reproduction.

It seems pretty simple rule. But it gives unexpectedly complex behavior.

Here are some common patterns:



As you can see from these examples, the first one changes from a vertical three-box pattern to a horizontal three-box pattern. It is going to oscillate between these two patterns forever if there is no other box interfering.

The second one is a pattern just like Tecky Logo. It is stable because of the life-box has 2-3 neighbors.

The third one is much more populate and it becomes four separated box with 0 neighbors. So the third pattern would disappear completely after two generations.

That is the game we need to implement. It seems pretty difficult to implement all these boxes by ourselves. Hence we are going to make use of a very useful library called `p5.js`.

Here is a demonstration that showcase how the neighbors are calculated.

p5.js

`p5.js` is a library that makes use of the `canvas` element in web. It aims at making coding easier for designer, artist and programming beginners. Hence it is a very nice library for us to build our Game of life with this library.

Installation of p5.js

Installation of p5.js is very simple. You can just put the following CDN link to your `index.html`.

```
<script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.2.0/p5.js"></script>
```

Our resulting `index.html` would look like the following with `bootstrap` installed as well.

```
<html>
    <head>
        <meta charset="utf-8">
        <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=
        <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.cs
        rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD
        crossorigin="anonymous">
        <link rel="stylesheet" href="index.css"/>
    </head>
    <body>
        <div id='canvas'></div>
        <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle
        integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM
        crossorigin="anonymous"></script>
        <script src="https://cdnjs.cloudflare.com/ajax/libs/p5.js/1.2.0/p5.js"></script>
        <script src="index.js"></script>
    </body>
</html>
```

Usage

Using `p5.js` is very straight forward, there are two functions that you can put inside your `index.js` to start with. They are the functions `setup()` and `draw()`.

```
function setup(){
   // Setup logic here
}

function draw(){
   // draw logic here
}
```

`setup()` function is for setting up the initial values. It is going to run **exactly once**. `draw()` is going to run many times. It is going to run **once per frame**. If you have a frame rate of 30 frames per second, then `draw()` runs 30 times per second!

`p5.js` makes writing `canvas` related code easier because it simplifies the common flow of drawing animations into `functions` like `setup()` and `draw()`. They also have different functions in their Reference. We would utilize many of them in the next section.