**Exercise: Think about how to keep the game going**

The Game of Life is generating cells for one round of the game; our next task is to keep the game going for multiple generations. Take another look at the code we've got so far; to play the game, we're calling play, which calls computeNextGen.

What do you think is the best way to have our game continue computing new generations? Should we iteratively loop using playing after the Start button is clicked?

1. function play() {
2. while (playing) {
3. computeNextGen();
4. }
5. }

Are there any downsides to doing it this way? Can you think of a better way?

Our answer

Conceptually, wrapping the computeNextGen function in a loop is the right idea; however, in practice, it is problematic. As it turns out JavaScript only has one thread of control, and so as soon as you begin the while loop, this code will consume most of your browser's computing resources and make the game controls unresponsive, or slow to act. This approach also doesn't allow you to control the speed at which each generation is computed, which may be too fast to see the Game of Life computation unfold.

We gave you a hint about how we're going to approach this when we gave you some research homework on the setTimeout method. By using setTimeout we'll be able to control the execution of each generation by scheduling it to run *asynchronously* (meaning the execution of the function to compute the next generation won't block everything else the browser is doing). Let's move on to the next section and see how this works.