Introduction

This game is similar to those that begin with a series of balls arranged into a triangle or square pattern on a board. Balls can only move when they jump over another ball (in real life, this might also be pegs on a wooden board). When a ball does jump over another ball, the jumped ball is eliminated. The objective is to end up with only one ball.

Goals

1. I wanted to generate all graphics using SVG so that there would be no assets to manage. This could show that one doesn't need anything but the tools built into the front end to build a fun game. No libraries, no outside graphic editors.
2. The code would be simple enough that I could design levels quickly.
3. The interface would be intuitive enough that one would pick up the controls of the game without any trouble.
4. The graphics would be responsive.

A straight-forward designed

When you click on a ball, it turns gold to signify that it's active. Doing so might cause red squares to highlight on the board. These are your legal moves. If there are no legal moves, there will be no red squares. You can deactivate a ball by clicking on it again, or selecting another ball. If you click on a red square, the active ball will make the move (jump), and the jumped ball will be eliminated.

The application keeps track of how many balls are on the board. When a jump is completed, it checks to see if there is only one ball left. If there is, the level ends and the player progresses.

How does this work?

This project was a valuable exercise in object-oriented programming, functional programming and scoping. Because the user can potentially interact with each element, every tile on the board is an individual object, as is each ball. Each one has access to functions that run loops to check what objects are close by. Tiles also have functions to check if they are occupied by a ball.

Because a game like this is sequential with simple rules, it allowed me to break down the whole process into steps. For example, here are some of the steps involved when you simply click on a ball:

1. The ball becomes active, causing the SVG shading to change.
2. The ball runs a loop that checks the squares where it could possibly jump, regardless if they are legal.
3. The squares that will be jumped over check if they are occupied. If they are, there is a ball to jump over and it is a legal move.
4. The destination square will highlight red to show that the jump is legal.

To accomplish this, I made liberal use of properties attached to each object, such as square.occupied and square.highlighted. To keep the scope local to the selected object, I also made frequent use of 'this,' as the object that owned the function was integral to the board state.