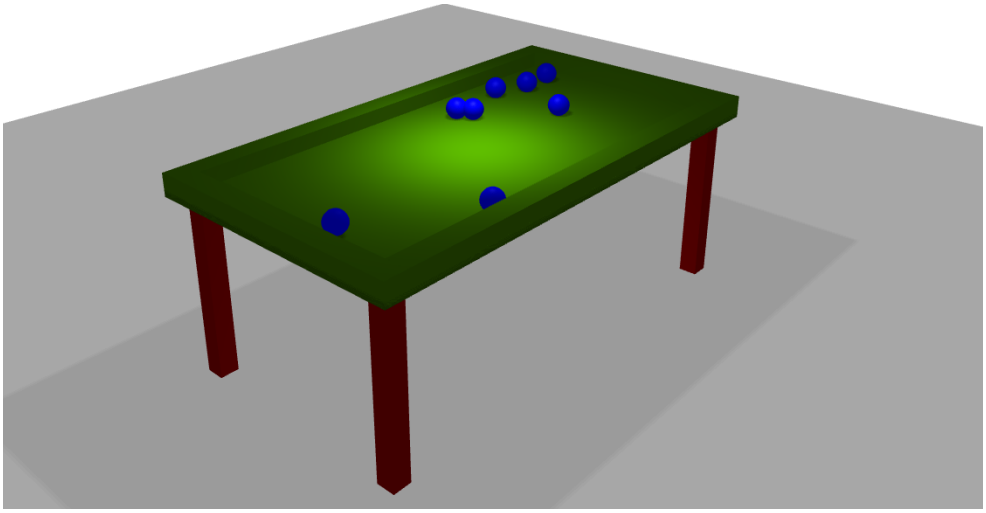
Assignment 3: Billiard Simulation (Part 1)

Write a billiard simulation. A possible solution could look like this:



Requirements:

- Create a table of realistic proportions. Add cushions and legs. See here for table dimensions: https://en.wikipedia.org/wiki/Billiard_table#Dimensions_2
- Add 8 billiard balls (ideally as wireframe models) of realistic size:
  - Place the balls initially at random, non-overlapping positions on the table.
  - Move the balls according to a random velocity vector assigned to each ball. Due to friction the speed of each ball drops by 20% each second.
  - Make sure the balls are rolling without slip and not just sliding.
  - Take into account reflection off the cushions. The speed of a ball drops by 20% at each reflection due to loss of energy.
  - Do not take into account collisions among the balls.
- Add a spot light above the table whose position is indicated by a yellow light bulb. Add shadows of the billiard balls visible on the table.

Handing in the code

- No need to hand in anything. Grades will be given after the second part of the assignment.
- You are encouraged to discuss with your colleagues. But everybody is supposed to write her or his own program. Copy-and-pasted code leads to subtraction of grade points.

Coding Style

Please stick to the following coding guidelines. Failure to do so leads to subtraction of grade points.

No errors: The code must not create errors in the browser console.

No junk files: The zip-file handed in must contain only those files that are necessary for the project.

Indentation: All code must be properly indented.

var: Always declare variables with the `var` keyword inside functions. See also here: https://google.github.io/styleguide/javascriptguide.xml?showone=var#var

Semicolon: Always use semicolons. See also here: https://google.github.io/styleguide/javascriptguide.xml?showone=Semicolons#Semicolons

Variable names: Use descriptive variable names.

```
var a = [1,2,3];    // bad
var ballSpeed = [1,2,3];    // better
```

An exception to this rule can be the naming of loop variables in short `for`-loops.

No magic numbers: Store numeric values in reasonably named variables instead of using them directly:

```
speed.multiplyScalar(1-0.2*dt);    // bad

rollFric = 0.2;
speed.multiplyScalar(1-rollFric*dt);    // better
```

Comments:

- You are encouraged to read also the rest of Google's Javascript coding guidelines: https://google.github.io/styleguide/javascriptguide.xml
- It is useful to check your code with *jshint* (or a similar tool) which is built into many editors or IDEs. Alternatively, see http://jshint.com/