Assignment 4 – Interactive 8-ball game

Version 1.01 (last update: Mar. 19)
Changes highlighted in yellow
Due date: Mon, Apr 1, 9:00 AM

Summary and Purpose

For this assignment, you will be creating a web-site that allows users to play a game of 8-ball.

Deliverables

You will be demonstrating your program to the instructor and/or TAs during a scheduled appointment on or after Apr. 1st. You do not need to submit your software.

Despite this, you are required to archive your code in gitlab as you work to prevent losing your work, to transfer between computers, to practise using appropriate software tools, and importantly to show that you completed your project by 9am on the 1st.

During your demo you will begin by download your software from gitlab and showing the TA that it was uploaded before 9am on the 4th.

You will need to demo you code in the THRN lab but can run it on your own laptop or remotely on some other machine.

Part I: Making a shot via UI

Develop a webpage where you can display a pool table and click on the cue-ball and drag the mouse. As you are dragging the mouse you should live add and update a thick (stroke-width) line from the centre of the cue ball to the mouse position. When you release the mouse button, the difference between the x and y components of the release position and the cue ball position should be calculated and used to compute the initial velocity of the cue ball. At this point the line should be removed. Acceleration should be computed the same way that we have all along. The tables resulting from the shot should be added to the database as in A3.

Leave enough space around the table and multiply the line parameters by appropriate constants, to be able to make shots at speeds up to, 10000mm/s.

Part II: Animating the shot

Develop a webpage which animates the shot created in Part I. The webpage should start a timer at time=0.0, and request the table corresponding to time=0.0 of the shot, which the server should send as an SVG file. After the page has displayed the SVG file, it should check the value of the timer and request the table at the new time. This process should be repeated, so

that the shot is displayed in real-time (regardless of the connection speed, internet traffic, browser speed, or computer speed).

Part III: Playing 8-ball

Create a website that allows two players to play a complete game of 8-ball. Both players will play on the same browser. Allow both players to enter their names. Have the server decide randomly who goes first. Have the server indicate who's turn it is.

One player will play the lower numbered balls (1-7) while the other player will play the higher numbered balls (9-15). Whoever sinks the first ball gets assigned that half of the balls. Once the first ball has been sunk, the server should display "high" or "low" after the player's name to indicate which balls that player is playing.

After sinking all their own balls, a player must sink the black (8) ball. Any player who sinks the 8 ball before sinking all their other balls automatically loses! Whoever sinks the 8 ball first (after sinking all their own balls) wins the game.

If a player sinks and opponent's ball, that ball is credited to their opponent, but there is no other penalty.

If a player sinks one of their own balls (even if they also sink an opponent's ball, or the white ball), they get another turn. Otherwise, the opponent gets a turn.

If the cue ball (white/0) falls into a hole it is returned to its origin position (as per A2Test01.py).

Simplifications: the following rules are not implemented to make server simpler.

1. Calling shots, players do not need to indicate which ball they intend to sink, nor which pocket they intend it to fall into.
2. Scratching, sinking the cue (while/0) ball incurs no penalty.
3. Missing all the balls incurs no penalty.
4. Sinking and opponent's ball incurs no penalty beyond leaving the opponent with one less ball to sink.

For full marks, set up a table with 15 balls. If this is too difficult you can set up a table with less balls, but must have an equal number of low and high balls, and include the 8 ball.

Nightmare Mode

When a player moves their mouse over the cue ball, make the cue ball much larger, so that they can select a point on the cue ball to click on. Use the position on the cue ball where the player pushes on the mouse button to apply spin. Above the centre, top spin (ball rolls further/decelerates more slowly). Below the centre, bottom spin (ball rolls less far/decelerates

more quickly). Left of centre, ball curves towards the right as it rolls. Right of centre, ball curves toward the left as it rolls.

A4 Demo Instruction and Rubric

There is a ***zero-tolerance*** policy in effect for unprofessional behaviour towards the TAs. If you have any issues with the demo or the grading of the demo, please contact the course instructor after the demo.

Practise all the following steps before your demo. You will only have 10 minutes to complete the demo. If something goes wrong, you will still only get 10 minutes (in the event something goes wrong beyond your control we will reschedule your demo rather than delaying all the other students' demos).

1. Book a time for your demo here via the link provided on CourseLink Announcements at the end of March.
2. Finish your work on A4 and deposit everything you need into the repo.
3. Include a database that you have created. You may want to put an initial table in the database and load it at the beginning of each game (instead of constructing it programmatically like in A3Test03.py).
4. Repeat the steps of "The demo", below, until everything works the way you want.
5. If you did not complete part of A4, modify your demo to skip those steps.
6. When you are confident that everything is working correctly and you can do the demo, DO NOT MODIFY YOUR GITLAB SUBMISSION.

The demo:

7. Arrive early enough to set up your computer. You can work on the machines in THRN 2418, or bring you own laptop.
8. Create an empty directory.
9. Retrieve your code from the GitLab repository.
10. Run `git log` in a terminal window. Leave that window showing the git log.
11. In a different terminal window, enter the command to start your server, but don't hit return until the TA arrives. If you are running on a shared machine use a port that contains part of your student ID like in assignment 2.
12. Wait for the TA to arrive.
13. Spell your login ID to the ID and tell them your student ID number.
14. Show them the window with the git log.
15. Start your server (make sure you know the port number).
16. Load the start page in your favourite browser.
17. Play a game. You can play against the TA, or play both players yourself (e.g. if you don't want the TA to touch your mouse).
18. Shut down your sever and logout.
19. Leave the lab room. Do not disturb other students doing demos or watch other demos.

Rubric

Students Login: Student ID: Date/time:

Ability to launch the server and add player names	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Ability to draw a cue	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Cue's direction and length determines balls direction and speed	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Animation shows balls moving on table	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Animation shows reasonable ball collisions	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Animation shows reasonable cushion collisions	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Balls fall into holes	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Players are correctly assigned to low or high	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Turns change correctly	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0
Winner is identified and reported	Outstanding, no issues 5	Good, Minor issues 4	Satisfactory, Some limitations 3	Unsatisfactory, Doesn't work properly 2	Not implemented 0