# Project 2 – JavaScript Game – The Purple Dot Game

## Expectations

- Instructor Guidance in Class – **High**
- Independent Effort – **Moderate**
- Originality – **Low**
- Teamwork – *None*

## Description

This project assumes that you have already done the Project 1 setup work for the course. ***To get started*** you will need, at the absolute minimum, VS Code and a mis3502workspace folder on your computer to store your work. ***By the end of the project,*** you need to share your work using the MIS3502 S3 bucket you created for all your web work this semester.

If you need help with those things, please see videos in Project 1.

A disclaimer – this project is meant to reacquaint students with client-side code. It does not illustrate the best approach to creating a JavaScript game. A proper approach to that topic would use an HTML5 canvas (see: w3schools "HTML5 canvas") and libraries/frameworks dedicated to creating interactive games.

***Another*** disclaimer – as important as A11y is, it is very difficult (perhaps impossible) to create a version of this game that a person with a serious vision impairment can use. So, for this project only, which is purely an educational exercise, we are setting our A11y concerns aside.

Let's get started.

## Together as a class

1) Students should start by downloading joke.zip and purpledot.zip. Copy the joke folder from joke.zip into your local mis3502workspace folder on your personal computer. Copy the purpledot folder from purpledot.zip into your workspace as well.
   (MORE BACKGROUND / INSTRUCTIONS: **https://youtu.be/SXnuIcLRCxs ~ 7 mins**)

2) Starting in the joke folder, open the index.html file in VS Code. Open the index.html in chrome. Follow the instructions on the screen.

3) As a class – discuss:

   a. How would you describe the behavior of this code to a non-technical person?

   b. While examining the code … we will review the following:

      i. What are some of the HTML elements used?

      ii. What are some HTML ***attributes***?

iii. What are some of the CSS elements used?

iv. What are some of the JavaScript elements used?

v. What are some of the jQuery elements used?

vi. What's the difference between JavaScript and jQuery?

vii. Is this code making any API calls?

viii. What jQuery ***events*** and ***methods*** are we using? Which is which?

c. Notice that there are some new-to-you jQuery methods in use here. Students are not expected to memorize these, but they are expected to take note of how they work, and then be able adapt and/or reapply them for their own purposes.

4) Now examine the contents of the purpledot folder.

5) As a class we will create the Purple Dot Game. An example of the finished product can be found here: https://youtu.be/FQL7snoMy4E (EXAMPLE)

Some things to notice:

a. There is a div with the id "panel". The "panel" div contains additional div tags with the following ids: "message", "dot" and "paddle". Not all of these are visible when the page loads initially.
b. Below "panel" there is another div used to hold your HTML buttons. It has an id of "controls". There you will find HTML buttons with the following ids: "btnGo", "btnStop" and "btnPlayAgain"
c. We have global variables to control the overall state of the game. It is usually best to use global variables sparingly or not at all. But in this case, it makes sense! Here are the variables:

```
let leftoffset = 50;
let topoffset = 50;
let leftoffsetpaddle = 25;
let blnContinue = false;
```

d. We have some supporting functions to control what's happing in the game. You might need to make at least one more before the project is done.
e. We will also have some event handlers. You might need to make at least more before the project is done.
f. The pattern of variable declarations at the top of the code, supporting functions in the middle, and one or more event handlers at the bottom of the script is a common one. You will see that pattern over and over in this class.

6) FOLLOW ALONG with your instructor.

a. We will start our work by adding a click event handler for the "Play Again" button. (Starting out easy here…)
(MORE INSTRUCTIONS: **https://youtu.be/cd35NjGZTxg ~6 mins**)

b. Next, we will add logic to prevent the paddle from sliding off the right-hand edge of the panel. (MORE INSTRUCTIONS: **https://youtu.be/Ti8_Bd3-HZU** **~10 mins**)

c. We will then edit the existing function dotAnimation and also create a supporting function continueDotAnimation.

   i. Start by creating logic to allow the dot to bounce around the panel. You do this by manipulation of the global variables leftoffset and topoffset.

   ii. Take note of the existing event handlers for the “Stop” and “Go” buttons.

   (MORE INSTRUCTIONS: **https://youtu.be/HBOSCApGjBs** **~19 mins**)

d. Finally, add in logic to let the dot bounce off the paddle. If the ball does not bounce of the paddle, and instead reaches the bottom of the panel, the game is over.
   (MORE INSTRUCTIONS: **https://youtu.be/HtqoUfYuIsE** **~18 mins**)

   (FULL DISCLOSURE – At 6:47 I make a mistake. I fix it at the end of the video!)

   (WRAP UP: **https://youtu.be/S19wzNfgivo** **~3 mins)**

**CONTINUED…**

## On Your Own

Complete the work we began in class. Test your work. Game play should be intuitive and fun.

Once the game is working to your satisfaction, add in the following elements.

1. Replace the Purple Dot with a picture of your face.
2. Every time the dot hits the paddle it should spin for 1 second.

See this video for an example: https://youtu.be/CQZ7hgGyQw0 (EXAMPLE)

## Hints

1. Here is a code sample of the CSS I used to put my picture on the dot:

```
#dot
{
        position:absolute;
        top:225px;
        left:375px;
        background-image: url("images/shafer.jpg");
        background-size: cover;
        border-radius:100px;
        height:50px;
        width:50px;
        }
```

- Notice that used the background-image feature of CSS. I did not change the #dot div tag to an HTML img tag. Keeping the tag a div was an easy way to achieve the desired effect without having to change much code.
- Also notice that the CSS code above implies that I created a folder called images and stored a file named shafer.png in that folder.

2. For the spinning effect, there is a Font Awesome CSS class called fa-spin.
   - To make the dot spin, add the class "fa=spin" to the tag with the id of "dot"
   - To stop the dot from spinning, remove the class.

3. You need to think about both adding and (after 1 second) removing the spin effect. To figure that out I used W3schools to research the setTimeout function in JavaScript.

## Turn in your work!

When you are done, upload your work to a "projec2" folder in your S3 bucket. Share the URL to your work with your instructor using the corresponding assignment out on canvas.

CONTINUED …

## How will this project be graded?

| Item | Points |
|---|---|
| I gave the instructor a good, working URL on canvas. (all or nothing) | 20 |
| I was able to reproduce the paddle behavior covered in class. (all or nothing) | 20 |
| I was able to reproduce the dot bounce behavior covered in class (10 or 20 points … must bounce off the paddle for 20 points) | 20 |
| Visual elements (buttons, messages) appear and disappear appropriately (0,5,10,15,20) points | 20 |
| I added my picture to replace the purple dot. (You must do this to ear these points, and the points that follow.) | 10 |
| I added the spin effect when the dot hit the paddle | 5 |
| I remove the spin effect from the dot after one second. | 5 |
| **TOTAL** | 100 |

Please see the syllabus for the late policy.