# Assignment09 – Hidden Hippos Game Part 2

In this assignment, students are going to build a simple game called “Hidden Hippos”.

***You may wish to reuse some, but not all, of your work from the first “Hidden Hippos” assignment.***

You will use your knowledge of Ajax and error trapping to write this code.

## Overview

Important notes:

* In this assignment you will write JavaScript code that renders an HTML table. You will do this using a **for** loop, nested inside another **for** loop.
* If you need help understanding HTML table syntax. See this video from earlier in the semester: <https://youtu.be/6xldLvon-yI>
* The size of the table will be controlled by the user. That is: is it a 5 x 5 table? 6 x 6? 10 x 10? Bigger?
* You will need to implement error trapping.
* The table will contain 5 hippos. The location of the hippos is determined by the API found here:
  + <https://misdemo.temple.edu/hippo/grid?N=5>
  + Notice that this URL allows you to specify different values of N.
* You may use concatenation to build a string of characters to represent the HTML table. Or you may wish to use the jQuery append method to systematically append HTML to the target tag. Either is fine.

## Instructions – Part A

1. Download assignment09.zip as provided by your instructor.
2. Take a moment to explore the start file hippogame2A.html. A lot has been built for you already. It will be helpful to know what is / isn’t in the start file.
3. Take a moment to explore the API . Notice that the API always returns exactly 5 hippos in random locations. It also won’t accept values of N smaller than 5, or greater than 40. Try these links:

* [https://misdemo.temple.edu/hippo/grid?N=**40**](https://misdemo.temple.edu/hippo/grid?N=40)
* [https://misdemo.temple.edu/hippo/grid?N=**4**](https://misdemo.temple.edu/hippo/grid?N=4)
* [https://misdemo.temple.edu/hippo/grid?N=**5**](https://misdemo.temple.edu/hippo/grid?N=5)
* [https://misdemo.temple.edu/hippo/grid?N=**6**](https://misdemo.temple.edu/hippo/grid?N=6)

**CONTINUED**

1. All your work should be done in the function hippoBuildGrid. This function accepts one parameter, N.
   1. Your first task is to complete the error trapping. In this assignment the error trapping is going to go inside the hippoBuildGrid function. ***Do not make the Ajax call if error trapping fails.***
   2. If N is not a natural number, use the jQuery html method to put “Bad data. Try again.” into the div tag with the id of #textDisplayed1. You should also terminate (that is: quit, stop, cease, abruptly end) the function at this point. You do this in ***two*** lines of code, like this:

$("#textDisplayed1").html("Bad data. Try again.");

return; //this makes the function stop!!!

* 1. N must be between 5 and 40 , inclusive. If that is not so, then put “N must be between 5 and 40.” in textDisplayed1 and stop the function.
  2. If the value of N gets past the error trapping, then prepare a variable called the\_serialized\_data and write it to the console.log as shown below.

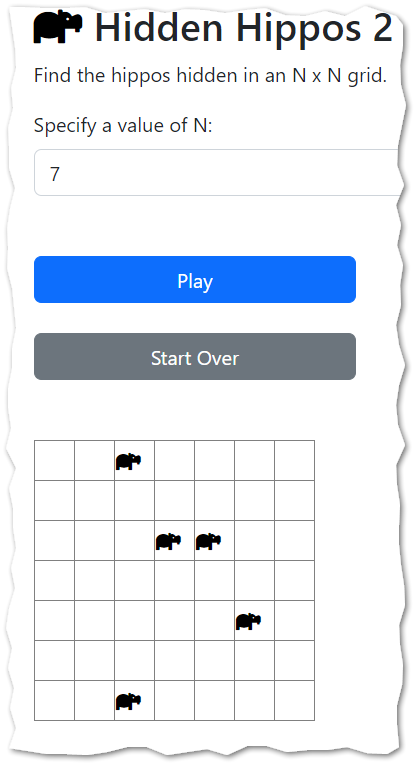
let the\_serialized\_data = "N="+N;

console.log(the\_serialized\_data);

|  |
| --- |
| Stop and read this: In previous lecture material, you were shown the following basic mechanism for making an Ajax call using jQuery:  $.getJSON(*someurl*, function(result) { *some things to do* });  In this assignment, we not only want to *retrieve* data from some url, but we also want to *send* some data. Specifically, we want to send data such as N=6 or N=10 or N=22. We will do that now by placing the variable the\_serialized\_data into the $.getJSON statement as shown below.  $.getJSON(*someurl*, the\_serialized\_data , function(result) { *some things to do* }); |

**CONTINUED**

* 1. Now, ***inside*** the callback function, console.log the data you got back from the API.
  2. Next, loop through the result returned by the API call. Use your loops to render an HTML table. When you are done you should be able to click on the “Play” button and see a grid like the following.



Of course, the size of the grid varies with user input, and the locations of the hippos change randomly.

## Instructions – Part B

1. Now work on hippogame2b.html. Start by copying your hippoBuildGrid work over from Part A. Be sure to copy over both your error trap logic, and your Ajax call.
2. A function named refreshHippoClicks has been written for you. This function should be called ***inside*** the callback function of your getJSON statement. should be called ***inside*** the callback function and ***after*** the tag textDisplayed1 has been fully populated.
3. Test your work. If you click on each one of the pink hippos, you should see a confirmation message that says “Good job! You found them all.” It will appear similar to what is shown below.

A black square with pink pigs in it

Description automatically generated

1. Once everything appears to be working ok, then change the color “pink” on line 25 to “black”. ***This is what makes the game interesting.*** Black hippos on a black background are hidden hippos!

## Check your work

1. Compare your work to these videos. These videos have no audio.

* Part A. <https://youtu.be/yERG7x4GXuQ>
* Part B. <https://youtu.be/p-Q6f0WTJcs>

## Turn in your work

1. Publish your work to misdemo. Check your URL.
2. Go to canvas and turn in your URL there. A correct URL will start with **https://misdemo**

## How will this assignment be graded?

|  |  |
| --- | --- |
| Error trap in Part A | 10 |
| Ajax call in Part A | 20 |
| HTML Table rendered in Part A | 20 |
| Error trap in Part B | 10 |
| Click refresh in Part B | 20 |
| Changed pink to black | 10 |
| Good Upload, Good URL | 10 |
| **TOTAL** | **100** |