**Javascript API**

**processInput()**

* Triggered when the user clicks the “Submit Values” button
* Gets the input values from the length and width input textboxes
* Validates the user's input
* Removes the old table and draws the new table based on the users input for length and width
* The number of cells drawn for the table are length\*width

**validateInput(length, width)**

* Checks if the user’s input is a numeric value above 0 and less than 256
* Does not allow non-numeric input
* Return true if the input is a number

**removeOldTable()**

* Removes the old grid and creates a new one based on the user's input
* Will erase any previous cell’s that were colored
* New grid has all cells set to a white background

**clickedCell()**

* If the background color of the clicked cell is white it will change it to the color selected from the color picker button (default color is Black)
* If the clicked cell is not white it will set it to white

**resetGrid()**

* Wipes the grid clean, removing all colored cells
* Resets all cells in the grid to a white background

**debug()**

* Runs through a series of tests to demonstrate correct functionality
* Tests invalid input to test error and input validation functionality

**function test\_1()**

* Tests creating tables of a given length and width
* Passes correct input into length and width: 20x20 and 50x50
* Creates a 20x20 table then creates a 50x50 table

**function test\_3()**

* Tests error and input validation
* Assigns incorrect values to length and width
* Displays error messages based on incorrect input

**drawTAMU()**

* Preset function
* Creates a 25x25 table
* Colors specific cells to create a A&M logo

**drawFlower()**

* Preset function
* Creates a 50x50 table
* Colors specific cells to create a flower
* Background is a light blue color

**drawChecker()**

* Preset function
* Creates a 100x100 table
* Colors cells based on a pre-determined pattern
* Colors include green, orange, blue, red, yellow, and purple

**sleep(milliseconds)**

* Takes in a value and determines how long to wait
* After the allotted amount of time has passed it will return