Javascript

- **debugging** (https://www.w3schools.com/js/js_debugging.asp)
  - **console.log()**
    - Use this to debug your javascript program by tracking and examining the values of points, vectors, and matrices when there is nothing showing on the canvas or when the picture is partially drawn correctly
    - Example: console.log(tMatrix);
      console.log("Point 1 is"); console.log(point1);
  - Set breakpoints in the debugger window using "**debugger**" keyword in program
  - In html, use: document.write("msg"); and document.writeln(value);

- **Variable**
  - var vs. let
    - let has block scope;
      - let is only accessible within the block it is declared
      - it can have global scope or local block scope depending on where it is declared
    - variable declared using let cannot be redeclared within the same scope;

    - var does not have scope, variable declared with var may be redeclared within the same scope

```
function checkLoopScope() {
 var i = 4;
 for (var i = 0; i < 10; i++) {
  // some statements
 }
  // updated value of i will display
  document.write('Final value of x outside of the loop: ' + i);
}
Output:
Final value of x outside of the loop: 10
```

  - Scope
    - A variable declared outside a function, becomes **GLOBAL**. A global variable has **global scope**: All scripts and functions can access it.
    - Automatically global — assign a value to a variable without declaring that variable makes that variable automatically global
  - == vs === (equal value and equal type)

- **array**
  - array literal
    - **syntax:** **var array-name = [item1, item2, ...];**
    - **Example:** **var points = [vec2(0. 0.5), vec2(0.5, 0.5)];**
  - array object
    - var cars = new Array("Saab", "Volvo", "BMW"); ← no need to use this form, slow
  - **access array element using index number**
    - **Example:** point1 = points[0]; point2 = points[1];
      p1_x = point1[0];
      p1_y = point1[1];
  - array elements can be of different types
  - Associative Arrays ← do not use, use number indexed array instead
  - **array methods**
    - **often useful in WebGL :**
      - **pop** (remove the last element)
        Example: points.pop();
      - **push** (adds a new element at the end)
        Example: points.push (vec2(0.5, 0.5));
      - **length** (returns number of items in the array)
        Example: points.length
    - valueOfarray, shift, delete, splice, slice
    - sort, reverse, defining the compare function: function(a,b)

- **Object** http://www.w3schools.com/js/js_objects.asp
  - define data and methods

```
var person = {
 firstName: "John",
 lastName : "Doe",
 id       : 5566,
 fullName : function() {
  return this.firstName + " " + this.lastName;
 } };
```

  - access data and methods by name: person.lastName
- **Math object**
  - Math.random() ← between 0 (inclusive) and 1 (exclusive);
  - **min, max**, round, ceil, floor,
    - Example: xValues=[3.4, 2.5, 6.7, 1.5];

```
minXvalue = Math.min(xValues);
roundedValue = Math.round(minXvalue);
```

- Pay attention to the trigonometry functions: http://www.w3schools.com/js/js_math.asp
  - Math.PI, Math.sin(), Math.cos(), etc.
  - The trigonometry functions operates on radians, not degree
    - Example:
```
angle = 60;
cosAngle = Math.cos(60*PI/180);
// PI constant needs to have been defined
```

- **function** http://www.w3schools.com/js/js_functions.asp
  - no need for function declaration
  - function is an object, it can be passed as parameter
  - number of parameters in the function call do not have to match the number in the function definition
  - parameters normally are passed by value only
    - To modify the parameter value, need to use an object, and put the parameter values in as members of the object (https://www.w3schools.com/js/js_object_definition.asp)

```
var person = new Object();
person.firstName = "John";
person.lastName = "Doe";
person.age = 50;
person.eyeColor = "blue";

call function:  Func(person);
```

- **Decision statements**
  - if statement, switch statement, if/else if/else ← exactly the same as C++

- **Loops**
  - for loop, while loop, do-while loops ←exactly the same as C++
  - for/in, loops through the properties of an object

- **string** http://www.w3schools.com/jsref/jsref_obj_string.asp
  - difference between string and string object? no need for string object (slows down execution), string literal can be used to call all the string member functions.
  - string methods: indexOf, lastIndexOf, search, slice, substring, substr, → not used much in WebGL programming