Single line comments- //
Multi-line comments- /* comment here */
var- The most common variable. Can be reassigned but only accessed within a function. Variables defined with var move to the top when code is executed.
const- Cannot be reassigned and not accessible before they appear within the code.
Numbers- var num = 17;
Variables- var x;
Text (strings)- var a = "algebra";
Operations- var b = 1 + 4+ 9;
True or false statements- var c = true;
Notation Scientific-
var num1=1e10, num2=4e-9;
Constant numbers- const PI = 3.1416;
Basic Operators
+ Addition, - Subtraction, * Multiplication, / Division
(...) Grouping operator, % Modulus (remainder)

Increment/Decrement Numbers
i++ Postfix Increment numbers
i-- Postfix Decrement numbers
++i Prefix Increment numbers
--i Prefix Decrement numbers

```
var var1 = 5, var2 = 5;
alert(var1++); //5
alert(++var2); //6.0
alert(var1--); //6.0
alert(--var2); //5.0
```

Assigment and Operators

```
a += b    //a = a + b        Add then assign
a -= b    //a = a – b        Subtract then assign
a *= b    //a = a * b        Multiply then assign
a /= b    //a = a / b        Divide then assign
```

Comparison Operators
== Equal to, != Not equal, > Greater than, < Less than,
>= Greater than or equal to, <= Less than or equal to
Ternary Operator ?- (expression)? ifTrue: ifFalse

```
function getTF(isMember) {
 return (isMember ? 'True' : 'False');
}
alert(getTF(true)+"\n"+getTF(false)+"\n"+getTF(1));
```

Logical Operators
&& Logical and-

```
alert((1 > 0) && (4 > 0));
```

|| Logical or-

```
alert((1 > 3) || (4 > 3));
```

! Logical not-

```
alert(!(1 == 1));
```

Bitwise Operators
& AND statement, | OR statement, ~ NOT, ^ XOR,
<< Left shift, >> Right shift, >>> Zero fill right shift

```
alert("5&1="+(5&1) +
"5|1="+(5|1) +
"~5="+(~5) +
"5<<1="+(5<<1) +
"5^1="+(5^1) +
"5>>1="+(5>>1) +
"5>>>1="+5>>>1);
```

Functions

function namefunction(parameter1, parameter2, parameter3)
{ // what the function does }

Outputting Data

alert()- Output data in an alert box in the browser window.

```
alert("Welcome to Geogebra");
```

prompt()- Creates an dialogue for user input.

```
var value=Number(prompt("Enter value: ",1));
```

Global Functions

eval()- Evaluates JavaScript code represented as a string.

```
alert(eval("x = 2+3"));
var x = 10;
var y = 20;
var a = eval("x * y") ;
var b = eval("2 + 2");
var c = eval("x + 17");
var res = a + b + c;
alert(a+"+"+b+"+"+c+"="+res)
```

isFinite()- Determines whether a passed value is a finite number.

isNaN()- Determines whether a value is NaN or not.

Number()- Returns a number converted from its argument.

parseFloat()- Parses an argument and returns a floating-point number.

parseInt()- Parses its argument and returns an integer.

JAVASCRIPT LOOPS

for- The most common way to create a loop in JavaScript.

for (before loop; condition for loop; execute after loop) {
// what to do during the loop }

```
var str="";
for (var i = 0; i < 10; i++) {
   str=str+i + ": " + i*3 + "\n";
}
alert(str);

var sum = 0;
for (var i = 0; i < a.length; i++) {
   sum + = a[i];
}  // parsing an array
```

while- Sets up conditions under which a loop executes.

```
var i = 1;  // initialize
var str=""; // initialize
while (i < 100) { // enters the cycle if statement is true
   i *= 2;  // increment to avoid infinite loop
   str=str+i + ", ";  // output
}
alert(str);
```

do while- Similar to the while loop, however, it executes at least once and performs a check at the end to see if the condition is met to execute again.

```
var i = 1;  // initialize
var str=""; // initialize
do { // enters cycle at least once
   i *= 2; // increment to avoid infinite loop
   str=str+(i + ", "); // output
} while (i < 100) // repeats cycle if statement is true at the end
alert(str);
```

break- Use to stop and exit the cycle at certain conditions.

```
var i = 1;  // initialize
var str=""; // initialize
for (var i = 0; i < 10; i++) {
   if (i == 5) { break; }        // stops and exits the cycle
  str=str+(i + ", ");     // last output number is 4
}
alert(str);
```

continue- Skip parts of the cycle if certain conditions are met.

```
var i = 1;  // initialize
var str=""; // initialize
for (var i = 0; i < 10; i++) {
   if (i == 5) { continue; }      // skips the rest of the cycle
   str=str+(i + ", ");   // skips 5
}
alert(str);
```

IF - ELSE STATEMENTS

```
if (condition) {
// what to do if condition is met
} else {
// what to do if condition is not met
}
if(n%2){
alert("Number is odd.");
}else{
alert("Number is even.");
}
```

IF - ELSE NESTED STATEMENTS

```
var number=Number(prompt("Enter number",0));
if (number > 0) {
alert("Positive");
}else if (number < 0) {
alert("Negative");
}else{
alert("Zero");
}
```

SWITCH STATEMENTS

```
var n=333;
switch (n%4) {
 case 0:
   result = "1"; break;
 case 1:
   result = "i"; break;
 case 2:
    result = "-1"; break;
 case 3:
   result = "-i"; break;
  default:
   result="Null";
}
alert("i^"+n+" = "+result);
```

Objects Definition and Properties

Initialize Objects- `var obj3D = {name:"Sphere", radius:2};`

```
alert("Object="+obj3D.name+"\nradius="+obj3D.radius);
```

Initialize Objects-

```
var obj2D = { name:"Triangle", area:20, perimeter:24,
type:"rectangle"};
alert("Object=" + obj2D.name+" "+ obj2D.type + "\nArea=" +
obj2D.area);
```

Initialize Objects-

```
var polygon = new Object();
polygon.type = "quadrilateral";
polygon.property = "trapezoidal";
polygon.area = 50;
polygon.perimeter = 60;
alert(polygon.type+"\n"+
polygon.property +"\n"+
polygon.area+"\n"+
polygon.perimeter);
```

Initialize Functions in Objects-

```
var obj = {name: "Straight",
type: "Angle",
getName: function() { alert (this.name);}
};
obj.getName();
```

Initialize String Array- "

```
var course = ["Geometry","Trigonometry","Calculus"];
var course= new Array("Geometry", "Trigonometry", "Calculus");
var list1 = [2,3,5,8];
var list1 = new Array(2,3,5,8);
```

Declare Array Dynamic

```
var arr=new Array();
```

Array Methods

arr.length- Returns the number of elements in an array.
arr.concat()- Join several arrays into one.
arr.indexOf()- Returns the primitive value of the specified object.
arr.join()- Combine elements of an array into a single string and return the string.
arr.lastIndexOf()- Gives the last position at which a given element appears in an array.
arr.pop()- Removes the last element of an array.
arr.push()- Add a new element at the end.
arr.reverse()- Sort elements in descending order.
arr.shift()- Remove the first element of an array.
arr.slice()- Pulls a copy of a portion of an array into a new array.
arr.sort()- Sorts elements alphabetically.
arr.splice()- Adds elements in a specified way and position.
arr.toString()- Converts elements to strings.
arr.unshift()- Adds a new element to the beginning.
arr.valueOf()- Returns the first position at which a given element appears in an array.

Initialization of a 2d Array Static

```
var c[2][3] = {{1, 3, 0}, {-1, 5, 9}};
```

Set Value Element in 2D Array- `c[1][2]=4;`
Get Value Element in 2D Array- `num=c[2][1];`

STRINGS

```
var CEO="Markus Hohenwarter";
var str= "javascript";
alert(str.length);
```

length- the length of the string

Escape Characters

\' - Single quote, \" - Double quote, \\ - Backslash
\r - Carriage return, \0 - NUL character
\n - A new line characte

String Concatenation

```
var nside = 3;
alert('Triangles have ' + nside + ' sides.');
```

String Methods

charAt()- Returns a character at a specified position inside a string.
charCodeAt()- Gives you the unicode of character at that position.
concat()- Concatenates (joins) two or more strings into one.
fromCharCode()- Returns a string created from the specified. sequence of UTF-16 code units.
indexOf()- Provides the position of the first occurrence of a specified text within a string.
lastIndexOf()- Same as indexOf() but with the last occurrence, searching backwards.
match()- Retrieves the matches of a string against a search pattern.
replace()- Find and replace specified text in a string.
search()- Executes a search for a matching text and returns its position.
slice()- Extracts a section of a string and returns it as a new string.
split()- Splits a string object into an array of strings at a specified position.
substr()- Similar to slice() but extracts a substring depended on a specified number of characters.
substring()- Also similar to slice() but cannot accept negative indices.
toLowerCase()- Convert strings to lower case.
toUpperCase()- Convert strings to upper case.
valueOf()- Returns the primitive value of a string object.

NUMBERS AND MATH

Number Properties

Math.MAX_VALUE- The maximum numeric value representable in JavaScript.
Math.MIN_VALUE- Smallest positive numeric value representable in JavaScript.
Math.NaN- The "Not-a-Number" value.
Math.NEGATIVE_INFINITY- The negative Infinity value.
Math.POSITIVE_INFINITY- Positive Infinity value Number Methods.

```
alert("MAX_VALUE="+Math.MAX_VALUE+
"\nMIN_VALUE="+Math.MIN_VALUE+
"\nNaN="+Math.NaN+
"\nNEGATIVE_INFINITY="+Math.NEGATIVE_INFINITY+
"\nPOSITIVE_INFINITY="+Math.POSITIVE_INFINITY);
```

Number Functions

```
var num=Number("2");
```

num.toExponential()- Returns a string with a rounded number written as exponential notation.
num.toFixed()- Returns the string of a number with a specified number of decimals.
num.toPrecision()- String of a number written with a specified length.
num.toString()- Returns a number as a string.
num.valueOf()- Returns a number as a number.

Math Properties

Math.E- Euler's number.
Math.LN2- The natural logarithm of 2.
Math.LN10- Natural logarithm of 10.
Math.LOG2E- Base 2 logarithm of E.
Math.LOG10E- Base 10 logarithm of E.
Math.PI- The number PI.
Math.SQRT1_2- Square root of 1/2.
Math.SQRT2- The square root of 2.

Math Methods and Functions

Math.abs(x)- Returns the absolute (positive) value of x
Math.acos(x)- The arccosine of x, in radians
Math.asin(x)- Arcsine of x, in radians
Math.atan(x)- The arctangent of x as a numeric value
Math.atan2(y,x)- Arctangent of the quotient of its arguments
Math.ceil(x)- Value of x rounded up to its nearest integer.
Math.cos(x)- The cosine of x (x is in radians).
Math.exp(x)- Value of exponential.
Math.floor(x)- The value of x rounded down to its nearest integer.
Math.log(x)- The natural logarithm (base E) of x.
Math.max(x,y,z,...,n)- Returns the number with the highest value.

```
alert(Math.max(2,3,4));
```

Math.min(x,y,z, ...,n)- Same for the number with the lowest value.

```
alert(Math.min(-2,-3,-4));
```

Math.pow(x,y)- X to the power of y.
Math.random()- Returns a random number between 0 and 1.
Math.round(x)- The value of x rounded to its nearest integer.
Math.sin(x)- The sine of x (x is in radians).
Math.sqrt(x)- Square root of x.
Math.tan(x)- The tangent of an angle.

Errors

try- Lets you define a block of code to test for errors.
catch- Set up a block of code to execute in case of an error.
throw- Create custom error messages instead of the standard JavaScript errors.
finally- Lets you execute code, after try and catch, regardless of the result.

```
var x =Number(prompt("Enter x"));
try{
if(x == "") throw "empty"; //error cases
if(isNaN(x)) throw "not a number";
x = Number(x);
if(x > 10) throw "too high";
}catch(err) { //if there's an error
alert("Input is " + err); //output error
}finally{
alert("Done"); //executed regardless of the try / catch result
}

var ver=parseInt(ggbApplet.getVersion());
try{
if(ver== 5.0){
var A=prompt("Enter Matrix:","0,1,0,1,4; -5,4,2,6,15; -3,2,1,3,8; 2,-1,0,-2,-5; -1,2,0,0,0");
}else{
var A = "" + ggbApplet.getValueString("Matrix");
}}catch(err){
alert("Error");
}finally{
alert("Version: "+ver);
}
```

Setting Dates

var d = new Date();

Date()- Creates a new date object with the current date and time
Date(2017, 5, 21, 3, 23, 10, 0)- Create a custom date object. The numbers represent year, month, day, hour, minutes, seconds, milliseconds. You can omit anything you want except for year and month.
Date("2017-06-23")- Date declaration as a string Pulling Date and Time Values.
d.getDate()- Get the day of the month as a number (1-31)

d.getDay()- The weekday as a number (0-6) Sunday-0, Monday-1, Tuesday-2, Wednesday-3, Thursday, Friday-5, and Saturday-6.
d.getFullYear()- Year as a four-digit number (yyyy)
d.getHours()- Get the hour (0-23)
d.getMilliseconds()- The millisecond (0-999)
d.getMinutes()- Get the minute (0-59)
d.getMonth()- Month as a number (0-11)
d.getSeconds()- Get the second (0-59)
d.getTime()- Get the milliseconds since January 1, 1970
d.getUTCDate()- The day of the month in the specified date according to universal time.
Date.now()- Returns the number of milliseconds elapsed since January 1, 1970, 00:00:00 UTC.
Date.parse()- Parses a string representation of a date, and returns the number of milliseconds since January 1, 1970.

```
const unixTimeZero = Date.parse('01 Jan 1970 00:00:00 GMT');
const javaScriptRelease = Date.parse('04 Dec 1995 00:12:00 GMT');
alert("unixTimeZero="+ unixTimeZero+
"\nJavaScriptRelease="+javaScriptRelease);
```

d.setDate()- Set the day as a number (1-31)
d.setFullYear()- Sets the year (optionally month and day)
d.setHours()- Set the hour (0-23)
d.setMilliseconds()- Set milliseconds (0-999)
d.setMinutes()- Sets the minutes (0-59)
d.setMonth()- Set the month (0-11)
d.setSeconds()- Sets the seconds (0-59)
d.setTime()- Set the time (milliseconds since January 1, 1970)
d.setUTCDate()- Sets the day of the month for a specified date according to universal time.

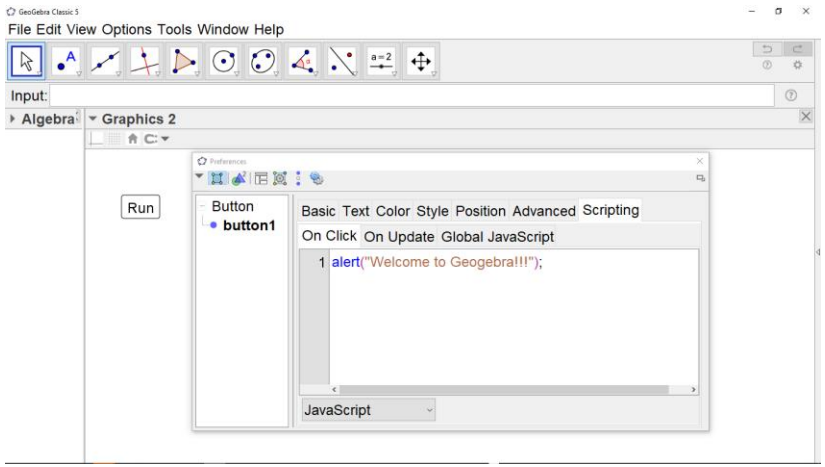
```
var d = new Date();
d.setFullYear(2008, 05, 15);
alert(d.getFullYear()+"-"+d.getMonth()+"-"+d.getDate());
```

ggbAppletMethods

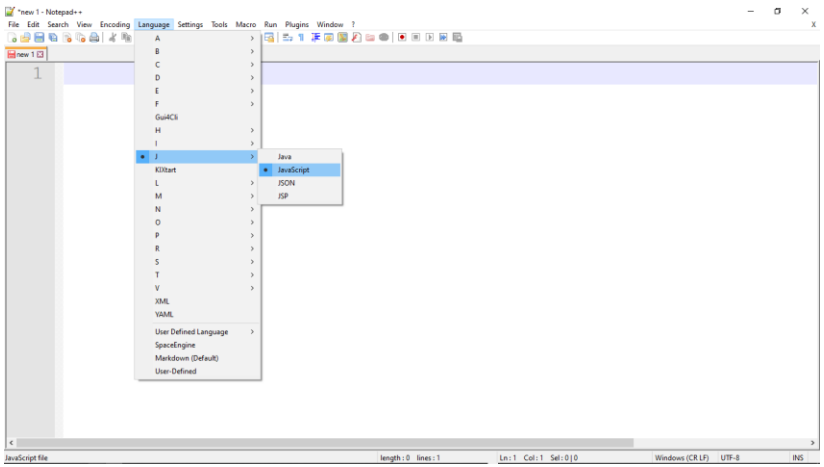
```
var n=ggbApplet.getValue("n");
ggbApplet.setValue("a",1);
ggbApplet.evalCommand("l1={}");
ggbApplet.setListValue("l1", i, Math.random() * (vmax - vmin) +
vmin);
```

Notepad++ is a text and source code editor to JavaScript.
https://notepad-plus-plus.org/downloads/
Use Notepad ++ to create and edit the Geogebra JavaScript code in Language-> J-> JavaScript and after finishing the code copy the code in Geogebra JavaScript and compile the code in some Geogebra object as a button to see if there are errors in the code or if the code works.

GeoGebra Scripting
https://wiki.geogebra.org/en/Scripting
Reference:GeoGebra Apps API
https://wik.geogebra.org/en/Reference:GeoGebra_Apps_API
JavaScript Tag Search in GeoGebra Web
https://www.geogebra.org/search/tags:JavaScript
Code Snipets of Javascript in Geogebra
https://ww.geogebra.org/m/paxjwmxh



Geogebra Scripting JavaScript



Notepad++ Source Code Editor JavaScript