Javascript quick reference:

2 things to remember:

1. Always put a ; at the end of each line if it doesn’t have a { or a }
2. Put your code between <script> </script> tags

Variable declarations:

var a = 12;

var name = “Alex”;

Operations:

a == b --> true if a equals b

a != b --> true if a is not equal to b

a < b --> true if a is STRICTLY inferior to b

a > b --> true if a is STRICTLY superior to b

a <= b --> true if a is inferior or equal to b

a > b --> true if a is superior or equal to b

a + b

a - b

a * b (a 'times' b)

a / b (a 'divided by' b)

Conditionals:

```
if(condition) {
//Do this
else { //condition is false
//Do this instead
}
```

Loops:

```
while(condition) {
//Do this until the condition is false
}
```

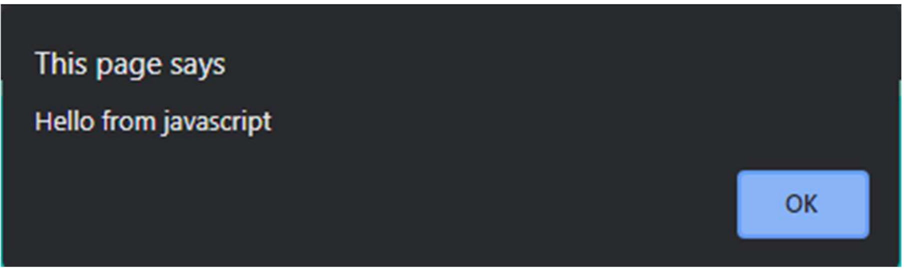
Ex:

```
var i = 0;
while(i < 5) {
//Do this
}
```

- ➔ Does whatever is in the {} 5 times

Useful functions:

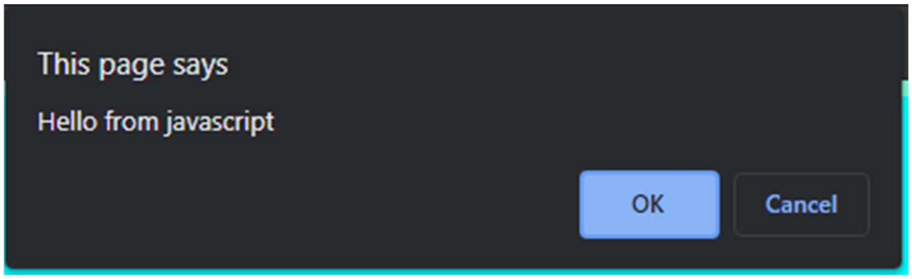
alert(message) -->



This page says

Hello from javascript

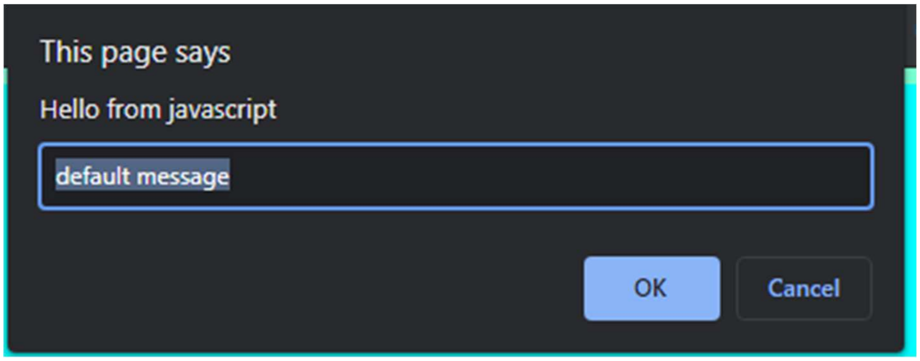
confirm(message)


This page says

Hello from javascript

returns true if the user clicks true and false if the user clicks cancel

prompt(question,default)


This page says

Hello from javascript

default message

returns the content of the input box after the user presses OK or false if the user presses cancel, by default is equal to the second argument (if you put one)

Styling your page:

Example:

```
<img src='pikachu.png' id='pikachu'>

…

<script>

var pikachu =  document.getElementById('pikachu');

pikachu.style.border = "2px red solid";

//Sets the border to 2px red solid

//Can be used on the border or any other style argument
```

</script>

Canvas:

```
<canvas height='500' width='500' id='mycanvas'></canvas>
```

➔ Creates a canvas of 500px by 500px with the ID ‘mycanvas’

```
var canvas = document.getElementById("mycanvas");

var context = canvas.getContext("2d");
```

➔ Include these 2 lines to start working on your canvas

The first one gets the canvas from the document

The second one declares it as a 2d surface (as opposed to 3d)

context.moveTo(x,y);

➔ Moves the pen to coordinates x,y (remember, in programming, if y goes up, the pen goes down → 0,0 is your top left corner)

context.lineTo(x,y);

➔ Traces a line from the current position of the pen to the x,y coordinates

context.stroke();

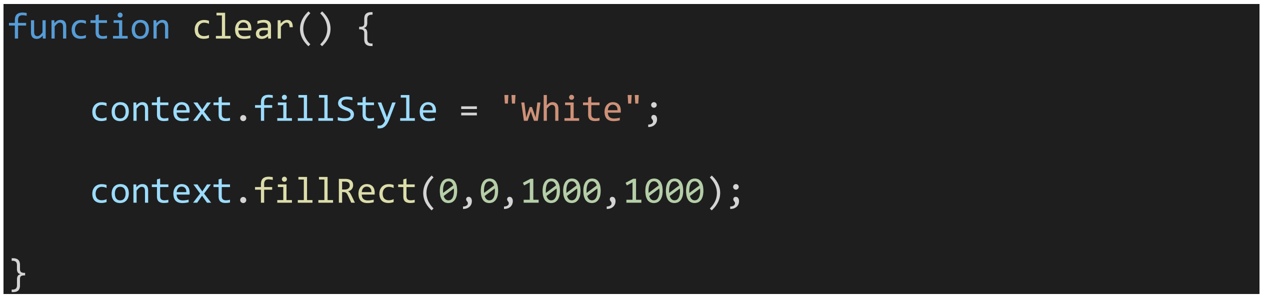
➔ Applies all changes

context.fillStyle = “color”;

➔ Changes the filling style to the color specified

➔ context.strokeStyle can be used to change the color of the line context.fillRect(beginX, beginY, endX, endY);

➔ Fill a rectangle between 2 points (beginX, beginY) and (endX, endY)



```
function clear() {

    context.fillStyle = "white";

    context.fillRect(0,0,1000,1000);

}
```

➔ Use this code to clear the screen (replace 1000 with the width and height of your canvas of course)

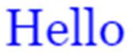
context.font = “25px Georgia”;

➔ Sets the font to Georgia and the size of it to 25px

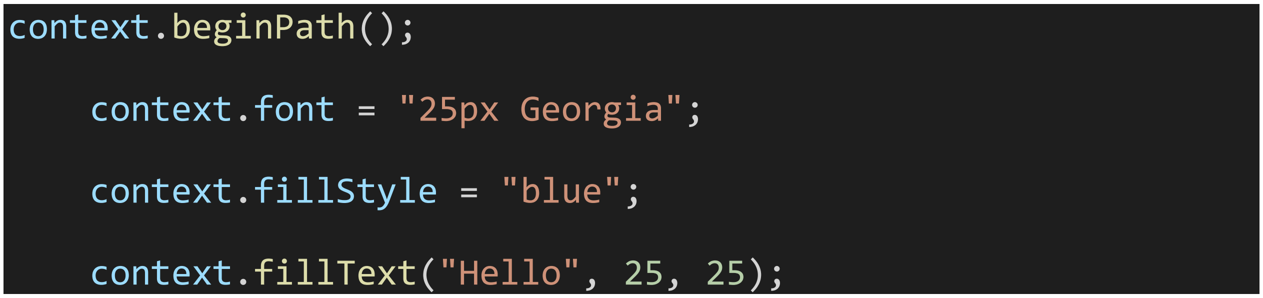
➔ It will be used in the following 2 functions

context.fillText(message, x, y);

➔ Draws the full text at the specified coordinates



Hello

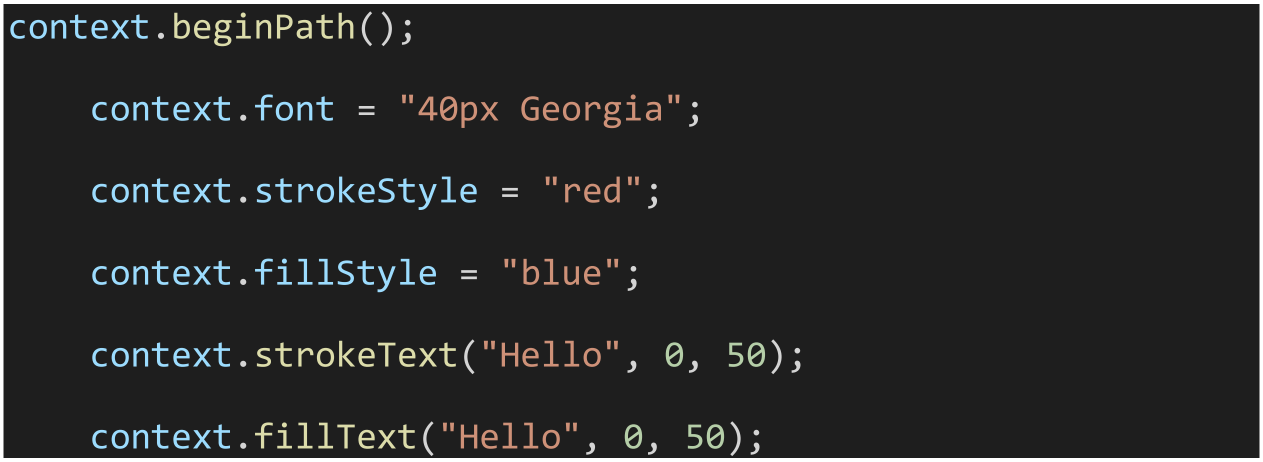


```
context.beginPath();

    context.font = "25px Georgia";

    context.fillStyle = "blue";

    context.fillText("Hello", 25, 25);
```

Note: context.beginPath() resets the pen position and color, this should be used when you change color

context.strokeText(message, x, y);

- ➔ Draws the outline of the characters (use strokeStyle to change the color you want)
  Associated with fillText can give the following:



```
context.beginPath();
    context.font = "40px Georgia";
    context.strokeStyle = "red";
    context.fillStyle = "blue";
    context.strokeText("Hello", 0, 50);
    context.fillText("Hello", 0, 50);
```

Note: the bottom left corner of the first letter corresponds to the x,y point