LET'S CODE TOGETHER IN JAVASCRIPT

ANITA BORG CELEBRATION MUNICH - A JAVASCRIPT BEGINNER'S WORKSHOP

Jessica Jordan / @jjordan_dev

WHAT IS JAVASCRIPT?

- language on the web
- cross-platform scripting language that works in your browser (e.g. Chrome)
- small and lightweight
- gain control of the elements and manipulate them

WHAT IS IT USED FOR?

Dynamic interactions on any website

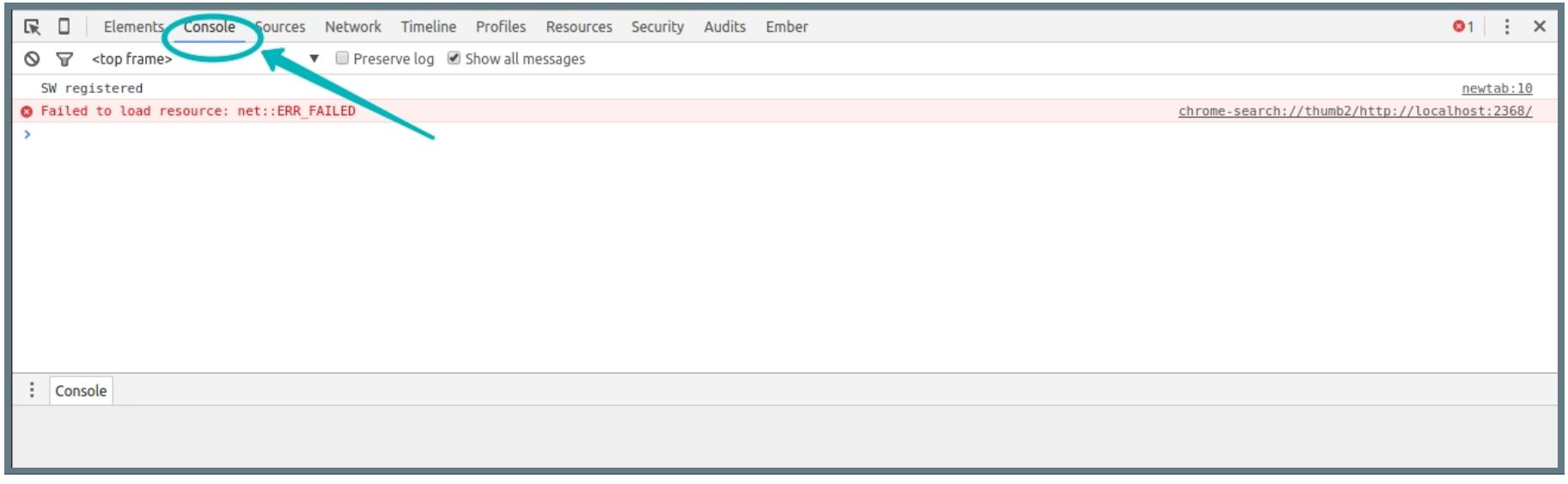
Lists, Search Fields, Google Maps, Animations and many more

USING YOUR BROWSER CONSOLE

Open your index.html file in the browser

Right-click on your browser window and select **Inspect**

Click on Console



YOUR FIRST LINE OF CODE

Share your thoughts with your console

```
console.log("Hello World!");
```

Let your console tell you something about the weather!

SOLUTIONS

This is how it could have looked like

```
console.log("The weather in Munich is sunny today.");
```

USE VARIABLES TO SAVE VALUES FOR LATER

var

```
var weather = "sunny";
console.log("The weather in Munich is " + weather + " today");
```

What will happen here?

```
var weather = "sunny";
console.log("The weather in Munich is " + weather + " today");
weather = "rainy";
console.log("The weather in Munich is " + weather + " today");
```

Try it out for yourself!

SOLUTIONS

This is how it could have looked like

```
var weather = "sunny";
console.log("The weather in Munich is " + weather + " today");
// The weather in Munich is sunny today
weather = "rainy";
console.log("The weather in Munich is " + weather + " today");
// The weather in Munich is rainy today
```

LOOPS

Loops help you to automate program instructions that have to be repeated in the same or in a similar way. Have a look at this example!

```
for (var i = 0; i < 3; i += 1){
// do something for 3 times
}
```

PRINT A LIST OF NUMBERS INTO YOUR CONSOLE

Print the list of numbers from 0 to 30 to your console

```
for (// insert the definition for your for loop here){
console.log(i);
}
```

SOLUTIONS

Print the list of numbers from 0 to 30 to your console

```
for (var i = 0; i < 31; i += 1){
console.log(i);
}
// 0
// 1
// 2
// 3
// ..
// 30
```

CONDITIONS

Sometimes you want to execute code only in specific situations

```
var weather = "sunny";
if (weather === "sunny"){
console.log("Let's go skating!");
}
```

And sometimes you even want to have a fallback

```
var weather = "rainy";
if (weather === "sunny"){
console.log("Let's go skating!");
}
else {
console.log("Let's watch a movie!");
}
// Let's watch a movie!
```

Imagine how this scenario could turn out: Print either "Let's celebrate your birthday!" or "Let's not celebrate your birthday!" to the console depending on the value of the variable hasBirthday

```
var hasBirthday = true;
// if (....
```

SOLUTIONS

This is how it could have looked like:

```
var hasBirthday = true;
if (hasBirthday === true){
console.log("Let's celebrate your birthday!");
}
else {
console.log("Let's not celebrate your birthday!");
}
// Let's celebrate your birthday!
```

EXTRAS

Extra task "FizzBuzz": Print all the numbers from 1 to 100 into your console. If the number is dividable by 3 print "Fizz", if it is dividable by 5 print "Buzz" and "FizzBuzz" if it is dividable by both numbers instead. Hint: Check for division by numbers with the % operator:

```
if (num % 3 === 0){
// do if num is dividable by 0
}
```

FUNCTIONS

Help you to seperate program instructions into seperate bits

```
myFunction(){
// Your instructions go here
}
```

You can then execute them anytime in your program by calling the name of your function

```
myFunction();
```

FUNCTIONS PRODUCE VALUES

You can use the **return value** of functions to set new variables

```
function myFunction (){
  return 3;
}
var number = myFunction();
console.log(number); // -> 3
```

FUNCTIONS ACCEPT PARAMETERS

You can provide values called **parameters** to your function and interact with those

```
function doubleMe (paramNumber) {
  return paramNumber * 2;
}
```

And later on use this with different values which you put in

```
var myNumber = 3;
var myDoubledNumber = doubleMe(myNumber);
console.log(myDoubledNumber); // -> 6
```

EXERCISE

Write a function called **multiply()**that returns the result of the multiplication of two input parameters! Hint: You can seperate several parameters by comma

```
function multiply(a,b){
  return  //....
}
```

SOLUTIONS

You can create the function with two input parameters

```
function multiply(a,b){
  return  a * b;
}
```

You can use this function now to create results for any multiplication

```
function multiply(a,b){
  return  a * b;
}
var firstNum = 3;
var secondNum = 4;
var product = multiply(firstNum, secondNum);
console.log(product); // -> 12
```

LET'S DRAW!

Check your index.html from the HTML introduction to get started and open the file drawing.js in your working example folder.

We want to draw on the canvas element that you already prepared in your **index.html** file

```
<canvas id="my-canvas" width="500" height="300"></canvas>
```

...which will be selected as the drawing area by the following code in **functions.js**

```
var canvas = document.getElementById('my-canvas');
var ctx = canvas.getContext('2d');
```

After this setup you can now use the **circle()** function we prepared for you to draw on your canvas in the file **drawing.js**

```
function drawOnCanvas(circle, randomColor){
  // drawing code here
  //    circle(x-coordinate,y-coordinate);
  circle(60,60);
}
```

You can add another circle next to it:

```
function drawOnCanvas(circle){
  // drawing code here
      circle(60,60);
      circle(120,60);
}
```

Try creating 5 circles in a row the using one of the loops we got familiar with!

```
function drawOnCanvas(circle){
  // drawing code here

}
```

SOLUTIONS

Try creating 5 circles in a row using one of the loops we got familiar with!

```
function drawOnCanvas(circle){
  // drawing code here
    for (var i = 1; i < 6; i += 1){
      circle(60 * i, 60);
    }
}
```

EXTRAS

How about 5 circles in a row and 4 in a column? Hint: You may use another counting variable as well!

```
function drawOnCanvas(circle){
  // drawing code here
    for (var i = 1; i < 6; i += 1){
    // ....
    }
}
```

Color your circles!

```
function drawOnCanvas(circle, randomColor){
  // drawing code here
  // circle function takes an optional parameter for the color
    circle(60,60,"#ff00ff");
  // you can even create random colors
   var myRandomColor = randomColor();
}
```

LET'S SEE WHAT ELSE YOU CAN DO WITH JAVASCRIPT AND VISUALIZATIONS!

- Visualization with D3.js

WHAT IS D3.JS?

Javascript library for visualizing data using HTML, CSS and SVG

HOW DO I USE D3.JS ON MY WEBSITE?

You can easily plug D3 into your website using the code from the project site:

```
<script src="//d3js.org/d3.v3.min.js" charset="utf-8"></script>
```

This makes your D3.js toolbox usable for your website straight away!

```
d3.selectAll("p").style("color", "white");
```

A LIVE EXAMPLE OF D3.JS

See the chart demo

See more examples of D3.js

Sunburst Example in D3.js

Geographic Bounding Boxes in D3.js