Cheat Sheet – Functions (Advanced)

Closures

Closure are also called anonymous functions. They have that name for a reason: They are not assigned a name.
Why would you use such functions?
A very important difference, is the different scope: The closure uses the scope of the function in which it is used, whereas the named function would not have access to that scope.

Here's an example:

```
function calculaterCreator(number) {
  return function() {
    console.log(number * 2); // has access to number
  };
}
```

Learn more about closures here:
https://developer.mozilla.org/en/docs/Web/JavaScript/Closures

IIFEs (Immediately invoked function expression)

IIFEs are functions which are invoked immediately when running the script.

```
(function () {
    console.log('Immediately invoked!');
})();
```

Why would you do that and not simply write the code without a function wrapping it?
The reason is the JavaScript scope. Remember, that everything outside of a function belongs to the global scope. With IIFEs you can create a local scope (and therefore some privacy) and still run the code immediately. Sometimes this might be the behavior you want.

Function Properties & Methods

Remember, that JavaScript Functions are of type Function? This type also has some properties and methods you can access.
If you want to find out which arguments were passed to a function, you may use the *arguments* keyword inside that function.

```
function myFunction(message) {

      console.log(arguments);

};
```

You can also find out the number of arguments a function expected as well as its name:

```
console.log(myFunction.length); // 1

console.log(myFunction.name); // 'myFunction'
```

A very important method, is the bind() method, which allows you to set to what *this* should refer to inside the function call.
Consider this example:

```
var name = 'Anna';
var obj = {
    name: 'Max'
};

function printName() {
    console.log(this.name);

}

printName(); // Anna
printName.bind(obj)(); // Max
```

Learn more here: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Function