Closure

Overview

```
/*
  - Returning functions from functions
  - Closure
*/
```

I am a string

Returning values from functions

```
1   /* we know that functions can return values */
2
3   function returnsAString() {
4     return 'I am a string';
5   }
6
7   let returnedValue = returnsAString();
8
9   console.log(returnedValue);
10
11
12
13
14
```

Returning values from functions

```
1   /* in the previous unit, we learned that functions are like any other
2      value in JS */
3
4   /* so if functions can return values, and functions are values… */
5
6   /* …functions can return other functions! */
7
8
9
10
11
12
13
14
```

Returning functions from functions

```
function
[Function: greetMe]
Hi!
```

```
1   function greetMe() {
2     console.log('Hi!');
3   }
4
5   function getGreeter() {
6     return greetMe; // note: we're returning the function without calling it
7   }
8
9   let greeter = getGreeter();
10
11  console.log(typeof greeter);
12  console.log(greeter);
13  greeter();
14
```

Returning functions from functions

```
Hi!
undefined
undefined
TypeError: greeter is not a function
```

```
1   function greetMe() {
2     console.log('Hi!');
3   }
4
5   function getGreeter() {
6     return greetMe(); // what if we did invoke it?
7   }
8
9   let greeter = getGreeter();
10
11  console.log(typeof greeter);
12  console.log(greeter);
13  greeter();
14
```

Hi!

Returning functions from functions

```
1   function getGreeter() {
2     // we can declare a new inner function and then return it
3     function greetMe() {
4       console.log('Hi!');
5     }
6
7     return greetMe;
8   }
9
10  let greeter = getGreeter();
11  greeter();
12
13
14
```

Hi!

Returning functions from functions

```
1   function getGreeter() {
2     // we can return anonymous function, too
3     return () => {
4       console.log('Hi!');
5     };
6   }
7
8   let greeter = getGreeter();
9   greeter();
10
11
12
13
14
```

Returning functions from functions

```
1   function getGreeter() {
2     // what if our returned function takes a parameter?
3     return function(name) {
4       console.log('Hi', name);
5     };
6   }
7
8   let greeter = getGreeter();
9
10  greeter('Marie');
11  greeter('Rosalind');
12  greeter();
13
14
```

```
Hi Marie
Hi Rosalind
Hi undefined
```

Closure: scope refresher

```
1   /* before we talk about closure, let's review how scope works with nested
2      functions */
3
4   /* inner functions can access variables defined in outer functions */
5
6   function greeter() {
7     let name = 'Marie';
8     let saysHi = function() {
9       console.log('Hi', name);
10    };
11    saysHi();
12  }
13
14  greeter();
```

Closure: definition

```
/* closure is the fact that an inner function can STILL access values
   defined in the outer function even after the outer function is finished
   running! */
```

Closure: examples

```
1   function getGreeter() {
2     console.log('getGreeter is running');
3
4     let name = 'Marie';
5
6     console.log('getGreeter is finishing');
7
8     return function() {
9       console.log('Hi', name);
10    };
11  }
12
13  let greeter = getGreeter();
14  greeter();
```

```
getGreeter is running
getGreeter is finishing
Hi Marie
```

Closure: examples

```
1   // what if getGreeter take a parameter?
2   function getGreeter(name) {
3     return function() {
4       console.log('Hi', name);
5     };
6   }
7
8   let greeter = getGreeter('Rosalind');
9   greeter();
10
11
12
13
14
```

Closure: examples

```
1   // what if both functions take parameters?
2   function getGreeter(name1) {
3     return function(name2) {
4       console.log('Hi ' + name1 + ', meet ' + name2);
5     };
6   }
7
8   let greeter = getGreeter('Marie')('Rosalind');
9
10
11
12
13
14
```

Closure: examples

```
Hi Marie, meet Dorothy
Hi Marie, meet Albert
Hi Rosalind, meet Barbara
Hi Rosalind, meet Isaac
```

```
1   function getGreeter(name1) {
2     return function(name2) {
3       console.log('Hi ' + name1 + ', meet ' + name2);
4     };
5   }
6
7   let introduceMarieTo = getGreeter('Marie');
8   let introduceRosalindTo = getGreeter('Rosalind');
9
10  introduceMarieTo('Dorothy');
11  introduceMarieTo('Albert');
12
13  introduceRosalindTo('Barbara');
14  introduceRosalindTo('Isaac');
```

Recap

```
/*
  - Returning functions from functions
  - Closure
*/
```