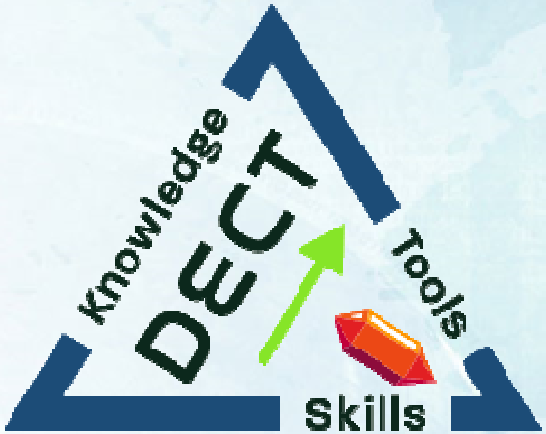
Web Engineering Front-end Pt. 3

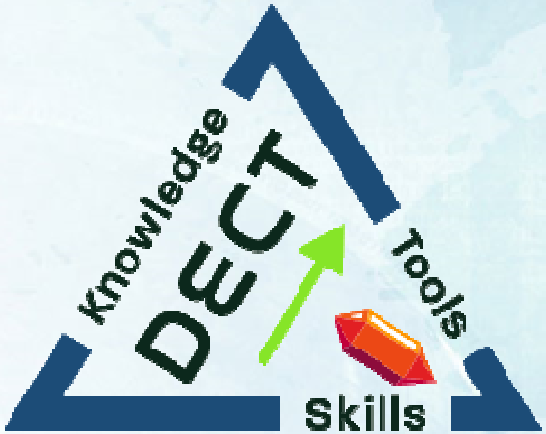
10. JavaScript: Functions

Presented by Krystal Educational Platform



Contents

10.1 What are functions?



What are functions?

Similar to methods, functions are blocks of code used to perform specific functions. The difference is that functions are not tied to an object.

Uses of functions

The main benefit of using functions is reusability. Once a function is defined, it can be invoked over and over again.

Uses of functions

Imagine if we wanted to convert the properties of objects into arrays.

No functions

```
var array1 = []
  for (i in object1) {
    array1.push(object1[i])
  }
  return array1;
}
var array2 = []
  for (i in object2) {
    array2.push(object2[i])
  }
  return array2;
}
var array3 = []
  for (i in object3) {
    array3.push(object3[i])
  }
  return array3;
}
```

Using functions


```
function arrayify(obj) {
  var array = []
  for (i in obj) {
    array.push(obj[i])
  }
  return array;
}

arrayify(object1);
arrayify(object2);
arrayify(object3);
```

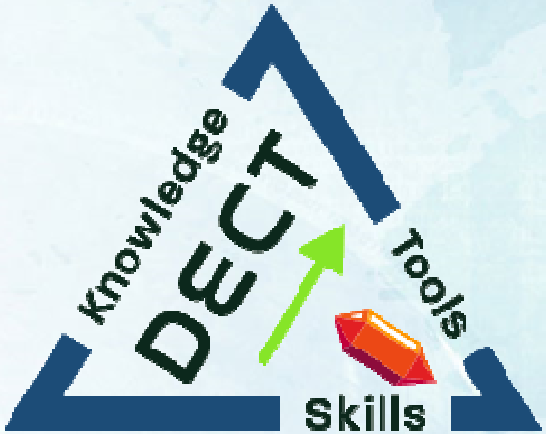
Uses of functions

The previous example may not seem to have saved a lot of lines, but when the function is more complex it can really make a difference.

Activity: Function ideas

Can you think of some commonly repeated code that you would want to write as a function?

10.2 Writing Functions



Function syntax

```
function name(parameters) {
  // code to be executed when invoked
}
```

Function example

```
function greet(name) {
   return ("Hello, "+name+"!");
}
```

function keyword

The function keyword is used to declare function objects.

Function name

The name of the function is used when trying to invoke it. A function without a name is an anonymous function.

Parameters

A function's parameters are variables to be used within the function when it is invoked. A function can have multiple parameters.

Multiple parameters example

```
function greet2(name1, name2) {
   return ("Hi, "+name1+" and "+name2+"!");
}
```

Default parameters

You can set a default value for parameters that will be used if no arguments are passed when the function is invoked.

Default parameters

```
function message(name = “anonymous”, msg = “Default message”) {
   return (name+”: “+message);
}

console.log(message(“James”, “What’s up?”));
// James: What’s up?

console.log(message(undefined, “Hello, there.”));
// anonymous: Hello, there.

console.log(message());
// anonymous: Default message
```

Local variables

Variables declared in a function cannot be accessed outside of it.

Local variables example

```
// x is not defined

function aFunction() {
   var x = 0;
   // x equals 0
}

// x is not defined
```

return keyword

By default, a function will output undefined. The return keyword is used to specify what the result is when the function is invoked.

return keyword example

```
function greet(name) {
   return (“Hello, ”+name+”!”);
}
```

Creating methods

Object methods can be created in a similar way to functions. The difference is that is placed within the object constructor.

Method example

```
var dog = {
   name: “Marley”,
   bark: function () {
      return (“Woof!”);
};
```

this keyword

this in a method refers to the object that it belongs to. It can be used to access the parameters of an object within a method.

this keyword example


```
var person = {
   firstName: “Joe”,
   lastName: “Schmoe”,
   introduce: function() {
      return (“I’m “+this.firstName+” “+this.lastName+”.”);
   }
}
```

Method shorthand

As of ES5, the function() expression can be omitted. As a result, the previous example can be shortened to...

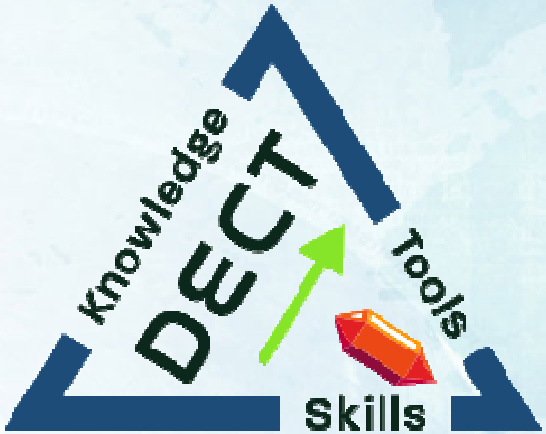
Method shorthand example

```
var person = {
   firstName: “Joe”,
   lastName: “Schmoe”,
   introduce() { // VS introduce: function()
      return (“I’m “+this.firstName+” “+this.lastName+”.”)
   }
}
```

Practice: Creating functions

Try creating some of the functions you came up with in the previous activity.

10.1 Invoking Functions



Invoking functions

Just like with methods, you can call a function using the () operators.

Invoking functions example

```
function hello() {
   return “Hello!”
}

console.log(hello);

// Hello!
```

Function object

Omitting the () will instead return the function object itself.

Function object example

```
function hello() {
   return “Hello!”
}

console.log(hello);

/*
function hello() {
   return “Hello!”
}
*/
```

Arguments

Any arguments (parameters) for the function should be placed between the (), separated by commas.

Arguments example

```
function greet2(name1, name2) {
   return (“Hi, “+name1+” and “+name2+”!”);
}

console.log(greet2(“Joe”, “Megan”));
// Hi, Joe and Megan!
```

Functions as variables

You can use the results of a function as the value of a variable.

Functions as variables example


```
function merge(num1, num2) {
   return (String(num1)+String(num2));
}

var x = merge(12, 24)
x; // 1224
```

Hoisting

JavaScript will move all declaration statements to the top of the script, which is why you can seem to use functions before they are even created.

Hoisting example

```
console.log(greet2(“Joe”, “Megan”));
// Hi, Joe and Megan!


function greet2(name1, name2) {
   return (“Hi, “+name1+” and “+name2+”!”);
}
```

Hoisting

Even if it isn't necessary, declaring functions before using them in the code is the best practice for readability.

Recursion

Recursion is when a function calls itself, which can be a more efficient way of iteration.

Recursion example

```
function countdown(x) {
  if (x > 0) {
    console.log(x);
    countdown(x-1);
}

countdown(5);
// 5, 4, 3, 2, 1
```

Recursion

Like iteration, you need to be careful not to accidentally create infinite loops.

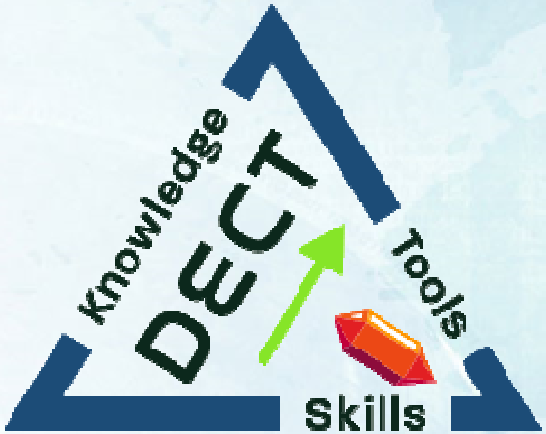
Recursion infinite loop example

```
function countdown(x) {
  if (x > 0) {
    console.log(x);
    countdown(x);
}
```

Practice: Invoking functions

Try using some of the functions you created in the previous activity. Are they working as you intended them to? Also try writing your own recursive functions.

The End



References

Reference 1: Code Academy Methods and Functions https://www.codecademy.com/articles/fwd-js-methods-functions

Reference 2: W3 Schools JavaScript Tutorial https://www.w3schools.com/js/default.asp

Reference 3: MDN JavaScript reference https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference