How To “Function()”

What it is, what it does, how to use them

Agenda

TIME	CONTENT
5 min	Intro: assumptions, definitions, and function basics
2 min	Look at code & verbally annotate (codepen)
5 min	Whiteboard activity (pseudocode - how to make coffee)
2 min	Review Best Practices

Assumptions:

Today we are going to talk about functions

We'll be using JAVASCRIPT

- You have made a cup of coffee
- You *may* be familiar with concept of 'variables'
- You *may* have completed *some* prep work or online-code exercises (online tutorials, GA workshop etc).

Functions - What are they?

Key Ideas

Functions are about action.

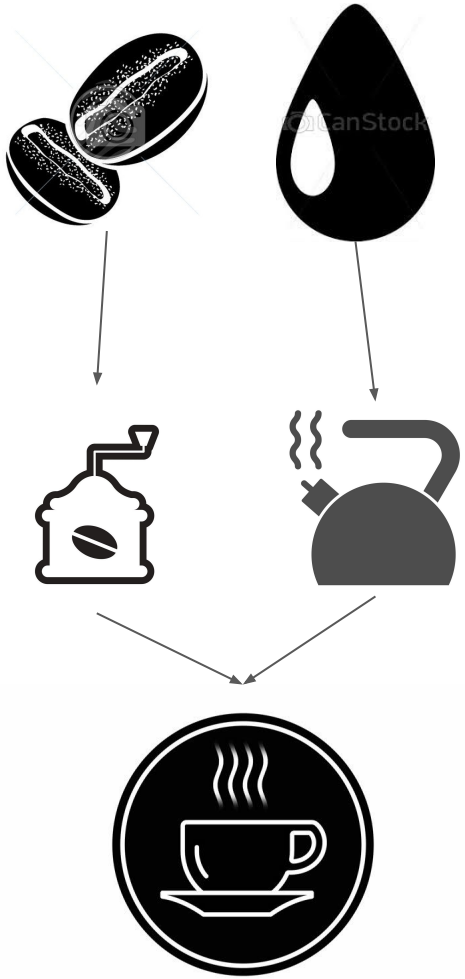
A function is a type of
object that does something

Key Steps:

INPUT:
arguments go in, think of them like ingredients the function can expect.

ACTIONS:
Instructions about what to do with **arguments** are interpreted.

OUTPUT :
A **value** is returned



Declaring a function: Function Definition

```
function theNameOfYourFunction ( string1, string2, …) {

    const aNewString = string1 + string2;

    // Step 1. assigning the arguments to a new variable
    // Step n.  the process of the code may continue...

    return aNewString;

    // Step 2. a new value is returned that “concatenates” the two
    argument strings into a new string whenever the function is called.

}
```

__ = **keyword**
a special word designated by JS to “construct” a function.

__ = **variable / identifier**
a string of characters that substitute for a set of values

__ = **arguments**
placeholders for values inside the function

__ = **operators**
how the function acts on the parameters values.

Declaring a function: “Function Expression”

```
const theNameOfYourFunction = function ( string1, string2, …) {
        const aNewString = string1 + string2;
        return aNewString
}
```

A function *expression* is a way to *store* the values returned by a function in a variable. This variable can then be ‘called’ by another function.

Calling a function

```
function theNameOfYourFunction (arg1, arg2){
        return console.log(arg1+" "+arg2);
}

const aNewWord = theNameOfYourFunction ( );
```

OUTPUT >> “undefined undefined”

// the functions value (undefined) is stored in a variable newWord

```
aNewWord("a duck,"  , "dill pickle");
```

OUTPUT >> “a duck, dill pickle”

// we are calling, or invoking by the function, by using the variable name and passing in values as “parameters”

SAMPLE CODE USED IN PREPARATION: [codepenIO]

Whiteboard Pseudocode Activity

As a group/s we will use natural language to make a cup of coffee.

Questions:

What are your arguments / ingredients?

What actions are needed to prepare the coffee?

What actions and tools are required to brew the coffee?

What is the expected outcome?

Functions - What are they?

THE BIG PICTURE

Functions groups instructions the computer/browser can understand.

Functions are built with keywords, statements and operations.

Function are **objects** *that is built to do something with an input.*

Calling a functions allow us to repeats these instructions without writing new code each time.

Functions - A Review

Declaring the function means to **Name** the function and define its properties.

INPUT:

arguments go in:

Variables are declared and will be used when the code is run.

ACTIONS & PROCESSES :

a block of instructions inside { } tells the computer how to act on your arguments.

Possible choices:
Operations (+ - * / %)
Logic Statements (if else)
Loops (for / while)

The code is interpreted line by line and your INPUT is read, evaluated, and modified according to the code inside the { }

OUTPUT:
a new value is **“returned”**

Note: The declared function is can only run when you “call” the function.

When declaring a function:

1. **Choose a good name**: functions do something so try to use words that describe the process of the function. (get, set, find, add, decrease, etc)

2. Arguments are placeholder names, they should be topical and concrete.

3. Try to keep your functions as simple as possible. **A good function does one thing and returns a predictable result.**

When calling a function:

More advanced things to remember:

1. **Code runs synchronously** - code runs line by line, one statement at a time. A later function cannot execute until the previous function has completed its process.

2. **Function definitions** may be defined anywhere in the code, even after the code is called (!)

3. **Function Expressions** should only be declared <u>before</u> the code is called. Otherwise an undefined variable will be called instead.