# Functions

Contents

[Functions 1](#_Toc25748421)

[Introduction 1](#_Toc25748422)

[Naming Conventions 2](#_Toc25748423)

[Calling Functions 2](#_Toc25748424)

[Using parameters 2](#_Toc25748425)

[Arguments vs Parameters 2](#_Toc25748426)

[Syntax 2](#_Toc25748427)

[Return Statements 3](#_Toc25748428)

## Introduction

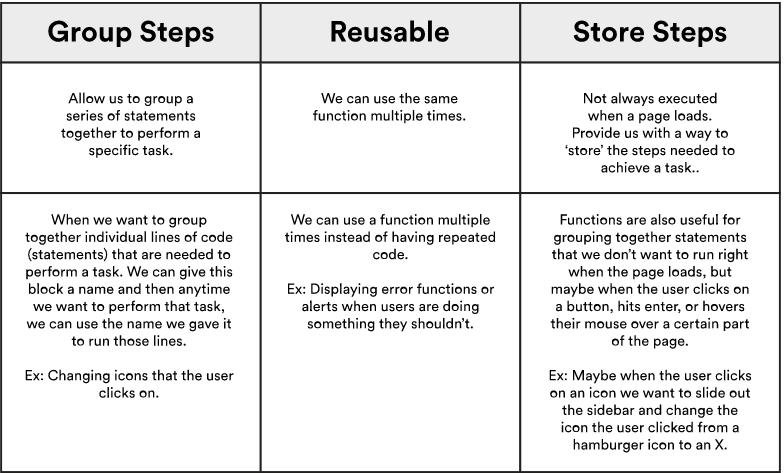
* If a statement (or statements) is to be re-used again and again, it should be represented as a function.
* Functions are critical components of programming because they allow us to execute a key tenet of engineering: Don’t Repeat Yourself DRY code

We avoid repetition for maintainability and performance.

* Maintainability: Easier to make changes and debug, because we only need to make a change at a single place, instead of, say, 10 places where that code is repeated.
* Performance: Longer scripys, take up more memory and more time to load.

##\*\* Functions may or may not include parameters.

Functions are useful because it allows us to:



## Naming Conventions

* camelCase
* no nouns
* describe what is does, use verbs!

Example: calculateArea, finalHeight,

## Calling Functions

We invoke/call/execute a function. The code in a function will only run when it is called.

Functions are called by appending () to the functionName,

var myFunction = function (x) {

var area = function (width, length) {

### Using parameters

Let’s say you have a function which calculates the total price of coffee (which is: base price of coffee \* (1 + sales tax rate)).

Now you want to calculate the total price of swiss roll.

We can write another function to do that, but the code will be very similar to the function which calculates the total price of coffee.

Since we should keep our code DRY(Don’t repeat yourself), we can make use of parameters.

We can write a function which calculates the total price, with a parameter which accepts the base price of the food item.

### Arguments vs Parameters

functionName(argument1,argument2);

When you call a function, you provide values for the parameters. These values are called arguments.

Arguments are actual values passed into the function when the function is called.

## Syntax

function functionName (parameter1,parameter2) {

//code here;

}

*This is a function declaration. It is loaded before any code is run.*

OR

var <name> = function (parameter1,parameter2) {

//code here;

}

*This is a function expression. It is loaded only when the interpreter reaches the line of code*

\*\* Don’t forget to add the word “function” before functionName!!

This is necessary for JavaScript but not in java

## Return Statements

We use return statements when we want the function to return a value.

This can be used to update a variable within a function.

const totalSum = add(3, 4);

// => 7

// The variable totalSum will now hold the value 7.

\*\* We can also use return; *by itself* as a way to exit the function and prevent any code after it from running.

|  |  |
| --- | --- |
| Javascript Syntax | Java Syntax |
| function add(x, y) {  return x + y;  } | public double getAverageLength(Shape s) {  double avg = getPerimeter(s) / getNumPoints(s);  return avg;  } |
| No need to define datatype | Need to define datatype, |