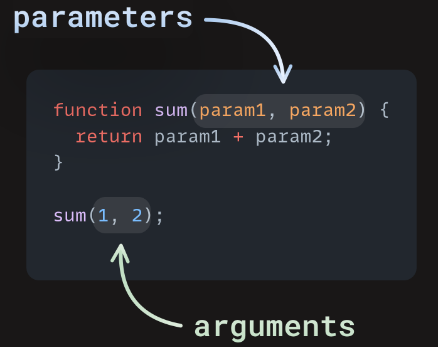
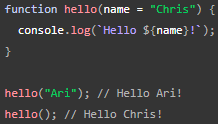
**FUNCTION BASICS**

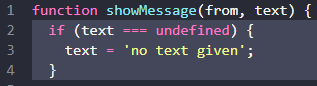
Functions are reusable pieces of code which perform specific tasks. parameters are the items listed between the parentheses (), arguments are the actual values we decide to pass to the function.

Note: Functions that are part of objects are called **methods**.

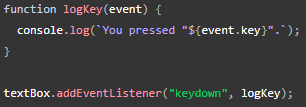
1. **Parameters**

* Without parameters, example:
* Needs parameters, example:
* Optional parameters, ex: & 
* Default parameters

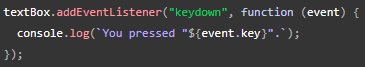
If you're writing a function and want to support optional parameters, you can specify default values by adding = after the name of the parameter, followed by the default value:

Alternative(in old JS code): 

1. **Anonymous Functions and Arrow Functions**
2. Anonymous functions is a function that has no name. AF usually become “function as arguments” in other function(i.e at addEventListener). Example:

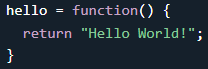
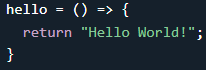


Instead,you can pass an anonymous function into addEventListener():



1. Arrow functions (<https://www.w3schools.com/js/js_arrow_function.asp>)

Arrow functions allow us to write shorter function syntax.



If the function has only one statement, and the statement returns a value, you can remove the brackets and the ‘return’ keyword:

Other example:



if you have only one parameter, you can skip the parentheses as well:

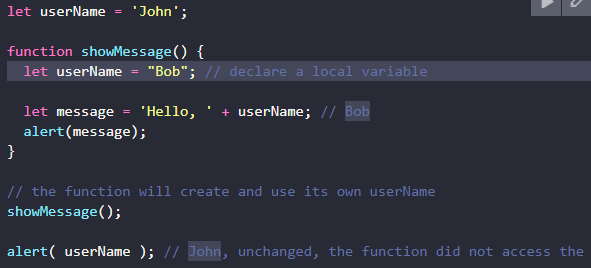


1. **Scope**
2. Global Variable

Variable declared outside the function (or loop) are called global variable, it can be accessed in entire part of document.

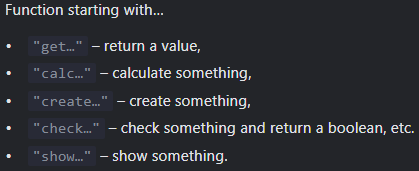
1. Local Variable

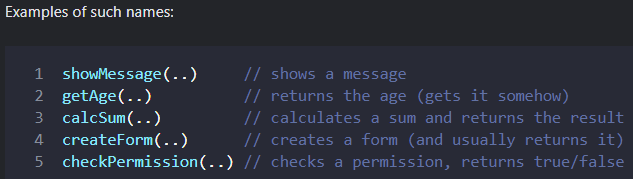
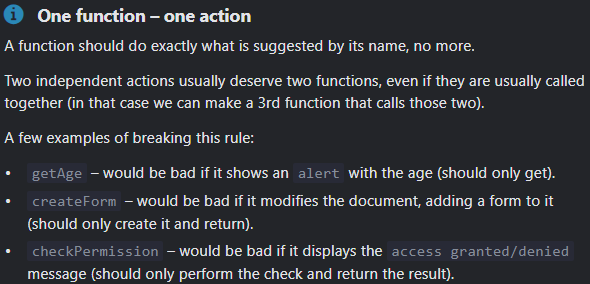
Variable declared inside te function (or loop) are called local variable, it can only be accessed inside the function (or loop) where they are declared.

NOTE: If a same-named variable is declared inside & outside the function, the function will igonre the outside variable. if there’s a command that execute outside the function, the inside variable will be ignored.

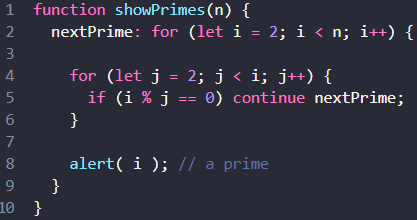
Reccomendation: To make the code clean and easy to understand, it’s recommended to use mainly local variables and parameters in the function, not outer variables.

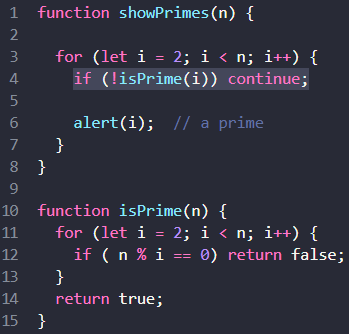
1. **Naming a Function**

Functions are actions. So their name is usually a verb.

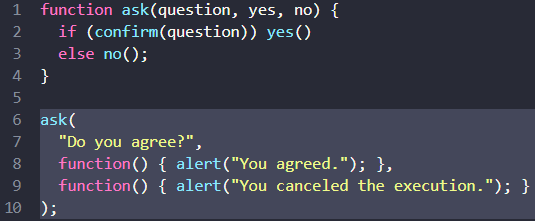
Note:

A separate function is not only easier to test and debug – its very existence is a great comment! This is the example where the second variant better the first variant:

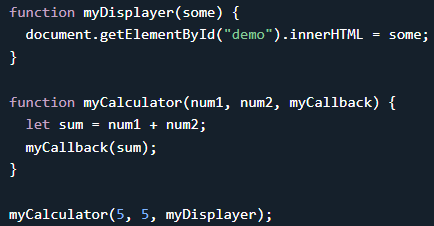
The first variant uses a label:

The second variant uses an additional function isPrime(n) to test for primality: (easier to understand, isn’t it?)

1. **Callback Functions (or just callbacks)**

A callback is a function passed as an argument to another function.

Two anonymous functions as arguments in the ask() function are callback functions.

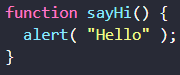
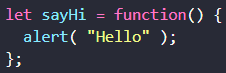
myDisplayer() is a callback function.

* When to Use a Callback?

Where callbacks really shine are in asynchronous functions, where one function has to wait for another function (like waiting for a file to load).

1. **Function Expression**

Function Expression adalah cara mendefinisikan fungsi di mana fungsi tersebut ditugaskan ke sebuah variabel.

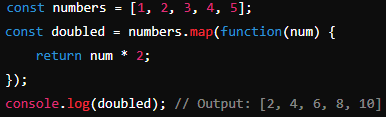
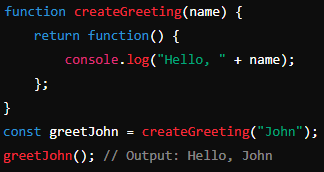
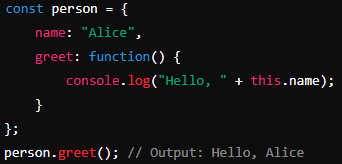
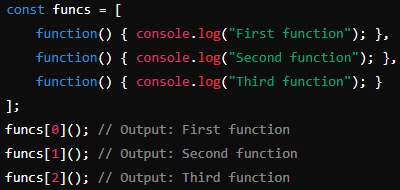
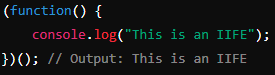
<= Function Declaration

Function Expression =>

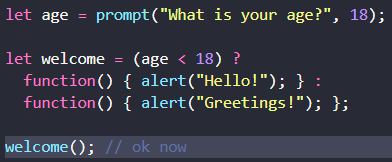
No matter how the function is created, a function is a value. Both examples above store a function in the sayHi variable.

Note: Function declaration dapat dipanggil sebelum dideklarasikan, tetapi Function expression tidak.

Function expression(anonim) dapat digunakan sebagai:

1. Argumen untuk fungsi lain, seperti dalam metode array ‘map’, ‘filter’, atau ‘forEach’.
2. Nilai yg direturn dari fungsi lain
3. Metode dalam objek
4. Elemen dalam array
5. Dalam IIFE(Immediately Invoked Function Expression)

* How we can make Function Expression visible outside of ‘if’ ?



We could simplify it:

* When to choose Function Declaration versus Function Expression?

When we need a conditional declaration (we’ve just seen an example above), then Function Expression should be used.

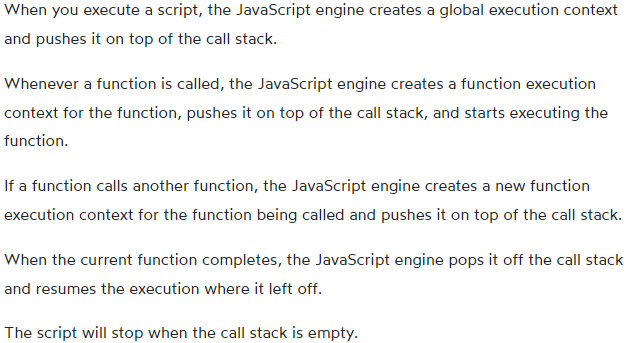
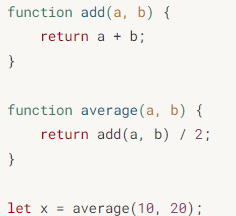
Full(see the last chat): <https://chatgpt.com/share/4249f5da-8180-45e7-99b6-b496fba96ad2>

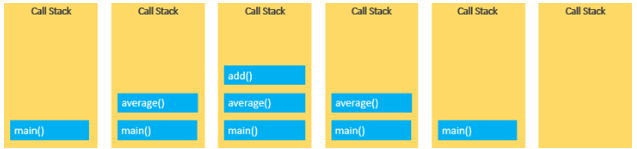
1. **JavaScript Call Stack (**<https://www.javascripttutorial.net/javascript-call-stack/>**)**

the JavaScript engine uses a ‘call stack’ to manage execution contexts:

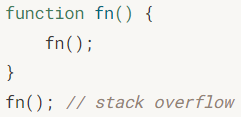
* Global execution context : denoted by main() or global() function on the call stack
* Function execution contexts : other than main() or global()

The call stack works based on the last-in-first-out (LIFO) principle.

JS Call Stack Example:



* Stack Overflow

If the number of execution contexts exceeds the size of the stack, a stack overflow error will occur. example, a recursive function that has no exit condition:

* Asynchronous JavaScript

Asynchronous means the JavaScript engine can execute other tasks while waiting for another task to be completed. Hal ini berarti beberapa operasi dapat dilakukan tanpa harus menunggu operasi sebelumnya selesai. Teknik ini sangat berguna dalam meningkatkan kinerja aplikasi web, terutama untuk operasi yang memakan waktu lama seperti pengambilan data dari server, pengolahan file, atau animasi yang kompleks.

1. **Note**

* Rewrite the function using ‘?’ or ‘||’

