Execution Context/Execution Stack

All javascript code is run on an environment and these environments are called execution context. You can imagine execution context like a box i.e it is a container in which variables are stored and our piece of code is executed.

The default execution context is the global execution context. In default execution context the code that is not inside any function. You can also consider execution context as a Object therefore global execution object is associated with the global objects.

Global Object is nothing but ‘window’ Object.

Hence declaring lastname and window.lastname is the same thing. So, the variable that is declared outside the functions will get automatically attached to the window but the variable that are inside the functions will have there brand new execution context.

We can associate execution context with execution context Object.This Object has three properties:

1. Variable Object : contains function, variable declaration as well as function declaration
2. Scope chain: contains current variable objects as well as variables of parent variable objects.
3. This variable

When a fn is called a new excution context is created on the top of the execution stack. And this happens in two phases:

1. Creation phase: creation of variable object , scope chain, detemine value of This variable.
2. Exection phase : Code if the fn runs line by line

hello();

function hello() {

    console.log("Hello World!");

}

Output: Hello World!

Above, function hello is called before its even made and still we could see output in console because, Even before the execution the function get stores into Variable object.

Some syntax

var hello = function test(some) {

    console.log("Hello World! " + some);

}

hello("Tanuj");

Hello World! Tanuj

test("Tanuj");

var hello = function(some) {

    console.log("Hello World! " + some);

}

test.js:8 Uncaught TypeError: hello is not a function

at test.js:8

Here, we did not get output as function is stored in variable and before execution the are *undefined*.