**GENERATOR FUNCTION**

* **RAVI SINGH**

In ES6, a generator was added as a new type of function. Once we execute a function in JavaScript, it has to be executed entirely. However, we may use generator functions to build functions that another code can use several times. Nothing can cause it to pause outside of the generator function. When the generator function encounters a yield expression, it stops. The generator cannot continue execution on its own after it reaches the yield expression. Something from the outside must continue the process.

Another important difference between generators and normal functions is that generator functions can produce multiple values during its execution. Hence, they can generate a sequence of values, not all at once, but on a per request basis. At every request, the generator function gives us a value until it reaches the end of its execution. Once that happens, the done flag will be set to done.

**Syntax:**

the syntax of declaring the generator function is quite similar to traditional functions. We declare a generator function by using the \* ( asterisk ) operator after the function keyword:

Graphical user interface, application

Description automatically generated

**Yield:-**

The yield keyword pauses the generator function execution, and the value of the expression following the yield keyword is returned to the generator's caller. It acts as a generator-based version of the return keyword. In the following example, to pause the generator's execution, and we use the statement yield.:

**next() method:-**

A generator gives us the next() method, which is used to resume the execution. This method returns an object with two properties. These are value and done:

{

value: [next value],

done: [true if we reach the end, else false]

}

Example:

Text

Description automatically generated

Output:

Text

Description automatically generated

Explanation:

The iterator object is returned when we first call the generator function. When we use next() for the first time, the generator function is invoked, and the first value is returned: 1.  Additionally, calling it a second time returns the second and final value 2. Finally, the third call returns undefined, and done is set to true, indicating that the generator has been iterated over.