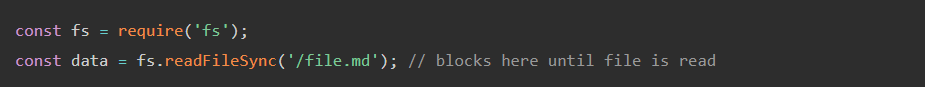
1. **Blocking and Non-Blocking: Asynchronous nature of Node.js**

**BLOCKING:**

* Blocking is when the execution of additional JavaScript in Node.js process must wait until a non-JavaScript operations complete. This happens because the event loop is unable to continue running JavaScript while a blocking operation is occurring.
* Blocking methods executes synchronously.
* Blocking counterparts have names that ends with Sync.

Example:



* It is simple to write a code for blocking but has the disadvantage, as we can see the second line blocking the execution of any additional JavaScript until the entire file is read.
* In the synchronous version if an error is thrown it will need to be caught or the process will crash.

**NON-BLOCKING:**

* In non-blocking, the additional JavaScript in Node.js doesn’t wait for the non-JavaScript operations to be completed.
* Non-Blocking methods executes asynchronously.

Example:

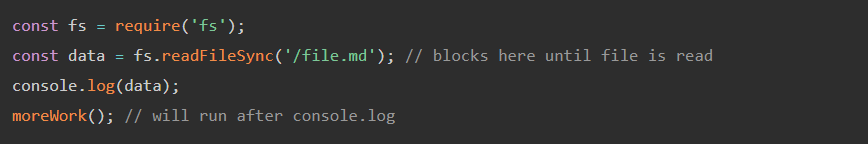


* In the asynchronous version, it is up to the author to decide whether an error should be thrown.

All of the I/O methods in the Node.js standard library provide asynchronous versions, which are non-blocking and accept callback functions.

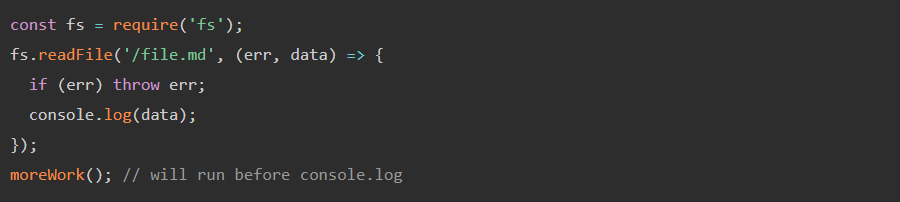
**Comparing Code:**

Synchronous file read:



In this example, console.log will be called before moreWork(). And morework() will be run after console.log. Fs.readFileSync is blocking method.

Asynchronous file read:



fs.readFile () is non-blocking so JavaScript execution can continue and moreWork() will be called first. The ability to run moreWork () without waiting for the file read to complete is a key design choice that allows for higher throughput.

1. **Difference between Var, let and const.**

**Var:** In older JavaScript codes, you will only see variables being declared with this keyword. Var is a global scoped or function scoped keyword depending on where it is declared. Variables declared outside a function have global scope whereas the ones declared inside a function have function scope. Global scoped variables can be accessed throughout the browser window. Although it is advisable to not make use of global variables, there are some applications in which they are needed.

**Let:** let is a major improvement from the var keyword. Variables declared with the let keyword are block scoped. A block is any code written in between the curly brackets { }. Block scoped variables are only accessible within that block.

If we declare a variable within a block and try to access it outside, it gives you an error.

**Const:** As the name suggests, const is used for variables with a constant value. The value of the const variable will never change throughout the lifetime of the program. Just like let, const is also block scoped, i.e. it is restricted to the block in which it is declared. When declaring variables with const, it is necessary to initialize it with a value.

Constants are written in all caps.

Let and const are the new alternatives to var for declaring variables and are safe to use nowadays. Although all three can be used. The use of var is strongly discouraged by most JS Experts since its error prone and large applications can be hard to maintain. I have personally replaced the use of var with let and const and is a good practice to follow.