# *JS essential Notes*

## Equality Operators

=== (or !==) is the strict way to compare (doesn’t preform type coercion).

== (or !=) in the opposite side is loose way to compare (preform type coercion) (avoid it because it sometimes lead to unexpected bugs so even when we need to compare deferent types it preferred to convert it manually.

## Functions

* There are two kinds to declare it expression (and special case the arrow function) or declaration.

### Declaration function

It’s more familiar to me, it’s once declared it you can call it whatever above or below. Example:



### Expression function

It’s little weird it stores the function into a variable and use it like normal, it must call only below it not above like the declaration way. Example:

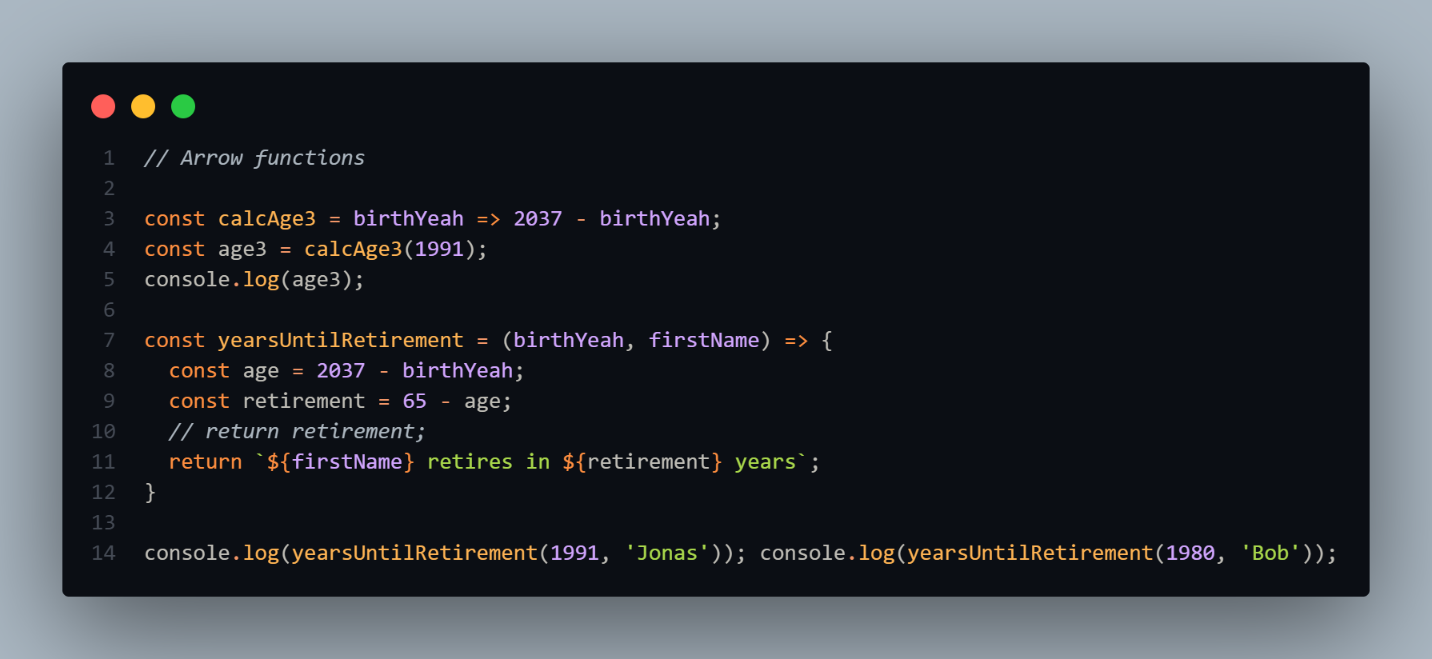


#### **Arrow function**

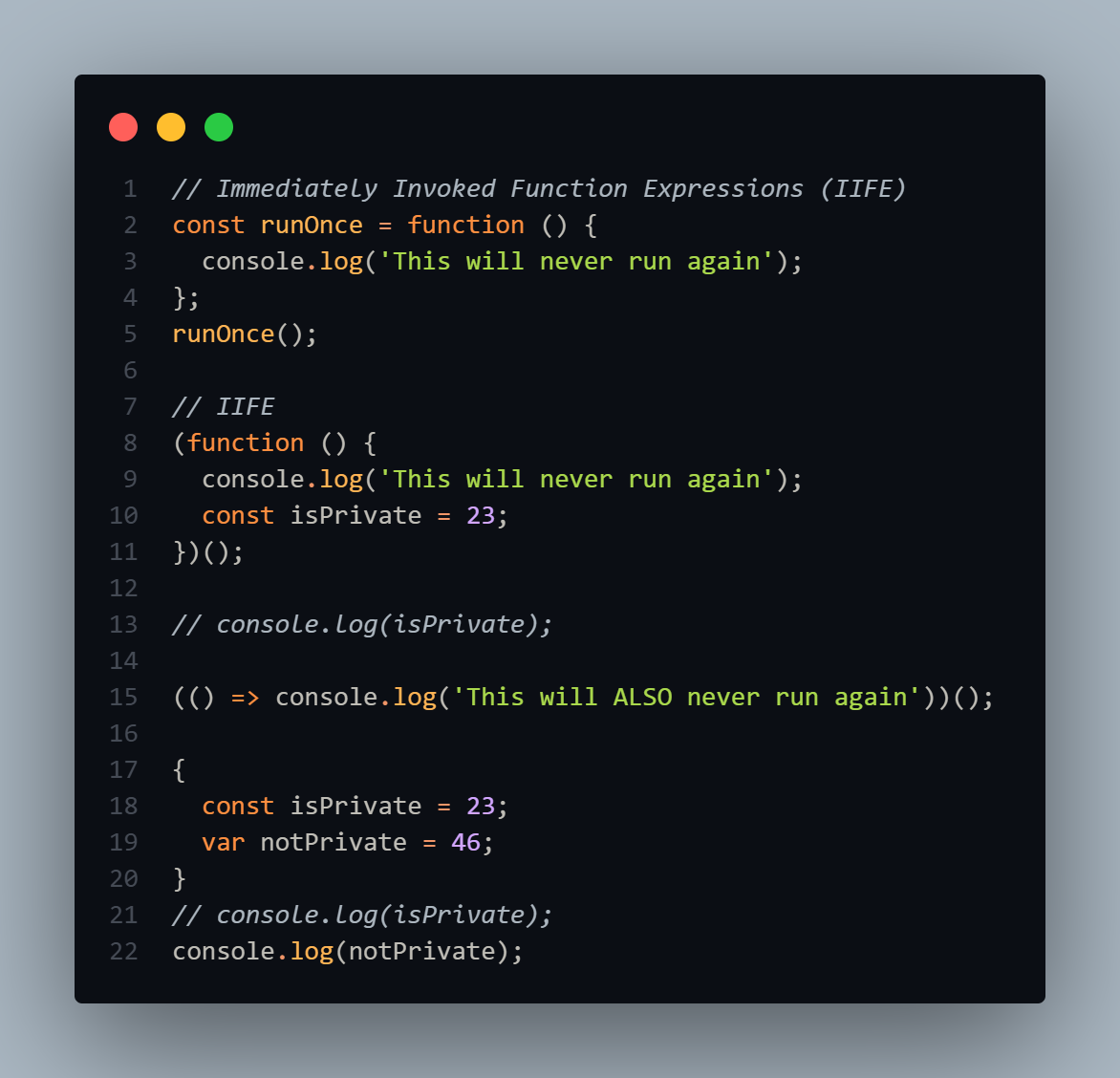
it’s a special case of expression function (added in ES6), It’s easy and faster for one line function.

It doesn’t have arguments object or this keyword.

Example:



### Immediately invoke function expression



### Closures

Graphical user interface, text, application, email

Description automatically generated

## Arrays

* It’s a data structure container that can contain any type of variables (not like in c++)and even an anther array, we can declare it in different ways, like: 

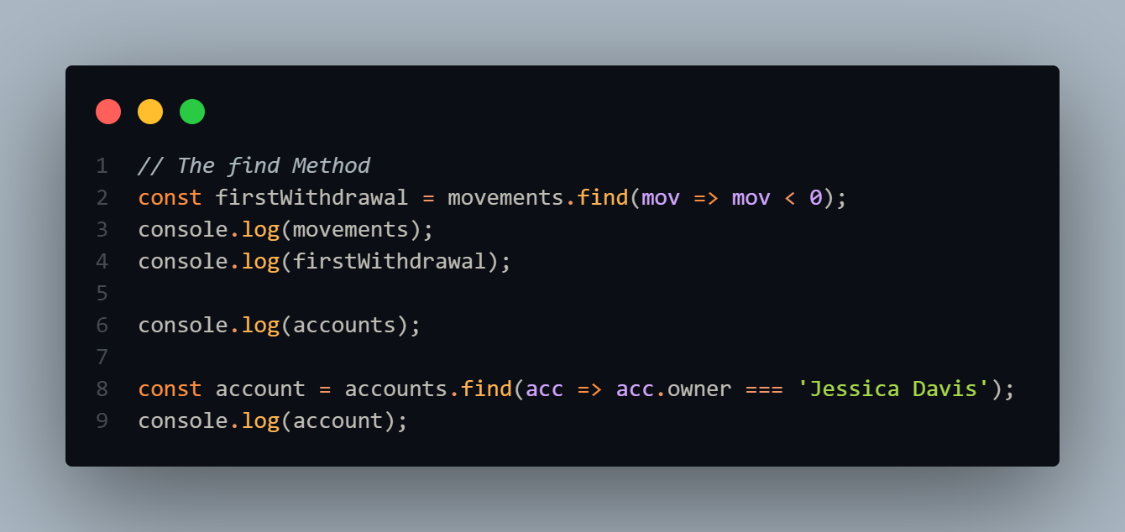
### Basic array methods



We can use method to access the array instead of square brackets it so much helpful when we want to use a negative index to access the array from the end.



Find method it returns the first element satisfies the condition



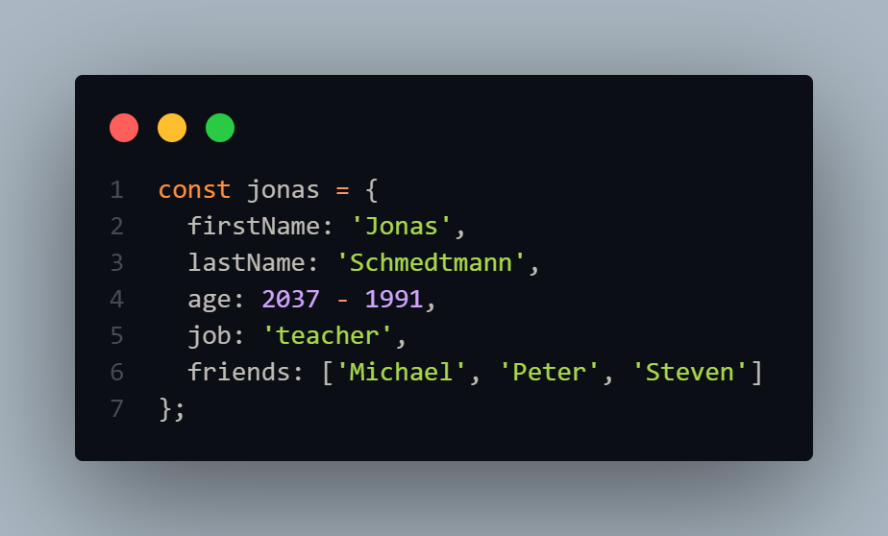
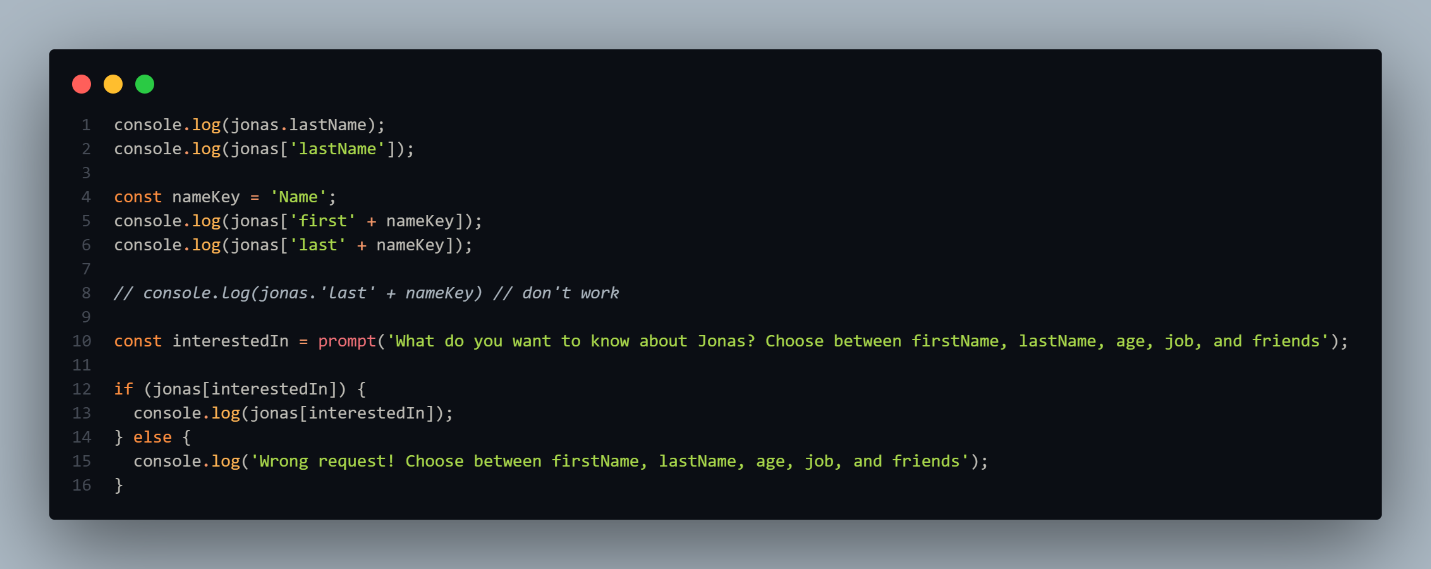
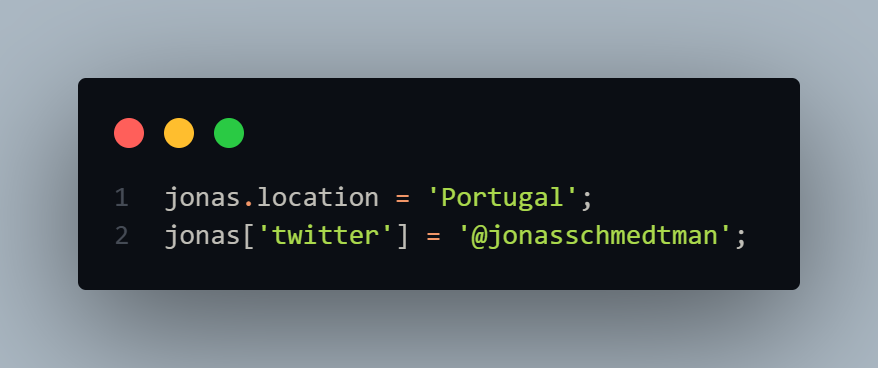
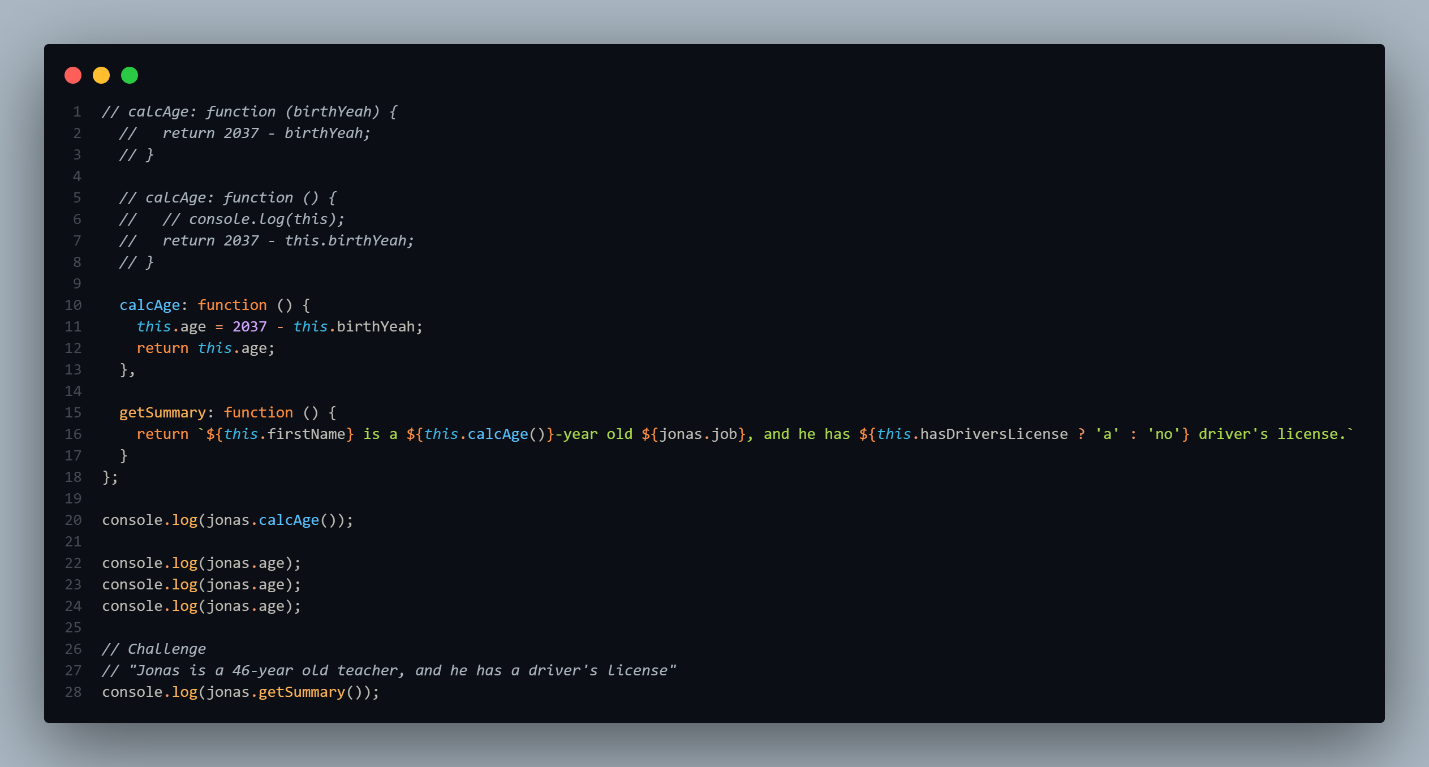
Sum for check if anyone satisfy the condition but every must all elements satisfy the condition



Flat method is used to flat the array of arrays to one flatted array, and we can pass how deeper we want to flat. Flatmap is a compilation of the flat and map and we can’t go deeper with it over one nested level.



## Object

* It’s like the map in c++ store the values and key for each. We can access the values by them keys in two ways in dot notation and bracket notation. 
* Dot notation: It accesses the value by only his name directly, but it cleaner and easier to use.
* Brackets notation: It can access the value by name and accepts expressions. 
* We can easily add new properties by the two ways: 
* We can as well add a function to an object in this case it call a method and it write only in expression way. 

## Scope and scope chain

* Text, application

  Description automatically generated

## Loop

* Text

  Description automatically generated

### ForEach

* It loops over the array and modify it like the normal loop but so much helpful for method chaining
* ForEach on arrays

  Description automatically generated
* Text

  Description automatically generated

### map

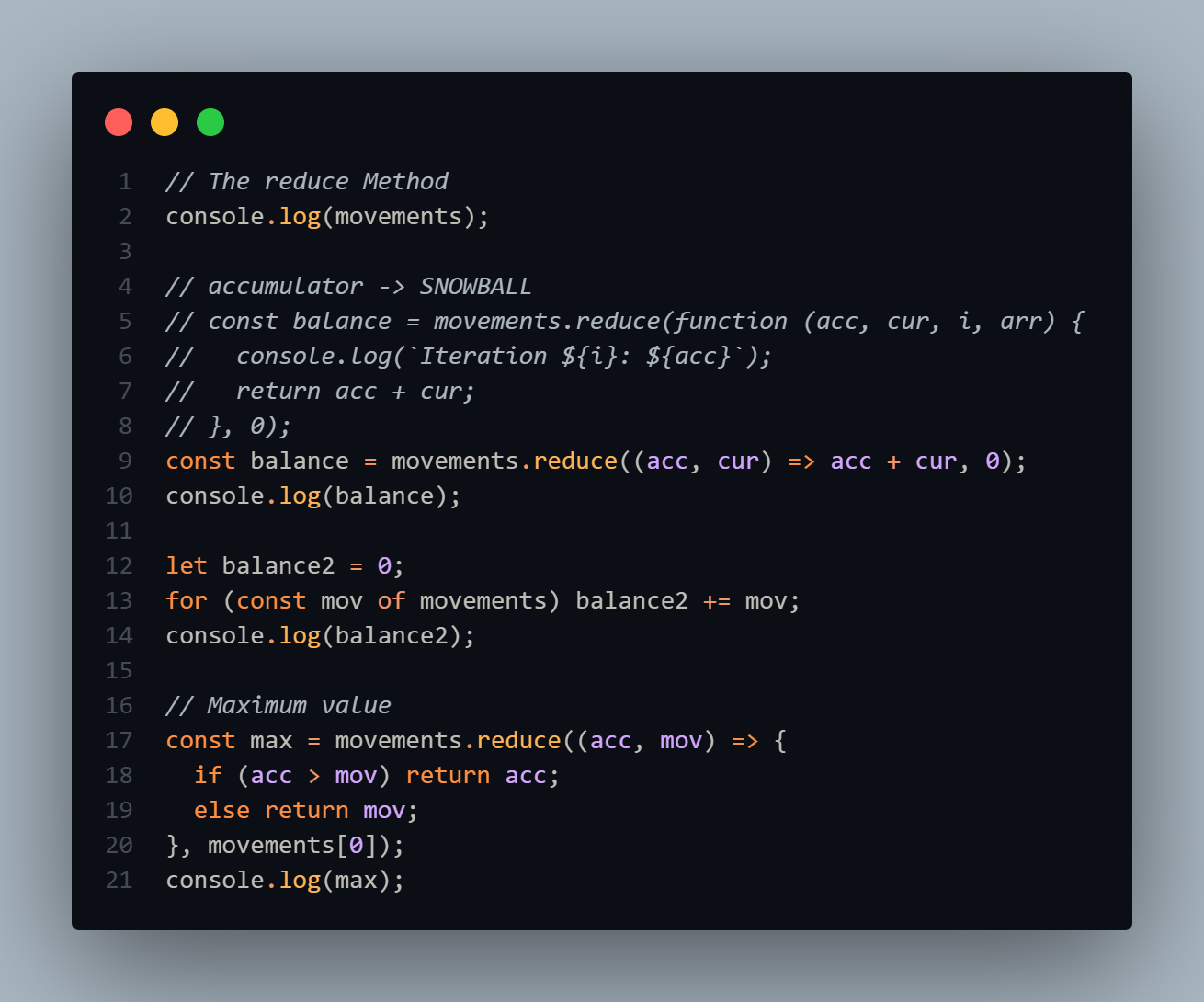
* It returns a new array with chance than happen in the callback function
* 

### Filter

* It filters the array base on boolen callback function
* 

### Reduce

It collect all array in singe value (whatever it type) through any process

* 

## Container

Graphical user interface, website

Description automatically generated

### Sets:

It like the array but her elements are unique and hasn’t index, it can be use to ignore the duplicate of any arrayText

Description automatically generated

### Maps:

Its anther data structure much like objects but the difference that it accept any type of keys even an object or array. Initialize

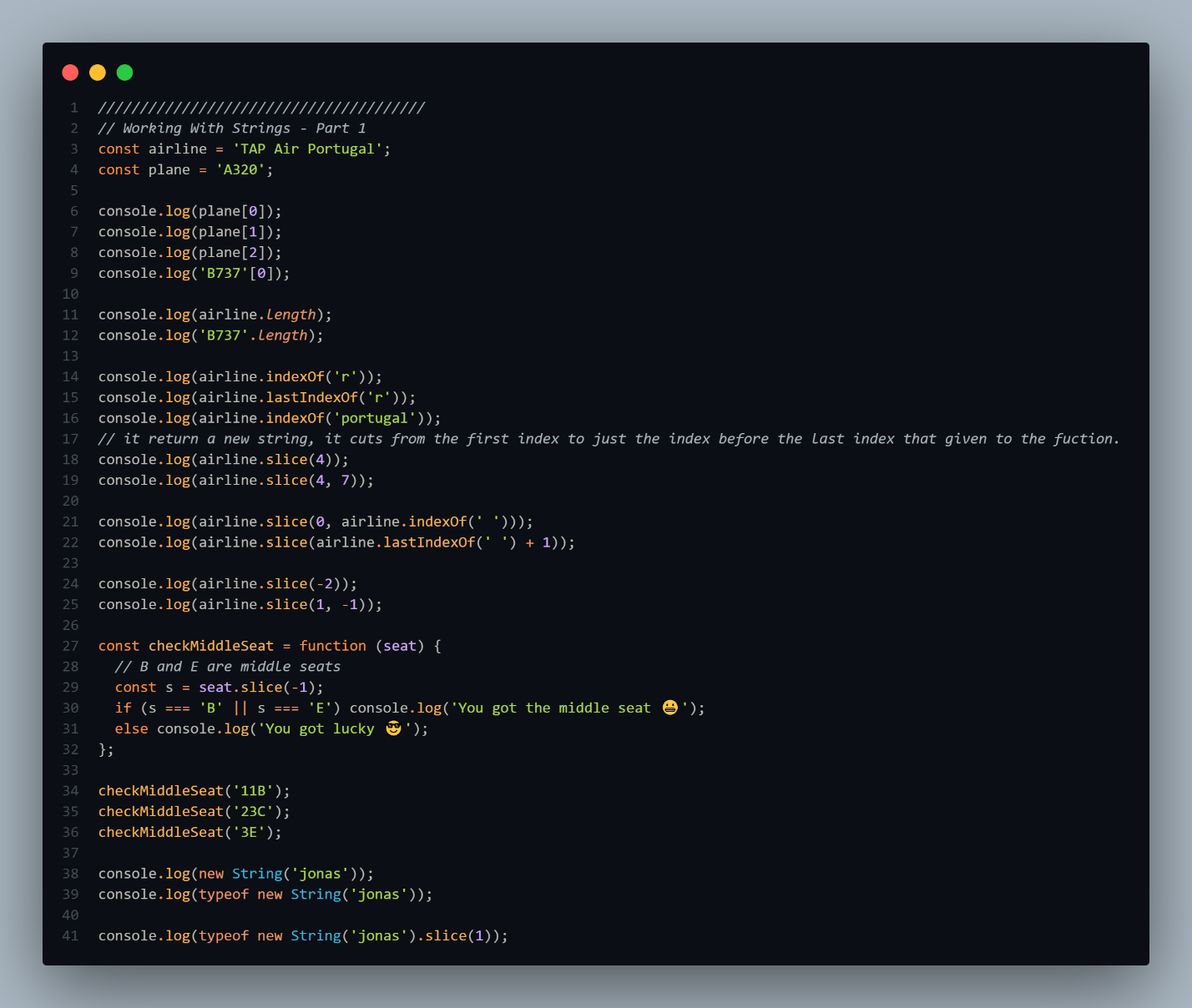
Text

Description automatically generated

Text

Description automatically generated

## Strings

* 
* 
* 