'const' Vs 'object'

a) Example -- Assignment with 'const'

The 'const' keyword ensures that the variable it creates is read-only. It doesn't mean that the actual value to which the 'const' variable reference is immutable.
We can change & modify the object properties created using 'const' keyword.

```
const person = { age: 20 };
person.age = 30; // OK
console.log(person.age); // 30

person = {age: 40}; // TypeError
```

b) Example -- Assignment with 'object'

If we want the value of the person object to be immutable, you have to freeze it by using the Object.freeze() method.

```
const person = Object.freeze({age: 20});
person.age = 30; // TypeError
```

'const' Vs 'arrays'

We can change / modify / delete the array's elements. However, we cannot reassign the array colors (or data) to another array.

Example

```
const colors = ['red'];
colors.push('green');
console.log(colors); // ["red", "green"]

colors.pop();
colors.pop();
console.log(colors); // []

colors = []; // TypeError
```

Template Literals / Template Strings

- Template literals are an improvement on string concatenation that you should rarely ever combine strings with traditional concatenation.
- In ES6, we can create a template literal by wrapping our text in backticks.
- Template literals allows us to easily implement variables with a very simple syntax (${ }) and embed expressions.
- With the help of Template literals we can embedd expressions & variables inside strings.
- Template literals mostly used for constructing API requests and nesting templates.

Syntax of Template Literals

```
let abc = `Hello Template Literals`
```

features of template literals

1) Multiline string -- a string that can span multiple lines.
2) String formatting -- the ability to substitute part of the string for the values of variables or expressions. This feature is also called string interpolation.
3) HTML escaping -- the ability to transform a string so that it is safe to include in HTML.

Example

```
let str = `Template literal Examples in ES6`;
console.log(str);// Template literal in ES6
console.log(str.length); // 23
console.log(typeof str);// string
```

Example

```
let name = "Rohit"
let result = `Hello ${name}`
console.log(result) // Hello Rohit
```

Example

```
let name = "Rohit"
const greeting = `Hello my name is ${name}`
console.log(greeting) // Hello my name is Rohit
```

Example

```
let result = `The additon is ${10+20}`
console.log(result)
```

Example -- Multi-line Strings

```
let greet = `Hello
Rohit
you are
awesome`
console.log(greet)
```

Example -- Inside of Functions

```
function displayName(name) {
   console.log(`${name.toUpperCase()}`)
}
displayName("Rohit")
```

Example -- With Object

```
let customer = { name: "Rohit" }
let card = { amount: 7, product: "Bar", unitprice: 42 }

let message = `Hello ${customer.name},
want to buy ${card.amount} ${card.product} for
a total of ${card.amount * card.unitprice} bucks?`

console.log(message)
```

Example -- Executing Raw Strings

```
console.log(String.raw`Hello \n World`) // Hello \n World
```

Ternary Operator in ES6

- Ternary Operators are the shorthand version of if...else statements.
- It is the only conditional operator in JavaScript that takes three operands.
- The difference betwwen the "ternary operator" & "if-else" is that the ternary operator is an expression whereas an "if..else" construct is a statement.

Syntax

Syntax => condition ? expression1 (if) : expression2 (else)

where,

- "condition" is the value to be tested/evaluated,
- "expression1" can be value(s) of any type to be executed if the condition is true
- "expression2" can be value(s) of any type to be executed if expression1 is false i.e fallback value commonly know as 'else'

- Simply, " ? " means "IF", and " : " means "else".

Advantages of Ternary Operator

- Syntactic Sugar Syntax Over "if-else".
- Flexibility and Miniature size.
- Less bugs.
- Ternaries don't need temporary variables, reducing load on working memory.

Comparison between "if-else" & "ternary" operator

Example -- With "if...else" Statement

```
let age = 18;
if (age >= 16) {
  console.log("You're allowed to drive!");
}
else {
  console.log("You should be 16 to drive!");
}
```

Example -- With "Ternary" Operator

```
let age = 18;
(age >= 16) ? console.log("You're allowed to drive!") : console.log("You
should be 16 to drive!");
```

Examples of "ternary" operator

Example

```
var day = true; //conditon
console.log(day ? 'It is day-time' : 'It is night-time')
```

Example

```
var age = 26;
var beverage = (age >= 21) ? "Beer" : "Juice";
console.log(beverage); // "Beer"
```

Example

```
let isMember = true;
console.log('The fee is ' + (isMember ? '200' : '100'));
```

Example

```
let age = 18;
let canDrive = (age >= 16) ? "You're allowed to drive!" : "You should be 16 to drive!";
console.log(canDrive); // "You're allowed to drive!"
```

Example

```
let isStudent = false;
let isSenior = true;
let price = isStudent ? 8 : isSenior ? 6 : 10;
console.log(price);
```

Example

```
let myName = false;
let age = false;
let message = myName ? "I have a name" : "I don't have a name, duh!"
console.log(message) // I have don't have a name, duh!
```

Example

```
let myName = true;
let message = myName ? age = true : 'Get out of here, nameless'
console.log(message) // true
```

Example

```
var email = false;
var phoneNumber = true;
var message = email ? 'Enter your number' : phoneNumber ? 'Enter you
phone number' : 'No Reply'
console.log(message) //Thanks for reaching out to us
```

Example

```
let home = true;
let dish = '';
home ? (
  console.log('Welcome to kitchen'),
  dish = prompt('What you wanna have'),
  console.log('Your ' + dish + ' is ready')
) : console.log('Check back when you get home' )
```

Example -- Simplify Ternary Operator

```
var locked = 1;
var canChange = locked != 1 ? true : false;
```

Example -- Simplify Ternary Operator

```
var locked = 1;
var canChange = locked != 1;
```

Example -- Chained Ternary Operator

```
let year = prompt('Which year was the 2018 World Cup?', '');
(year < 2018) ? console.log('Too early') : (year > 2018) ? console.log('Too
late') : console.log('Exactly!');
```

Example -- Nested Ternary Operator

```
let statement1 = true;
let statement2 = true;
let check = statement1 ? (statement2 ? "True, Yes!" : "True, False!") : 'False';
console.log(check); // True, Yes!
```

Example

```
let stop = false, age = 23;
age > 18 ? (
    console.log('OK, you can go.')
) : (
    console.log('Sorry, you are much too young!')
);
```

Example -- Handling 'null' Values

```
let greeting = person => {
    let name = person ? person.name : `stranger`
    return `Howdy, ${name}`
}
console.log(greeting({name: `Rohit`}));  // "Howdy, Rohit"
console.log(greeting(null));             // "Howdy, stranger"
```

Example -- Multiple operations

```
var marks = 43
var grade;
(marks < 30) ? (grade ='Fail', console.log("Better luck next time "))
  : (grade ='Pass', console.log("Congratulations "));
console.log(grade)
```

Example -- Multiple Operations

```
var home = true;
var myLocation = 'India';
myLocation = home ? ('New Delhi', 'Noida', 'Gurugram', 'Agra' ) : 'New
York'
console.log(myLocation)
```

Example -- Multiple operations per case

```
let age = 16;
let auth = age > 18 ? (
    'OK, you can go.',
    'APPROVED'
) : (
    'You are much too young!',
    'Sorry :-(',
    'DISAPPROVE'
);
console.log(auth); // "DISAPPROVE"
```

Example -- With Functions

```
const running = true;
function start(){
  console.log('True Executed')
}
function stop(){
  console.log('False Executed')
}
(running === true) ? start() : stop()
```

Example -- With Functions

```
let marvel = true;
function movieCharacter(marvel) {
  return (marvel === true) ? "I am Iron Man" : "I am Superman";
}
let output = movieCharacter(marvel);
console.log(output);
```

Example -- With Function

```
function getFee(isMember) {
  return (isMember ? 'yes' : 'no');
}
console.log(getFee(true));
console.log(getFee(false));
console.log(getFee(null));
```

Example -- With Functions

```
function foo(bar) {
    bar = typeof(bar) !== 'undefined' ? bar : 10;
    console.log(bar);
}
foo(); // 10
foo(20); // 20
```

Example -- With Functions

```
let authenticated = true;
let nextURL = authenticated ? (
    alert('You will redirect to admin area'),
        '/admin'
) : (
    alert('Access denied'),
        '/403'
);
// redirect to nextURL here
console.log(nextURL); // '/admin'
```

Example -- With Objects

```
let person = {
  name: 'Neha',
  age: 20,
  driver: null
};
(person.age >= 16) ? person.driver = 'Yes' : person.driver = 'No';
console.log(person)
```