# JAVASCRIPT CHEATSHEETS

## How to add JavaScript

## On-page JavaScript

Adding internal JavaScript to HTML

```
<script> //JS code goes here </script>
```

## External JavaScript

Adding external JavaScript to HTML

```
<script src="script.js"></script>
```

## JS Output

```
console.log(“Hello, World!!”);
```

## JSAlert

JavaScript `alert()` method displays a pop-up alert box on the browser with a specified message such as text etc.,

```
alert(“Hello,World!!”);
```

## Operators

An operator is a special symbol in JavaScript that performs operations on operands.

## Arithmetic Operators

The assignment operators are: `+,-,*,/,%,++,--`

```
Let x = 6;
let y = 4;

console.log('x + y = ', x + y);
console.log('x - y = ', x - y);
console.log('x * y = ', x * y);
console.log('x / y = ', x / y);
```

## Assignment Operators

The assignment operators are: =, +=, -=, *=, /=

```
console.log(“Hello, World!!”);
```

## ComparisonOperators

The comparison operators are: ==, ===, !=, !==, >, <, >=, <=

```
Let x = 5;
let y = 5;

console.log(x == y);
console.log(x!= y);
console.log(x > y);
console.log(x >= y);
console.log(x < y);
console.log(x <= y);
```

## Logical Operators

The logical operators are: **&&**(AND), **||**(OR), and**!**(NOT)

```
Let x = 5;
let y = 5;

console.log(x&&y);
console.log(x||y);
```

# Conditional Statements

Conditional statements are used to perform operations based on some conditions.

# If Statement

The if statement is used to execute a block of code if a given condition is true.

```
if(condition){
  // block of code to be executed if the condition is true
}
```

# If...else statement

The  If...else statement is used to execute a block of code if a specified condition is true and another block of code if the condition is false.

```
if(condition){
  // block of code to be executed if the condition is true
}else{
  // block of code to be executed if the condition is false
}
```

# Ifelseladder

The "if-else ladder" is a control structure in JavaScript that allows you to execute a different block of code depending on multiple conditions.

```
if(condition){
  // block of code to be executed if condition1 is true
}else if(condition2){
  // block of code to be executed if the condition1 is false and
condition2 is true
}else{
  // block of code to be executed if the condition 1 is false and
condition2 is false
}
```

# Switch Statement

The "switch" statement in JavaScript is another control structure that allows you to execute a different block of code depending on a specific value.

```
switch(expression){
  case value 1:
    // block of code
    break;
  Case value 2:
    // block of code
    break;
  Case value 3:
    // block of code
    break;
  default:
    // block of code
}
```

# Loops

Loops are used to execute a block of code until the specified condition is met.

# For Loop

A for loop In JavaScript is used to execute a piece of code a specified number of times.

```
for(initialization; testExpression; increment/decrement){
// block of code
}
```

Example:

```
Let x = 5;
for (let i = 1; i <= 5; i++) {
   console.log("Hello JavaScript");
}
```

# For Loop

A for loop In JavaScript is used to execute a piece of code a specified number of times.

```
while(condition){
  // block of code
}
```

Example:

```
Let i = 1;

while (i <= 8) {
  console.log(i);
  i++;
}
```

# Strings

A string in JavaScript is a primitive data type that represents a sequence of characters. It is enclosed by single quotes (''), double quotes (""),or backticks ('').

```
let name1 = "Peter";
let name2 = 'Perker';
let greeting = `Hello${name}!`;
```

# String Methods

# length

The length method returns the length of a string.

```
Let str = "Hello JavaScript";
console.log(str.length);
```

## concat

The concat() method is used to concatenate two or more strings.

```
Let str1 = "Hello";
let str2 = " JavaScript";
console.log(str1.concat(str2));
```

## indexof

The indexOf() method is used to find the indexof a specific character in a string.

```
let str = "Hello JavaScript";
console.log(str.indexOf("J"));
```

## replace

The replace() method replaces a string with another string.

```
let str = "Hello JavaScript";
console.log(str.replace("Hello", "I Love"));
```

## Number

Numbers are a fundamental data type in JavaScript that represents numerical values. They can be either integer or floating-point numbers.

```
let x = 4;
let y = 4.13;
```

## Arrays

Arrays in JavaScript area fundamental data structure used to store multiple values  in a single variable.

Example:

```
Let cars = ["Thar", "Scorpio", "BMW", "Ferrari"];
```

# Array Methods

## push

The push() method is used to add an element to the end of an array.

```
Let myArr = ["Thar", "Scorpio", "Ferrari"];
myArr.push("Audi");
console.log(myArr);
```

## pop

The pop() method is used to remove the last element of an array.

```
Let myArr = ["Thar", "Scorpio", "Ferrari"];
myArr.pop();
console.log(myArr);
```

## Date

The `Date` object in JavaScript is used to interact with dates and times. You can create a new `Date` object to represent either the current date and time or a specific date and time.

Example:

```
Let currentDate = new Date();
console.log(currentDate);
```

# Date Methods

| Method | Description |
|---|---|
| getDate() | Gets the day of the month(1-31) according To local time |
| getFullYear() | Gets the year according to local time |
| getMonth() | Gets the month,from 0 to11 according to local time |
| setDate() | Sets the day of the month according to local time |
| setFullYear() | Sets the full year according to local time |
| setMonth() | Sets the month according to local time |

**Example:**

```
let date = new Date();
let day = date.getDate();
console.log(day);

let date = new Date();
let year = date.getFullYear();
console.log(year);

let date = new Date();
let month = date.getMonth()+ 1;
console.log(month);

let date = new Date();
date.setDate(26);
let D = date.getDate();
console.log(D);
```

# Function

JavaScript Function syntax

```
function nameOfFunction(){

    //function body

}
```

# Mouse Events

## click

Worked when an element is clicked

```
element.addEventListener('click',()=>{
    //Code to be executed when the event is fired
});
```

## oncontextmenu

Worked when an element is right-clicked

```
element.addEventListener('contextmenu',()=> {
    //Code to be executed when the event is fired
});
```

# Keyboard Events

## keydown

Worked when the user is pressing a key on the keyboard

```
element.addEventListener('keydown',()=>{
    //Code to be executed when the event is fired
});
```

## keyup

Worked when the user releases a key on the keyboard

```
element.addEventListener('keyup',()=>{
    //Code to be executed when the event is fired
});
```

# Query/GetElements

## querySelector

Selector to select first matching element

```
document.querySelector('css-selectors')
```

## querySelectorAll

A selector to select all matching elements

```
document.querySelectorAll('css-selectors',…)
```

## getElementsByClassName

A selector to select all matching elements

```
document.getElementsByClassName('class-name')
```

## GetElementbyId

Select an element by its id

```
document.getElementbyId('Id')
```