JavaScript

JavaScript For Everything

Introduction to JavaScript

JavaScript is

- lightweight and interpreted programming language
- Most popular programming language in the world
- Height level programming language
- Dynamic programming language

Statements

- Programming Language တစ်ခုခုနဲ့ ရေးထားတဲ့ လုပ်ဆောင်ရမယ့် ညွှန်ကြားချက်တွေကို Statements လို့ခေါ်တယ်။
- JavaScript မှာ Statements တွေကို Values, Operators, Expressions, Keywords, Comments တွေနဲ့ပါဝင်ဖွဲ့စည်းထားတယ်.

Syntax

- White space
  - Space တွေနရာအများကြီးယူခြင်းကို လျှစ်လျူရှုပေးတယ်။ Code တွေလှပအောင် character တွေကို Space ခြားပြီးရေးနိုင်။ Eg. x = 1 + 2;
- Case sensitive
  - Case Sensitive ဖြစ်တယ်။ ဥပမာ variable something နဲ့ Something မတူသလိုပါ။ တခြားသက်မှတ်ချက်များလဲ ထိုနည်းတိုင်းပင်ဖြစ်သည်။
- Comments
  - Single line comment -> //
  - Multi line comment -> /* ……. */
- Semicolons
  - Statement တစ်ကြောင်းရဲ့ အဆုံးမှာ Semicolons (;) ကိုသုံးလဲရတယ်။ မသုံးလဲရတယ်။

Variables

$$\overbrace{\texttt{var}}^{\text{Variable Keyword}}\ \underbrace{\texttt{courseFee}}_{\text{Variable Name}} = \overbrace{\texttt{280000}}^{\text{Variable Value}};$$

Variable Keywords

- Var
  - Global Variable
- Let
  - Block Scope Variable
- Const
  - Constant variable

Rule for naming variables

Variable name ပေးတဲ့အခါမှာ

- number နဲ့လုံးဝ မစရဘူး။ letter, dollar sign($), underscore (_) တို့နဲ့စရမယ်။ သူတို့တွေရဲ့နောက်မှာ Number ထည့်လို့ရတယ်။

- dash (-)နဲ့ (.) ပါလို့မရဘူး။

- Keywords တွေမသုံးရဘူး။

- variable အားလုံးဟာ case sensitive ဖြစ်တယ်။

- variable ထဲကိုသိမ်းမယ့် Value နဲ့သက်ဆိုင်တဲ့ name ပေးရမယ်။

- စကားလုံး တစ်လုံးထက်ပိုရင် camelCase ပုံစံပေးရမယ်။

Data Types

- String data type
- Numeric data type
- Boolean data type
- Null
- Undefined
- Array
- Object

Expressions

Two type of expressions

- Expression that just assign a value to a variable
    - var animal = ‘Panda’;
- Expression that use two or more values to return a single value
    - var volume = 12 * 3 * 4;

Operators

- The addition operator (+)
  - let three = 1+2;
  - let four = three + 1;
  - let threeOne = ‘three’ + 1;
- The subtraction operator (-)
  - let two = 4 - 2;
- The division operator (/)
  - let div1 = 20/5;
  - let div2 = 20/7;
  - let div3 = 1/0;

Operators

- The remainder operator (%)
  - let rem1 = 20/5;
  - let rem2 = 20/7;
  - let rem3 = 1/0;
- The multiplication operator (*)
  - let mul = 1 * 2;
- The exponentiation operator (**)
  - let exp = 1**2;
  - let exp1 = 2**8;

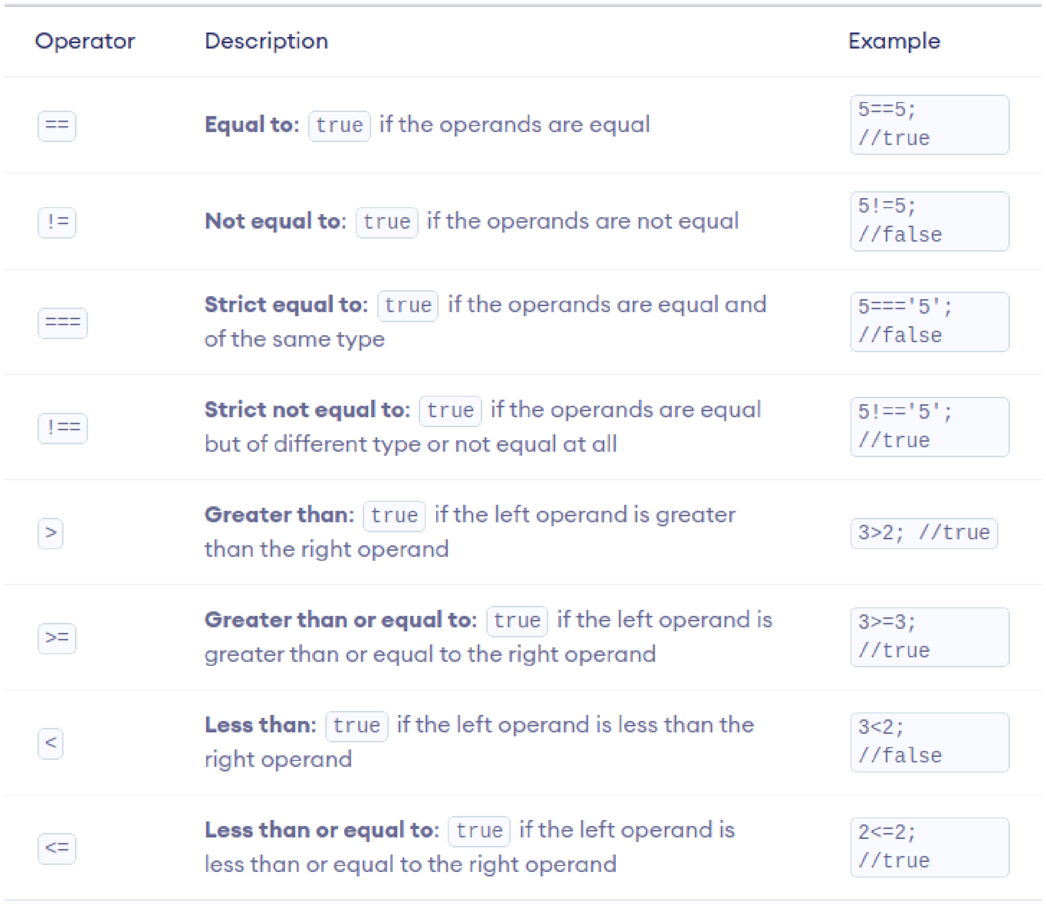
Operators

- Precedence
  - * / %
  - + -
  - =
  - let a = 1 * 2 + 5 / 2 % 2;

Comparisons

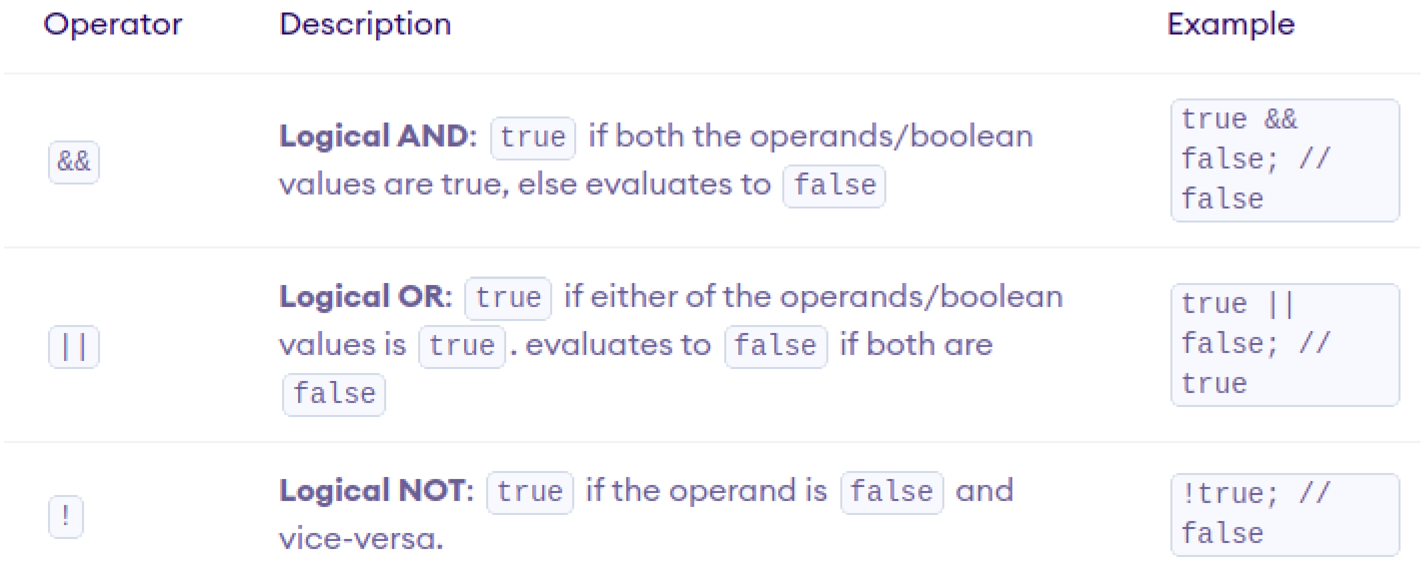
- < “less than”
- <= “less than or equal”
- > “greater than”
- >= “greater than or equal”
- == “equality value”
- === “equality value and data type”
- != “not equality value”
- !== “not equality value and data type”

Comparisons



Operator	Description	Example
`==`	**Equal to:** `true` if the operands are equal	`5==5; //true`
`!=`	**Not equal to:** `true` if the operands are not equal	`5!=5; //false`
`===`	**Strict equal to:** `true` if the operands are equal and of the same type	`5==='5'; //false`
`!==`	**Strict not equal to:** `true` if the operands are equal but of different type or not equal at all	`5!=='5'; //true`
`>`	**Greater than:** `true` if the left operand is greater than the right operand	`3>2; //true`
`>=`	**Greater than or equal to:** `true` if the left operand is greater than or equal to the right operand	`3>=3; //true`
`<`	**Less than:** `true` if the left operand is less than the right operand	`3<2; //false`
`<=`	**Less than or equal to:** `true` if the left operand is less than or equal to the right operand	`2<=2; //true`

Logical Operator

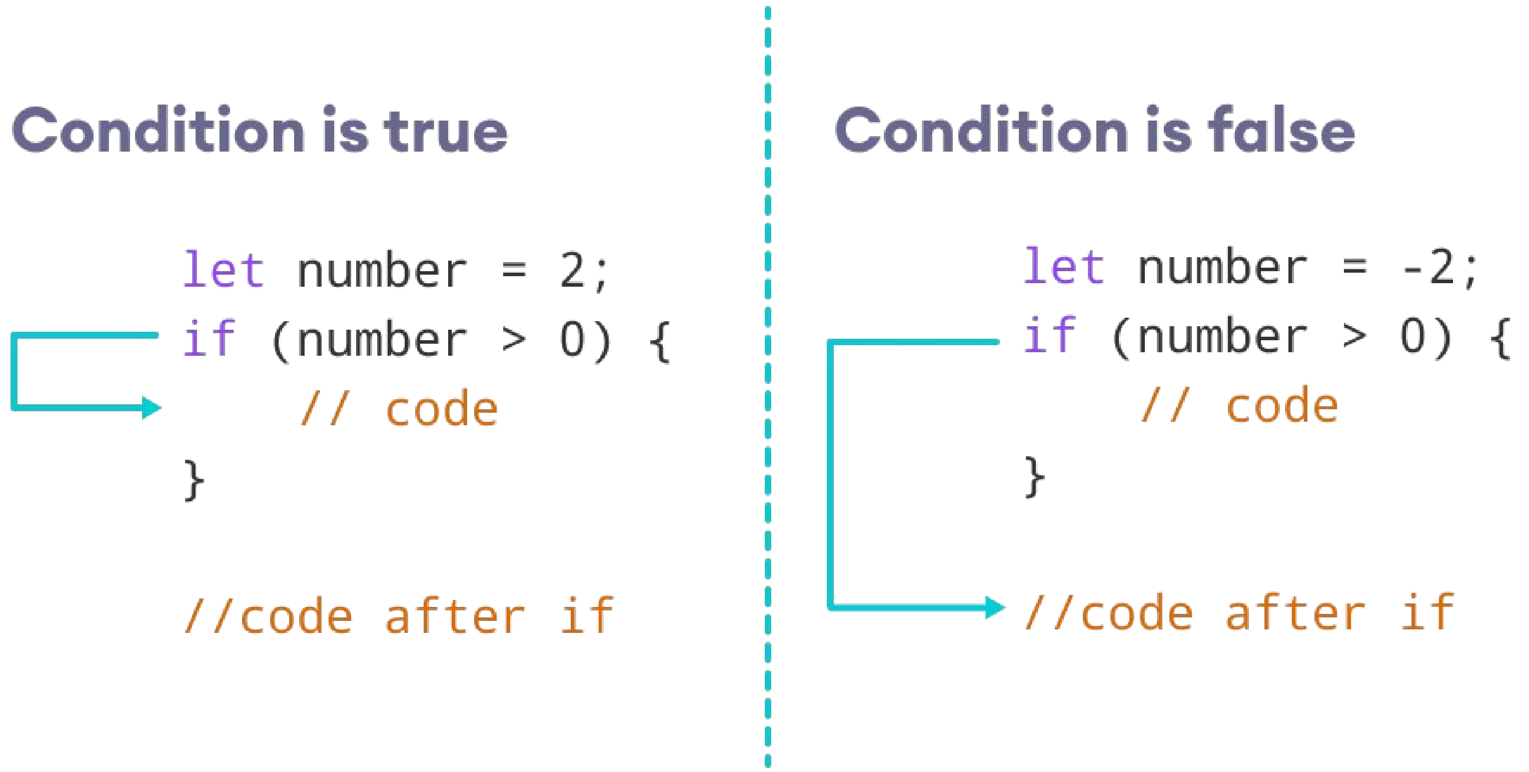


Operator	Description	Example
`&&`	**Logical AND:** `true` if both the operands/boolean values are true, else evaluates to `false`	`true && false; // false`
`||`	**Logical OR:** `true` if either of the operands/boolean values is `true`. evaluates to `false` if both are `false`	`true || false; // true`
`!`	**Logical NOT:** `true` if the operand is `false` and vice-versa.	`!true; // false`

Conditionals

```
if (true) {
    //Do something
} else if (true) {
    //Do something
}else {
    //Do default
}
```

if Condition



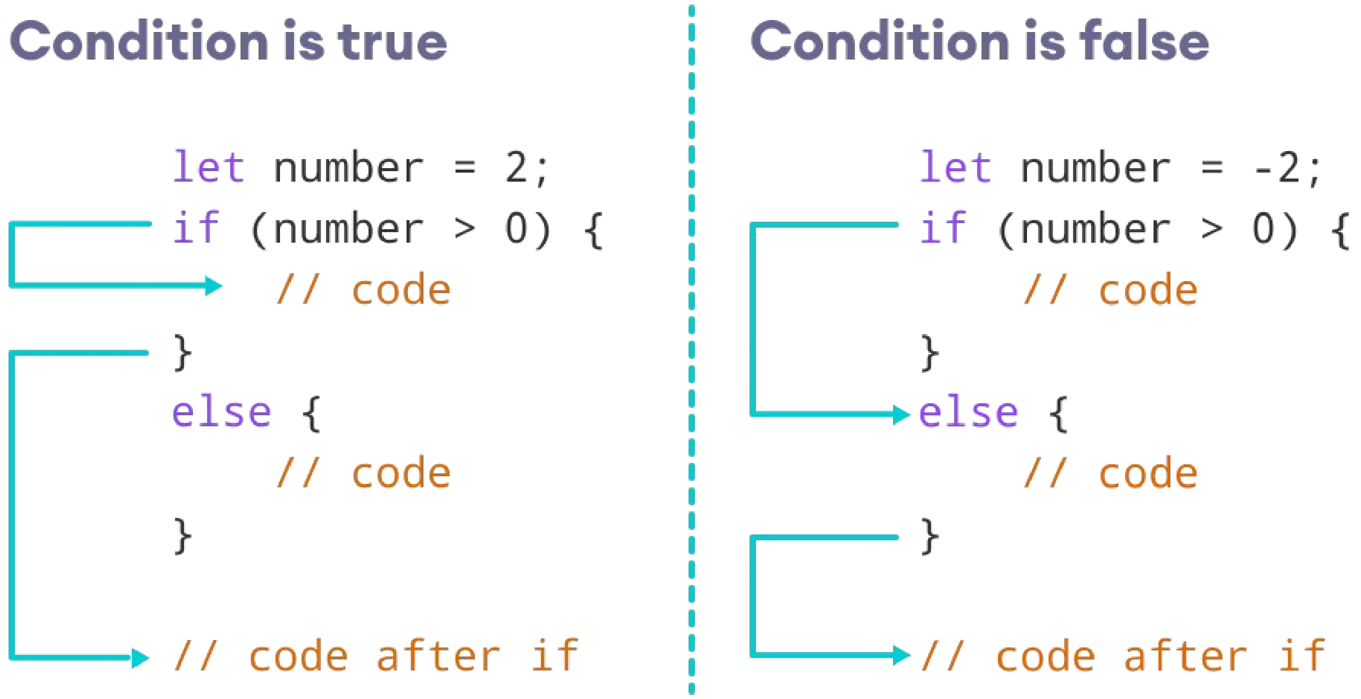
Condition is true

```
let number = 2;
if (number > 0) {
    // code
}

//code after if
```

Condition is false

```
let number = -2;
if (number > 0) {
    // code
}

//code after if
```

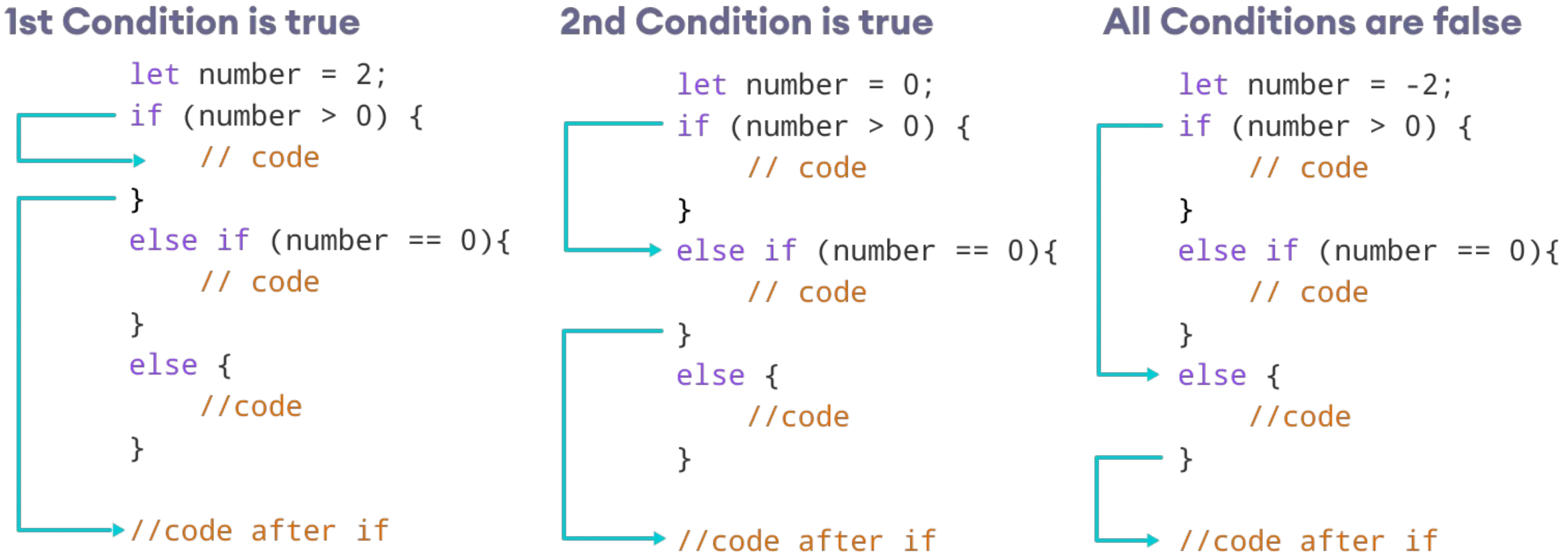
if else Condition



```
let number = 2;
if (number > 0) {
    // code
}
else {
    // code
}

// code after if
```

```
let number = -2;
if (number > 0) {
    // code
}
else {
    // code
}

// code after if
```

if esle if esle Condition



1st Condition is true

```
let number = 2;
if (number > 0) {
    // code
}
else if (number == 0){
    // code
}
else {
    //code
}

//code after if
```

2nd Condition is true

```
let number = 0;
if (number > 0) {
    // code
}
else if (number == 0){
    // code
}
else {
    //code
}

//code after if
```

All Conditions are false

```
let number = -2;
if (number > 0) {
    // code
}
else if (number == 0){
    // code
}
else {
    //code
}

//code after if
```

Strings

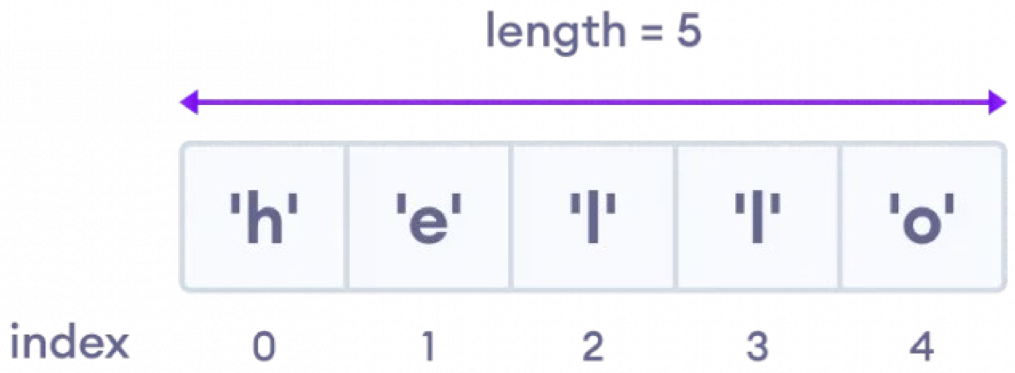
- ‘String’ or “String”
- String and variable concatenation with + operator
- Length
- toUpperCase()
- toLowerCase()
- substr(start, length) -> start နေရာကနေစယူ
- substring(start, end) -> အစကနေစရေပြီးယူ
- trim() -> remove whitespace
- search(“search”)
- replace(“searchvalue”,”newValue”)

Strings

- Backtick sign ` string ${vairable} string`
- Line break “ \n “
- Tab “ \t “
- Back slash “\”

Array( )

let myArray = ['h', 'e', 'l', 'l', 'o'];



Array

- [ ] or Array.of ( )
- Array value different type -> number, string, array
- Multi-dimensional arrays -> array ထဲက array
- length -> array အခန်းအရေအတွက်
- push(“value”) -> array အခန်းထဲကို နောက်နေထည့်
- unshift(“value”) -> array အခန်းထဲကို ရှေ့ကနေထည့်
- pop( ) -> နောက်ဆုံး array အခန်းကို ဖယ်
- shift( ) -> ရှေ့ဆုံး array အခန်းကိုဖယ်

Array

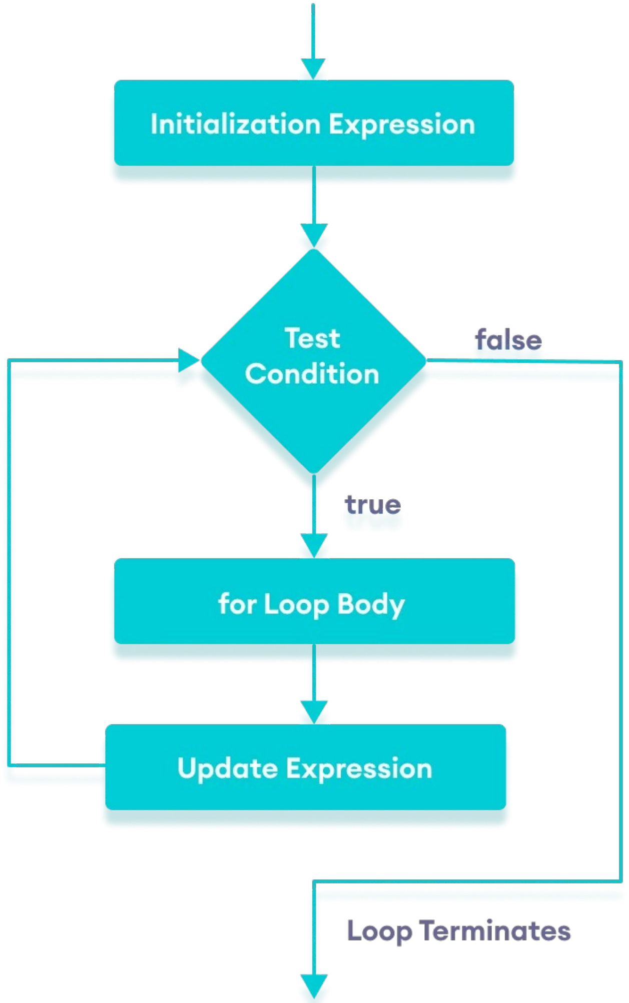
- srot( ) -> အစဉ်လိုက်စဉ်ပေး
- reverse( ) -> ပြောင်းပြန်စဉ်ပေး
- toString( ) -> array ကနေ string ပြောင်း
- split(“ “) -> string ကနေ array ပြောင်း
- concat(sec array) -> နှစ်ခုနဲ့ အထက် array တွေကိုပေါင်းခြင်း
- [...array1,...array2,...array3]

Loops

- for loop
- while loop
- do while loop

for Loop

```
for (initialExpression; condition; updateExpression) {
    // for loop body
}
```



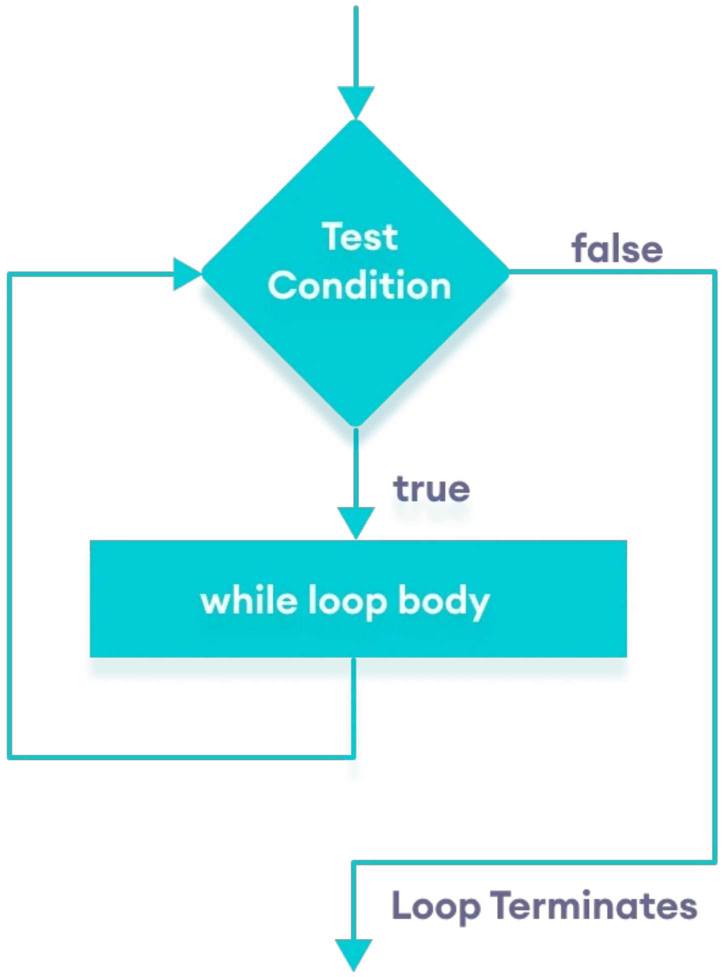
for Loop

```
for (let i=1; i<=5; i++) {

        document.write(`<h1>Hello JS</h1>`);

}
```

i	i <= 5	Output: Hello JS	i++
1	1 <= 5 -> true	Hello JS	2
2	2 <= 5 -> true	Hello JS	3
3	3 <= 5 -> true	Hello JS	4
4	4 <= 5 -> true	Hello JS	5
5	5 <= 5 -> true	Hello JS	6
6	6 <= 5 -> false	-	-

while Loop

```
while (condition) {
    // body of loop
}
```

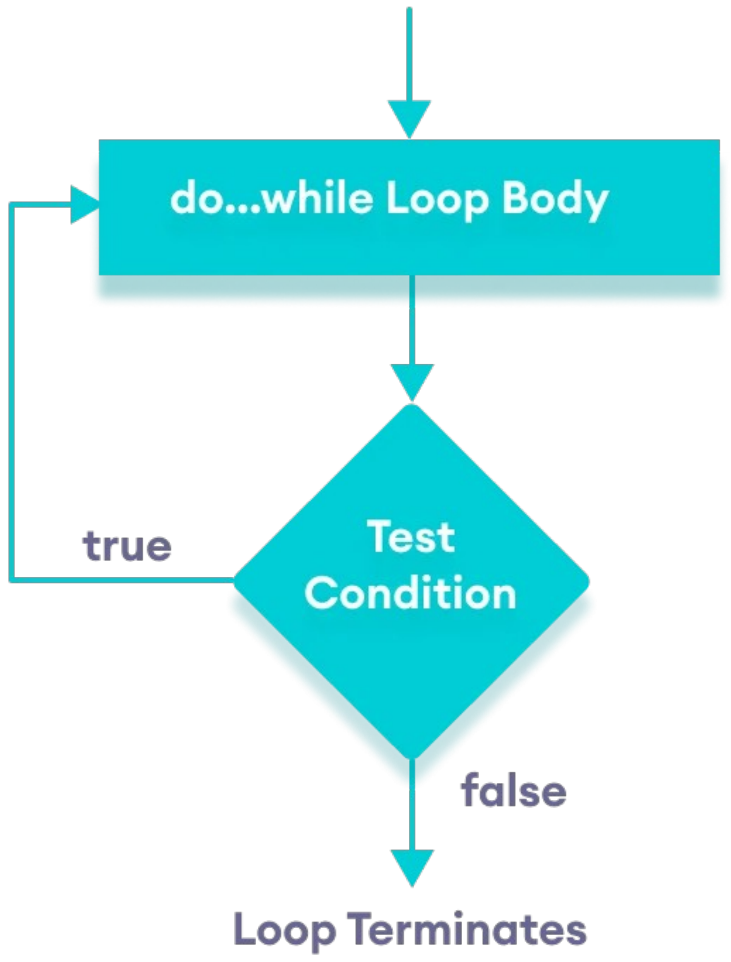


While Loop

```
let num = 1;
while(num<0){
    document.write(`<h1>This is Number ${num}</h1>`);
    num++;
}
```

do while Loop

```
do {
    // body of loop
} while(condition)
```

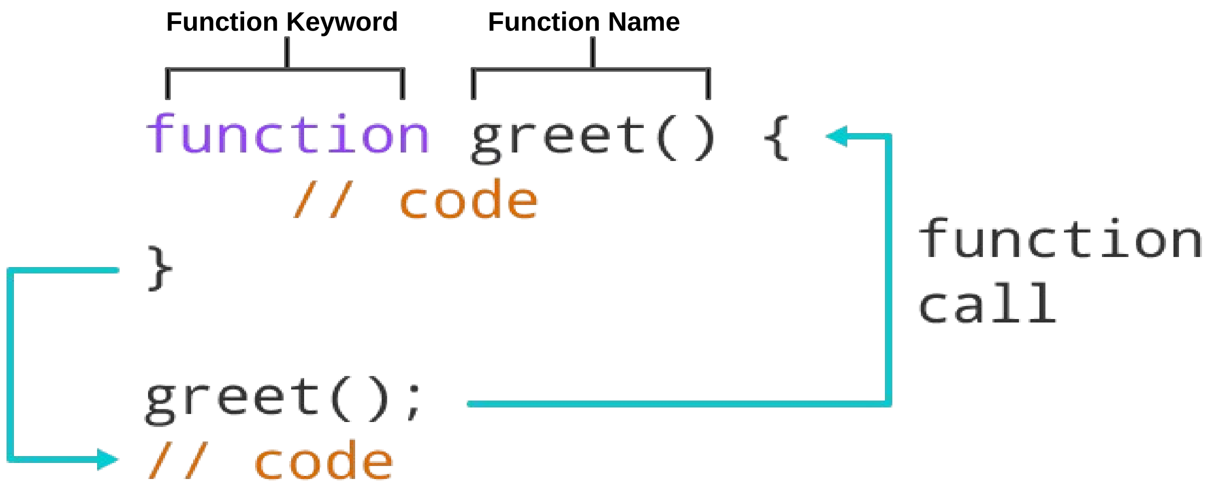


Do While Loop

```
let no1 = 1;
do{
    document.write(`<h1>Number is ${no1}</h1>`);
    no1++;
}while(no1<0);
```

function ()

- Function ဆိုတာ ထပ်ခါတလဲလဲလုပ်ဆောင်ရမယ့် လုပ်ဆောင်ချက်တစ်ခုကို လုပ်ဆောင်ဖို့ ကုဒ်တွေစု ရေးထားတဲ့ ကုဒ်အစုအဝေးဖြစ်သည်။။
- Function name ကိုပြန်ခေါ်မှသာလျှင်အလုပ်လုပ်မည်။

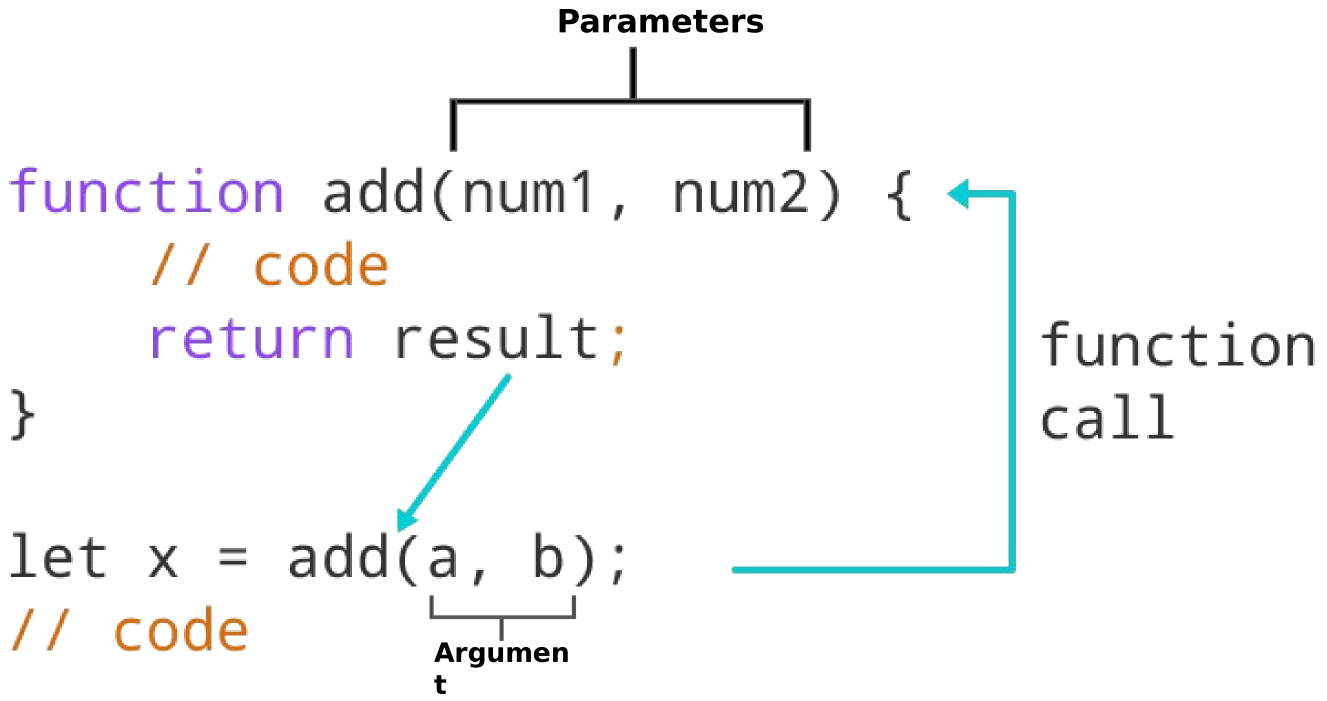


```
function greet() {
    // code
}

greet();
// code
```

functions ()

```
function getData() {
//do something
}
function getData(color) {
//do something
}
function getData(color, age) {
//do something
}
getData('green', 24)
getData('black')
```

function return



```
function add(num1, num2) {
    // code
    return result;
}

let x = add(a, b);
// code
```

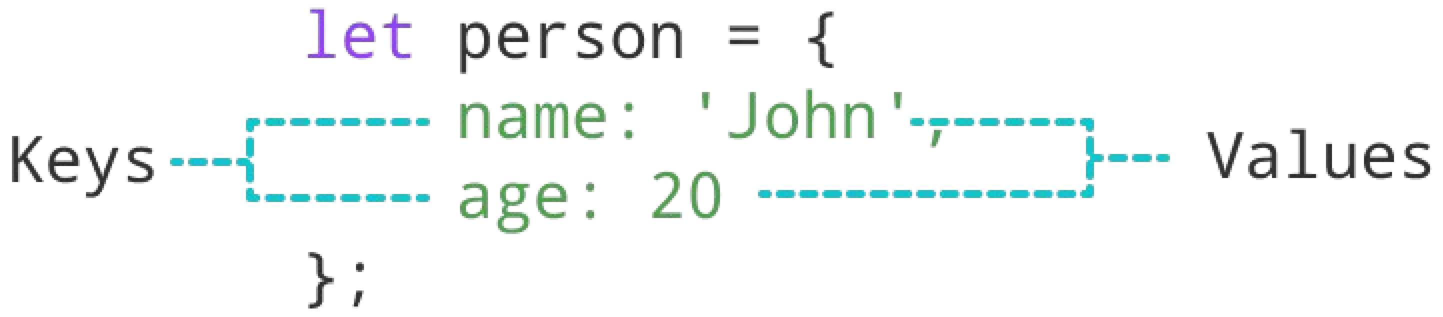
function return

```
function add(a, b) {
    return a + b;
}
let result = add(32, 23);
console.log("The sum is " + result);
```

Objects

let person = {name:John, age:'20'}

//person.name



```
let person = {
    name: 'John',
    age: 20
};
```

Objects

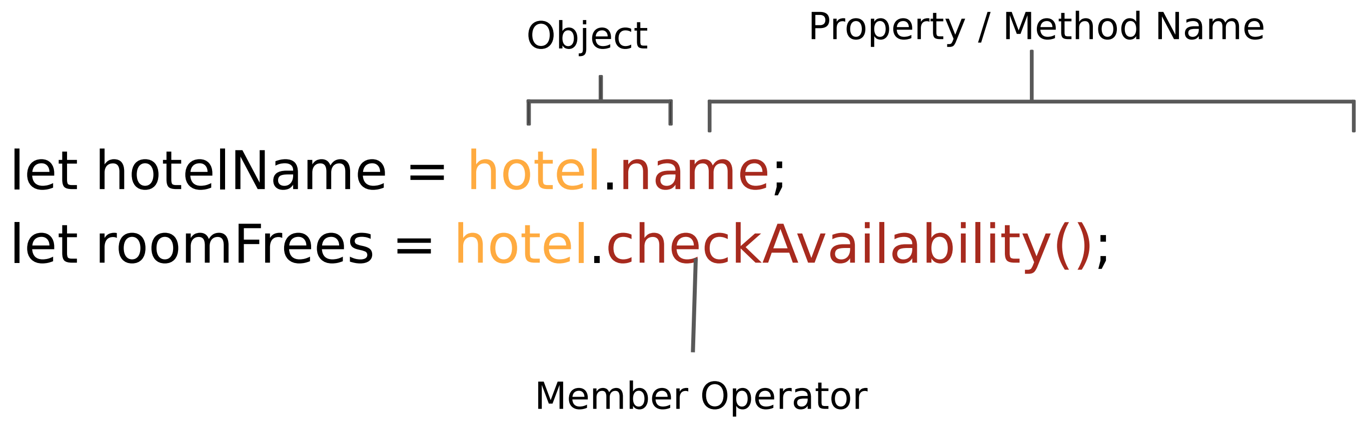
```
let hotel = {
        name: 'XYZ Hotel',
        rooms: 40,
        booked: 25,
        gym: true,
        roomTypes: ['twin','double','suite'],

        checkAvailability: function(){
            return this.rooms - this.booked;
        }
    }
```

IN AN OBJECT:
VARIABLES BECOME
KNOWN AS PROPERTIES

IN AN OBJECT:
FUNCTIONS BECOME
KNOWN AS METHODS

● Object
● Key
● Value

Objects



```
let hotelName = hotel.name;
let roomFrees = hotel.checkAvailability();
```

Document Object Model - DOM

- DOM ဆိုတာ HTML Document တွေကို JavaScript သုံးပြီး စီမံနိုင်တဲ့နည်းပညာ
- DOM ဆိုတာ JavaScript ရဲ့ အရေးကြီးဆုံး အစိတ်အပိုင်းဖြစ်
- အဓိကအကျဆုံး Object သုံးခု
- Global JavaScript Objects
- Browser Object Model
    - Navigator
    - Window
- Document Object Model

Navigator

Browser တစ်ခုလုံးနဲ့ သက်ဆိုင်တဲ့ အချက်အလက်

- navigator.connection
- navigator.deviceMemory
- navigator.languages
- navigator.online
- navigator.userAgent

Window

ဖွင့်ထားတဲ့ Tab ရဲ့ အချက်အလက်

- window.clientInformation
- window.location
- window.location.href
- window.outerHeight
- window.outerWidth
- window.screen

Document Object Model

DOM is specifies

- Making the model of the HTML page

- Accessing and changing the HTML page

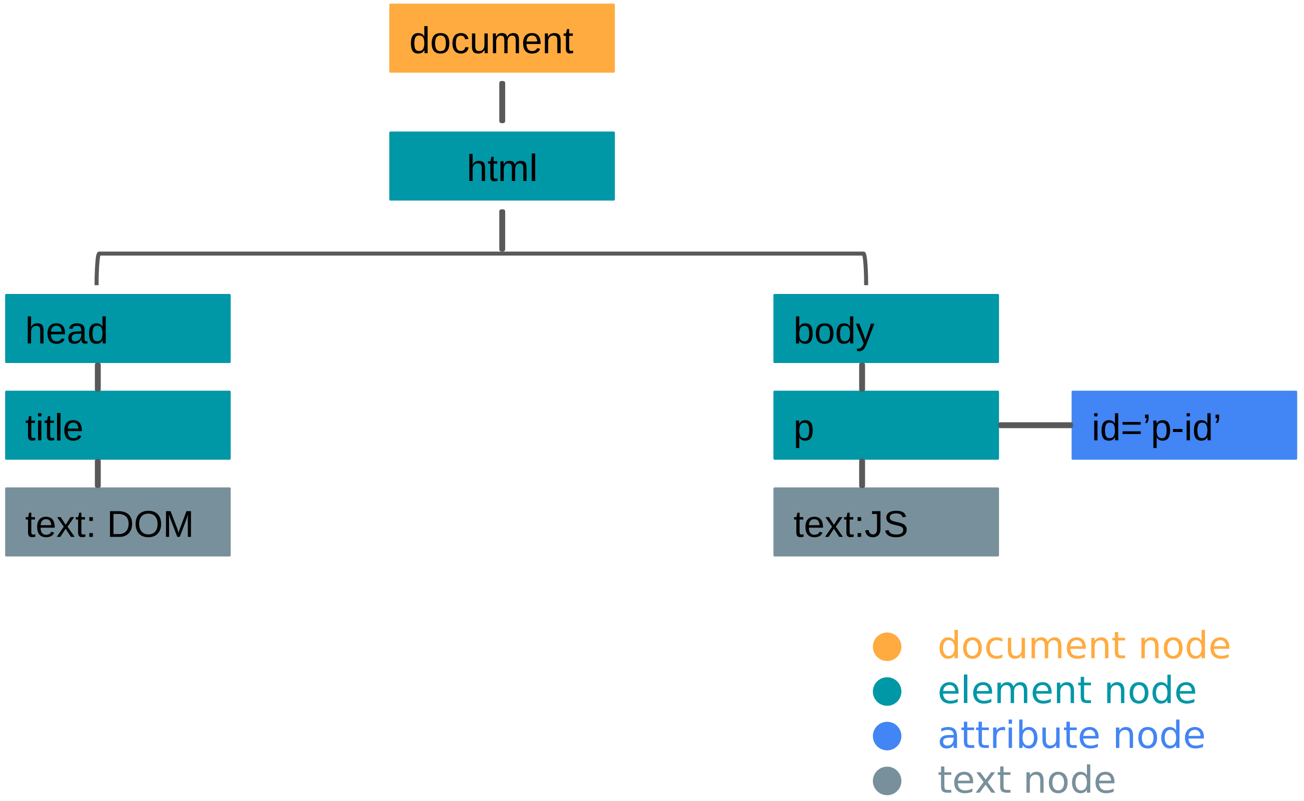
DOM Node

Browser မှာ Web page တစ်ခုကို Run လိုက်တဲ့အခါ browser memory မှာ Model တစ်ခုကို တည်ဆောက်တယ်။ အဲ့ Model ကို DOM Tree လို့ခေါ်တယ်။ DOM Tree မှာ အဓိက Nodes လေးမျိုးပါဝင်တယ်။

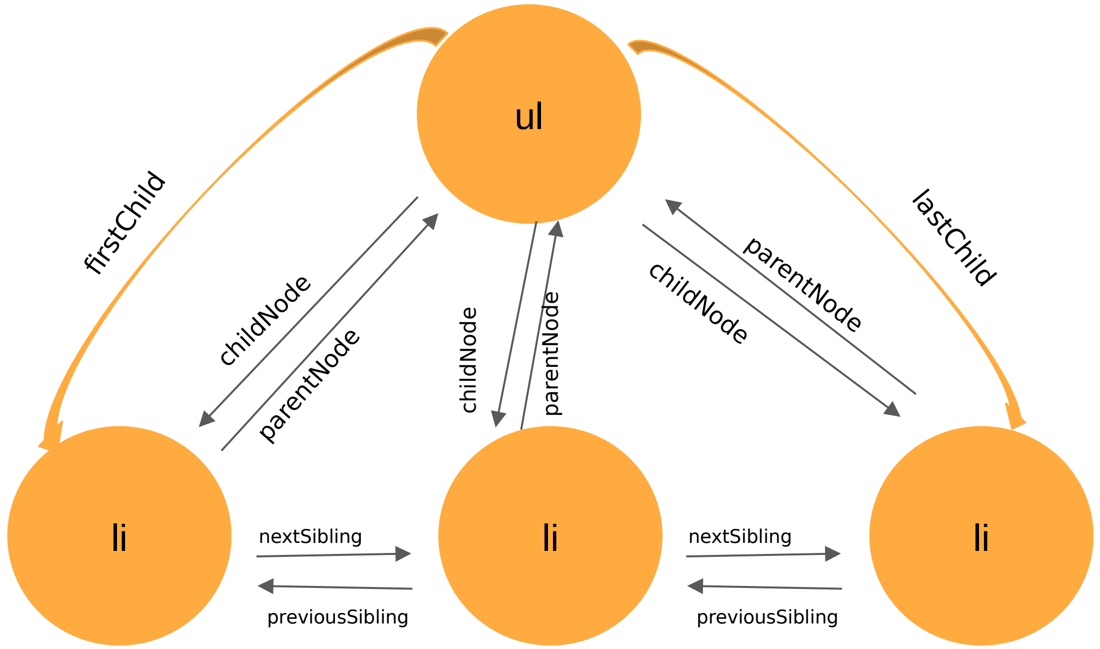
HTML document ထဲမှာရှိနေတဲ့အရာမှန်သမျှကို node လို့ခေါ်တယ်

- Document Node - document တစ်ခုလုံးကို DN လို့ခေါ်
- Element Node - HTML Element တိုင်းကိုခေါ် EN လို့ခေါ်
- Attribute Node - HTML Attribute တိုင်းကို AN လို့ခေါ်
- Text Node - HTML Element တွေထဲမှာရှိတဲ့ စာသားတွေကို TN လို့ခေါ်

DOM Tree


document
html
head
title
text: DOM
body
p
id='p-id'
text:JS
document node
element node
attribute node
text node

Node Relationships

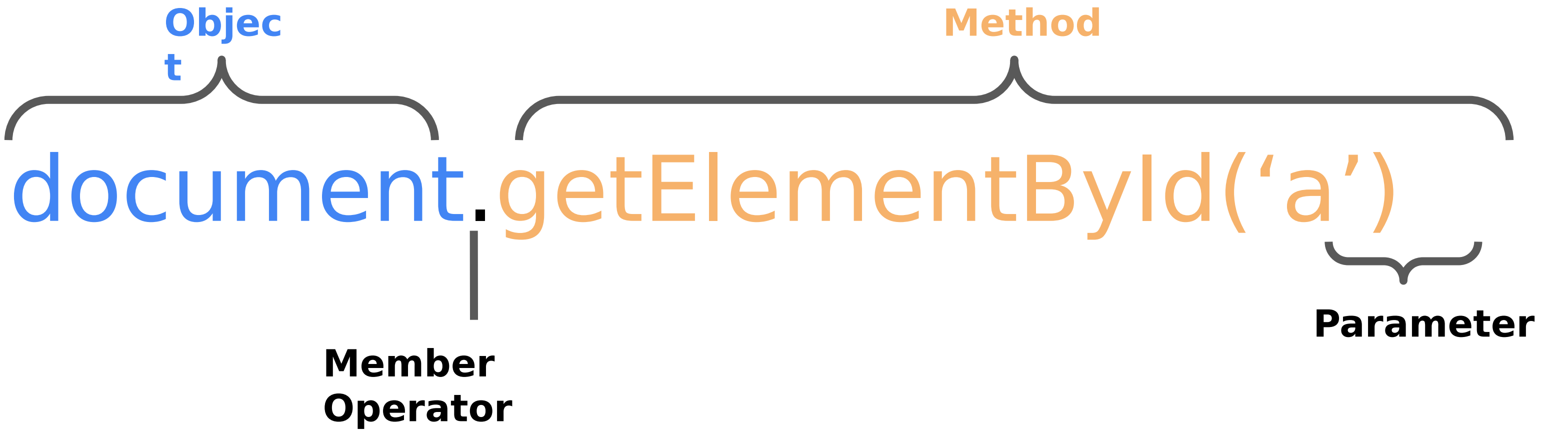


Working With DOM Tree

Step 1 : Select the element

Step 2 : Select element Working with DOM Property and Method

DOM Selector



Get & Update Element Content

- innerHTML - Add and remove HTML content
- innerText - Access and update text

Step 1 : Select the Element

Selector Methods

- document.getElementById(‘id’)
- document.getElementsByClassName(‘class’)
- document.getElementsByTagName(‘tagName’)
- document.querySelector(‘css selector’)
- document.querySelectorAll(‘css selector’)

Step2 : Select element Working with DOM Property and Method

- **selector.property**
- **selector.method()**

DOM Property and Method

- firstChild
- lastChild
- firstElementChild
- lastElementChild
- nextSibling
- previousSibiling
- nextElementSibiling
- previousElementSibiling

DOM Property and Method

- createTextNode(‘text)
- createElement(‘h3’)
- setAttribute(‘class’)
- getAttribute(‘class’)
- removeAttribute(‘class’)
- style

Thank You