06016323 Mobile Device Programming

CHAPTER 4 : JAVASCRIPT FOR MOBILE DEVICE PROGRAMMING

JavaScript (JS)

- JavaScript คือ ภาษาคอมพิวเตอร์ สำหรับการเขียนโปรแกรมบนระบบอินเทอร์เน็ต ที่เกิดมาเพื่อเป็นส่วนเสริมสำคัญของภาษา HTML(Hypertext Markup Language) เปรียบเทียบอย่างง่าย เมื่อ HTML คือตัวโครงสร้างของเว็บไซต์, CSS (Cascading Style Sheets) ช่วยตกแต่งโครงสร้างให้สวยงาม Javascript ทำการเสริมให้เว็บไซต์สามารถโต้ตอบกับผู้ใช้งานได้มากขึ้น

- การเขียน JavaScript นั้น Code ของ JS อยู่ระหว่างแท็ก <script> และ </script> โดยตัวคำสั่ง JavaScript นี้จะอยู่ในส่วนแท็ก <head> และ <body> ของเอกสาร HTML เช่น JavaScript ในแท็ก <body>

ตัวอย่าง

```
<html>
<body>

<script>
document.write("<p> JavaScript with React Native</p>");
document.write("<h1>www.it.kmitl.ac.th</h1>");
</script>

</body>
</html>
```

```
<!doctype html>
<html>
   <head>
          <meta charset="UTF-8">
          <title>ชื่อหัวข้อ</title>
   </head>
   <body>
   </body>
   <script type="text/javascript">
          alert("สวัสดีชาวไอทีลาดกระบัง");
   </script>
</html>
```

```
<meta charset="UTF-8">
<script src="./maccha.js"></script>
```

JavaScript Comment

- การจดบันทึก เพื่ออธิบายคำสั่งหรือขยายความ Statement นั้น ส่วนที่เป็น Comment ไป

ใน JavaScript จะใช้ Comment แบบเดียวกับภาษา C++ และ Java นั่นคือ ใช้เครื่องหมาย // เพื่อประกาศว่า ทุกสิ่งที่อยู่หลัง // เป็น Comment จนจบบรรทัด ซึ่งตัว comment นี้จะไม่แสดงผลบนหน้า web browser เช่น

```
<html>
<body>

<script >
document.write('Hello I am JavaScript') ; //comment:this is a first paragraph
</script>

</body>
 </html>
```

ประเภทของตัวแปรของ JavaScript (Data Types)

- **Primitive Type** - ชนิดตัวแปรมาตราฐาน เก็บค่า value ของ เลข สตริง และ บูลีน แบบภาษาอื่นๆ
  - **Number** - สำหรับภาษานี้จะไม่แยก integer/float (จำนวนเต็ม/ทศนิยม) เลขก็คือเลข ตัวเลขทั้งหมดนับเป็น number แต่ถ้าจะลงลึกถึงรายละเอียดมันคือ double-precision ขนาด 64 bits
  - **String** - เช่นกันกับตัวอักษร ภาษานี้ไม่แยก char/string คือตัวอักษรไม่ว่า 1 ตัวหรือยาวเป็นประโยคจะเรียก string ทั้งหมด และ quote ที่ใช้กับ string ก็สามารถเลือกได้ด้วยว่าจะใช้แบบไหน ระหว่า ' (single quote) กับ " (double quote)
  - **Boolean** - อันนี้ปกติมันก็เป็นค่าได้แค่ true/false อยู่แล้ว ในภาษานี้ก็ไม่มีอะไรผิดปกติไปจากภาษาอื่น
- **Object Type** - ชนิดตัวแปรแบบวัตถุ วิธีใช้เช่นเดียวกับภาษาระดับสูงอื่นๆ ตัวเด่นๆ ที่เจอบ่อยก็เช่น
  - Object
  - Function
  - Array
  - Date
  - RegExp

การประกาศตัวแปรของ JavaScript

- *var* คือ การประกาศตัวแปรสำหรับใช้ใน code ในส่วนที่ถูกรันในส่วนนั้น ๆ อาจจะหมายถึงทั้ง function หรือทั้งไฟล์เลยก็เป็นได้
- *let* คือ การประกาศตัวแปรสำหรับใช้เฉพาะใน scope นั้นหรือเฉพาะใน block นั้น ๆ
- *const* คือ การประกาศตัวแปรแบบค่าคงที่ หรือพูดง่ายๆคือตัวแปรแบบ Read Only

```
var x;
x = 18;
x = 18.75;

x = 245.2e-50;
x = 0xff;

x = NaN;
x = Infinity;
x = -Infinity;
```

```
x = parseInt('1.2'); // 1
x = parseFloat('1.2'); // 1.2
x = Math.pow(2, 3); // 8
x = Math.ceil(1.5); // 2
x = Math.floor(1.5); // 1
x = Math.abs(-10); // 10

x = Math.PI; // <span class="objectBox objectBox-number
">3.141592653589793</span>

x = Math.random(); // <span class="objectBox objectBox-
number ">0.753489534278567</span>

isNaN(x); // true
```

```
var a = ‘I have a iPhone';
var b = "It's gonna be OK.";
var c = 'นี่คือ \'JS\' ล่ะ';
```

```
'this is a pen'.length; // 13
'this is a pen'.indexOf('is'); // 2
'this is a pen'.charAt(1); // h
'this is a pen'.replace('pen', 'apple'); //this is a
apple
'this is a pen'.substring(2,3); // i
```

```
12 == '12' // true
12 === '12' // false
12 != '12' // false
12 !== '12' // true
```

```
var x;
console.log(typeof x); // undefined

var y = null;
console.log(typeof y); // null

console.log(undefined == null); // true
console.log(undefined === null); // false

console.log(typeof z === 'undefined');
console.log(z); // ERROR! ReferenceError: z is not defined
```

```
// ประกาศค่าคงที่ หรือ read only
const MY_VAR = 7;

// error
MY_VAR = 20;

// log 7
console.log(MY_VAR);

// error เพราะเป็น read only ประกาศใหม่ไม่ได้
const MY_VAR = 20;

// error
var MY_VAR = 20;
```

```
// error เช่นกัน
let MY_VAR = 20;
if (MY_VAR === 7) {
  // ใช้ได้ครับ มันจะมองเป็นตัวแปร scoped แบบ let
  const MY_VAR = 20;

  // log 20
  console.log(MY_VAR);

  // error เพราะมันจะพยายามไปเปลี่ยน MY_VAR ของ global
  var MY_VAR = 20;
}
// log 7 เหมือนเดิม
console.log(MY_VAR);
```

```
// ประกาศตัวแปรเป็นตัวเลข
let myVar = 1;
// print ตัวแปรออกมาใน console จะได้ 1
console.log(myVar);
// เปลี่ยนค่าเป็น string
myVar = "My name is noomerzx";
// print ตัวแปรออกมาใน console จะได้ My name
is noomerzx
console.log(myVar);
```

```
const person = {
name: 'Max',  age: 29,
greet() {
 console.log('Hi, I am ' + this.name);
  }
};
const printName = ({ name }) => {
 console.log(name);
};
printName(person);
```

```
const { name, age } = person;
console.log(name, age);

// const copiedPerson = { ...person };
// console.log(copiedPerson);

const hobbies = ['Sports', 'Cooking’];
const [hobby1, hobby2] = hobbies;
console.log(hobby1, hobby2);
```

```
<html>
<body>

<script>
var age= 20 ;
var name=“นายฉลาด รอบครอบเสมอ";
var answer="ถูกต้อง";

document.write(age + "<br>");
document.write(name + "<br>");
document.write(answer + "<br>");
</script>

</body>
</html>
```

สรุปประเภทของการประกาศตัวแปร

ประเภทตัวแปร	ขอบเขต	Assign ค่าใหม่ได้	สามารถเปลี่ยนแปลงค่าได้
const	Block	No	Yes
let	Block	Yes	Yes
var	Block	Yes	Yes

Objects DataType in JS

- JavaScript Object นั้นจะพยายามสมมุติให้ทุกๆอย่าง เป็นวัตถุ (Objects)ให้หมดเลย ทั้งนี้ก็เพื่อความง่ายในการเขียนโปรแกรม โดย ที่วัตถุ (Objects) แต่ล่ะอย่างนั้นจะประกอบด้วย 3 องค์ประกอบหลักคือ
- Object Name
- Property คือ คุณสมบัติของวัตถนั้น (Variables)
- Function ที่ใช้กับ object โดยมีผลกับ object นั้นๆ

```
var pokemon1 = {chue: "ฟุชิงิดาเนะ", sung: 0.7, nak: 6.9};
```

```
var pokemon2 = {
        "chue": "พีคาชู",
        "chue len":
        "พีคา",
        "suan-sung": 0.4,
        "nam-nak-tua": 6.0 };
```

```
alert(pokemon1.chue); // ได้ ฟุชิงิดาเนะ
```

```
alert(pokemon2["chue"]); // ได้ พีคาชู
```

```
var h = {1: "a", 2: "b"};
alert(h[1]); // ได้ a
alert(h["2"]); // ได้ b
```

```
var obji = {a: 1};
alert(obji.b); // ได้ undefined
```

```
var obja = {b: 5};
var prop = "b";
alert(obja[prop]); // ได้ 5
```

Delete Props

```
var obra = {k: 10};
alert(obra.k); // ได้ 10
alert("k" in obra); // ได้ true
delete obra.k; // ลบ
alert(obra.k); // ได้ undefined
alert("k" in obra); // ได้ false
```

```
<html>
 <body>
 <script>
var fruit=new Object();
            fruit.name="apple";
            fruit.color="green";
            fruit.weight=10;
            fruit.price="20 bath";
document.write("This is an" +fruit.name + " it is " + fruit.color + " eight" +fruit.weight+ "and price" +fruit.price);
document.write ( str.link( "http://www.Lotus.com" ) );
</script>
</body>
</html>
```

การสร้าง Method

```
var maeo = {
    chue: "ทามะ",
    rongmiao: function() {
        alert("เหมียวๆ");
    }
};
maeo.rongmiao(); // ได้ เหมียวๆ
```

```
rongmiao = function() {
 alert("เหมียวๆ");
};
var maeo = { chue: "ทามะ", rongmiao: rongmiao };
```

การใช้ this ใน Methods

- this เป็นตัวแปรพิเศษตัวหนึ่งในจาวาสคริปต์ ซึ่งสามารถถูกเรียกใช้จากที่ไหนก็ได้ในโปรแกรมโดยไม่ต้องนิยามขึ้นเอง
- หากเรียกใช้ในเมธอดของออบเจ็กต์ ดังนั้น this ก็คือตัวแปรที่แทนออบเจ็กต์นั้น

```
var Sawasdee = function() {
        alert("สวัสดี ฉันชื่อ" + this.name);
};
var A = { name: ”สุพรรณษา",
            Sawasdee: Sawasdee };
var B = { name: ”สุนิสา",
            Sawasdee: Sawasdee };
A.Sawasdee(); // ได้ สวัสดี ฉันชื่อสุพรรณษา
B.Sawasdee(); // ได้ สวัสดี ฉันชื่อสุนิสา
```

การใช้ this นอก Method

- this จะแทนออบเจ็กต์ global
- global เป็นออบเจ็กต์ที่ห่อหุ้มตัวโปรแกรมทั้งหมด

```
var x = 1.1;
alert(this.x); // ได้ 1.1
alert(this.x === x); // ได้ true
```

ตัวแปรอาร์เรย์

- ตัวแปรแบบอาร์เรย์ (Array) หมายถึงตัวแปรซึ่งมีค่าได้หลายค่าโดยใช้ชื่ออ้างอิงเพียงชื่อเดียว ด้วยการใช้หมายเลขลำดับเป็นตัวจำแนกความแตกต่างของค่าตัวแปรแต่ละตัว

- การเรียกใช้ค่าที่อยู่ในตัวแปรอาร์เรย์ จะระบุด้วยหมายเลขลำดับ หรือ ดัชนี(index) ตามหลังชื่อตัวแปร โดยเริ่มต้นของดัชนีที่ 0 ต่อไปเรื่อย ๆ ตามลำดับ เช่น name[0], name[1],...name[n] เป็นต้น

```
var arr = new Array();
arr[0] = 'a’;
arr[1] = 'b'; // ["a", "b"]
//กำหนดขนาดเริ่มต้นก่อนก็ได้
arr = new Array(3); // [undefined, undefined, undefined]
//แต่ถ้าใส่ไปมากกว่า 1 ตัวจะเป็นการกำหนดสมาชิกเริ่มต้นแทน
arr = new Array(1, 2); // [1, 2]
//หรือรูปย่อ
arr = [1, 2]; // [1, 2]
console.log(typeof arr); // object
console.log(arr instanceof Array); // true
```

```
var ar = [5, 1, 3, 19, 11, 2];
ar.sort();
alert(ar); // ได้ 1, 11, 19, 2, 3, 5
```

```
f = function(a, b) {
return a < b; };
 var ar = ["d", "c", "e", "a", "b"];
ar.sort(f);
alert(ar); // ได้ e,d,c,b,a
```

```
var KMITL = [“กระบัง”, “ลาด”, “ที”, “ไอ”];
var KMITL0 = [];
var i = KMITL.length - 1;
    while (i >= 0) {
        KMITL0.push(KMITL[i]);
        i--;
    }
alert(KMITL0); // ได้ ไอ,ที,ลาด,กระบัง
alert(KMITL.reverse()); // ??
alert(KMITL);  // ??
```

```
var arr = [1, 2, 3, 4];

x = arr.pop(); // x = 4, arr = [1, 2, 3]
arr.push(5); // arr = [1, 2, 3, 5]

x = arr.shift(); // x = 1, arr = [2, 3, 5]
arr.unshift(0); // arr = [0, 2, 3, 5]
```

```
var arr = [1, [2, 3], [4, [5, 'X']]];

arr[0] // 1
arr[1][0] // 2
arr[2][1][1] // 'X'
```

ชนิด	เมธอด	ความหมาย	ตัวอย่าง
การคืนค่าใหม่	slice	หยิบเอาบางส่วนในแถวลำดับ	ar.slice(1,4)
	concat	เชื่อมแถวลำดับเข้าด้วยกัน	ar1.concat(ar2)
	join	เชื่อมสมาชิกในแถวลำดับเข้าด้วยกันเป็นสายอักขระ	ar.join(" ")
การแก้แถวลำดับ (เดิม)	push	ใส่ข้อมูลเพิ่มต่อท้ายให้แถวลำดับ	ar.push(x)
	unshift	ใส่ข้อมูลเพิ่มต่อด้านหน้าแถวลำดับ	ar.unshift(x)
	pop	เอาข้อมูลตัวท้ายสุดออกจาแถวลำไดับ	ar.pop()
	shift	เอาข้อมูลตัวแรกออกจากแถวลำดับ	ar.shift()
	splice	เอาข้อมูลในลำดับที่ระบุออกจากแถวลำดับแล้วแทรกข้อมูลใหม่ลงไป	ar.splice(1,1)
	sort	เรียงข้อมูลในแถวลำดับตามค่า	ar.sort(f)
	reverse	กลับลำดับข้อมูลในแถวลำดับจากหลังมาหน้า	ar.reverse()

Function in JavaScript

- **ฟังก์ชัน** หรือชุดคำสั่งที่รวม Statement การทำงานเอาไว้ด้วยกัน และสามารถเรียกมาใช้งานตามที่เราต้องการได้ การสร้างฟังก์ชันสามารถทำได้ 2 วิธี
  - การสร้างฟังก์ชันมีชื่อ ด้วยคำสั่ง function
  - การสร้างฟังก์ชันไม่มีชื่อด้วยคำสั่ง function

- **fist-class type** คือ ตัวแปรที่ยอมให้ประกาศเป็นตัวแปรได้ ปกติเราจะเจอแต่พวก int string เช่น int x หรือ string s คือสร้างตัวแปรมาเก็บจำนวนเต็มและประโยค แต่สำหรับ js นั้น function ก็สามารถประกาศเป็นตัวแปรได้ด้วยเช่นกัน

Normal Case (Create Funciton)

```
function ชื่อฟังก์ชั่น( ตัวแปรพารามิเตอร์ถ้ามี ) {
Statement  คำสั่งเพื่อให้โปรแกรมทำงาน เช่น การเปรียบเทียบค่า การคำนวณ เป็นต้น
return <- ถ้าต้องการส่งค่าบางอย่างกลับไปนอกฟังก์ชั่น เมื่อการทำงานมาถึงจุดนี้
}
```

การสร้างฟังก์ชันมีชื่อ ด้วยคำสั่ง function

```
function ชื่อตัวแปร(พารามิเตอร์){
    เนื้อหาในฟังก์ชัน
}
```

```
function f(x) {
    return "2x = " + x * 2; }
alert(f(1.1)); // ได้ 2x = 2.2
```

```
function plus(x, y){
    return x + y;
}

var sum = plus(1, 2);

function T(){
    //do nothing
}

var x = T(); // undefined
```

```
function plus(x, y){
    return x + y;
}
function minus(x, y){
    return x - y;
}
plus(10,4); // 14
minus(10,4); // 6
var MP= plus;
MP(10,4); // 14
MP = minus;
MP(10,4); // 6
```

การสร้างฟังก์ชันไม่มีชื่อ ด้วยคำสั่ง function

```
var ชื่อตัวแปร = function(พารามิเตอร์) {
  เนื้อหาในฟังก์ชัน
}
```

```
var f = function(x) {
        alert((x + 8) / 2);
 };
f(4); //  6
f(7); // 7.5
```

```
var f = function(x) {
return (x + 8) / 2; };
var a = f(2);
alert(a); // ได้ 5
```

ฟังก์ชันที่มีพารามิเตอร์หลายตัว

```
var f = function(x, y, z) {
return "x=" + x + " y=" + y + " z=" + z; };
alert(f(2, 3, 4)); // ได้ x=2 y=3 z=4
```

คำสั่ง return

```
var f = function() {
    s = ”นายเก่งได้เกรดมากกว่า";
    return s;
    s += ”นายฉลาดได้เกรดน้อยกว่าเขา";
    alert(s);
};
alert(f()); // นายเก่งได้เกรดมากกว่า
```

```
var f = function(x, y) {
    if (x > y) return "x มากกว่า y";
    else if (x == y) return "x เท่ากับ y";
    else return "x น้อยกว่า y";
};
alert(f(4, 6)); // ได้ x น้อยกว่า y
alert(f(19, 7)); // ได้ x มากกว่า y
```

คำสั่ง return

```
var f = function(x, y) {
    if (x > y) return "x มากกว่า y";
    if (x == y) return "x เท่ากับ y";
    return "x น้อยกว่า y";
};
```

```
var f = function(x) {
return;
};
var a = f(29);
alert(a); // ได้ undefined
```

การค่าคืนของ return

```
var f = function(x) {
    return x / 2 - (x * x) / 10 + 5; };
var a = f(29);
alert(a); // undefined
```

```
var f = function (x) {
    return ( (x / 2) - x * x / 10 + 5 );
};
var a = f(20);
alert(a); // ได้ -25
```

```
function vTest() {
  var x = 1;
  if (true) {
    var x = 2;  // มองเป็นตัวเดียวกันกับด้านนอก
    console.log(x);  // 2
  }
  console.log(x);  // 2
}

function lTest() {
  let x = 1;
  if (true) {
    let x = 2;  // มองเป็นคนละตัวกับด้านนอก
    console.log(x);  // 2
  }
  console.log(x);  // 1
}
vTest()
lTest()
console.log(x) // error
```

```
function gTest() {
  x = 1;
  if (true) {
    x = 2;
    console.log(x);  // 2
  }
  console.log(x);  // 2
}
gTest()
console.log(x) // 2
```

Flow Control

- if...else
- while...do
- for....loop (for..in /for..of)

```
if( x > 0 ){
    //TODO
}
if( x > 0 ){
    //TODO
}
else{
    //TODO
}
for( i = 0; i < 100; i++ ){
    //TODO
}
while( i < 100 ){
    //TODO
}
do{
    //TODO
} while( i < 100 );
```

```
var data = [1, 2, 3, 4];

//standard
for( i = 0; i < data.length; i++ ){
        console.log(data[i]);
}

//for..in
for( i in data ){
        console.log(data[i]);
}

//ES6: for..of
for( i of data ){
        console.log(i);
}
```

```
while (เงื่อนไขที่จะให้ทำซ้ำต่อไป) {
        สิ่งที่ต้องการให้ทำซ้ำ

}
```

```
var i = 1;
var s = "";
        while (i < 12) {
                s += i + "~";
                i += 2;
        }
alert(s); // ได้ 1~3~5~7~9~11~
```

```
var i = 1;
var s = "";
        while (i < 12) {
                s += i + "฿";
                if (i == 7) break;
                i += 2;
        }
alert(s); // ได้ 1฿3฿5฿7฿
```

```
var i = 1;
var s = "";
        while (i < 27) {
                i += 3;
                if (i > 7 && i < 21) continue;
                s += i + “ก";
}
alert(s); // ได้ 4ก7ก22ก25ก28ก
```

```
var s = "";
var i = 0;
do {
        s += "|" + i + "|";
        i++;
} while (i < 9);
alert(s); // ได้ |0||1||2||3||4||5||6||7||8|
```

Apply JS to React Native

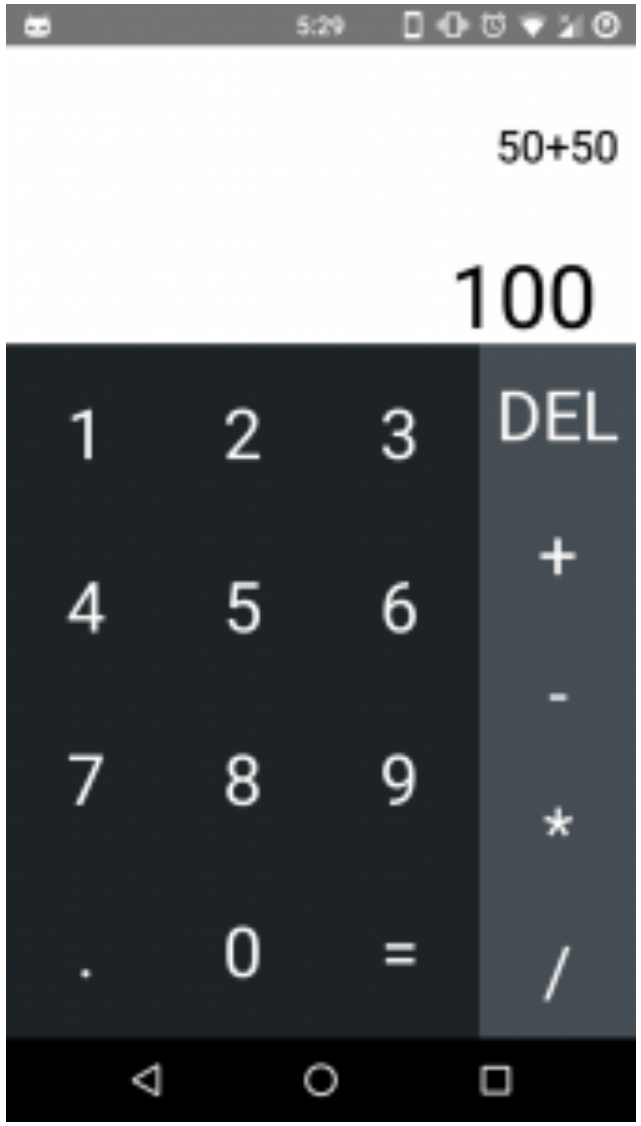
```
import React, { Component } from 'react'
import {
  StyleSheet,
  TouchableOpacity,
  Text,
  View,
} from 'react-native'

class App extends Component {
  state = {
    count: 0
  }

  onPress = () => {
    this.setState({
      count: this.state.count + 1
    }) }
```

```
render() {
    return (
      <View style={styles.container}>
        <TouchableOpacity
         style={styles.button}
         onPress={this.onPress}
        >
         <Text>Click me</Text>
        </TouchableOpacity>
        <View>
          <Text>
            You clicked { this.state.count } times
          </Text>
        </View>
      </View>
    )
  } }
```

```
const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
  button: {
    alignItems: 'center',
    backgroundColor: '#DDDDDD',
    padding: 10,
    marginBottom: 10
  }
})
export default App;
```


5:29
50+50
100
1
2
3
DEL
+
4
5
6
-
7
8
9
*
.
0
=
/