CHAPTER 1

JAVASCRIPT (JS)

342-266 Mobile Web Application Development

Dr.Wachirawut Thamviset

Department of Computer Science, Khon Kaen University

JavaScript : Overview

- ชื่อเป็นทางการคือ ECMAScript
- JavaScript ออกแบบเพื่อเป็น ภาษา Script สำหรับรันภายใต้ Host Environment (เช่น Web Browser)
  - Lightweight, Interpreted, Dynamic, Object-Oriented, Functional ...
  - Script Language for Web
  - Standard : ES6(2015), ES2020 (ปัจจุบัน)

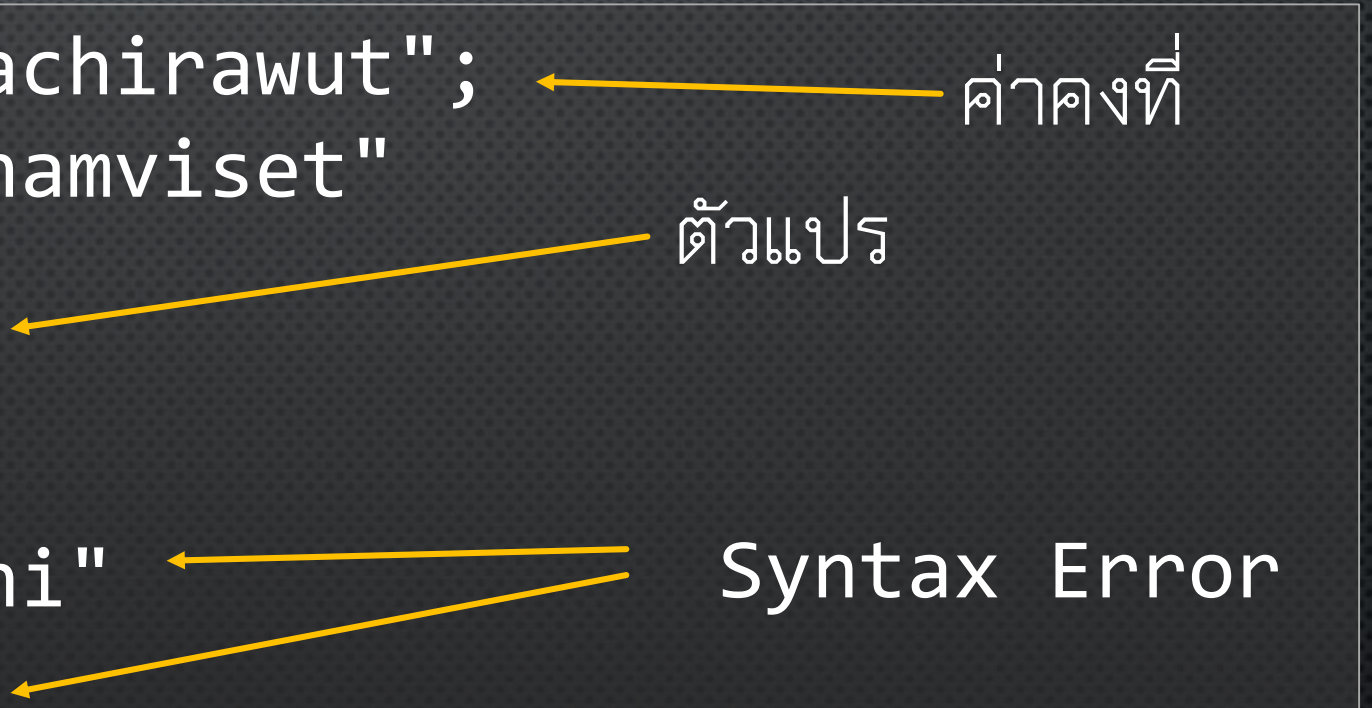
JavaScript Engine

- แต่ละ Web Browser พัฒนา JavaScript Engine ของตนเอง
  - V8 : Google Chrome, NodeJS
  - SpiderMonkey : Firefox
  - JavaScriptCore : Safari, WebKit, Apple
  - Chakra : Microsoft
- ทั้งหมดพัฒนาตามมาตรฐาน ECMAScript แต่ว่า แต่ละค่ายจะมีการเพิ่มความสามารถบางอย่างที่ไม่มีในมาตรฐาน ทำให้ JavaScript Engine แต่ละตัวมีคำสั่งบางอย่างที่ต่างกัน

Syntax : Variables

- สร้างตัวแปร const, var, let

```
const  fname="Wachirawut";
const  lname="Thamviset"
var age = 40
let nick = "W"
var age = 41
let nick = "Wachi"
fname="Wachi"
```



Variable types

- Number
- String
- Boolean
- Array
- Object

Dynamic Typing

ตัวแปรไม่ต้องระบุชนิดข้อมูล

ชนิดข้อมูล เป็นไปตาม ข้อมูล

```
var x = 100;
```

x จะมีชนิดเป็นตัวเลข

```
x = "100";
```

x จะมีชนิดเป็น String

Numbers and Operators

- Numbers : Integer, Floating Point, Double

  Binary ฐาน 2, Octal ฐาน 8, Hex ฐาน 16

```
let a = 5;
let b = 5.521;
typeof  a;
typeof  b;
let c = "56";
c = Number(c) + 3;
```

การปัดจุดทศนิยม

```
b.toFixed(2)
```

```
c=c+3;
```

จะได้ '563'

จะได้ 59

operators

\+ - * / %

** ยกกำลัง

Strings

ข้อความต้องอยู่ในเครื่องหมาย " ' หรือ `

double quote, single quote, grave

```
let firstName = "Wachirawut";
let lastName = 'Thamviset';
let fullName = firstName +' '+ lastName;
let welcome = `Welcome, ${fullName}`;
let out = `${5+10}`;
```

Template String สามารถแทรกตัวแปร หรือ นิพจน์ใน string ได้

String methods

- ความยาว string let name = "Computer";
  name.length; ได้ 8
- ตัวอักษรใน string name[2] ตัวอักษรตัวที่ 3
- ค้นหาคำใน string name.indexOf("TER") ได้ -1
  name.indexOf("Ter") ได้ 5

Arrays

- สร้าง Array ใช้เครื่องหมาย [ ]

```
let  cities   = ["เมืองขอนแก่น","ศิลา","บ้านเป็ด","สำราญ","ท่าพระ"];
let  khonkaen = ["เมืองขอนแก่น", 40000, [16.425,102.821] ];
```

- เข้าถึงข้อมูล

cities[1] ได้ "เมืองขอนแก่น"

khonkaen[2][0] ได้ 16.425

จำนวนข้อมูล

```
cities.length
```

เพิ่มข้อมูลต่อท้าย

```
cities.push("บ้านฝาง")
```

เพิ่มด้านหน้า

```
cities.unshift("บ้านค้อ")
```

Control

- `if( condition ) {  }`
- `if( condition ) {  } else {  }`
- `while (condition) {  }`
- `do {  } while (condition);`
- `for(var i=0;i<10;i++) {  }`
- `for(var x in cities) {  }` x จะเป็น 0,1,2,3,4 …
- `for(var x of cities) {  }` x จะเป็น ชื่อเมือง
- `var a = (condition) ? 'yes' : 'no';`

Objects

- ทุกอย่างของ JS เป็น Object
- สร้าง Standard Object
  - `var a = new Object( );`
  - `var b = { };`
  - `var c = {`
    `name : "Carrot",`
    `color : "#FFAA00",`
    `price : 10`
    `};`

`c['name']` จะได้ `"Carrot"`

`c.name` จะได้ `"Carrot"`

```
for(var x in c){
  console.log(x)
}
```

x จะเป็น name,color,price

```
for(var x of c){
   console.log(x)
}
```

error
for of ใช้ได้กับ array

`c.toString()` แปลงเป็น string

`JSON.stringify(c)` แปลงเป็น JSON string

Functions and Methods

- Basic Function

```
function add(x, y) {
  var total = x + y;
  return total;
}
```

```
function add(...a) {
  var total = 0;
  for(x of a){
    total += x;
  }
  return total;
}
```

```
function add() {
  var sum = 0;
  var j = arguments.length;
  for (var i = 0; i < j; i++) {
    sum += arguments[i];
  }
  return sum;
}
```

Functions and Methods

- Anonymous Function

```
var add = function (x, y) {
  var total = x + y;
  return total;
}
```

- Inner Function

```
var add = function (x, y) {
  function abs(a){
    return (a>0)? a: -a;
  }
  var total = abs(x + y);
  return total;
}
```

- Arrow Function

```
var add = (x, y)=> x + y;
```

- Closures

```
function makeAdd(x) {
  return function(a){
    return x + a;
  }
}

var add10 = makeAdd(10);
var add50 = makeAdd(50);
add10(5);
add50(5);
```

Java Script : Class

- นิยาม Class

```
class Person {
    constructor(name) {
        this.name = name;
    }
    describe() {
        return 'Person called '+this.name;
    }
}
```

Java Script : Class

- นิยาม Class สืบทอดจาก Class ที่มีอยู่

```
class Employee extends Person {
    constructor(name, title) {
        super(name);
        this.title = title;
    }
    describe() {
        return super.describe() + ' (' + this.title + ')';
    }
}
```

Asynchronous JavaScript

- Asynchronous functions

  เป็นฟังก์ชันที่สามารถทำงานไปพร้อมกัน

  สามารถไปเรียกคำสั่งอื่นต่อได้ โดยไม่จำเป็นต้องรอให้เสร็จ

- ตัวอย่าง
  - setTimeout( ) ใช้ตั้งเวลาเรียก function
  - fetch( ) การรับส่งข้อมูล

```
setTimeout(myFunction, 5000);

function myFunction() {
   console.log("I LOVE YOU");
}
```

Asynchronous JavaScript

- Promise: เป็น Object ที่จะใช้เข้าถึง function ที่กำลังรันแบบ Asynchronous

```
function myFunc(onDone, onError) {
// ส่วนของ code ที่จะประมวลผลอะไรบางอย่างที่ใช้เวลานาน
  onDone(data);    // เรียกเมื่อทำงานสำเร็จ
  onError(data);   // เรียกเมื่อเกิด error
}
let myPromise = new Promise(myFunc);

// myFunc จะทำงานและเมื่อเสร็จ คำสั่งใน function then จะถูกเรียก
myPromise.then(
  function(value) { /* code if successful */ },
  function(error) { /* code if some error */ }
);
```

Asynchronous JavaScript

- async สร้างฟังก์ชันที่จะทำงานแบบ Asynchronous
- await เรียกฟังก์ชันแบบ Asynchronous แบบหยุดรอ

```
async function myFunc() {
// ส่วนของ code ที่จะประมวลผลอะไรบางอย่างที่ใช้เวลานาน
  return data; // คืนค่าเมื่อทำงานสำเร็จ
}
async function waiting(){
  console.log("เริ่มทำงาน");
  var res = await myFunc();  // จะหยุดรอให้ myFunc ทำงานให้เสร็จก่อน
  console.log("ทำงานเสร็จแล้ว");
}
waiting(); // จะไปทำคำสั่งต่อไป โดยไม่หยุดรอให้ทำงานเสร็จ
console.log("Not wait");
```