# Basic Javascript

## การประกาศตัวแปร

|  |
| --- |
| var x = 10;  let y = 20;  const z = 30; |

แก้ไขค่าได้, global scope

var x = 10;

แก้ไขค่าได้, local scope

let y = 20;

แก้ไขค่าไม่ได้, local scope

const z = 30;

การแก้ไขค่าตัวแปร

|  |
| --- |
| var x = 10;  let y = 20;  const z = 30;  x = 20;  y = 50;  z = 100; //error |

global scope vs local scope

|  |
| --- |
| {  var x = 10;  let y = 20;  const z = 30;  }  console.log(x);  console.log(y); //error  console.log(z); //error |

Best practice!

ใช้เฉพาะ const และ let

## ตัวแปรประเภท number

|  |
| --- |
| const i = 10;  const f = 1.256;  const n = -10; |

Operators

|  |
| --- |
| const i = 10;  const f = 1.256;  const n = -10;  const sum = i + 10;  const remain = i - 20;  const mul = i \* 5;  const div = i / 3;  const mod = i % 2;  const pow = i \*\* 2;  console.log(sum, remain, mul, div, mod, pow);  //output  // 20 -10 50 3.3333333333333335 0 100 |

Functions

|  |
| --- |
| const pow = Math.pow(2, 5); // 2 ยกกำลัง 5  const floor = Math.floor(5 / 3); // ปัดทศนิยมทิ้ง  const ceil = Math.ceil(5 / 3); // ปัดขึ้น ถ้ามีทศนิยม  const round = Math.round(5 / 3); // ปัดขึ้นถ้าทศนิยมเป็น .5 ขึ้นไป  const rand = Math.random() // สุ่มเลข 0 - 1  const rand1 = Math.random() \* 5; // สุ่มเลข 0 - 5  const rand2 = Math.random() \* 5 + 1 // สุ่มเลข 1 - 6  console.log(pow, floor, ceil, round);  //output  //32 1 2 2 |

## ตัวแปรประเภท string

|  |
| --- |
| const str = 'Hello';  const str2 = "World";  const str3 = `Hahaha`; |

Operators

|  |
| --- |
| const str = 'Hello';  const str2 = "World";  const str3 = `Hahaha`;  const concat = str + " " + str2;  const concat2 = `${str} ${str2} ${str3}`;  console.log(concat + "\n" + concat2);  //output  //Hello World  //Hello World Hahaha |

Functions

|  |
| --- |
| const str = 'Hello';  const str2 = "World";  const str3 = `Hahaha`;  const substr = str.substr(0, 4); // Hell  const indexOf = str.indexOf('ll'); // 2  const splitted = str3.split('a'); // ['H', 'h', 'h', '']  const replaceAll = str3.split('a').join('o') // Hohoho |

## ตัวแปรประเภท boolean

|  |
| --- |
| const correct = true;  const fail = false;  const more = 5 > 3; // true  const less = 3 < 1; // false  const gte = 5 >= 3; // true  const lte = 6 <= 1; // false  const not = !fail; // true  const equal = 10 == '10'; //true  const notEqual = 10 === '10'; //false  const isNotEqual = 10 != '10'; //false  const isNotEqual2 = 10 !== '10' //true  const and = correct && fail; // false  const or = correct || fail; // true |

## If/Else, Switch case

If/Else

|  |
| --- |
| const score = 50;  if (score >= 80)  console.log('A')  else if (score >= 70)  console.log('B')  else if (score >= 60)  console.log('C')  else if (score >= 50)  console.log('D')  else  console.log('E') |

Switch case

|  |
| --- |
| let light = 'red';  switch (light) {  case 'red': console.log("STOP!!"); break;  case 'yellow': console.log('Slow down..'); break;  default: console.log('GO!');  } |

## Loop

For loop

|  |
| --- |
| for (let i = 0; i < 20; i++) {  console.log(i);  } |

While loop

|  |
| --- |
| let i = 0;  while (i < 20) {  console.log(i);  i++;  } |

Do while

|  |
| --- |
| let i = 0;  do {  console.log(i);  i++;  } while (i < 20); |

## ตัวแปรประเภท Array

|  |
| --- |
| const arr1 = [1, 'str', true, [1, 2, 3]]  const commonArray = [1, 2, 3, 4, 5] |

Operators

|  |
| --- |
| const arr = [1, 2, 3, 4, 5];  //accessing  console.log(arr[0]); // 1  console.log(arr[1]); // 2  // assigning  arr[0] = 10;  console.log(arr); // [10, 2, 3, 4, 5];  // destructuring  const [elem1, elem2, ...rest] = arr;  console.log(elem1); // 10  console.log(elem2); // 2  console.log(rest); // [3, 4, 5]  // spread operator  const copy = [...arr]; // [10, 2, 3, 4, 5]  const merged = [...arr, 6, 7, 8] // [10, 2, 3, 4, 5, 6, 7, 8] |

Functions

|  |
| --- |
| const arr = [1, 2, 3, 4];  // push เพิ่มค่าต่อท้ายในอาเรย์  arr.push(5);  console.log(arr); // [1, 2, 3, 4, 5]  // ตรวจสอบว่ามีค่าอยู่ในอาเรย์หรือไม่  arr.includes(1); // true  // forEach วนลูปทุกตัวในอาเรย์  arr.forEach((item, index) => {  console.log(`index: ${index} = ${item}`);  })  //output  // index: 0 = 1  // index: 1 = 2  // index: 2 = 3  // index: 3 = 4  // index: 4 = 5  // map วนลูปแปลงอาเรย์เดิมแล้ว return เป็นอาเรย์ใหม่  const powArr = arr.map((item, index) => {  return item \* item;  });  // [1, 4, 9, 16, 25]  // filter วนลูปกรองค่าที่ผิดเงื่อนไขออกจากอาเรย์  const filtered = arr.filter((item, index) => {  return index % 2 === 0  });  // [1, 3, 5]  // reduce วนลูปในอาเรย์แล้ว return ออกมาเป็นค่าใหม่ตามที่คำนวณ  const sum = arr.reduce((prev, item, index) => {  return prev + item;  }, 0);  // 15  // find วนลูปหาค่าแรกสุดที่ตรงเงื่อนไข ถ้าไม่พบจะมีค่าเป็น null  const found = arr.find((item, index) => {  return item % 2 === 0;  });  // 2  // findIndex วนลูปหา index แรกสุดที่ตรงเงื่อนไข ถ้าไม่พบจะมีค่าเป็น -1  const foundIndex = arr.findIndex((item, index) => {  return item % 2 === 0;  });  // 1  const sorted = arr.sort((a, b) => {  return b - a;  });  // [ 5, 4, 3, 2, 1 ] |

## ตัวแปรประเภท Object

|  |
| --- |
| const cat = { color: 'orange', age: 1 }  const p1 = { name: 'John Doe', married: false }  const p2 = { name: 'John Doe', married: true, children: [p1] } |

Operators

|  |
| --- |
| const p1 = { name: 'John Doe', age: 10, height: 160 };  // accessing  console.log(p1.age); // 10  console.log(p1.name); // John Doe  console.log(p1['age']); // 10  console.log(p1['name']);// John Doe  const key = 'age';  console.log(p1[key]); // 10  // assigning  p1.age = 18;  p1['name'] = p1['name'] + ' 123';  // destructuring  const { name, age, ...rest } = p1;  console.log(name); // John Doe 123  console.log(age); // 18  console.log(rest); // { height: 160 }  // spread operator  const copy = { ...p1 }; // { name: 'John Doe', age: 15, height: 160 }  const append = { ...p1, weight: 60 }; // { name: 'John Doe', age: 15, height: 160, weight: 60 }  const userData = { weight: 60 }  const merged = { ...p1, ...userData }; // { name: 'John Doe', age: 15, height: 160, weight: 60 } |

Functions

|  |
| --- |
| const obj = {  'DQX1': {  name: 'Tanakorn Karode',  age: 24  },  'DQX2': {  name: 'Madsoffee Yako',  age: 23  },  'DQX3': {  name: 'Usman Sulong',  age: 23  },  }  // Object.values แปลง object เป็นอาเรย์ เก็บค่าของ object  console.log(Object.values(obj));  // output  // [  // { name: 'Tanakorn Karode', age: 24 },  // { name: 'Madsoffee Yako', age: 23 },  // { name: 'Usman Sulong', age: 23 }  // ]  // Object.keys แปลง object เป็นอาเรย์ เก็บ key ของ object  console.log(Object.keys(obj));  // output  // [ 'DQX1', 'DQX2', 'DQX3' ]  // Object.entries แปลง object เป็นอาเรย์ เก็บ key และ value ของ object  // output  //[  // [ 'DQX1', { name: 'Tanakorn Karode', age: 24 } ],  // [ 'DQX2', { name: 'Madsoffee Yako', age: 23 } ],  // [ 'DQX3', { name: 'Usman Sulong', age: 23 } ]  //] |

## Date / Time

การสร้าง object วันที่

|  |
| --- |
| const now = new Date(); // เวลาปัจจุบัน  const specY = new Date('2020'); // 2020-01-01T00:00:00.000Z  const specYM = new Date('2020-11'); // 2020-11-01T00:00:00.000Z  const specYMD = new Date('2020-11-20'); // 2020-11-20T00:00:00.000Z  const specYMDHm = new Date('2020-11-20T12:10'); // 2020-11-20T05:10:00.000Z  const specYMDHms = new Date('2020-11-20T12:10:30'); // 2020-11-20T05:10:30.000Z |

Functions

|  |
| --- |
| const now = new Date();  const specY = new Date('2019');  // แปลง date object เป็น millisec (number)  console.log(now.valueOf()) // 1605779249993  // เปรียบเทียบเวลา  console.log(now.valueOf() > specY.valueOf()) // true  // เช็คว่าวันเดียวกันหรือไม่  const sameDate = now.getFullYear() === spec.getFullYear() &&  now.getMonth() === spec.getMonth() &&  now.getDate() === spec.getDate()  // แสดงข้อความเวลาตาม format  const options = { weekday: 'long', year: 'numeric', month: 'long', day: 'numeric' };  const usFormat = specY.toLocaleDateString('en-US', options)  // Tuesday, January 1, 2019  const thFormat = specY.toLocaleDateString('th-TH', options)  // อังคาร, มกราคม 1, 2563 |

## Function

การสร้างฟังก์ชัน

|  |
| --- |
| // ใช้คีย์เวิร์ด function และกำหนดชื่อฟังก์ชัน  function tripplePrint() {  console.log('one');  console.log('two');  console.log('three');  }  // เก็บ anonymous function ในตัวแปร  const tripplePrintValues = function (a, b, c) {  console.log('one', a, b, c);  console.log('two', a, b, c);  console.log('three', a, b, c);  }  // เก็บ arrow function ในตัวแปร  const addValues = (a, b, c) => {  return a + b + c;  }  // ถ้าไม่มีปีกกา { } จะถือว่าเป็นการ return ค่า  const addValuesShort = (a, b, c) => a + b + c; |

การเรียกใช้

|  |
| --- |
| // เรียกใช้เหมือนกันทุกแบบ  tripplePrint();  tripplePrintValues(1, 2, 3);  const result = addValues(1, 2, 3);  const result2 = addValuesShort(1, 2, 3); |

Callback function

|  |
| --- |
| // ฟังก์ชันสามารถรับค่าฟังก์ชันเข้าไปเรียกใช้ได้  const cbAdd = (a, b, cb) => {  return cb(a + b);  }  const answer = cbAdd(1, 2, (result) => result \* result);  console.log(answer);  // answer = cbAdd(1, 2, (result) => result \* result);  // cb(1+2)  // cb(3)  // (3) => 3 \* 3  // return cb(3)  // return 3 \* 3  // output = 9 |

Curried function

|  |
| --- |
| // ฟังก์ชันสามารถ return ฟังก์ชันได้  const curriedAdd = (a) => {  return (b) => {  return (c) => {  return a + b + c;  }  }  }  const firstAdd = curriedAdd(1); // (b) => (c) => 1 + b + c;  const secondAdd = firstAdd(2); // (c) => 1 + 2 + c;  const thirdAdd = secondAdd(3); // 1 + 2 + 3  console.log(thirdAdd); // 6  console.log(curriedAdd(1)(2)(3)) //6 |

## Asynchronous functions

Built-in Asynchronous functions

|  |
| --- |
| // รอเวลา 1000 millisec แล้วแสดงคำว่า Hello  setTimeout(() => {  console.log('Hello')  }, 1000);  // แล้วแสดงคำว่า Hello ทุก ๆ 1000 millisec  setInterval(() => {  console.log('Hello')  }, 1000); |

การสร้าง Promise

|  |
| --- |
| // ไม่สามารถ return ค่าออกมาได้  const asyncAdd = (a, b) => {  setTimeout(() => {  return a, b;  }, 1000);  }  // ไม่สามารถ return ค่าออกมาได้  asyncAdd(1, 2);  // ส่งค่าออกโดยการใช้ callback function อาจเกิดปัญหา callback hell  const asyncAddCb = (a, b, cb) => {  setTimeout(() => {  cb(a + b);  }, 1000);  }  // เก็บผลลัพธ์ใน result  asyncAddCb(1, 2, (result) => { })  // ใช้ Promise ที่สามารถแก้ปัญหา callback hell ได้  const asyncAddCb = (a, b) => {  return new Promise((resolve, reject) => {  setTimeout(() => {  resolve(a + b)  }, 1000);  })  }  // ใช้ .then เพื่อดึงค่า  asyncAddCb(1, 2).then(result => { }) |

ดึงค่าจาก Promise ด้วยวิธีที่สั้นกว่า โดยใช้ async/await

|  |
| --- |
| // ใช้ async/await เพื่อดึงค่า  const asyncFn = async () => {  // await ต้องถูกเรียกในฟังก์ชันที่มี async เท่านั้น  const result = await asyncAddCb(1, 2);  } |

## Module

การ export ค่า

|  |
| --- |
| // export ย่อย ใช้ได้หลายครั้งต่อไฟล์  const abc = 'abc'  // es5 (node js)  exports.d = abc;  // es6 (react, angular, vue, nest)  export const d = abc;  // export ระดับไฟล์ ใช้ได้ครั้งเดียวต่อไฟล์  const data = {  x: 10,  a: 20  }  // es5 (node js)  module.exports = data;  // es6 (react, angular, vue, nest)  export default data; |

การ import ค่า

|  |
| --- |
| // ดึงตัวแปรมาจากไฟล์ lib ที่อยู่ในโฟลเดอร์เดียวกัน  // es5  // ดึงค่าระดับไฟล์  const data = require('./lib');  // ดึงค่าระดับย่อย  const { d } = require('./lib');  // es6  // ดึงค่าระดับไฟล์  import data from './lib';  // ดึงค่าระดับย่อย  import { d } from './lib'; |

## Links

Problems

<https://github.com/tanakorn0314/js-challenge.git>

React

[Basic React](https://docs.google.com/document/d/1-Tq4F3CjscvKC81eDZ4txk_7GxmkmBqJGAjqlgxWo3I/edit#heading=h.bunbyt91oi0l)

Demo Project

<https://coc-training-workshop.web.app/>