

Javascript ES6, ES7

BY PAIBOON PANUSBORDEE
FRONT-END PROGRAMMER BOOTCAMP



Shorthand Syntax



The Ternary Operator

```
const x = 20;  
let answer;  
if (x > 10) {  
  answer = 10;  
} else {  
  answer = x;  
}
```



```
const x = 20;  
let answer = x > 10 ? 10 : x
```

Short-circuit Evaluation Shorthand

```
let variable2;  
if (variable1 !== null || variable1 !== undefined || variable1 !== '') {  
  variable2 = variable1;  
} else {  
  variable2 = 'new';  
}
```



```
let variable2 = variable1 || 'new';
```

Short-circuit Evaluation Shorthand

```
let variable1;  
let variable2 = variable1 || '';  
console.log(variable2 === ''); // prints true
```

```
variable1 = 'foo';  
variable2 = variable1 || '';  
console.log(variable2); // prints foo
```

Declaring Variables Shorthand

```
let x = 123;  
let y = "I'm string";  
let z = [1,2,3];
```



```
let x = 123,  
    y = "I'm string",  
    z = [1,2,3];
```

Arrow Function – หลาย parameters

```
let addSync = function (a, b) {  
  return a + b;  
}
```



```
let addSync = (a, b) => { return a + b };
```



```
let addSync = (a, b) => a + b ;
```

Arrow Function – parameter เดียว

```
let powerOfTwo = function(a) {  
  return a * a;  
};
```



Function หลาย
บรรทัดได้

```
let powerOfTwo = a => (  
  a * a  
);
```



Function
บรรทัดเดียว
เท่านั้น!!

```
let powerOfTwo = a => a * a ;
```


Arrow Function – parameter เดี่ยว

```
list.forEach(function(item) {  
  console.log(item);  
});
```



```
list.forEach(item => console.log(item));
```

Arrow Function – ไม่มี parameter

```
let printHello = function() {  
  console.log("Hello World");  
};
```



```
let printHello = () => console.log("Hello World");
```

Arrow Function – Return Object

```
let getEmployees = function () {  
  return {  
    "id" : 1,  
    "name" : "Somchai"  
  };  
}
```



```
let getEmployees = () => ({ "id" : 1, "name" : "Somchai" });
```

Multi-line String Shorthand

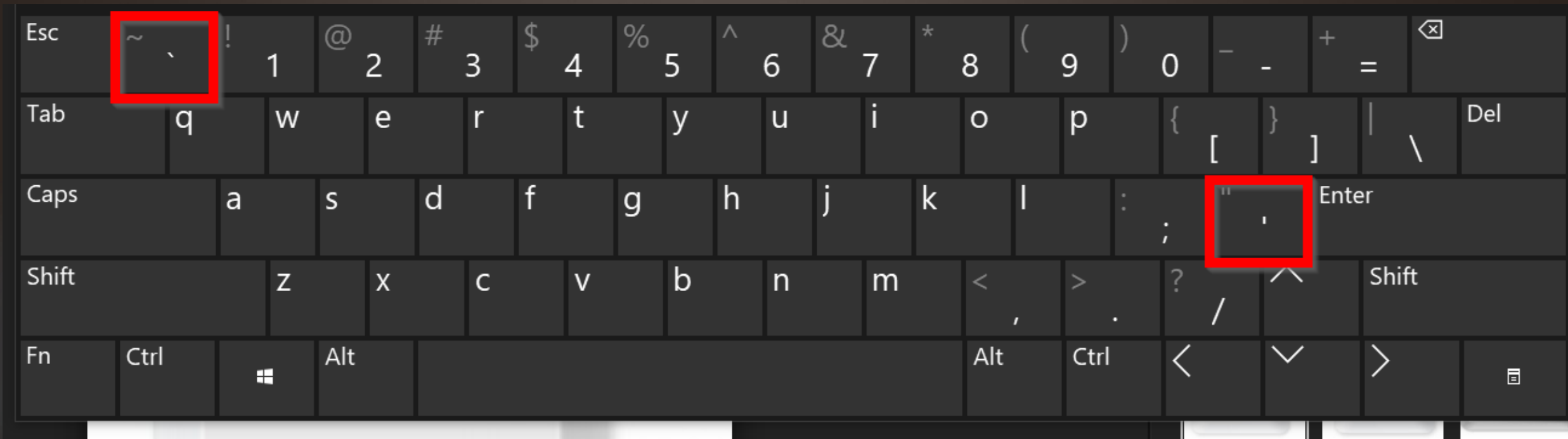
```
const description = 'JavaScript is the programming language of HTML and the Web.\n'+ 'JavaScript is easy to learn.\n'+ 'This tutorial will teach you JavaScript from basic to advanced.\n';
```



```
const description = `JavaScript is the programming language of HTML and the Web.
JavaScript is easy to learn.
This tutorial will teach you JavaScript from basic to advanced.`;
```

ไม่ต้องใช้ Single Quote!!

Multi-line String Shorthand



สำหรับ **Windows** กรุณาเปลี่ยนปุ่มเปลี่ยนภาษาของท่านจาก **Grave Accent (`)** เป็น **Shift+Alt !!**

Template Literals

```
const welcome = 'You have logged in as ' + first + ' ' + last + '.' ;  
const db = 'http://' + host + ':' + port + '/' + database;
```



```
const welcome = `You have logged in as ${first} ${last}`;  
const db = `http://${host}:${port}/${database}`;
```

Destructuring Assignment Shorthand

```
let foo = ['one', 'two', 'three'];  
let [one, two, three] = foo;
```

```
console.log(one); // "one"  
console.log(two); // "two"  
console.log(three); // "three"
```



Destructuring Assignment Shorthand

```
let a, b;
```

```
[a, b] = [1, 2];  
console.log(a); // print 1  
console.log(b); // print 2
```



Destructuring Assignment Shorthand

```
let {c,d} = {c:1,d:2};  
console.log(c); // print 1  
console.log(d); // print 2
```



Spread Operator Shorthand

```
// joining arrays
const odd = [1, 3, 5];
const nums = [2, 4, 6].concat(odd);
console.log(nums);
// [ 2, 4, 6, 1, 3, 5 ]
```

```
// cloning arrays
const arr = [1, 2, 3, 4];
const arr2 = arr.slice()
```



```
// joining arrays
const odd = [1, 3, 5 ];
const nums = [2, 4, 6, ...odd];
console.log(nums);
// [ 2, 4, 6, 1, 3, 5 ]
```

```
// cloning arrays
const arr = [1, 2, 3, 4];
const arr2 = [...arr];
```

Spread Operator Shorthand

```
const odd = [1, 3, 5];  
const nums = [2, ...odd, 4, 6];  
const { a, b, ...z } = { a: 1, b: 2, c: 3, d: 4 };
```

```
console.log(nums) // [2, 1, 3, 5, 4, 6]  
console.log(a) // 1  
console.log(b) // 2  
console.log(z) // { c: 3, d: 4 }
```

Lab 1: Shorthand Syntax

- ▶ ทดลอง Ternary Operator
- ▶ ทดลอง Short-circuit Evaluation
- ▶ ทดลอง Declaring Variables
- ▶ ทดลอง Arrow Function
- ▶ ทดลอง Template Literals
- ▶ ทดลอง Destructuring Assignment
- ▶ ทดลอง Spread Operator



การ Debug Javascript



Debugger

debugger;



```
let arr2 = [2,4,6,8,10];  
for (let i = 0; i < 10; i++) {  
    console.log(arr2[i]);  
}
```

Debug Screen

Debugger paused

Watch

- i: <not available>
- arr2[i]: <not available>

Call Stack

- (anonymous) index.html:37

Scope

- Script
 - arr: (3) [2, 4, 6]
- Global Window

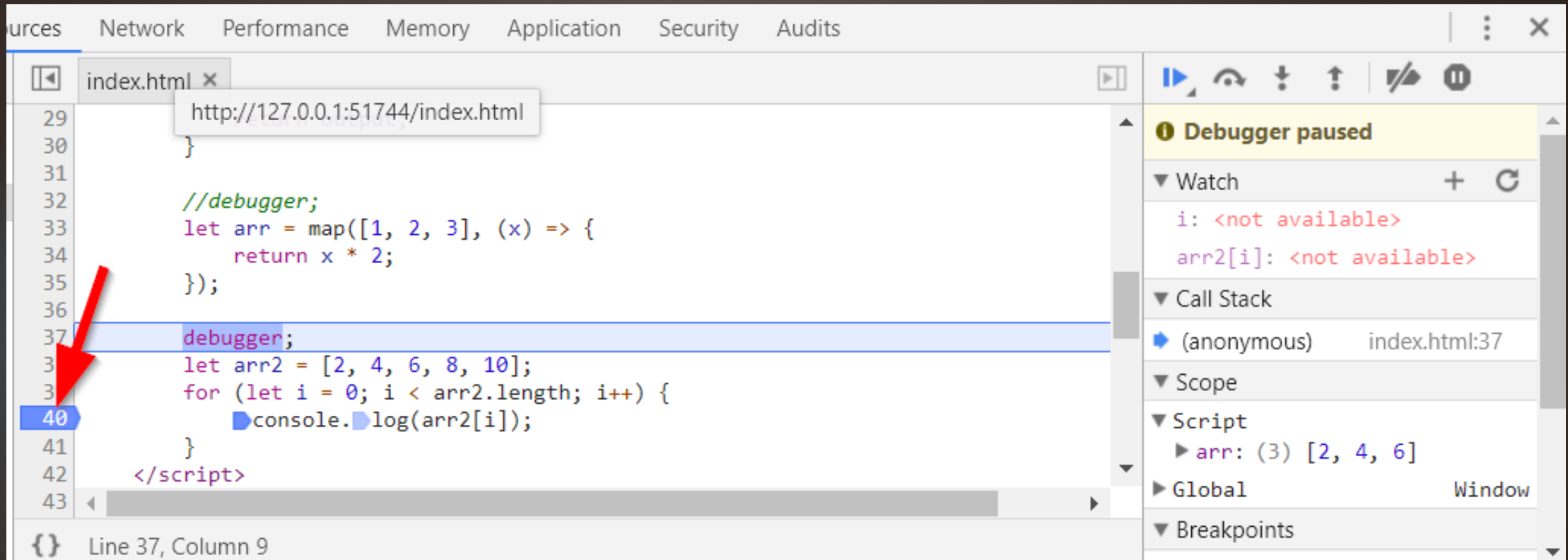
Breakpoints

```
29     return output;
30   }
31
32   //debugger;
33   let arr = map([1, 2, 3], (x) => {
34     return x * 2;
35   });
36
37   debugger;
38   let arr2 = [2, 4, 6, 8, 10];
39   for (let i = 0; i < arr2.length; i++) {
40     console.log(arr2[i]);
41   }
42   </script>
43
```

Line 37, Column 9

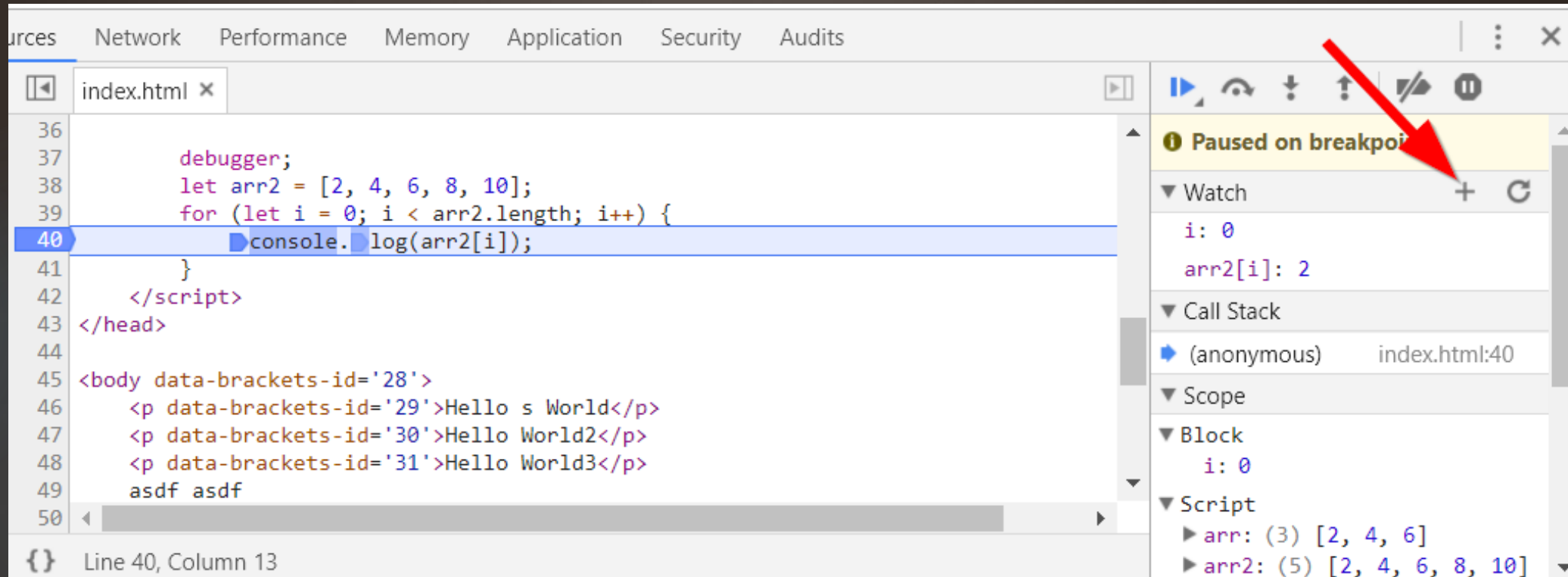
Add Breakpoint

- ▶ เพิ่มจุดที่จะให้ Debugger หยุดบรรทัดถัดไป



Add Watch

► ตรวจสอบค่าตัวแปรที่ต้องการ



The screenshot shows the Chrome DevTools interface. The main pane displays a JavaScript file named 'index.html' with the following code:

```
36  
37     debugger;  
38     let arr2 = [2, 4, 6, 8, 10];  
39     for (let i = 0; i < arr2.length; i++) {  
40         console.log(arr2[i]);  
41     }  
42     </script>  
43 </head>  
44  
45 <body data-brackets-id='28'>  
46     <p data-brackets-id='29'>Hello s World</p>  
47     <p data-brackets-id='30'>Hello World2</p>  
48     <p data-brackets-id='31'>Hello World3</p>  
49     asdf asdf  
50
```

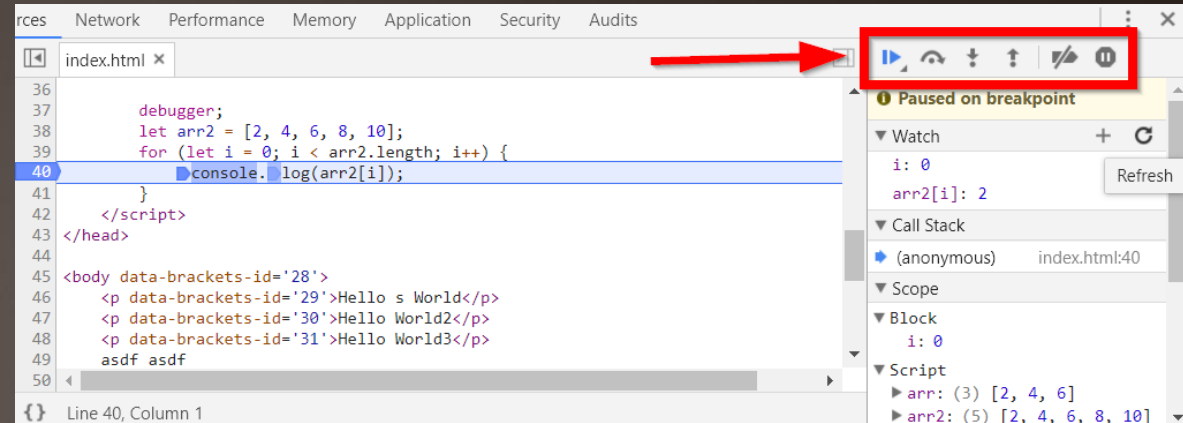
The code is paused at line 40, where a breakpoint is set. The right-hand sidebar shows the 'Paused on breakpoint' message. Below this, the 'Watch' panel is expanded, showing the following variables and their values:

- `i`: 0
- `arr2[i]`: 2

The 'Call Stack' panel shows the current call frame as '(anonymous) index.html:40'. The 'Scope' panel shows the current scope as 'i: 0'. The 'Block' panel shows the current block as 'i: 0'. The 'Script' panel shows the current script as 'arr: (3) [2, 4, 6]' and 'arr2: (5) [2, 4, 6, 8, 10]'. A red arrow points to the 'Watch' panel.

Debug Control Command

- ▶ Continue – วิ่งไปที่จุดที่มี Breakpoint ลำดับถัดไป
- ▶ Step Over – วิ่งข้ามจุดที่มี Function
- ▶ Step Into – เข้าไปใน Function
- ▶ Step Out – วิ่งออกจาก Function
- ▶ Deactivate Breakpoint – นำ Breakpoint ออกชั่วคราว
- ▶ Pause on exceptions – หยุดเมื่อเจอ exceptions



Debug Demo



Functional Programming



Array.map

```
let array1 = [1, 4, 9, 16];

const map1 = array1.map(function (x) {
  return x * 2;
});

console.log(map1);
// expected output: Array [2, 8, 18, 32]
```

Array.map - เบื้องหลังการทำงาน

```
function map(arr, callbackFunction) {  
  let output = [];  
  let result;  
  for (let i=0; i<arr.length; i++) {  
    result = callbackFunction(arr[i]);  
    output.push( result );  
  }  
  return output;  
}
```

Array.map - shorthand

```
let array1 = [1, 4, 9, 16];  
  
const map1 = array1.map(x => x * 2);  
  
console.log(map1);  
// expected output: Array [2, 8, 18, 32]
```

Array.filter

```
let words = ['spray', 'limit', 'elite', 'exuberant', 'destruction', 'present'];

const result = words.filter(word => {
  return word.length > 6
});

console.log(result); // ["exuberant", "destruction", "present"]
```



Array.filter - เบื้องหลังการทำงาน

```
function filter(arr, callbackFunction) {  
  let output = [];  
  for (let i=0; i<arr.length; i++) {  
    if ( callbackFunction( arr[i] ) ) {  
      output.push( arr[i] );  
    }  
  }  
  return output;  
}
```

Array.reduce

```
const numbers = [10, 20, 30, 40]
const result = numbers.reduce((sum, number) => {
  return sum+number
}, 0)
console.log(result) // 100
```



Array.reduce - เบื้องหลังการทำงาน

```
function reduce(arr, callbackFunction, initialize = null) {  
  let result;  
  let i;  
  if (initialize == null) {  
    result = arr[0];  
    i = 1;  
  } else {  
    result = initialize;  
    i = 0;  
  }  
  for (; i < arr.length; i++) {  
    result = callbackFunction(result, arr[i]);  
  }  
  return result;  
}
```

Chaining

```
let animals = ["cat", "dog", "fish"];
function studlyCaps(words, word) {
  return words + word;
}
function exactlyThree(word) {
  return (word.length === 3);
}
function capitalize(word) {
  return word.charAt(0).toUpperCase() + word.slice(1);
}
const threeLetterAnimals = animals
  .filter(exactlyThree)
  .map(capitalize)
  .reduce(studlyCaps);
console.log(threeLetterAnimals); // "CatDog"
```

Lab 2: Functional Programming

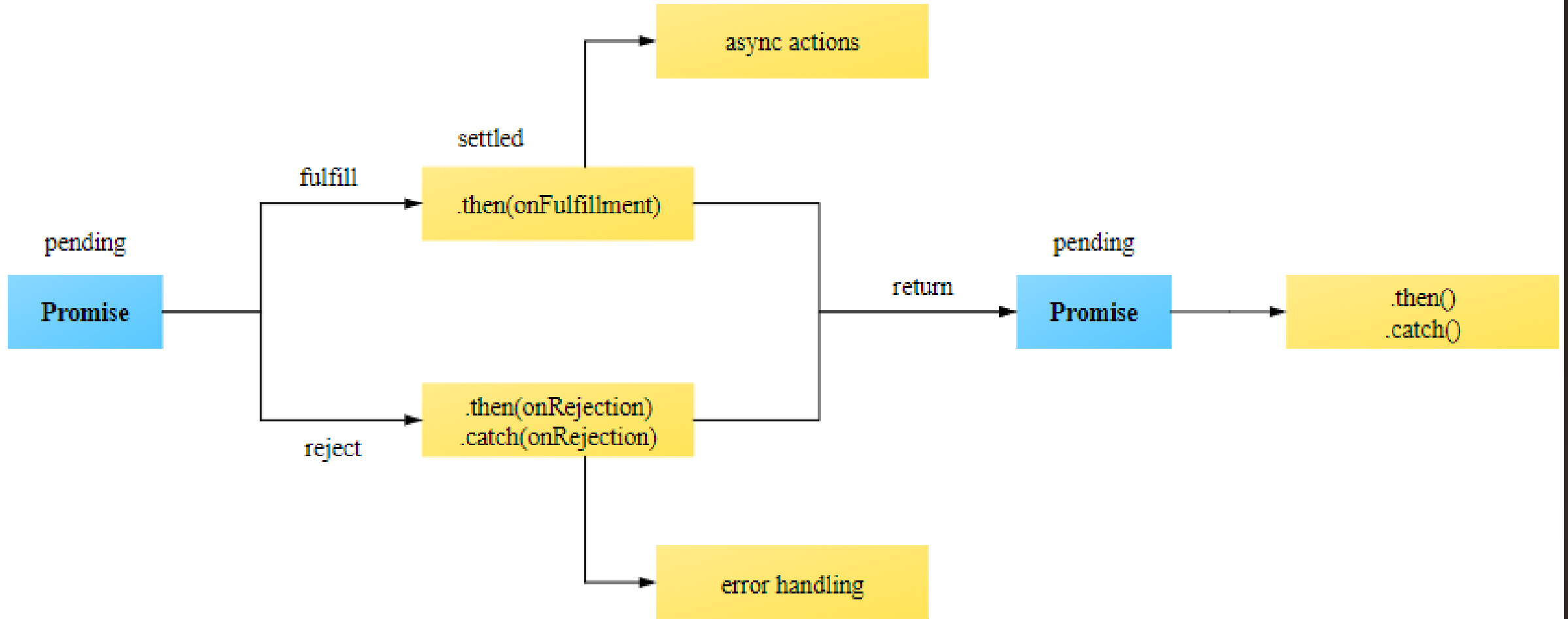
- ▶ ทดลอง map, filter, reduce
- ▶ ทดลอง chaining



Promise



Promise คืออะไร?



Function หลักของ Promise

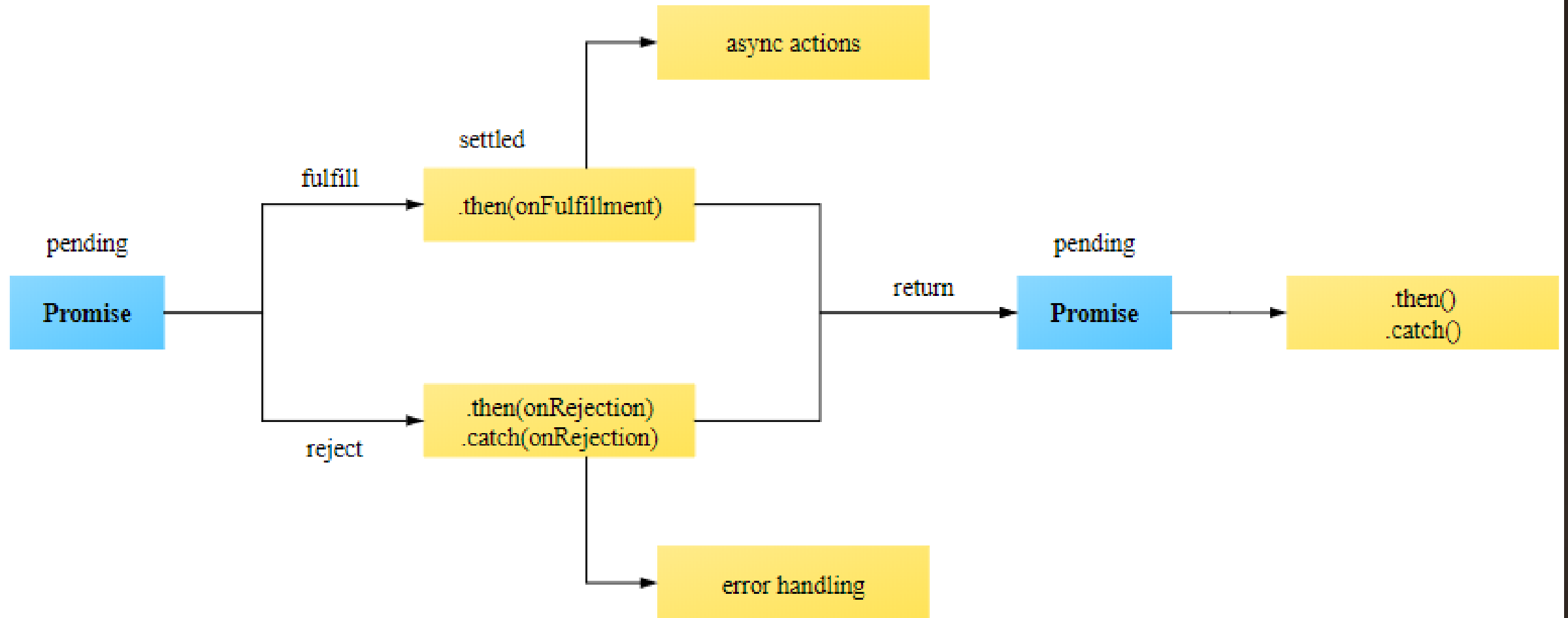
- ▶ Reject -> เรียกเมื่อมีบางอย่างผิดพลาด
- ▶ Resolve -> เรียกเมื่อทำงานเสร็จสมบูรณ์และส่งผลลัพธ์ของ Function ไป

ตัวอย่างการใช้งาน Promise

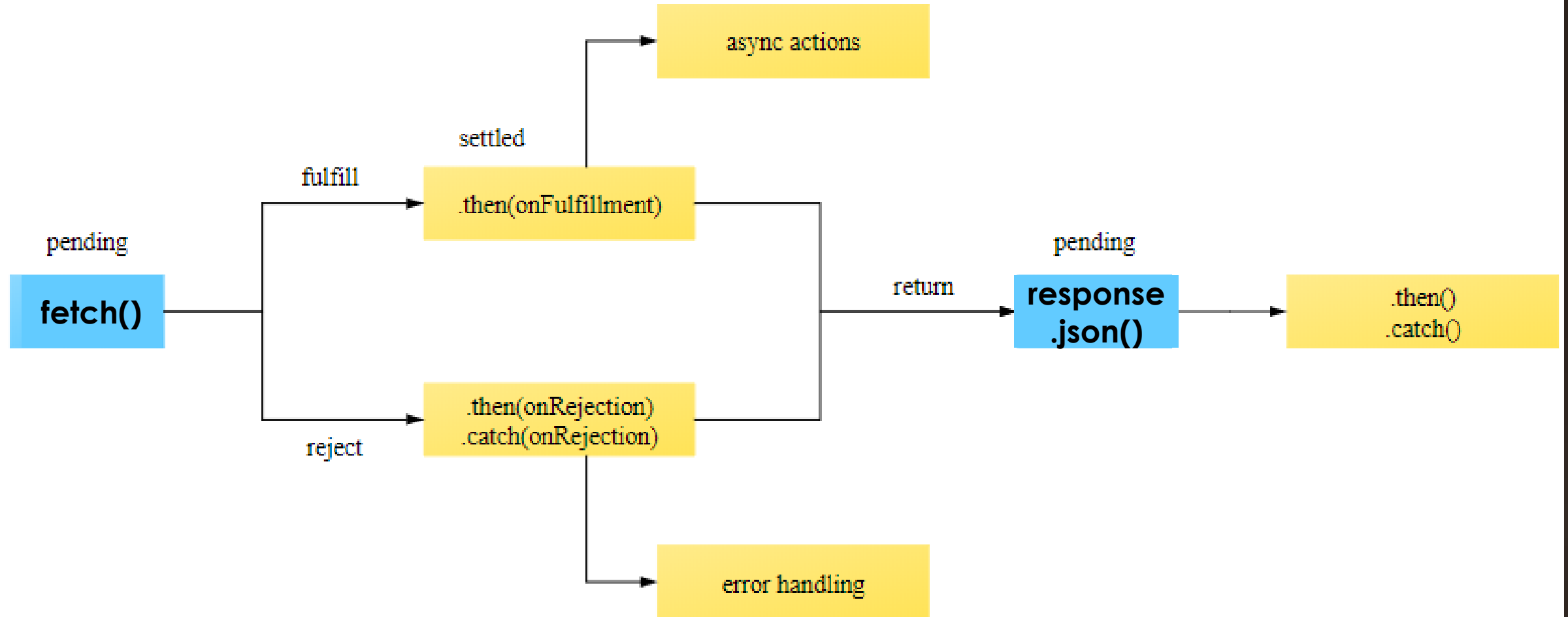
```
fetch('homework1.json').then(response => {  
  return response.json();  
})  
.then(myJson => {  
  console.log(myJson);  
})  
.catch(error => {  
  console.error('Error:', error);  
});
```



ตัวอย่างการใช้งาน Promise



ตัวอย่างการใช้งาน Promise



Async/Await

```
async function getHomework1Json() {  
  const response = await fetch('homework1.json');  
  const myJson = await response.json();  
  console.log(myJson);  
}  
getHomework1Json();
```

ข้อควรระวัง

- ▶ Fetch, Promise, Async/Await เป็น Syntax Javascript ES7 ไม่สามารถใช้กับ Internet Explorer เวอร์ชันใดๆ ได้เลย
- ▶ แก้ไขได้ด้วยการใช้ [Babel](#) เพื่อช่วยแปลงจาก Syntax Javascript ES7, ES6 เป็น Syntax Javascript ES5 ได้



Javascript Class (ES6)



Class Structure

```
1 class Employee {  
2   constructor(firstname, lastname, salary) {  
3     let _salary = salary; // simulate private variable  
  
4     this.firstname = firstname; // public property  
      this.lastname = lastname; // public property  
5     this.getSalary = function() { // simulate public method  
        return _salary;  
      };  
  }  
6  hello() { // simulate public method  
    console.log("Hello "+this.firstname+"!");  
  }  
}  
7 let dang = new Employee('Dang', 'Red', 10000);
```

Class Structure

1. ชื่อ class
2. Constructor function เป็นจุดเริ่มต้นของ class หลัง new รับ parameter มาจากตอนสร้าง Instance
3. `_salary` คือ property (ตัวแปร) ของ class ชนิด private มี scope เห็นเพียงภายใน method `constructor()` เท่านั้น
4. `this.firstname` คือ property (ตัวแปร) ของ class ชนิด public
5. `getSalary()` คือ method (function) ของ class ชนิด public แบบประกาศใน `constructor()`
6. `hello()` คือ method (function) ของ class ชนิด public แบบประกาศภายใน class นอก `constructor()` ส่วนใหญ่จะเจอบ่อยกว่าข้อ 5
7. สร้าง Instance ของ class `Employee` ให้กลายเป็น object เข้าไปยังตัวแปรชื่อ `dang`

Class Structure (extend)

```
class CEO extends Employee {  
  constructor(firstname, lastname, salary) {  
    1 super(firstname, lastname, salary);  
  }  
  getSalary(){ // simulate public method  
    2 return super.getSalary()*2;  
  };  
  3 hello() { // simulate public method  
    console.log("Hi, nice to meet you. "+this.firstname+"!");  
  }  
}  
4 let ceo = new CEO('Somchai', 'Sudlor', 30000);
```

Class Structure (extend)

1. `super()` เรียก constructor ของ class แม่ ต้องเรียกเสมอ หาก class ลูกประกาศ constructor เอาไว้
2. `super.getSalary()` เรียก method `getSalary()` จากใน class แม่ เพื่อที่จะนำมาคำนวณเพิ่มเติมใน class ลูก และเขียน method `getSalary()` ทับของ class แม่ (override) ไปเลย
3. method `hello` เขียนทับของ class แม่ (override) ไปเลย โดยไม่มีการอ้างอิงถึง method เก่าใน class แม่
4. สร้าง Instance ของ class `CEO` ซึ่งเป็น class ลูกให้กลายเป็น object เข้าไปยังตัวแปรชื่อ `ceo`



ข้อควรระวังเมื่อเทียบกับ class ในภาษาอื่น

- ▶ Javascript ไม่มี method overloading
- ▶ หากใช้ this ใน callback function ที่ถูกเรียกจากที่อื่น โดยไม่ใช่ Arrow function จะไม่ใช่ this ที่ใช้อ้างถึงตัวแปร หรือ method ใน class

```
class CEO extends Employee {  
  constructor(firstname, lastname, salary) {  
    super(firstname, lastname, salary);  
    let self = this;  
    setTimeout(function(err, res) {  
      console.log(self.lastname); // God  
      console.log(this.lastname); // undefined  
    }, 1000);  
  }  
}  
  
let ceo = new CEO('CEO', 'God', 10000);
```



ข้อควรระวังเมื่อเทียบกับ class ในภาษาอื่น

```
class CEO extends Employee {
  constructor(firstname, lastname, salary) {
    super(firstname, lastname, salary);
    let self = this;
    setTimeout((err, res) => {
      console.log(self.lastname); // God
      console.log(this.lastname); // God
    }, 1000);
  }
}
let ceo = new CEO('CEO', 'God', 10000);
```

Bind

```
let module = {  
  x: 42,  
  getX: function() {  
    return this.x;  
  }  
}
```

```
let retrieveX = module.getX;  
console.log(retrieveX()); // The function gets invoked at the global scope  
// expected output: undefined
```

```
let boundGetX = retrieveX.bind(module);  
console.log(boundGetX());  
// expected output: 42
```

Lab 3: Fetch, Async/Await, Class

- ▶ ทดลอง Fetch ไฟล์ homework1.json จากการบ้านวันที่ 1 แบบ Promise
- ▶ ทดลอง Fetch ไฟล์ homework1.json จากการบ้านวันที่ 1 แบบ Async/Await
- ▶ ทดลอง Fetch <https://jsonplaceholder.typicode.com/posts/1> แทนไฟล์ homework1.json
- ▶ ทดลองเขียน class และ extend อย่างง่าย

Homework



การบ้าน #1

- ▶ บรรทัดแรกของไฟล์การบ้านมีดังนี้ `let arr = [1,2,3,4,5,6,7,8,9,10];`
- ▶ จงคัดลอกสมาชิกใน array ให้เหลือเพียงสมาชิกที่หาร 2 ลงตัว และนำสมาชิกเหล่านั้นมาคูณ 1,000 ทุกตัว
- ▶ เงื่อนไข : ห้ามใช้ loop for, while ในโจทย์ข้อนี้เด็ดขาด ใช้ได้แค่ map, reduce, filter เท่านั้น
- ▶ เซฟไฟล์ชื่อ homework2-1.js



การบ้าน #2

- ▶ อ่าน JSON จากไฟล์ homework1.json จากการบ้านวันที่ 1 ด้วย fetch เพื่อนำ json มาใช้งานเก็บค่าลงตัวแปร peopleSalary
- ▶ สร้างตัวแปร array ชื่อ peopleLowSalary ที่คัดเอาเฉพาะพนักงานที่มีเงินเดือนน้อยกว่า 100,000 ออกมา และเพิ่มเงินเดือนให้พนักงานเหล่านั้น 2 เท่า
- ▶ หาผลรวมของเงินเดือนที่ต้องจ่ายในแต่ละเดือนทั้งหมดหลังขึ้นเงินเดือนเก็บลงตัวแปร sumSalary โดยรวมพนักงานที่ไม่ได้ขึ้นเงินเดือนด้วย (คำใบ้คือ ไม่ต้องหาทางรวมก่อน array ชุดเก่ากับชุดใหม่กันก็ได้ สามารถหาผลรวมจาก array ที่แก้ไขค่าแล้วได้)
- ▶ เงื่อนไข : ห้ามใช้ loop for, while ในโจทย์ข้อนี้เด็ดขาด ใช้ได้แค่ map, reduce, filter เท่านั้น
- ▶ Console.log ค่า peopleLowSalary, และ sumSalary ออกมา
- ▶ สร้างไฟล์ชื่อ homework4-3.js

การบ้าน #3

- ▶ จงเข้า Download file ด้านล่างมาเซฟเก็บไว้

<https://drive.google.com/open?id=13ejCsIsXKkLx9XUTMRmxqOhMG0tNlcHS>

- ▶ ใช้ fetch อ่านค่าไฟล์ “homework1-4.json” ที่ download มา มาเก็บค่าไว้
- ▶ จงสร้าง array ตัวใหม่ที่มีเพียงเพศชาย, มีเพื่อนอย่างน้อย 2 คน และกรอง field ออก โดยให้เหลือเพียง field name, gender, company, email, friends, balance เท่านั้น
- ▶ ลดเงินในบัญชี (balance) ของทุกคนลง 10 เท่า (อย่าลืมเติม \$ กลับที่ด้านหน้าด้วย)
- ▶ เงื่อนไข: ห้ามใช้ loop for, while ในโจทย์ข้อนี้เด็ดขาด ใช้ได้แค่ map, reduce, filter เท่านั้น
- ▶ เซฟชื่อ homework2-3.js



การบ้าน #4 - Javascript Class

▶ จงแก้ไขไฟล์

<https://drive.google.com/open?id=11RVkkdM3KjpozKTSYKOuzx1U2wVKBTsF>

ให้มีผลลัพธ์ทาง terminal ดังนี้ และให้แยกเป็นไฟล์ละ class (ชื่อไฟล์ตั้งตามชื่อ class) โดยมี homework2-4.js เป็นตัวรัน code จริงๆ

Hey Somsri, Today is very cold!

Somsri has been fired! Dress with :tshirt

Somsri has been hired back! Dress with :tshirt

He is going to seminar Dress with :suit

He goes to golf club to find a new connection. Dress with :golf_dress

Somsri's salary is less than before!!

Somsri's salary has been set to 25000

▶ เชพไฟล์ตามไฟล์ด้านบน

