



Software Park Thailand
</Code Camp>

JAVASCRIPT

Nuttachai Kulthammanit

Slide Design By Patteera Banlangthammas



CHULA ENGINEERING
Foundation toward Innovation

นายณัฐฐชัย กุลธรรมนิตย์ (ชั้นเต๋อ) จบจากคณะวิศวกรรมศาสตร์ ภาควิชาวิศวกรรม คอมพิวเตอร์ จุฬาลงกรณ์มหาวิทยาลัย

ประสบการณ์เขียนเว็บ 2 ปี

ประสบการณ์ด้านการสอน 3 ปี

ภาษาที่เคยเขียน

- C++, C, C#, Java, JavaScript, HTML, CSS, Verilog, Prolog, PHP, Python, SQL



หัวข้อ



Software Park Thailand
«/Code Camp»

- JavaScript คืออะไร
- JavaScript เบื้องต้น
- ตัวแปรและประเภทของข้อมูล
- ตัวดำเนินการเบื้องต้น
- การเปรียบเทียบ
- การเขียนเงื่อนไข

JavaScript คืออะไร

1. JavaScript คืออะไร

โครงสร้างของเว็บไซต์ประกอบด้วย 3 ส่วนหลัก ๆ



1. JavaScript คืออะไร (HTML)



- เปรียบเสมือน โครงสร้าง ของเว็บไซต์
- เช่น ปุ่ม ข้อความ รูปภาพ เป็นต้น

1. JavaScript คืออะไร (HTML)



HTML Tags ทั้งหมด:

[HTML Reference](#)

1. JavaScript คืออะไร (CSS)



- ใช้ตกแต่งโครงสร้าง บนเว็บไซต์
- เช่น ทำให้ข้อความมีสีเหลือง, ทำให้ปุ่มมีขนาดเล็ก-ใหญ่ เป็นต้น

1. JavaScript คืออะไร (CSS)



CSS Properties ทั้งหมด:

[CSS Reference](#)

2. JavaScript เบื้องต้น

ECMAScript หรือ ECMA-262 (Standard ECMA-262)

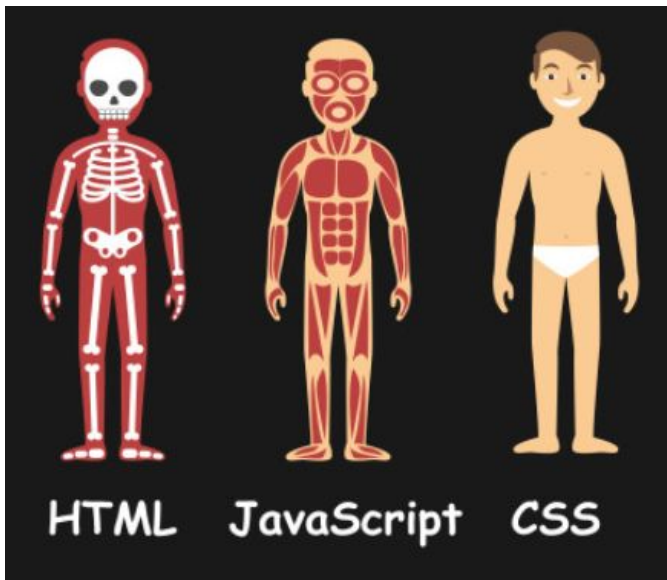
- เป็นมาตรฐานของภาษา JavaScript
- ประกาศโดย ECMA International
- ปัจจุบันมีถึง Version 10 แล้ว
- ในคลาสนี้จะสอน ES5 และ ES6 (ES 2015) เป็นหลัก

1. JavaScript คืออะไร (JavaScript)



- ทำหน้าที่กำหนดพฤติกรรมต่าง ๆ ของโครงสร้างบนเว็บไซต์
- เช่น เมื่อกดปุ่มทำให้พื้นหลังเปลี่ยนสี

1. JavaScript คืออะไร (3 Friends)



https://res.cloudinary.com/teepublic/image/private/s--cXHqFHVb--/t_Preview/b_rgb:262c3a,c_limit,f_jpg,h_630,q_90,w_630/v1561935848/production/designs/5202028_0.jpg

1. JavaScript คืออะไร (3 Friends)

Enter URL

Fetch Main Content Only ☐

Keep Javascript ☐

Compress Output ☐

HTML

Contact form

Your Name

Your Email

Phone*

Message Subject*

Message*

Verification* ☐ I am a robot

CSS

Classic validation

Enter your name

Enter your

The field is

Enter your password

The field is required

JavaScript



JavaScript resource

<https://developer.mozilla.org/en-US/docs/Web/JavaScript>

<https://javascript.info/>

<https://www.w3schools.com/js/default.asp>

*** EcmaScript standard : <https://www.ecma-international.org/ecma-262/>*

หัวข้อ



Software Park Thailand
«/Code Camp»

- JavaScript คืออะไร
- JavaScript เบื้องต้น
- ตัวแปรและประเภทของข้อมูล
- ตัวดำเนินการเบื้องต้น
- การเปรียบเทียบ
- การเขียนเงื่อนไข

JavaScript เบื้องต้น

2. JavaScript เบื้องต้น

ECMAScript compatibility table

Edition	Date published	Name	Changes from prior edition	Editor
1	June 1997		First edition	Guy L. Steele Jr.
2	June 1998		Editorial changes to keep the specification fully aligned with ISO/IEC 16262 international standard	Mike Cowlishaw
3	December 1999		Added regular expressions , better string handling, new control statements, try/catch exception handling, tighter definition of errors, formatting for numeric output and other enhancements	Mike Cowlishaw
4	<i>Abandoned</i>		Fourth Edition was abandoned, due to political differences concerning language complexity. Many features proposed for the Fourth Edition have been completely dropped; some were incorporated into the sixth edition.	
5	December 2009		Adds "strict mode," a subset intended to provide more thorough error checking and avoid error-prone constructs. Clarifies many ambiguities in the 3rd edition specification, and accommodates behaviour of real-world implementations that differed consistently from that specification. Adds some new features, such as getters and setters, library support for JSON , and more complete reflection on object properties. ^[11]	Pratap Lakshman, Allen Wirfs-Brock
5.1	June 2011		This edition 5.1 of the ECMAScript standard is fully aligned with third edition of the international standard ISO/IEC 16262:2011.	Pratap Lakshman, Allen Wirfs-Brock
6	June 2015 ^[12]	ECMAScript 2015 (ES2015)	See 6th Edition - ECMAScript 2015	Allen Wirfs-Brock
7	June 2016 ^[13]	ECMAScript 2016 (ES2016)	See 7th Edition - ECMAScript 2016	Brian Terlson
8	June 2017 ^[14]	ECMAScript 2017 (ES2017)	See 8th Edition - ECMAScript 2017	Brian Terlson
9	June 2018 ^[15]	ECMAScript 2018 (ES2018)	See 9th Edition - ECMAScript 2018	Brian Terlson
10	June 2019 ^[10]	ECMAScript 2019 (ES2019)	See 10th Edition - ECMAScript 2019	Brian Terlson, Bradley Farias, Jordan Harband

2. JavaScript เบื้องต้น

ตัวอย่าง ความแตกต่างระหว่าง ES5 และ ES6

```
var foo;  
var bar;
```

ES5

```
const foo  
let bar
```

ES6

2. JavaScript เบื้องต้น

2.1. การเขียน Java Script ทำได้สองวิธี

1. Internal JavaScript
2. External JavaScript

2. JavaScript เบื้องต้น

2.1. Internal JavaScript

```
<script>  
    console.log("Codecamp 5");  
</script>
```

2. JavaScript เบื้องต้น

2.2 External JavaScript

```
<script src="script.js"></script>
```

JS script.js ×

1.JavaScript > 1.2. External Javascript > JS script.js

```
1 console.log('Codecamp 5');
```

2. JavaScript เบื้องต้น

**** ใน script tag ถ้า src ถูกกำหนดแล้ว คำสั่งที่เขียนใน tag นั้นจะไม่ทำงาน ****

```
<script src="script.js">  
  console.log('Codecamp Internal') // ตรงนี้จะไม่ถูกทำงานเนื่องจากค่า src ถูกกำหนดแล้ว  
</script>  
  
<script>  
  console.log('Codecamp Internal') // ตรงนี้จะถูกทำงานเนื่องจากไม่มีค่า src กำหนดไว้  
</script>
```

2. JavaScript เบื้องต้น

2.3. การเขียน Statement (Ref: [Code structure](#))

- (สำหรับมือใหม่) ใส่ Semi-colon ทุกครั้ง หลังจากจบ คำสั่ง (Statement) ต่าง ๆ

```
console.log('welcome to');  
console.log('Codecamp 5');
```


2. JavaScript เบื้องต้น

2.4. การเขียน Comment

- Comment คือสิ่งที่เขียนไว้ในโปรแกรม เพื่ออธิบาย Code โดยจะเขียนไว้ตรงไหนก็ได้ของโปรแกรม เพราะ Comment นั้นจะไม่มีผลต่อโปรแกรม เนื่องจาก Computer จะไม่อ่านบรรทัดที่เป็น Comment

```
// ประกาศตัวแปรแบบค่าคงที่นะ  
const foo  
// ประกาศตัวแปรแบบไม่คงที่นะ  
let bar
```

2. JavaScript เบื้องต้น

2.4. การเขียน Comment มีทั้งหมด 2 แบบ

- Comment แบบบรรทัดเดียว
- Comment แบบหลายบรรทัด

```
// คอมเมนต์แบบบรรทัดเดียว
```

```
/*  
คอมเมนต์แบบหลายบรรทัด  
สามารถ  
เขียนได้  
หลายบรรทัด  
*/
```

2. JavaScript เบื้องต้น

2.4.1. Comment แบบบรรทัดเดียว

- จะเริ่มด้วย // และตามด้วยข้อความ

```
// คอมเมนต์แบบบรรทัดเดียว
```

2. JavaScript เบื้องต้น

2.4.2. Comment แบบหลายบรรทัด

- จะเริ่มด้วย /* และตามด้วยข้อความ สุดท้ายปิดด้วย */

```
/*  
คอมเมนต์แบบหลายบรรทัด  
สามารถ  
เขียนได้  
หลายบรรทัด  
*/
```



หัวข้อ



Software Park Thailand
«/Code Camp»

- JavaScript คืออะไร
- JavaScript เบื้องต้น
- ตัวแปรและประเภทของข้อมูล
- ตัวดำเนินการเบื้องต้น
- การเปรียบเทียบ
- การเขียนเงื่อนไข

ตัวแปรและประเภทของ ข้อมูล

3. ตัวแปรและประเภทของข้อมูล

3.1. ตัวแปร

- ตัวแปรคือสิ่งที่เอาไว้ใช้เก็บข้อมูล
- และสามารถเปลี่ยนแปลงข้อมูลข้างในได้

```
12 console.log('Codecamp #5');
```

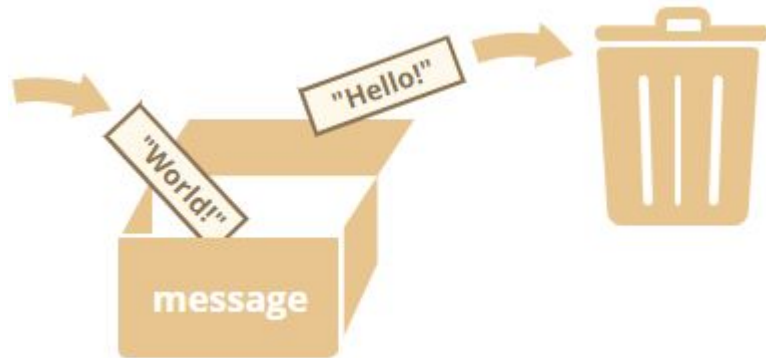
```
12 var message = 'Codecamp #5'  
13 console.log(message);
```


3. ตัวแปรและประเภทของข้อมูล

3.1. ตัวแปร(ตัวอย่าง)

- ตัวแปรคือสิ่งที่เอาไว้ใช้เก็บข้อมูล

```
let message = "Hello!";  
message = "World!";
```



3. ตัวแปรและประเภทของข้อมูล

3.2. การประกาศตัวแปร

- การประกาศตัวแปรก่อนการใช้งานตัวแปร
- การประกาศตัวแปรคือการที่บอกให้โปรแกรมรู้ว่าเราจะ ใช้ตัวแปร นั้น
- จะกำหนดค่าตอนประกาศ หรือกำหนดค่าตอนหลังก็ได้

3. ตัวแปรและประเภทของข้อมูล

3.2. การประกาศตัวแปร

- สามารถประกาศตัวแปรหลายตัวพร้อมกัน โดยเขียน var, let หรือ const แค่ครั้งเดียว

```
var name = 'sonter';  
var height = 176;  
var message = 'Hello World!';
```

```
var name = 'sonter',  
    height = 176,  
    message = 'Hello World!';
```

```
1 var name = 'sonter', height = 176, message = 'Hello World!';
```

3. ตัวแปรและประเภทของข้อมูล

3.2. การประกาศตัวแปร

- ความแตกต่างระหว่าง var, let, const
- var เป็น JavaScript เวอร์ชันเก่า
- let และ const มาใน ES6 (เวอร์ชันใหม่)

```
var foo;  
var bar;
```

```
const foo  
let bar
```

ES6

3. ตัวแปรและประเภทของข้อมูล

3.2. การประกาศตัวแปร

- **const** ใช้กับตัวแปรที่เปลี่ยนค่าไม่ได้ และต้องใส่ค่าเริ่มต้นให้เสมอ
- ~~var~~ ใช้กับตัวแปรที่เปลี่ยนค่าได้ (แต่เราจะใช้ let แทน var)
- **let** ใช้กับตัวแปรที่เปลี่ยนค่าได้ ไม่จำเป็นต้องใส่ค่าเริ่มต้นให้

3. ตัวแปรและประเภทของข้อมูล

3.2. การประกาศตัวแปร (ตัวอย่าง - const)

```
// แบบนี้จะเกิด Error ขึ้น  
const foo  
// ถ้าเป็น const ต้องกำหนดค่าเริ่มต้นด้วย  
const foo = 'bar'
```

3. ตัวแปรและประเภทของข้อมูล

3.2. การประกาศตัวแปร (ตัวอย่าง - let)

```
// สามารถประกาศแบบไม่กำหนดค่าเริ่มต้นได้  
let foo  
  
// สามารถมาใส่ค่าที่หลังได้  
foo = 'bar'  
  
// หรือจะประกาศและใส่ค่าไปเลยก็ได้  
let message = 'Hello World!'
```

3. ตัวแปรและประเภทของข้อมูล

3.3. ชื่อของตัวแปร

- ประกอบด้วย ตัวอักษร ตัวเลข หรือ สัญลักษณ์ \$ และ _ เท่านั้น
- ตัวแรกต้องไม่เป็นตัวเลข
- ต้องไม่เป็นคำสงวน เช่น if, function, break เป็นต้น (Keyword ทั้งหมด)

3. ตัวแปรและประเภทของข้อมูล

3.3. ชื่อของตัวแปร (Exercise)

- c4mp ถูกต้อง
- _codecamp45 ถูกต้อง
- _ ถูกต้อง
- \$code ถูกต้อง
- 5codecamp ผิด เพราะขึ้นต้นด้วยตัวเลข
- codecamp#5 ผิด เพราะมี #
- code camp ผิด เพราะมีเว้นว่าง (space)

3. ตัวแปรและประเภทของข้อมูล

3.3. ชื่อของตัวแปร (Exercise) - ต่อ

- function1 ถูกต้อง
- function ผิด เพราะ “function” เป็นคำสงวน
- if ผิด เพราะ “if” เป็นคำสงวน
- _if ถูกต้อง
- else ผิด เพราะ “else” เป็นคำสงวน
- codecamp-5 ผิดเพราะ มี - (ขีด)
- โค้ดแคมป์5 ไม่ผิด แต่ไม่ควรใช้ภาษาไทย

3. ตัวแปรและประเภทของข้อมูล

3.4. การตั้งชื่อตัวแปรที่ดี

- ชื่อตัวแปรที่ตั้งมนุษย์ควรอ่านรู้เรื่อง
- ไม่ควร ตั้งชื่อด้วย อักษรตัวเดียว เช่น a, b, c
- ควรตั้งชื่อที่ สื่อความหมาย เช่น newUser, myCareer เป็นต้น
 - ไม่ควรตั้งชื่อที่ไม่สื่อความหมาย เช่น aaa, bb, af, za เป็นต้น
- ถ้าชื่อตัวแปรมีช่องว่างให้ใช้ camelCase (คำถัดไปให้เริ่มด้วยตัวใหญ่)
 - current user ให้เขียนเป็น currentUser
 - my home ให้เขียนเป็น myHome

3. ตัวแปรและประเภทของข้อมูล

3.4. การตั้งชื่อตัวแปรที่ดี (Lab)

Lab 1

- ให้ประกาศตัวแปรชื่อ `human` และ `name`
- ใส่ชื่อตัวเองลงในตัวแปร `name`
- นำค่าที่อยู่ในตัวแปร `name` ไปใส่ให้ `human`
- เมื่อ `console.log(human)` ออกมาต้องเป็นชื่อตัวเอง

3. ตัวแปรและประเภทของข้อมูล

3.4. การตั้งชื่อตัวแปรที่ดี (Lab)

Lab 2

- ตั้งชื่อตัวแปรที่ใช้เก็บจำนวนเงินในกระเป๋าตังของคุณ
- ตั้งชื่อตัวแปรที่ใช้เก็บชื่อของ พ่อและแม่ของคุณ
- ตั้งชื่อตัวแปรที่ใช้เก็บที่อยู่ของบ้านคุณ
- ตั้งชื่อตัวแปรที่ใช้เก็บอายุของจักรวาล

3. ตัวแปรและประเภทของข้อมูล

3.5. Case-Sensitive

- Javascript เป็นภาษาที่ชื่อตัวแปรเป็น case-sensitive
- การพิมพ์ตัวใหญ่และตัวเล็กถือว่าเป็นตัวแปรคนละตัว
เช่น
 - name และ Name ถือว่าเป็นตัวแปรคนละตัว
 - codecamp และ CodeCamp ถือว่าเป็นตัวแปรคนละตัวกัน

3. ตัวแปรและประเภทของข้อมูล

3.5. Case-Sensitive (ตัวอย่าง)

```
let name, Name;  
name = 'Codecamp';  
Name = 'sonter';  
console.log(name);  
console.log(Name);
```

3. ตัวแปรและประเภทของข้อมูล

3.6. ประเภทของข้อมูล

ใน JavaScript มีข้อมูลที่เป็น ชนิดพื้นฐาน ทั้งหมด 5 ประเภท

- **Number:** ข้อมูลประเภทตัวเลข
- **String:** ข้อมูลประเภทข้อความ
- **Boolean:** ข้อมูลที่มีแค่ true และ false
- **null:** ค่าว่าง
- **undefined:** ข้อมูลที่ยังไม่ได้ใส่ค่า

3. ตัวแปรและประเภทของข้อมูล

3.6. ประเภทของข้อมูล

JavaScript เป็น Dynamic-type

- ไม่ต้องบอกประเภทตัวแปรตอนประกาศตัวแปร
- สามารถเปลี่ยนแปลงประเภทได้

```
let name = 'Sonter';  
  
let age = 18;  
age = "Eighteen";
```

3. ตัวแปรและประเภทของข้อมูล

3.6.1. Number: ข้อมูลประเภทตัวเลข

- ทั้งจำนวนเต็ม(Integer) และ ทศนิยม(Float)
- Special numeric values (ค่าพิเศษที่นอกเหนือจาก Integer และ Float)
 - NaN (Not a number)
 - Infinity (∞)
 - -Infinity ($-\infty$)
- สามารถ บวก, ลบ, คูณ, หาร ได้

3. ตัวแปรและประเภทของข้อมูล

3.6.1. Number: ข้อมูลประเภทตัวเลข

- `console.log(1 / 0)` จะได้ Infinity
- `console.log(Infinity)` จะได้ Infinity
- `console.log("Codecamp" / 2)` จะได้ NaN
- ไม่ว่าจะทำอะไร(บวก, ลบ, คูณ,หาร) กับ NaN ก็จะได้ NaN เสมอ

3. ตัวแปรและประเภทของข้อมูล

3.6.2. String: ข้อมูลประเภทข้อความ

- ข้อมูลประเภทข้อความจะครอบด้วย

Original

- Double quote “”
- Single quote ‘’

ES6

- Backticks ``

3. ตัวแปรและประเภทของข้อมูล

3.6.2. **String**: ข้อมูลประเภทข้อความ

- ทั้ง Double quote และ Single quote เหมือนกันทุกอย่างใช้ในการเขียน
ครอบข้อความ

```
let message1 = 'Software Park';  
let message2 = "Codecamp";
```

3. ตัวแปรและประเภทของข้อมูล

3.6.2. String: ข้อมูลประเภทข้อความ

- แต่ Backticks สามารถแทรกตัวแปรระหว่าง String ได้
ดังตัวอย่าง (`message4`) หรือจะใส่แค่ข้อความอย่างเดียว
แบบ Single quote และ double quote ก็ได้ (`message3`)

```
let message1 = 'Software Park';  
let message2 = "Codecamp";  
  
let message3 = `Sonter`;  
  
let message4 = `Sonter at ${message1} #5 ${message2}`;  
// Sonter at Software Park #5 Codecamp
```

3. ตัวแปรและประเภทของข้อมูล

3.6.2. String: ข้อมูลประเภทข้อความ

- แต่ Double quout หรือ Single quote ใส่ตัวแปรไม่ได้

```
let message5 = 'Software Park';  
let message6 = "Codecamp";  
  
let message7 = "Sonter at ${message1} #5 ${message2}";  
// Sonter at ${message1} #5 ${message2}
```

3. ตัวแปรและประเภทของข้อมูล

3.6.3. Boolean: ข้อมูลประเภทค่าความจริง

ทบทวนตรรกศาสตร์เบื้องต้น

- And (และ) - มี **false** แค่ตัวเดียวจะกลายเป็น **false** เลย
 - **true** และ **true** = **true**
 - **true** และ **false** = **false**
 - **false** และ **true** = **false**
 - **false** และ **false** = **false**

3. ตัวแปรและประเภทของข้อมูล

3.6.3. Boolean: ข้อมูลประเภทค่าความจริง

ทบทวนตรรกศาสตร์เบื้องต้น

- Or (หรือ) - มี **true** แค่ตัวเดียวจะกลายเป็น **true** เลย
 - **true** หรือ **true** = **true**
 - **true** หรือ **false** = **true**
 - **false** หรือ **true** = **true**
 - **false** หรือ **false** = **false**

3. ตัวแปรและประเภทของข้อมูล

3.6.3. Boolean: ข้อมูลประเภทค่าความจริง

- มีเพียงสองค่าเท่านั้นคือ **True**(จริง) และ **False**(เท็จ)
- โดยปกติ True จะเปรียบเสมือนคำว่า ใช่ และ False จะเปรียบเสมือนคำว่า ไม่ใช่

3. ตัวแปรและประเภทของข้อมูล

3.6.3. Boolean: ข้อมูลประเภทค่าความจริง (ตัวอย่าง)

```
let x = 5;  
  
console.log( x > 5 )    // false  
console.log( x == 5 )   // true  
console.log( x >= 5 )   // true
```

3. ตัวแปรและประเภทของข้อมูล

3.6.4. `null`: ค่าว่าง

- หมายถึง ค่าว่าง, ไม่มี, หรือไม่ทราบค่า ก็ได้

3. ตัวแปรและประเภทของข้อมูล

3.6.5. `undefined`: ยังไม่ได้กำหนดค่า

- `undefined` คือการที่ค่าของตัวแปรนั้นยังไม่ได้ถูกกำหนด
เช่น

```
let x;  
  
console.log(x); // undefined
```

3. ตัวแปรและประเภทของข้อมูล

3.7. `typeof`: การ Check ประเภทของตัวแปร

- เป็น Operator ที่จะบอกประเภทของข้อมูล
- เขียนเป็นแบบ Operator: `typeof x`

```
console.log(typeof 15); // number
```

- เขียนเป็นแบบ Function: `typeof(x)`

```
console.log(typeof(15)); // number
```

3. ตัวแปรและประเภทของข้อมูล

3.7. `typeof`: การ Check ประเภทของตัวแปร (ตัวอย่าง) [Data types](#)

```
console.log(typeof 0) // "number"  
console.log(typeof 10n) // "bigint"  
console.log(typeof true) // "boolean"  
console.log(typeof "foo") // "string"  
console.log(typeof Math) // "object"  
console.log(typeof null) // "object"  
console.log(typeof alert) // "function"
```

ปล. ที่จริง null ไม่ใช่ object

3. ตัวแปรและประเภทของข้อมูล

ตัวแปรและประเภทของข้อมูล (Exercise 1)

1.1. ผลลัพธ์ที่ `console.log` ทั้งสามคืออะไร

```
let name = "Codecamp";  
  
console.log(`hello ${1}`);  
console.log(`hello ${"name"}`);  
console.log(`hello ${name}`);
```


3. ตัวแปรและประเภทของข้อมูล

ตัวแปรและประเภทของข้อมูล (Exercise 2)

- 2.1. กำหนดตัวแปรสำหรับเก็บชื่อ และกำหนดค่าเริ่มต้นเป็นชื่อของผู้เรียน
- 2.2. กำหนดตัวแปรสำหรับเก็บอายุ และกำหนดค่าเริ่มต้นเป็นอายุของผู้เรียน
- 2.3. กำหนดตัวแปรสำหรับเก็บที่อยู่ และกำหนดค่าเริ่มต้นเป็นที่อยู่ของผู้เรียน
- 2.4. กำหนดตัวแปรสำหรับเก็บประวัติของนักเรียน โดยใช้ตัวแปรทั้ง 3 ตัวด้านบนประกอบกรเขียนประวัตินี้ด้วย



หัวข้อ



Software Park Thailand
Code Camp

- JavaScript คืออะไร
- JavaScript เบื้องต้น
- ตัวแปรและประเภทของข้อมูล
- **ตัวดำเนินการเบื้องต้น**
- การเปรียบเทียบ
- การเขียนเงื่อนไข

ตัวดำเนินการเบื้องต้น

4. ตัวดำเนินการเบื้องต้น

4.1. การเปลี่ยนแปลงประเภทของข้อมูล (Type Conversion)

- ข้อความ
- ตัวเลข
- Boolean

4. ตัวดำเนินการเบื้องต้น ข้อความ

4.1. การเปลี่ยนแปลงประเภทของข้อมูล(Type Conversion)

- สามารถแปลงค่าเป็น String ได้โดยการ ใส่ใน String()
- boolean ที่มีค่าเป็น true ถ้าแปลงเป็น String จะกลายเป็น String ที่มีค่าเป็น “true” (ไม่ใช่ Boolean)

```
let var1 = true;
console.log(typeof var1) // boolean
var1 = String(var1);
console.log(var1); // กลายเป็น "true"
console.log(typeof var1); // string
```

4. ตัวดำเนินการเบื้องต้น ข้อความ

4.1. การเปลี่ยนแปลงประเภทของข้อมูล (Type Conversion)

- boolean ที่มีค่าเป็น false ถ้าแปลงเป็น String จะกลายเป็น String ที่มีค่าเป็น “false” (ไม่ใช่ Boolean)

```
let var2 = false;  
console.log(typeof var2) // boolean  
var2 = String(var2);  
console.log(var2); // กลายเป็น "false"  
console.log(typeof var2); // string
```

4. ตัวดำเนินการเบื้องต้น ข้อความ

4.1. การเปลี่ยนแปลงประเภทของข้อมูล(Type Conversion)

- null (ค่าว่าง) ถ้าแปลงเป็น String จะกลายเป็น String ที่มีค่าเป็น “null” (ไม่ใช่ค่าว่างอีกต่อไป)

```
let var3 = null;  
console.log(typeof var3) // object  
var3 = String(var3);  
console.log(var3); // กลายเป็น "null"  
console.log(typeof var3); // string
```


4. ตัวดำเนินการเบื้องต้น ตัวเลข

4.1. การเปลี่ยนแปลงประเภทของข้อมูล(Type Conversion)

- ถ้านำ String ที่เป็นตัวเลข มาหารกัน

JavaScript จะแปลงให้เป็น Number อัตโนมัติก่อนการหาร

```
let value = "9" / "4.5";  
console.log(typeof value); // number  
console.log(value); // ได้เป็น number = 2
```

4. ตัวดำเนินการเบื้องต้น ตัวเลข

4.1. การเปลี่ยนแปลงประเภทของข้อมูล(Type Conversion)

- สามารถแปลงค่าเป็น number ได้โดยการ ใส่ใน Number()

```
let str = "240";  
console.log(typeof str); // string  
let num = Number(str); // กลายเป็น number ที่มีค่า 240  
console.log(num)  
console.log(typeof num); // number
```

4. ตัวดำเนินการเบื้องต้น ตัวเลข

4.1. การเปลี่ยนแปลงประเภทของข้อมูล (Type Conversion)

- การแปลง String ที่ไม่ใช่ตัวเลข จะกลายเป็น NaN (Not a number)

```
let str1 = "ร้อยหกสิบ";  
console.log(typeof str1); // string  
let num1 = Number(str1); // แปลงข้อความเป็นตัวเลขไม่ได้จะได้ NaN แทน  
console.log(num1); // NaN  
console.log(typeof num1); // number (number อยู่ใน data type ของ number)
```

4. ตัวดำเนินการเบื้องต้น

ค่า	กลายเป็น
undefined	NaN
null	0
true	1
false	0
""	0
" 154 "	154
" 1 5 4 "	NaN
"240z"	NaN
"240.24"	240.24

ตัวเลข

ถ้า string ที่เป็นตัวเลขและมีช่องว่างข้างหน้าและข้างหลัง JavaScript จะตัดออกให้อัตโนมัติ

แต่ถ้ามีช่องว่างระหว่างตัวเลขด้วยจะกลายเป็น NaN แทนที่

ถ้าเป็น string ที่มีตัวเลขผสมตัวอักษรผสมจะกลายเป็น NaN แทนที่

ถ้าเป็น string ที่มีตัวเลขอย่างเดียวสามารถแปลงได้ปกติ

4. ตัวดำเนินการเบื้องต้น ตัวเลข

4.1. การเปลี่ยนแปลงประเภทของข้อมูล(Type Conversion)

- สามารถแปลงค่าเป็น boolean ได้โดยการ ใส่ใน Boolean()

```
console.log(Boolean(1)); // true  
console.log(Boolean(0)); // false  
console.log(Boolean("hello")); // true  
console.log(Boolean("")); // false
```

4. ตัวดำเนินการเบื้องต้น Boolean

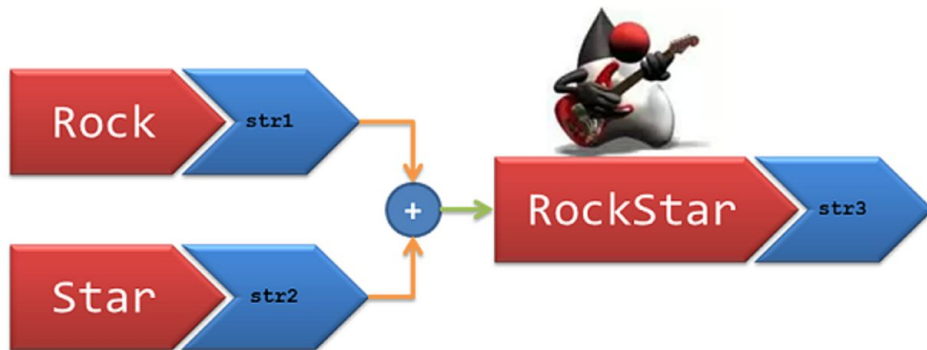
ค่า	กลายเป็น
0	false
""	false
null	false
undefined	false
NaN	false
"0"	true
" "	true
นอกเหนือจากที่เขียนมา	true

Boolean

4. ตัวดำเนินการเบื้องต้น

4.2. การต่อ String ด้วยเครื่องหมาย (+)

- การต่อ String เรียกว่า “Concatenates”



<https://1.bp.blogspot.com/-S-wOUjRIKTc/VWR6TVyJPvI/AAAAAAAAACzg/-nDp5Fj17cE/s1600/String%2Bconcatenation%2Bin%2BJava.png>

4. ตัวดำเนินการเบื้องต้น

4.2. การต่อ String ด้วยเครื่องหมาย (+) ตัวอย่าง

- การต่อ String ทำได้โดยนำ “String1” + “String2” จะได้ “String3” ที่เกิดจากการต่อกันของ “String1” และ “String2”

```
let str1 = "code" + "camp";  
console.log(str1); // "codecamp"
```


4. ตัวดำเนินการเบื้องต้น

4.2. การต่อ String ด้วยเครื่องหมาย (+) ตัวอย่าง

- การต่อ String สามารถทำได้ มากกว่า 2 Operand

```
let str2 = "Hello" + " " + "World!";  
console.log(str2); // "Hello World!"
```

4. ตัวดำเนินการเบื้องต้น

4.2. การต่อ String ด้วยเครื่องหมาย (+) ตัวอย่าง

- การนำ Operand ประเภท Number และ Operand ประเภท String มา + กัน JavaScript จะแปลง Number เป็น String ให้อัตโนมัติ และทำการต่อ String ทั้งสองให้อัตโนมัติ

```
console.log(1 + '2'); // "12"  
console.log('2' + 1); // "21"
```

ปล. สังเกตว่า ไม่ว่า Operand ที่เป็น String จะอยู่หน้าหรือหลัง
พุดง่าย ๆ ก็คือถ้ามี Operand ที่เป็น String อีกตัวหนึ่งจะถูกแปลงเป็น String โดยอัตโนมัติ

4. ตัวดำเนินการเบื้องต้น

4.2. การต่อ String ด้วยเครื่องหมาย (+) ตัวอย่าง

- แต่อย่างไรก็ตามการดำเนินการจากซ้ายไปขวา
- เพราะฉะนั้นจากตัวอย่างข้างล่างจะได้ 240

```
console.log(20 + 4 + "0"); // "240"
```

20 + 4 + "0" "240"

4. ตัวดำเนินการเบื้องต้น

4.2. การต่อ String ด้วยเครื่องหมาย (+) ตัวอย่าง

- แต่สำหรับ Operands ที่เป็น String มาดำเนินการกับ เครื่องหมาย ลบ(-), คูณ(*) และ หาร(/) JavaScript จะแปลง Operands ให้เป็น Number อัตโนมัติ

```
console.log("2" - "1"); // 1  
console.log("2" * "3"); // 6  
console.log("6" / "3"); // 2
```

4. ตัวดำเนินการเบื้องต้น

4.3. การดำเนินการของ Operands ประเภท Number

- มีการดำเนินการเบื้องต้นทั้งหมด บวก(+), ลบ(-), คูณ(*),หาร(/)
- มีการดำเนินการ Remiander (%)
- มีการดำเนินการ ยกกำลัง (**)
- มีการดำเนินการ เพิ่มขึ้น/ลดลง (++/--)

4. ตัวดำเนินการเบื้องต้น

4.3. การดำเนินการของ Operands ประเภท Number

- % คือการหารเอาเศษ
- เศษจากการที่ได้จากการหาร
- ถ้าหารลงตัวจะได้เป็น 0

```
console.log(4 % 3); // 1
console.log(15 % 3); // 0 (เศษ 0 ก็คือหารลงตัว)
console.log(27 % 5); // 2
console.log(19.5 % 1); // 0.5
```

4. ตัวดำเนินการเบื้องต้น

4.3. การดำเนินการของ Operands ประเภท Number

- ** คือการยกกำลัง (เพิ่มเข้ามาใน ES6)

```
console.log(2 ** 5); // 32
console.log(3 ** 2); // 9
console.log(0.2 ** 3); // 0.008
console.log(0.1 ** 3); // 0.001
```

ปล. ถ้าใครรันแล้วจะเห็นว่า บรรทัดที่ 3 และ 4 จะไม่ได้ 0.008 และ 0.001 เป๊ะ ๆ เนื่องจากการคำนวณในคอมพิวเตอร์ที่อยู่ในฐานสองทั้งหมดเมื่อคำนวณแล้วแปลงเป็นฐานสิบแสดงให้เราเห็น การแปลงทศนิยมฐานสองเป็นฐานสิบจะไม่ลงตัวเสมอไปจึงเป็นแบบที่เห็น
ตอนนี้ ภาษาต่าง ๆ ก็มีการแก้ไขปัญหานี้ สามารถหาข้อมูลเพิ่มเติมได้ใน Google เลยครับ

4. ตัวดำเนินการเบื้องต้น

4.3. การดำเนินการของ Operands ประเภท Number

- ++ คือการเพิ่มขึ้น 1 `5++; // error`
- ใช้ได้กับตัวแปรเท่านั้น ไม่สามารถใช้กับ ตัวเลข ได้
- การเขียนทั้งสองแบบได้ ผลเหมือนกัน แต่การเขียนด้วย ++ จะสั้นกว่า

```
let num = 2;  
num = num + 1;  
console.log(num); // 3
```

```
let num = 2;  
num++;  
console.log(num); // 3
```


4. ตัวดำเนินการเบื้องต้น

4.3. การดำเนินการของ Operands ประเภท Number

- -- คือการลดลง 1
- ใช้ได้กับตัวแปรเท่านั้น ไม่สามารถใช้กับ ตัวเลข ได้
- การเขียนทั้งสองแบบได้ผลเหมือนกัน แต่การเขียนด้วย -- จะสั้นกว่า

```
let num = 2;  
num = num - 1;  
console.log(num); // 1
```

```
let num = 2;  
num--;  
console.log(num); // 1
```

4. ตัวดำเนินการเบื้องต้น

4.3. การดำเนินการของ Operands ประเภท Number

*** Optional *** (ไม่ต้องจำ)

- การใส่ ++ ข้างหน้า และ ข้างหลังตัวแปรให้ผลต่างกัน
- การใส่ ++ ข้างหน้าจะทำให้การเพิ่มค่าก่อนนำตัวแปรไปใช้
- การใส่ ++ ข้างหลังจะนำตัวแปรไปใช้การแล้วค่อยเพิ่มค่าขึ้นหนึ่ง

```
let number = 1;  
console.log(++number); // 2  
console.log(number); // 2
```

```
let number = 1;  
console.log(number++); // 1  
console.log(number); // 2
```

4. ตัวดำเนินการเบื้องต้น

4.4. Bitwise Operators

- AND &
- OR |
- XOR (^) → เหมือนกันเป็น 0 ต่างกันเป็น 1
- NOT (~)
- LEFT SHIFT (<<)
- RIGHT SHIFT (>>)
- ZERO-FILL RIGHT SHIFT (>>>)

4. ตัวดำเนินการเบื้องต้น

4.5. การดำเนินการแบบย่อ

- การบวก ลบ คูณ หาร ให้กับตัวแปร และนำไปใส่ตัวแปรเดิมสามารถเขียนแบบย่อได้

```
let num = num + 5;
```

- การเขียนทั้งสองเหมือนกัน แต่แบบหลังจะเป็นแบบย่อ

```
let num = 5;  
num = num + 5; // 10
```

```
let num = 5;  
num += 5; // 10
```

4. ตัวดำเนินการเบื้องต้น

4.5. การดำเนินการแบบย่อ

- ทำได้ทั้ง บวก(+), ลบ(-), คูณ(*) และหาร(/)

```
let num = 5;  
num = num + 5; // 10  
num += 5; // 15  
num *= 2; // 30  
num /= 3; // 10  
num -= 7; // 3  
console.log(num);
```

4. ตัวดำเนินการเบื้องต้น

4.6. แบบฝึกหัด การดำเนินการเบื้องต้น

1. ให้ระบุค่าของ a, b, c และ d หลังจากจบ statements ทั้งสามบรรทัด

```
let a = 1, b = 1;  
  
let c = ++a;  
let d = b++;
```

4. ตัวดำเนินการเบื้องต้น

4.6. แบบฝึกหัด การดำเนินการเบื้องต้น

2. จงหาผลลัพธ์ของ Statement ต่อไปนี้

- | | |
|------------------------------|--------------------------------|
| 1. <code>"" + 1 + 0</code> | 11. <code>" -9 " + 5</code> |
| 2. <code>"" - 1 + 0</code> | 12. <code>" -9 " - 5</code> |
| 3. <code>true + false</code> | 13. <code>null + 1</code> |
| 4. <code>6 / "3"</code> | 14. <code>undefined + 1</code> |
| 5. <code>"2" * "3"</code> | 15. <code>" \t \n " - 2</code> |
| 6. <code>4 + 5 + "px"</code> | |
| 7. <code>"\$" + 4 + 5</code> | |
| 8. <code>"4" - 2</code> | |
| 9. <code>"4px" - 2</code> | |
| 10. <code>7 / 0</code> | |



หัวข้อ



Software Park Thailand
</Code Camp>

- JavaScript คืออะไร
- JavaScript เบื้องต้น
- ตัวแปรและประเภทของข้อมูล
- ตัวดำเนินการเบื้องต้น
- การเปรียบเทียบ
- การเขียนเงื่อนไข



Software Park Thailand
</Code Camp>

การเปรียบเทียบ

5. การเปรียบเทียบ

5.1. การเปรียบเทียบ

- การเปรียบเทียบมีทั้งหมด 4 แบบหลัก ๆ
 1. มากกว่า($>$) และ น้อยกว่า($<$)
 2. มากกว่าเท่ากับ($>=$) และ น้อยกว่าเท่ากับ($<=$)
 3. ไม่เท่ากับ (\neq)
 4. เท่ากับ ($=$)
- ผลลัพธ์ของการเปรียบเทียบจะได้ Boolean

5. การเปรียบเทียบ

5.1. การเปรียบเทียบ

- การเปรียบเทียบ String สามารถเก็บค่าไว้ในตัวแปรได้ด้วย

```
let vars = 5 < 2;  
console.log(vars); // false
```

5. การเปรียบเทียบ

5.1. การเปรียบเทียบ - ตัวอย่าง

```
console.log(3 < 5); // true (correct)
console.log(4 == 2 + 2); // false (incorrect)
console.log(1 > 2); // false (incorrect)
```

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String

- การเรียงของ String ใช้การเรียงที่มีชื่อว่า “Dictionary” หรือ “Lexicographical”
- หรือพูดง่าย ๆ ก็คือเรียงแบบพิจารณาโดยดูจากอักษรทุกตัว

```
console.log('A' < 'Z');           // true
console.log('A' < 'a');           // true
console.log('coco' < 'code');     // true
console.log('1' < '2');           // true
console.log('Be' < 'Bee');        // true
```

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String

วิธีเรียงของ String

1. พิมพ์ใหญ่มีค่าน้อยกว่า(มาก่อน)พิมพ์เล็ก

“A” < “a”

2. A มีค่าน้อยกว่า(มาก่อน) Z

“b” < “c”

3. ตัวเลขมาก่อนตัวอักษร

“1” < “A”

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String

วิธีเรียงของ String

- การเรียงมาจาก Unicode Order
- ดูรหัสของตัวอักษรได้จาก List of Unicode characters

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String

วิธีการเปรียบเทียบสตริง (String comparison algorithm)

1. เปรียบเทียบตัวอักษรตัวแรก
2. ถ้าตัวอักษรตัวแรกมีค่ามากกว่าตัวที่สอง จะทำให้ String ตัวแรกมีค่ามากกว่าตัวที่สองทันที
3. ถ้าตัวแรกมีค่าเท่ากัน เปรียบเทียบตัวที่สองต่อ
4. ทำไปเรื่อย ๆ จนกว่าจะจบตัวอักษรทุกตัวใน String นั้น
5. ถ้า String ทั้งสองมีความยาวเท่ากัน ก็ให้ String สองตัวนั้นเท่ากัน
6. ถ้าความยาวไม่เท่ากัน ตัวที่มีความยาวมากกว่ามีค่ามากกว่า

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String - ตัวอย่างที่ 1

วิธีการเปรียบเทียบสตริง (String comparison algorithm)

1. เปรียบเทียบตัวอักษรตัวแรก
2. ถ้าตัวอักษรตัวแรกมีค่ามากกว่าตัวที่สอง จะทำให้ String ตัวแรกมีค่ามากกว่าตัวที่สองทันที
3. ถ้าตัวแรกมีค่าเท่ากัน เปรียบเทียบตัวที่สองต่อ
4. ทำไปเรื่อย ๆ จนกว่าจะจบตัวอักษรทุกตัวใน String
5. ถ้า String ทั้งสองมีความยาวเท่ากัน ก็ให้ String สองตัวนั้นเท่ากัน
6. ถ้าความยาวไม่เท่ากัน ตัวที่มีความยาวมากกว่ามีค่ามากกว่า

“A” < “Z”

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String - ตัวอย่างที่ 2

วิธีการเปรียบเทียบสตริง (String comparison algorithm)

1. เปรียบเทียบตัวอักษรตัวแรก
2. ถ้าตัวอักษรตัวแรกมีค่ามากกว่าตัวที่สอง จะทำให้ String ตัวแรกมีค่ามากกว่าตัวที่สองทันที
3. ถ้าตัวแรกมีค่าเท่ากัน เปรียบเทียบตัวที่สองต่อ
4. ทำไปเรื่อย ๆ จนกว่าจะจบตัวอักษรทุกตัวใน String
5. ถ้า String ทั้งสองมีความยาวเท่ากัน ก็ให้ String สองตัวนั้นเท่ากัน
6. ถ้าความยาวไม่เท่ากัน ตัวที่มีความยาวมากกว่ามีค่ามากกว่า

“Good” > “Gold”

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String - ตัวอย่างที่ 3

วิธีการเปรียบเทียบสตริง (String comparison algorithm)

1. เปรียบเทียบตัวอักษรตัวแรก
2. ถ้าตัวอักษรตัวแรกมีค่ามากกว่าตัวที่สอง จะทำให้ String ตัวแรกมีค่ามากกว่าตัวที่สองทันที
3. ถ้าตัวแรกมีค่าเท่ากัน เปรียบเทียบตัวที่สองต่อ
4. ทำไปเรื่อย ๆ จนกว่าจะจบตัวอักษรทุกตัวใน String
5. ถ้า String ทั้งสองมีความยาวเท่ากัน ก็ให้ String สองตัวนั้นเท่ากัน
6. ถ้าความยาวไม่เท่ากัน ตัวที่มีความยาวมากกว่ามีค่ามากกว่า

“Be” = “Be”

5. การเปรียบเทียบ

5.2. การเปรียบเทียบ String - ตัวอย่างที่ 4

วิธีการเปรียบเทียบสตริง (String comparison algorithm)

1. เปรียบเทียบตัวอักษรตัวแรก
2. ถ้าตัวอักษรตัวแรกมีค่ามากกว่าตัวที่สอง จะทำให้ String ตัวแรกมีค่ามากกว่าตัวที่สองทันที
3. ถ้าตัวแรกมีค่าเท่ากัน เปรียบเทียบตัวที่สองต่อ
4. ทำไปเรื่อย ๆ จนกว่าจะจบตัวอักษรทุกตัวใน String
5. ถ้า String ทั้งสองมีความยาวเท่ากัน ก็ให้ String สองตัวนั้นเท่ากัน
6. ถ้าความยาวไม่เท่ากัน ตัวที่มีความยาวมากกว่ามีค่ามากกว่า

“Be” < “Bee”

5. การเปรียบเทียบ

5.3. การเปรียบเทียบข้อมูลต่างประเภทกัน

1. ถ้ามีการเปรียบเทียบคนละประเภทกัน JavaScript จะแปลงข้อมูลนั้นเป็น Numbers ทั้งหมด

```
console.log( '3' < 5 ); // true, string '2' จะถูกแปลงเป็น number ที่มีค่า 2  
console.log( '01' == 1 ); // true, string '01' จะถูกแปลงเป็น number ที่มีค่า 1
```

5. การเปรียบเทียบ

5.3. การเปรียบเทียบข้อมูลต่างประเภทกัน

2. สำหรับข้อมูลประเภท Boolean ค่า true จะถูกแปลงเป็น number ที่มีค่าเป็น 1 และ ค่า false จะถูกแปลงเป็น number ที่มีค่าเป็น 0

```
console.log( true == 1); // true, เนื่องจาก ค่า true จะถูกแปลงเป็น number ที่มีค่า 1  
console.log( false == 0); // true, เนื่องจาก ค่า false จะถูกแปลงเป็น number ที่มีค่า 0
```

5. การเปรียบเทียบ

5.4. ความแตกต่างระหว่าง == และ ===

- == จะเช็คแค่ Value อย่างเดียว
- === จะเช็คทั้ง Value และ ประเภทของข้อมูล ด้วย

5. การเปรียบเทียบ

5.4. ความแตกต่างระหว่าง == และ ===

- ตัวอย่าง

```
console.log(1 == "1"); // true, เนื่องจากค่าเท่ากัน  
console.log(1 === "1"); // false, ถึงแม้ค่าจะเท่ากัน แต่ประเภทไม่ตรงกัน
```

5. การเปรียบเทียบ

5.5. ความแตกต่างระหว่าง $!=$ และ $!==$

- $!=$ จะเช็คแค่ Value อย่างเดียว
- $!==$ จะเช็คทั้ง Value และ ประเภทของข้อมูล ด้วย

5. การเปรียบเทียบ

5.5. ความแตกต่างระหว่าง `!=` และ `!==`

- ตัวอย่าง

```
console.log(1 != "1"); // false, เพราะมันเท่ากัน เนื่องจากไม่สนใจประเภท  
console.log(1 !== "1"); // true, แต่เนื่องจากคนละประเภท จึงไม่เท่ากัน
```

5. การเปรียบเทียบ

5.6. การเปรียบเทียบกับค่า null และ undefined

- null และ undefined จะเท่ากัน เมื่อเทียบกับ ==
- null และ undefined จะไม่เท่ากัน เมื่อเทียบกับ ===

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

5. การเปรียบเทียบ

5.6. การเปรียบเทียบกับค่า null และ undefined

- ไม่ควรใช้ มากกว่าหรือน้อยกว่า กับ null และ undefined

```
console.log(null > 0); // false  
console.log(null == 0); // false  
console.log(null >= 0); // true
```

```
console.log(undefined > 0); // false  
console.log(undefined < 0); // false  
console.log(undefined == 0); // false
```

5. การเปรียบเทียบ

5.7. การเปรียบเทียบ - แบบฝึกหัด

1. จงหาค่าของการเปรียบเทียบต่อไปนี้

- $5 > 4$ **true**
- `"apple" > "pineapple"` **false**
- `"2" > "12"` **true**
- `undefined == null` **true**
- `undefined === null` **false**
- `"bee" < "be"` **false**
- `"bee" > "Bee"` **true**
- `"Bee" < "be"` **true**



หัวข้อ



Software Park Thailand
</Code Camp>

- JavaScript คืออะไร
- JavaScript เบื้องต้น
- ตัวแปรและประเภทของข้อมูล
- ตัวดำเนินการเบื้องต้น
- การเปรียบเทียบ
- การเขียนเงื่อนไข

การเขียนเงื่อนไข

6. การเขียนเงื่อนไข

6.1. การเขียนเงื่อนไข if

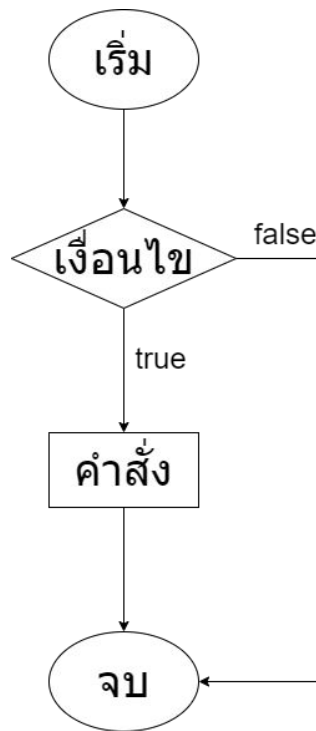
- if จะถูกใช้เมื่อในโปรแกรมมีทางเลือกเกิดขึ้น เช่น ถ้าอายุมากกว่า 18 ให้แสดงภาพข้อความเตือนผู้ใช้ เป็นต้น
- คำสั่ง(Statement) สามารถใส่ได้มากกว่า 1 คำสั่ง

```
if( เงื่อนไข ){  
    // คำสั่ง  
}
```

6. การเขียนเงื่อนไข

6.1. การเขียนเงื่อนไข if

- การทำงานของ if



6. การเขียนเงื่อนไข

6.1. การเขียนเงื่อนไข if

- การทำงานของ if (ตัวอย่าง)

```
let year = prompt('ปีแรกที่จัด Codecamp คือปีอะไร');  
  
if (year == 2018) {  
    alert('ถูกต้อง');  
    alert('คุณเก่งมาก ๆ');  
}
```

6. การเขียนเงื่อนไข

6.2. การเขียนเงื่อนไข if-else

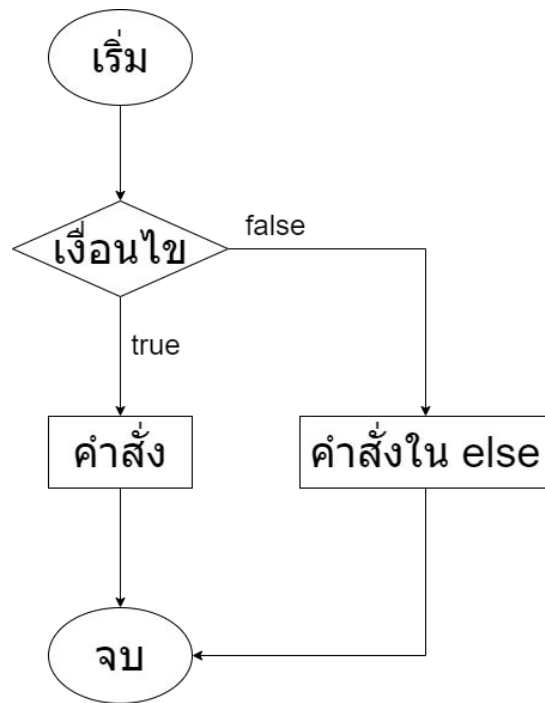
- เขียนเมื่อเกิดทางเลือก เช่นเดียวกับ if
- ถ้าไม่ตรงเงื่อนไขใน if คำสั่งใน else จะทำงาน

```
if (เงื่อนไข) {  
    //คำสั่งเมื่อเงื่อนไขเป็นจริง  
} else {  
    //คำสั่งเมื่อเงื่อนไขเป็นเท็จ  
}
```

6. การเขียนเงื่อนไข

6.2. การเขียนเงื่อนไข if-else

- การทำงานของ if-else



6. การเขียนเงื่อนไข

6.2. การเขียนเงื่อนไข if-else

- การทำงานของ if-else (ตัวอย่าง)

```
let year = prompt('ปีแรกที่จัด Codecamp คือปีอะไร');

if (year == 2018) {
  alert('ถูกต้อง');
  alert('คุณเก่งมาก ๆ');
} else {
  alert('ผิดนะ');
  alert('กด F5 ลองตอบใหม่นะ');
}
```

6. การเขียนเงื่อนไข

6.3. การเขียนเงื่อนไข else-if

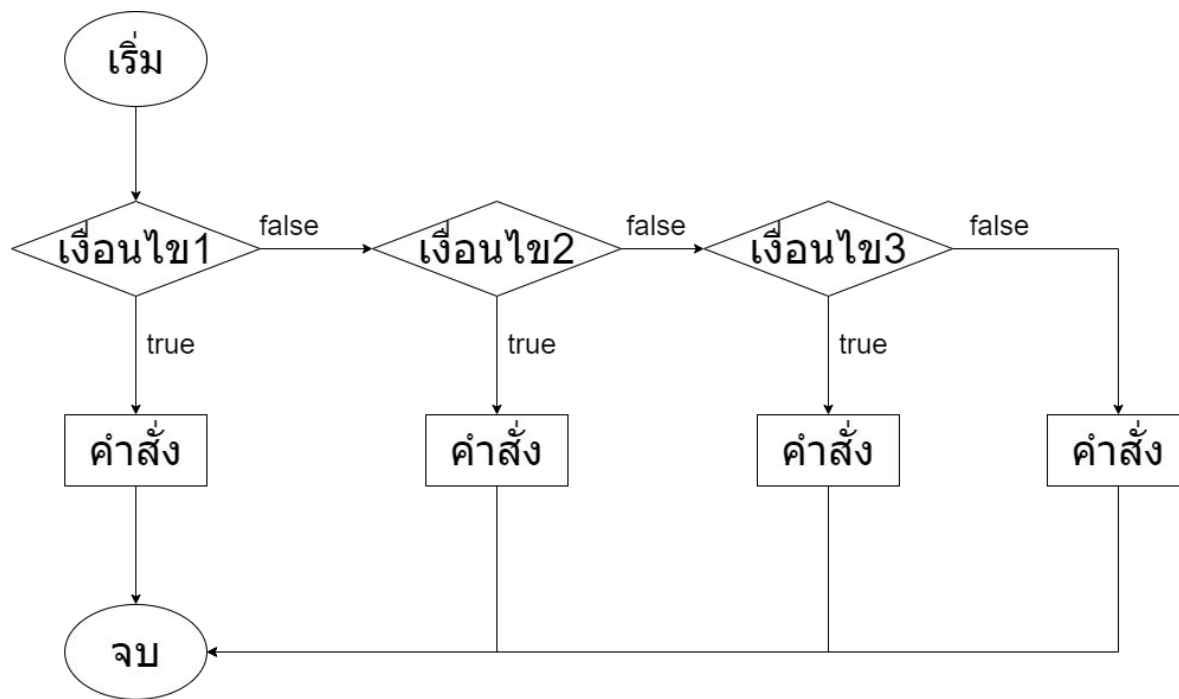
- โปรแกรมจะเช็คเงื่อนไขที่ 1 ก่อน
ถ้าเงื่อนไขเป็นจริงก็ทำคำสั่งของ
เงื่อนไขแรกแล้วจบเลย
- ถ้าเงื่อนไขแรกไม่ตรงโปรแกรมจะ
เช็คเงื่อนไขที่สอง
- ถ้าเงื่อนไขที่สองไม่ตรงโปรแกรม
จะเช็คเงื่อนไขที่สาม ทำแบบนี้ไป
เรื่อย ๆ จนครบทุกเงื่อนไข
- ถ้าไม่ตรงเงื่อนไขใด ๆ เลย โปรแกรมจะทำคำสั่งของ else

```
if (เงื่อนไขที่1) {  
    // คำสั่งเมื่อตรงเงื่อนไขที่ 1  
} else if (เงื่อนไขที่2) {  
    // คำสั่งเมื่อตรงเงื่อนไขที่ 2  
} else if (เงื่อนไขที่3) {  
    // คำสั่งเมื่อตรงเงื่อนไขที่ 3  
} else {  
    // คำสั่งเมื่อไม่ตรงเงื่อนไขใด ๆ เลย  
}
```


6. การเขียนเงื่อนไข

6.3. การเขียนเงื่อนไข else-if

- การทำงาน
ของ else-if



6. การเขียนเงื่อนไข

6.3. การเขียนเงื่อนไข else-if

- การทำงานของ else-if (ตัวอย่าง)

```
if (year == 2018) {  
    alert('ถูกต้อง');  
} else if (year == 2017 || year == 2019) {  
    alert('เกือบแล้วนะ อีกนิดนึง');  
} else {  
    alert('ไม่ใกล้เคียงเลยนะ')  
}
```

6. การเขียนเงื่อนไข

6.4. การเขียนเงื่อนไขแบบ Ternary-Operator

- เป็นการเขียน if-else แบบย่อ
- ใช้สำหรับการกำหนดค่าเท่านั้น

```
let age = prompt("โปรดใส่อายุคุณ");  
let message;  
  
if (age < 18) {  
    message = 'คุณเข้าไปไม่ได้'  
} else {  
    message = 'คุณเข้าไปได้'  
}  
  
alert(message)
```

```
let age = prompt("โปรดใส่อายุคุณ");  
let message;  
  
message = (age < 18) ? 'คุณเข้าไปไม่ได้' : 'คุณเข้าไปได้';  
  
alert(message)
```

6. การเขียนเงื่อนไข

6.4. การเขียนเงื่อนไขแบบ Ternary-Operator

- ถ้าเงื่อนไขเป็นจริงค่าที่อยู่ข้างหน้าจะถูกกำหนดให้กับตัวแปร
- ถ้าเงื่อนไขเป็นเท็จค่าที่อยู่ข้างหลังจะถูกกำหนดให้กับตัวแปร

ตัวแปร = (เงื่อนไข) ? ค่าเมื่อเป็นจริง : ค่าเมื่อเป็นเท็จ

```
let age = prompt("โปรดใส่อายุคุณ");  
let message;  
  
message = (age < 18) ? 'คุณเข้าไปไม่ได้' : 'คุณเข้าไปได้';  
  
alert(message)
```

6. การเขียนเงื่อนไข

6.5. แบบฝึกหัด

1. Browser จะโชว์ข้อความ “Hello Codecamp #5” ไหม

```
if ("0") {  
    alert('Hello Codecamp #5');  
}
```

6. การเขียนเงื่อนไข

6.5. แบบฝึกหัด

2. ใช้ if else ในการเขียน ถามชื่อ ของคุณ
 - ถ้าตอบ ถูก ให้แสดงคำว่า “เก่งมาก”
 - ถ้าตอบ ผิด ให้แสดงคำว่า “คุณไม่รู้จักชื่อนั้น”

6. การเขียนเงื่อนไข

6.5. แบบฝึกหัด

3. ใช้ prompt ในการรับคะแนนมาคำนวณเกรด

ถ้าคะแนน มากกว่าเท่ากับ 80 ได้ A

ถ้าคะแนน อยู่ระหว่าง 70 - 79 ได้ B

ถ้าคะแนน อยู่ระหว่าง 60 - 69 ได้ C

ถ้าคะแนน อยู่ระหว่าง 50 - 59 ได้ D

ถ้าคะแนน น้อยกว่า 50 ได้ F

6. การเขียนเงื่อนไข

6.5. แบบฝึกหัด

4. เปลี่ยน if-else ข้างล่างให้อยู่ในรูปของ Ternary Operators

```
let age = prompt('How old are you?');  
let price;  
if (age < 18) {  
  price = 2000;  
} else {  
  price = 3500;  
}
```

