

Javascript The Complete Reference

JS



รวบรวมตัวอย่างการใช้งาน
javascript ไว้มากที่สุด

Version beta



กฤษณวัฒน์ แก้วแสนเมือง

ผู้สนับสนุน

www.pathosting.co.th
P&T HOSTING
P&T HOSTING

Hosting ทดลองฟรี 30 วัน
บริการติดตั้งโปรแกรมต่างๆฟรี

99
บาท/ปี



DigitalOcean

Built for developers. Deploy an SSD cloud server in 55 seconds!



ติดปัญหาเกี่ยวกับ HTML Css JavaScript PHP หรือ SQL

โพสถามตามกระทู้หรือ Facebook Group

ได้คำตอบบ้าง ไม่ได้บ้าง

ถามทีนี้ได้คำตอบ **99%**



RUK-COM
Hosting

Roadmap

1. เพื่อเนื้อหาส่วนของ weakest, iterator, generator
2. แสดงว่าคำสั่งนี้ใช้กับบราวเซอร์เวอร์ชันไหนได้บ้าง
3. เพิ่มตัวอย่างการใช้งานกับพารามิเตอร์ทุกตัว
4. เพิ่มตัวอย่างการจัดการ dom
5. เพิ่มตัวอย่างการจัดการ css
6. เพิ่มตัวอย่างการใช้งานกับ event



GETLINKS

ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

สารบัญ

Array	1
Concat	1
copyWithin	1
entries.....	2
every	2
Fill	2
filter.....	3
find.....	4
findIndex.....	4
forEach	5
From	6
includes	6
indexOf	7
isArray	7
Join	8
lastIndexOf	8
Length.....	9
Map.....	10



RUK-COM
Hosting

Of	10
Pop.....	11
Push.....	11
Reverse.....	12
Shift	12
Slice	13
Some.....	13
Sort	14
Splice	15
toString.....	16
Unshift.....	16
Values.....	17
Date.....	18
getDate	18
getDay.....	19
getFullYear	19
getHours.....	20
getMilliseconds.....	20
getMinutes	21
getMonth	21
getSeconds	22
getTime	22
getUTCDate	23



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

getUTCDay	23
getUTCFullYear	24
getUTCHours.....	24
getUTCMilliseconds.....	25
getUTCMinutes	25
getUTCMonth	26
getUTCSeconds	26
Now.....	27
setDate	27
setFullYear	28
setHours	28
setMilliseconds	29
setMinutes	29
setMonth	30
setSeconds.....	30
setUTCDate	31
setUTCFullYear	31
setUTCHours.....	32
setUTCMilliseconds.....	32
setUTCMinutes	33
setUTCMonth	33
setUTCSeconds	34
toDateString	34



toISOString	35
toJSON	35
toLocaleDateString	36
toLocaleString	36
toLocaleTimeString	37
toString	37
toTimeString	38
toUTCString	38
Errors	39
Message	39
Name	39
toString	40
Global Objects	40
decodeURI	40
encodeURI	41
isFinite	41
isNaN	42
NaN	43
Null	44
parseFloat	44
parseInt	45
Undefined	46
Escape	46



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

unescape	47
Intl	48
Collator	48
Compare	48
resolvedOptions	49
DateTimeFormat	50
supportedLocalesOf	50
NumberFormat	51
JSON	51
Parse	51
Stringify	52
Map	52
Clear	52
Delete	53
Entries	54
forEach	54
Get	55
Has	55
Keys	56
Set	57
Size	58
Values	58
WeakMap	59



Weakest.....	60
Weakdelete.....	60
Weakget.....	61
Weakhas.....	61
Math.....	62
Abs.....	62
Acos.....	63
Acosh.....	63
Asin.....	63
Asinh.....	64
Atan	64
Atanh	65
Cbrt.....	65
Ceil.....	66
Cos.....	66
Cosh.....	67
E.....	67
Floor.....	68
log10.....	68
log2.....	69
Max	69
Min.....	70
PI.....	70



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

Pow	71
Random	71
Round	71
Sin	72
Sqrt	72
Tan	73
Tanh	73
Miscellaneous	74
Eval	74
Number	75
isFinite	75
isInteger	75
isNaN	76
parseFloat	77
parseInt	78
toExponential	79
toFixed	79
toLocaleString	80
toString	80
Object	81
Assign	81
Create	81
defineProperties	82



Freeze	83
getOwnPropertyDescriptor	84
hasOwnProperty	85
Is	86
isExtensible	87
isFrozen	88
isPrototypeOf	89
isSealed	90
Keys	91
preventExtensions	91
Seal	92
toString	93
Promise	94
All	94
Catch	95
Race	96
Reject	96
Resolve	97
then	97
RegExp	98
Match	98
Replace	99
Search	99



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

Split.....	100
lastIndex	100
Exec	101
Flags	101
Global.....	102
ignoreCase	102
Multiline.....	103
Source	103
Test.....	104
toString.....	104
Unicode.....	105
Set.....	105
Add.....	105
Clear.....	106
Delete.....	106
Entries.....	107
forEach	107
Has.....	108
values	109
size	110
String	110
Anchor	110
charAt.....	111



Concat	111
endsWith	112
Includes	112
indexOf	113
Length.....	114
Link	114
Match.....	115
Normalize	115
padEnd.....	116
padStart.....	116
Raw	117
repeat.....	117
Replace	118
Search.....	118
Slice	119
Split.....	119
startsWith	120
Substr	120
substring	121
toLowerCase	121
toString.....	122
toUpperCase	122
trim	123



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

valueOf	123
Proxy.....	124
Construct.....	124
defineProperty	125
deleteProperty.....	125
Get	126
Apply.....	127
getOwnPropertyDescriptor	128
getPrototypeOf.....	128
Has.....	129
isExtensible	130
ownKeys	131
preventExtensions	132
Set.....	133
setPrototypeOf.....	134





GETLINKS

ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย 

Array

Concat

ใช้สำหรับรวม array หลายๆ ตัวเข้าด้วยกัน

ตัวอย่างที่ 1

```
var alpha = ['a', 'b', 'c'],  
    numeric = [1, 2, 3];  
  
var alphaNumeric = alpha.concat(numeric);  
document.getElementById('output').innerHTML = alphaNumeric;
```

ผลลัพธ์

a,b,c,1,2,3

ผลลัพธ์แบบ [live](#)

copyWithin

คัดลอกอาเรย์แล้ววางทับลงในตัวของมันเอง

รูปแบบการใช้งาน

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];  
fruits.copyWithin(2, 0);
```

array.copyWithin(target,start,end)

target คือ ตำแหน่งในอาเรย์ที่ต้องการคัดลอก

start คือ ตำแหน่งที่ต้องการแทน



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

entries

สร้างอาร์เรย์อ็อบเจกต์ขึ้นมาจากอาร์เรย์ธรรมดา

ตัวอย่างที่ 1

```
var arr = ['a', 'b', 'c'];
var eArr = arr.entries();

console.log(eArr.next().value);
console.log(eArr.next().value);
console.log(eArr.next().value);
```

ผลลัพธ์

```
▶ [0, "a"]
▶ [1, "b"]
▶ [2, "c"]
```

every

ใช้ตรวจสอบว่าค่าที่กำหนดไว้ผ่านเงื่อนไขทั้งหมดหรือไม่

ตัวอย่างที่ 1 ตรวจสอบว่าอาร์เรย์มีค่ามากกว่า 10 ทุกตัวหรือไม่

```
function isBigEnough(element, index, array) {
    return element >= 10;
}

[12, 5, 8, 130, 44].every(isBigEnough);
[12, 54, 18, 130, 44].every(isBigEnough);
```

ผลลัพธ์

```
false
true
```

Fill

ใช้เปลี่ยนแปลงค่าในอาร์เรย์



RUK-COM
Hosting

ตัวอย่างที่ 1 แปลงทุกค่าเลย

```
var animal = ["elephant", "tiger", "cat", "dog"];
animal.fill("dolphin");
```

ผลลัพธ์

```
< ["dolphin", "dolphin", "dolphin", "dolphin"]
```

ตัวอย่างที่ 2 มีจุดเริ่มต้น

```
var animal = ["elephant", "tiger", "cat", "dog"];
animal.fill("dolphin", 1);
```

ผลลัพธ์

```
< ["elephant", "dolphin", "dolphin", "dolphin"]
```

ตัวอย่างที่ 3 มีจุดสิ้นสุด

```
var animal = ["elephant", "tiger", "cat", "dog"];
animal.fill("dolphin", 1, 3);
```

ผลลัพธ์

```
< ["elephant", "dolphin", "dolphin", "dog"]
```

filter

ใช้สร้างอาร์เรย์จากค่าที่ผ่านเงื่อนไข

ตัวอย่างที่ 1



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

```
var ages = [32, 33, 16, 40];

function checkAge(age) {
  return age <= 18;
}

ages.filter(checkAge);
```

ผลลัพธ์

< [16]

find

ใช้สำหรับค้นหาข้อมูลในอาเรย์โดยใช้เงื่อนไขเป็นฟังก์ชัน

ตัวอย่างที่ 1

```
var ages = [32, 33, 16, 40];

function checkAge(age) {
  return age >= 18;
}

ages.find(checkAge);
```

ผลลัพธ์

< 32

findIndex

ใช้สำหรับค้นหาข้อมูลในอาเรย์โดยใช้เงื่อนไขเป็นฟังก์ชัน โดยจะส่งเป็นลำดับของค่าที่ตรงกลับมาแทน

ตัวอย่างที่ 1



RUK-COM
Hosting

```
var ages = [32, 33, 16, 40];

function checkAge(age) {
    return age >= 18;
}

ages.findIndex(checkAge);
```

ผลลัพธ์

< 0

forEach

ใช้สำหรับเรียกใช้ฟังก์ชันเพื่อส่งค่าเป็นสมาชิกในอาเรย์ในครั้งเดียว โดยฟังก์ชัน **callback** ที่จะใช้
งานร่วมนั้นต้องมีรูปแบบดังนี้

function callback (value, index, array)

value ค่าของอาเรย์

index ลำดับของสมาชิกในอาเรย์

array ค่าที่ต้องการเสริมเข้าไป

ตัวอย่างที่ 1

```
<p>กดที่ปุ่ม</p>
<button onclick="numbers.forEach(formular)">ลองดู</button>

<p>หาผลรวมของค่าในอาเรย์: <span id="summon"></span></p>

<script>
    var sum = 0;
    var numbers = [34, 232, 55, 232, 285];

    function formular(item) {
        sum += item;
        summon.innerHTML = sum;
    }
</script>
```



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ผลลัพธ์

กดที่ปุ่ม

ลองดู

หาผลรวมของค่าในอาเรย์: 838

ผลลัพธ์แบบ [live](#)

From

สร้างอาเรย์โดยรับค่าอาทิวเมนต์ที่รับเข้ามาเป็นอ็อบเจกต์ เช่น set, map,string

ตัวอย่างที่ 1

```
var numbers = [4, 16, 9, 25];
var summ = numbers.map(Math.sqrt);
myvar = Array.from(summ);
console.log(myvar);
```

ผลลัพธ์

`Array [2, 4, 3, 5]`

includes

ตรวจสอบว่ามีค่าที่ต้องการอยู่ในอาเรย์ หรือไม่



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var arr = [1,2,3,4,5,6,7,8,9];
var myvar = arr.includes(2);
console.log(myvar);
```

ผลลัพธ์

true

indexOf

ตรวจหาตำแหน่งของค่าที่ต้องการในอาเรย์

ตัวอย่างที่ 1

```
var arr = [1,2,3,4,5,6,7,8,9,9];
var myvar = arr.indexOf(5);
console.log(myvar);
```

ผลลัพธ์

ถ้าหากว่าหาค่าไม่เจอจะคืนค่า -1

4

isArray

ตรวจว่าข้อมูลเป็นอาเรย์หรือไม่

ตัวอย่างที่ 1

```
var arr = [];
Array.isArray(arr);
```

ตัวอย่างที่ 2

```
var arr = 33;
Array.isArray(arr);
```

ผลลัพธ์

false

1. ผลลัพธ์
- 2.
3. รับโบนัส 10,000 บาท



ผลลัพธ์

```
true
```

Join

แปลงอาเรย์ให้เป็นประโยคโดยใส่ตัวขึ้นให้ด้วย

ตัวอย่างที่ 1

```
var a = ['one', 'two', 'three'];
var myVar1 = a.join();
console.log(myVar1);
```

ผลลัพธ์

Wind,Rain,Fire

ตัวอย่างที่ 2

```
var a = ['one', 'two', 'three'];
var myVar2 = a.join(', ');
console.log(myVar2);
```

ผลลัพธ์

one, two, three

ตัวอย่างที่ 3

```
var a = ['one', 'two', 'three'];
var myVar3 = a.join(' + ');
console.log(myVar3);
```

ผลลัพธ์

one + two + three

ตัวอย่างที่ 4

```
var a = ['one', 'two', 'three'];
var myVar4 = a.join('');
console.log(myVar4);
```

ผลลัพธ์

onetwothree

lastIndexOf

หาตำแหน่งที่ข้อมูลที่ต้องการจากหลังมาหน้า หรือจากตำแหน่งที่กำหนด



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "A", "B", "C", "D", "E"];
var myvar = arr.lastIndexOf("B");
console.log(myvar);
```

ผลลัพธ์

6

ตัวอย่างที่ 2

ถ้าหากว่าไม่เจอค่าที่ค้นหาจะคืนค่า -1 หรือ 1

```
var arr = [1,2,3,4,5,6,7,8,9,9];
var myvar = arr.lastIndexOf(99);
console.log(myvar);
```

ผลลัพธ์

-1

Length

ค้นหาจำนวนสมาชิกของอาเรย์นั้น ๆ



ตัวอย่างที่ 2

```
var arr = ["A", "B", "C", "D", "E", "A", "B", "C", "D", "E"];
var myvar = arr.length;
console.log(myvar);
```

ผลลัพธ์

10

Map

สร้างอาเรย์ใหม่โดยดึงข้อมูลในอาเรย์แล้วส่งไปให้ฟังก์ชัน แล้วได้ค่าใหม่กลับออกมาโดยไม่ทำให้ค่าเดิมเสียรูป

ตัวอย่างที่ 1

นำเอาค่าในอาเรย์มาหารากที่สองโดยที่ค่าของอาเรย์ตัวเดิม ไม่เปลี่ยนแปลงหลังจากผ่านฟังก์ชันใหม่

```
var numbers = [4, 9, 16, 25];
myvar = numbers.map(Math.sqrt);
console.log(myvar);
console.log(numbers);
```

ผลลัพธ์

Array [2, 3, 4, 5]

Array ["4", "9", "16", "25"]

Of

สร้างอาเรย์โดยรับอาร์กิวเมนต์เข้ามา ก็ได้ตัวก็ได้



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var myvar = Array.of(1,2,3,4,5,6,7,8,9,9);
console.log(myvar);
```

ผลลัพธ์

Array [1, 2, 3, 4, 5, 6, 7, 8, 9, 9]

Pop

ดึงข้อมูลตัวสุดท้ายออกมาจากอาเรย์และลบในคราวเดียวกัน

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H", "G"];
var myvar = arr.pop();
console.log(myvar);
```

ผลลัพธ์

G

Push

เพิ่มค่าลงไปในอาเรย์โดยเพิ่มจากท้ายสุด



ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];  
arr.push("I");  
console.log(arr);
```

ผลลัพธ์

```
Array [ "A", "B", "C", "D", "E", "F", "G", "H", "I" ]
```

Reverse

จัดเรียงลำดับในอาเรย์ใหม่โดยเรียงจากด้านหลังขึ้นมา

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];  
arr.reverse();  
console.log(arr);
```

ผลลัพธ์

```
Array [ "H", "G", "F", "E", "D", "C", "B", "A" ]
```

Shift

ดึงข้อมูลในลำดับแรกออกจากอาเรย์และใส่เข้าไปในตัวแปร



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];
arr.shift();
console.log(arr);
```

ผลลัพธ์

```
Array [ "B", "C", "D", "E", "F", "G", "H" ]
```

Slice

ดึงข้อมูลออกจากอาเรย์โดยกำหนดจุดเริ่มต้น และสิ้นสุด

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];
myvar = arr.slice(1,5);
console.log(myvar);
```

ผลลัพธ์

```
Array [ "B", "C", "D", "E" ]
```

Some

ใช้ตรวจสอบว่าสมาชิกของอาเรย์ผ่านการทดสอบจากฟังก์ชันอื่น ๆ มาแล้ว



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ตัวอย่างที่ 2

```
var numbers = [1, 2, 5, 11];
function chk(num) {
    return num >= 10;
}
myvar = numbers.some(chk);
console.log(myvar);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var numbers = [1, 2, 5, 11];
function chk(num) {
    return num >= 13;
}
myvar = numbers.some(chk);
console.log(myvar);
```

ผลลัพธ์

false

Sort

เรียงลำดับสมาชิกในอาเรย์โดยอ้างอิงจากตัวเลขและตัวอักษร



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var numbers = [1, 12, 58, 11];
myvar = numbers.sort();
console.log(myvar);
```

ผลลัพธ์

```
Array [ 1, 11, 12, 58 ]
```

ตัวอย่างที่ 2

```
var arr = ["G", "B", "D", "A", "I", "C"];
myvar = arr.sort();
console.log(myvar);
```

ผลลัพธ์

```
Array [ "A", "B", "C", "D", "G", "I" ]
```

Splice

เพิ่มข้อมูลลงในอาเรย์ตามตำแหน่งที่กำหนดโดยกำหนดได้ว่าจะเพิ่มจากตำแหน่งไหน และจะลบออกกี่ตัว



ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];
arr.splice(2, 3, "X", "Y");
console.log(arr);
```

ผลลัพธ์

Array ["A", "B", "X", "Y", "F", "G", "H"]

toString

แปลงอาเรย์ให้เป็นตัวอักษร

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];
myvar = arr.toString();
console.log(myvar);
```

ผลลัพธ์

A,B,C,D,E,F,G,H

Unshift

เพิ่มสมาชิกเข้าไปด้านหน้าของอาเรย์



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D", "E", "F", "G", "H"];
arr.unshift("x", "y");
console.log(arr);
```

ผลลัพธ์

```
Array [ "X", "Y", "A", "B", "C", "D", "E", "F", "G", "H" ]
```

Values

ดึงข้อมูลในอาเรย์ออกมาสร้างเป็นอาเรย์ใหม่ โดยไม่ต้องคีย์

ตัวอย่างที่ 1

```
var arr = ["A", "B", "C", "D"];
var myvar = arr.values();
console.log(myvar.next().value);
console.log(myvar.next().value);
console.log(myvar.next().value);
console.log(myvar.next().value);
```



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ผลลัพธ์

A

B

C

D

Date

โดยปกติแล้ว การเรียกใช้ฟังก์ชันเกี่ยวกับวันเวลา จะถูกเซตให้เรียกวันเวลาปัจจุบันออกมาเสมอๆ แต่ว่าก็สามารถกำหนดให้ดึงค่าออกมาจากเวลาที่ต้องการได้เหมือนกัน โดยการกำหนดให้อ็อบเจกต์วันเวลาเป็นเวลาที่ต้องการให้อยู่ในรูปแบบที่ต้องการ เช่น

```
Date("July 21, 1983 01:15:00");
```

getDate

ตัวอย่างที่ 1

ดึงค่าวันออกมาจากอ็อบเจกต์ของวันเวลา

```
var d = new Date();
var myvar = d.getDate();
console.log(myvar);
```



RUK-COM
Hosting

ผลลัพธ์

แสดงผลเป็นวันที่ปัจจุบันคือวันที่ 4 ตุลาคม 2559

4

getDay

ดึงลำดับของวันในสัปดาห์จาก อ็อบเจกต์วันเวลา โดยนับวันอาทิตย์เป็นวันแรก

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getDay();
console.log(myvar);
```

ผลลัพธ์

แสดงผลเป็นวันอังคาร โดย ที่ปัจจุบันคือวันอังคาร ที่ 4 ตุลาคม 2559

2

getFullYear

ดึงค่าเลข ค.ศ. 4 หลัก ออกมาจากอ็อบเจกต์วันเวลา

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getFullYear();
console.log(myvar);
```

ผลลัพธ์

แสดงผลเลขปีปัจจุบันคือ ค.ศ. 2016



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

getHours

ดึงค่าชั่วโมงออกมาจากอ็อบเจกต์วันเวลา

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.getHours();  
console.log(myvar);
```

ผลลัพธ์

แสดงเวลาปัจจุบันคือ 18.06 นาฬิกา

18

getMilliseconds

ดึงค่าเวลาที่ไมโครวิ ออกมาจากอ็อบเจกต์วันเวลา โดยจะนับใหม่ทุก ๆ 1วินาที ซึ่งค่าที่ได้จะมีค่าตั้งแต่ 0 - 1000

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.getMilliseconds();  
console.log(myvar);
```

ผลลัพธ์

495



RUK-COM
Hosting

getMinutes

ดึงค่าเวลาส่วนของนาฬิกา ออกมาจากอ็อบเจกต์วันเวลา

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getMinutes();
console.log(myvar);
```

ผลลัพธ์

แสดงผลเวลาปัจจุบัน 18.18 นาฬิกา

18

getMonth

ดึงค่าเดือนออกมาจากอ็อบเจกต์วันเวลา

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getMonth();
console.log(myvar);
```

ผลลัพธ์

แสดงหมายเลขประจำเดือนปัจจุบันที่นับจากเดือนมกราคม

9



getSeconds

ดึงค่าวินาทีออกมาจากอ็อบเจกต์วันเวลา

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getSeconds();
console.log(myvar);
```

ผลลัพธ์

แสดงเลขวินาทีปัจจุบัน เวลา 18.18 :34

34

getTime

ดึงค่ามิลลิวินาที ออกมาจากอ็อบเจกต์วันเวลาซึ่งฟังก์ชันนี้เริ่มนับเวลาตั้งแต่ 1970/01/01 00.00

นาฬิกา แล้วนับมาเรื่อยๆ จนถึงปัจจุบัน ซึ่งนับเป็นวินาที

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getTime();
console.log(myvar);
```

ผลลัพธ์

1475580032155



RUK-COM
Hosting

getUTCDate

ดึงค่าวันที่ ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getUTCDate();
console.log(myvar);
```

ผลลัพธ์

แสดงวันที่ปัจจุบันของเวลามาตรฐานสากล ซึ่งเป็นวันที่ 4 ตุลาคม 2016

4

getUTCDay

ดึงค่าลำดับของวันในสัปดาห์ ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล และเริ่มนับจากวันอาทิตย์

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getUTCDay();
console.log(myvar);
```

ผลลัพธ์

แสดงเลขวันประจำสัปดาห์ซึ่งในปัจจุบันวันนี้ตามเวลาสากลเป็นวันอังคาร

2



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

getUTCFullYear

ดึงค่าปี ค.ศ. ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getUTCFullYear();
console.log(myvar);
```

ผลลัพธ์

แสดง ค.ศ. ปัจจุบันของเวลามาตรฐานสากล ซึ่งเป็นวันที่ 4 ตุลาคม 2016

2016

getUTCHours

ดึงค่าชั่วโมง ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getUTCHours();
console.log(myvar);
```

ผลลัพธ์

แสดงเวลา ชั่วโมงปัจจุบัน ของเวลามาตรฐานสากล

11



RUK-COM
Hosting

getUTCMilliseconds

ดึงค่ามิลลิวินาที ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.getUTCMilliseconds();  
console.log(myvar);
```

ผลลัพธ์

364

getUTCMinutes

ดึงค่านาที ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.getUTCMinutes();  
console.log(myvar);
```

ผลลัพธ์

24



getUTCMonth

ดึงค่าเดือน ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getUTCMonth();
console.log(myvar);
```

ผลลัพธ์

แสดงเลขประจำเดือน ของเดือนปัจจุบันตามเวลามาตรฐานสากล ซึ่งเป็นวันที่ 4 ตุลาคม 2016

9

getUTCSeconds

ดึงค่าวินาที ออกมาจากอ็อบเจกต์วันเวลาโดยอ้างอิงจากเวลามาตรฐานสากล

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.getUTCSeconds();
console.log(myvar);
```

ผลลัพธ์

แสดงค่าเลขนาที่ปัจจุบันที่ 18.30.57

57



RUK-COM
Hosting

Now

ดึงค่ามิลินาทีโดยเริ่มนับจาก วันที่ 1 มกราคม 1970

ตัวอย่างที่ 1

```
var myvar = Date.now();
console.log(myvar);
```

ผลลัพธ์

1475580626650

setDate

ใช้กำหนดวันในอ็อบเจกต์ Date

ตัวอย่างที่ 1

```
var d = new Date();
d.setDate(25);
console.log(d);
```

ผลลัพธ์

Date 2016-10-25T11:32:45.934Z



setFullYear

ใช้กำหนดปีในอ็อบเจกต์ Date

ตัวอย่างที่ 1

```
var d = new Date();
d.setFullYear(2111)
console.log(d);
```

ผลลัพธ์

Date 2111-10-04T11:34:02.101Z

setHours

ใช้กำหนดชั่วโมงในอ็อบเจกต์ Date

ตัวอย่างที่ 1

```
var d = new Date();
d.setHours(12)
console.log(d);
```

ผลลัพธ์

เนื่องจากว่าเวลาที่ตัวเซิร์ฟเวอร์ในการรันโปรแกรมทดสอบนั้นเป็นเวลาของประเทศไทย จึงทำให้ค่าที่ได้จะต้องลบกับเวลาตามเขตประเทศไทยอีก ซึ่งก็คือเวลา 05 นาฬิกา (12 - 07)

Date 2016-10-04T05:37:20.093Z



RUK-COM
Hosting

setMilliseconds

ใช้กำหนดมิลลิวินาทีในอ็อบเจกต์ Date ซึ่งจะมีค่าตั้งแต่ 0 - 1000

ตัวอย่างที่ 1

```
var d = new Date();
d.setMilliseconds(555);
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T11:42:39.555Z

setMinutes

ใช้กำหนดนาฬิกาในอ็อบเจกต์ Date

ตัวอย่างที่ 1

```
var d = new Date();
d.setMinutes(22);
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T11:22:43.159Z



setMonth

ใช้กำหนดเดือนในอ็อบเจกต์ Date

ตัวอย่างที่ 1

```
var d = new Date();  
d.setMonth(11);  
console.log(d);
```

ผลลัพธ์

Date 2016-12-04T11:44:36.463Z

setSeconds

ใช้กำหนดวินาทีในอ็อบเจกต์ Date

ตัวอย่างที่ 1

```
var d = new Date();  
d.setSeconds(33);  
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T11:50:33.554Z



RUK-COM
Hosting

setUTCDate

ใช้กำหนดวันในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();
d.setUTCDate(20);
console.log(d);
```

ผลลัพธ์

Date 2016-10-20T11:51:50.552Z

setUTCFullYear

ใช้กำหนดปีในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();
d.setUTCFullYear(2222);
console.log(d);
```

ผลลัพธ์

Date 2222-10-04T11:53:00.199Z



setUTCHours

ใช้กำหนดชั่วโมงในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();  
d.setUTCHours(00);  
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T00:53:46.768Z

setUTCMilliseconds

ใช้กำหนดนาฬิกาในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();  
d.setUTCMilliseconds(0);  
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T11:54:30.000Z



RUK-COM
Hosting

setUTCMinutes

ใช้กำหนดนาทีในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();
d.setUTCMinutes(55);
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T11:55:13.896Z

setUTCMonth

ใช้กำหนดเดือนในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();
d.setUTCMonth(5);
console.log(d);
```

ผลลัพธ์

Date 2016-06-04T11:55:55.510Z



setUTCSeconds

ใช้กำหนดวินาทีในอ็อบเจกต์ Date ตามมาตรฐาน UTC

ตัวอย่างที่ 1

```
var d = new Date();
d.setUTCSeconds(01);
console.log(d);
```

ผลลัพธ์

Date 2016-10-04T11:56:01.824Z

toString

แปลงอ็อบเจกต์ Date ให้เป็นตัวอักษร

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.toString();
console.log(myvar);
```

ผลลัพธ์

Tue Oct 04 2016 18:58:21 GMT+0700 (SE Asia Standard Time)



RUK-COM
Hosting

toISOString

แปลงอ็อบเจกต์ Date ให้เป็นตัวอักษรในรูปแบบของ UTC

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.toISOString();  
console.log(myvar);
```

ผลลัพธ์

2016-10-04T11:59:16.221Z

toJSON

แปลงอ็อบเจกต์ Date ให้เป็นตัวอักษรในรูปแบบของ json

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.toJSON();  
console.log(myvar);
```

ผลลัพธ์

2016-10-04T12:00:56.430Z



toLocaleDateString

กำหนดรูปแบบของวันให้เป็นตามรูปแบบที่ประเทศนั้นใช้

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.toLocaleDateString();  
console.log(myvar);
```

ผลลัพธ์

2016-10-04T12:01:37.729Z

toLocaleString

กำหนดรูปแบบของวันเวลาให้เป็นตามรูปแบบที่ประเทศนั้นใช้

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.toLocaleString();  
console.log(myvar);
```

ผลลัพธ์

10/4/2016, 7:02:38 PM



RUK-COM
Hosting

toLocaleTimeString

กำหนดรูปแบบของเวลาให้เป็นตามรูปแบบที่ประเทศนั้นใช้

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.toLocaleTimeString();
console.log(myvar);
```

ผลลัพธ์

7:03:17 PM

toString

แปลงอ็อบเจกต์ Date ให้เป็นประโยค

ตัวอย่างที่ 1

```
var d = new Date();
var myvar = d.toString();
console.log(myvar);
```

ผลลัพธ์

Tue Oct 04 2016 19:04:14 GMT+0700 (SE Asia Standard Time)



toString

แปลงอ็อบเจกต์ Date ให้เป็นประโยค แต่เลือกแสดงเฉพาะค่าเวลา

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.toString();  
console.log(myvar);
```

ผลลัพธ์

Tue Oct 04 2016 19:05:42 GMT+0700 (SE Asia Standard Time)

toUTCString

แปลงอ็อบเจกต์ของ Date ให้อยู่ในรูปแบบของ UTC

ตัวอย่างที่ 1

```
var d = new Date();  
var myvar = d.toUTCString();  
console.log(myvar);
```

ผลลัพธ์

Tue, 04 Oct 2016 12:06:36 GMT



RUK-COM
Hosting

Errors

เป็นออบเจกต์ที่ใช้จัดการข้อผิดพลาด

Message

ใช้ตั้งค่าข้อความแสดงข้อผิดพลาด

ตัวอย่างที่ 1

```
try {
    allllert("ok not error");
}
catch(err) {
    console.log('you write alert wrong');
}
```

ผลลัพธ์

you write alert wrong

Name

ตั้งชื่อให้ error

ตัวอย่างที่ 1

```
var name_err = new Error('my name is Error');
name_err.name = 'allllllllert';
throw name_err;
```

ผลลัพธ์

Error: my name is Error



toString

แปลงอ็อบเจกต์ให้กลายเป็นข้อความ

ตัวอย่างที่ 1

```
var myobj = new Error('error error xxx');
console.log(myobj.toString());
```

ผลลัพธ์

Error: error error xxx

Global Objects

decodeURI

ถอดรหัสค่าที่ถูกเข้ารหัสโดย encodeURI

ตัวอย่างที่ 1

```
var tt = "A%20B%20C%20D%201%202%203";
var myvar = decodeURI(tt);
console.log(myvar);
```

ผลลัพธ์

A B C D 1 2 3



RUK-COM
Hosting

encodeURIComponent

ใช้เข้ารหัสค่า URI เพื่อป้องกันในกรณีมีการส่งอักขระพิเศษ

ตัวอย่างที่ 1

```
var tt = "A B C D 1 2 3";  
var myvar = encodeURIComponent(tt);  
console.log(myvar);
```

ผลลัพธ์

A%20B%20C%20D%201%202%203

isFinite

ใช้ตรวจสอบว่าตัวเลขเป็นค่าที่มีที่สิ้นสุดไหม

ตัวอย่างที่ 1

```
var tt = "97567";  
var myvar = isFinite(tt);  
console.log(myvar);
```

ผลลัพธ์

true



ตัวอย่างที่ 2

```
var tt = "ABCD";  
var myvar = isFinite(tt);  
console.log(myvar);
```

ผลลัพธ์

false

isNaN

ใช้ตรวจสอบว่าตัวเลขเป็นค่า NaN หรือไม่

ตัวอย่างที่ 1

```
var tt = "ABCD";  
var myvar = isNaN(tt);  
console.log(myvar);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var tt = "123";  
var myvar = isNaN(tt);  
console.log(myvar);
```



RUK-COM
Hosting

ผลลัพธ์

false

NaN

ย่อมาจาก Not A Number แปลว่าค่าที่ไม่ใช่ตัวเลข

ตัวอย่างที่ 1

```
var a = 9;  
var b = 2;  
var c = a - b;  
console.log(c);
```

ผลลัพธ์

7

ตัวอย่างที่ 2

```
var a = 9;  
var b = "ABCD";  
var c = a - b;  
console.log(c);
```

ผลลัพธ์

NaN



Null

ใช้เพื่อกำหนดให้ตัวแปรหรือตรวจสอบว่าตัวแปรนั้นเป็นค่าว่าง หรือไม่

ตัวอย่างที่ 1

```
var tt;
if(tt == null){
    console.log('empty');
}else{
    console.log(' exist');
```

ผลลัพธ์

empty

parseFloat

แปลงตัวอักษรทุกอย่างให้กลายเป็นเลขที่มีจุดทศนิยม

ตัวอย่างที่ 1

```
var tt = "ABCD123";
myvar = parseFloat(tt);
console.log(myvar);
```



RUK-COM
Hosting

ผลลัพธ์

NaN

ตัวอย่างที่ 2

```
var tt = "84766";  
myvar = parseFloat(tt);  
console.log(myvar);
```

ผลลัพธ์

84766

parseInt

แปลงตัวเลขระบบฐานเลข

ตัวอย่างที่ 1

```
var myvar = parseInt(" 0xF", 16);  
console.log(myvar);
```

ผลลัพธ์

15



Undefined

ตรวจสอบว่าตัวแปรถูกกำหนดค่าหรือพร้อมใช้งานหรือเปล่า

ตัวอย่างที่ 1

```
var x;  
if(x === undefined){  
    console.log('var is undefined');  
}else{  
    console.log('exist');  
}
```

ผลลัพธ์

var is undefined

Escape

แปลงอักขระพิเศษให้เป็นเลขฐานสิบหก

ตัวอย่างที่ 1

```
var tt = "ABC 123 / * ? .";  
var myvar = escape(tt);  
console.log(myvar);
```



RUK-COM
Hosting

ผลลัพธ์

ABC%20123%20/%20*%20%3F%20.

unescape

ถอดรหัสค่าที่ถูกเข้ารหัสโดย escape

ตัวอย่างที่ 1

```
var tt = "ABC%20123%20/%20*%20%3F%20.";
var myvar = unescape(tt);
console.log(myvar);
```

ผลลัพธ์

ABC 123 / * ? .



Intl

เป็นออบเจกต์ที่ใช้จัดการรูปแบบของข้อมูลให้สอดคล้องกับภาษาและท้องถิ่นของประเทศนั้นๆ เช่น ภาษา, ค่าเงิน, วันเวลา, ตัวเลข

Collator

ใช้ในการเปรียบเทียบระบบภาษาซึ่งสามารถเปรียบเทียบได้ทั้งภาษาเดียวกัน หรือภาษาคนละภาษาก็ได้ เช่น **a** กับ **b** จะได้ -1 เนื่องจาก ตามลำดับแล้ว **a** มาก่อน **b** แต่ถ้าเปรียบเทียบ **b** กับ **a** จะได้ค่า 1 เนื่องจาก **b** มาทีหลัง **a** แต่ว่าถ้าเราเปรียบเทียบ **a** กับ **a** จะได้ 0 เนื่องจากว่าเป็นค่าเดียวกัน

ตัวอย่างที่ 1

```
var col = new Intl.Collator().compare('a', 'b');  
console.log(col);
```

ผลลัพธ์

-1

Compare

ใช้เปรียบเทียบระบบภาษาจะเป็นภาษาเดียวกัน หรือคนละภาษาก็ได้



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var col = new Intl.Collator('en');
var chk = col.compare('a', 'A');
console.log(chk);
```

ผลลัพธ์

เนื่องจากว่า **a** มาก่อน **A** จึงได้ค่า -1

-1

resolvedOptions

ดึงตัวแปรที่ใช้ตั้งค่าต่างๆ ของอีอบเจกต์ collator มาแสดง

ตัวอย่างที่ 1

```
var col = new Intl.Collator('en', { sensitivity: 'base' });
var chk = col.resolvedOptions();
console.log(chk);
```

ผลลัพธ์

```
Object { locale: "en", usage: "sort",
sensitivity: "base", ignorePunctuation:
false, collation: "default", numeric: false
}
```



DateTimeFormat

Format

จัดรูปแบบวันเวลาให้อยู่ในรูปแบบ ภาษาที่ประเทศนั้นใช้ จากตัวอย่างจะแปลงให้เป็นวันเวลาของประเทศสหรัฐอเมริกา ซึ่งสังเกตได้จาก en-US

ตัวอย่างที่ 1

```
var date = new Date(Date.UTC(2016, 10, 6));
console.log(new Intl.DateTimeFormat('en-US').format(date));
```

ผลลัพธ์

11/6/2016

supportedLocalesOf

ดึงค่าภาษาที่อ็อบเจกต์สนับสนุนการใช้งาน ซึ่งในตัวอย่างจะใช้ภาษาเยอรมัน

ตัวอย่างที่ 1

```
var loc = ['ban', 'id-u-co-pinyin', 'de-ID'];
var opt = { localeMatcher: 'lookup' };
var test = Intl.DateTimeFormat.supportedLocalesOf(loc, opt);
console.log(test);
```

ผลลัพธ์

Array ["id-u-co-pinyin", "de-ID"]



RUK-COM
Hosting

NumberFormat

Format

จัดรูปแบบตัวเลขให้อยู่ในรูปแบบ ภาษาที่ประเทศนั้นใช้

ตัวอย่างที่ 1

```
var num = 1123456;  
var test = new Intl.NumberFormat('en-IN').format(num);  
  
console.log(test);
```

ผลลัพธ์

11,23,456

JSON

Parse

แปลงสตริงให้กลายเป็นสคริปต์ที่ใช้ในภาษา json

ตัวอย่างที่ 1

```
var myvar = JSON.parse('[1, 8, "A", "B", "false"]');  
console.log(myvar);
```

ผลลัพธ์

Array [1, 8, "A", "B", "false"]



Stringify

แปลงอ็อบเจกต์ต่างๆ ใน javascript ให้เป็น json

ตัวอย่างที่ 1

```
var obj = {A:"ant", b:"birt", c:"cat"};
JSON.stringify(obj);
console.log(obj);
```

ผลลัพธ์

```
Object { A: "ant", b: "birt", c: "cat" }
```

Map

เป็นออบเจกต์ที่เก็บข้อมูลเป็น key และ value

Clear

ล้างข้อมูลทั้งหมดออกจากอ็อบเจกต์ของ Set

ตัวอย่างที่ 1

```
var map = new Map();
map.set("a", "b");
map.size;
map.clear();
console.log(map);
```



ผลลัพธ์

`Map { }`

Delete

ลบข้อมูลที่กำหนดออกจากอ็อบเจกต์ ซึ่งถ้าสำเร็จคืนค่า **true** ไม่สำเร็จจะคืนค่า **false**

ตัวอย่างที่ 1

```
var map = new Map();
var set = map.set("A", "B");
var del = map.delete("A");
console.log(del);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var map = new Map();
var set = map.set("A", "B");
var del = map.delete("C");
console.log(del);
```

ผลลัพธ์

false



Entries

แปลงอ็อบเจกต์ให้เป็น Iterator

ตัวอย่างที่ 1

```
var map = new Map();
map.set("A", "x");
map.set("B", "y");
map.set("C", "z");
var test = map.entries();
console.log(test.next().value);
console.log(test.next().value);
console.log(test.next().value);
```

ผลลัพธ์

Array ["A", "x"]

Array ["B", "y"]

Array ["C", "z"]

forEach

ช่วยให้จัดการสมาชิกอาเรย์โดยการวนลูปได้ง่ายขึ้น

ตัวอย่างที่ 1

```
function test(ind, valu, map) {
    console.log(ind + '=>' + valu);
}
new Map([["A", 'x'], ["B", 'y'], ["C", 'z']]).forEach(test);
```



RUK-COM
Hosting

ผลลัพธ์

x=>A

y=>B

z=>C

Get

ดึงข้อมูลของสมาชิกในอ็อบเจกต์ Map

ตัวอย่างที่ 1

```
var map = new Map();
map.set("A", "x");
map.set("B", "y");
var test = map.get("A");
console.log(test);
```

ผลลัพธ์

x

Has

ตรวจสอบว่ามีสมาชิกที่กำหนดไหม

ตัวอย่างที่ 1

```
var map = new Map();
map.set("A", "x");
map.set("B", "y");
var test = map.get("A");
console.log(test);
```



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var map = new Map();  
map.set("A", "x");  
map.set("B", "y");  
var test = map.has("C");  
console.log(test);
```

ผลลัพธ์

false

Keys

เข้าถึงค่าคีย์ในอ็อบเจกต์



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var map = new Map();
map.set("A", "x");
map.set("B", "y");
map.set("C", "z");
var test = map.keys();
console.log(test.next().value);
console.log(test.next().value);
console.log(test.next().value);
```

ผลลัพธ์

A

B

C

Set

เพิ่มสมาชิกลงในอ็อบเจกต์

ตัวอย่างที่ 1

```
var map = new Map();
map.set("A", "x");
map.set("B", "y");
map.set("C", "z");
console.log(map);
```

ผลลัพธ์

Map { A: "x", B: "y", C: "z" }



Size

ใช้หาขนาดของอ็อบเจกต์ Map

ตัวอย่างที่ 1

```
var map = new Map();  
map.set("A", "1");  
map.set("B", "2");  
map.set("C", "3");  
map.set("D", "4");  
map.set("E", "5");  
map.set("F", "6");  
console.log(map.size);
```

ผลลัพธ์

6

Values

ใช้ดึงค่าในตำแหน่งที่กำหนด



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var map = new Map();  
map.set("A", "1");  
map.set("B", "2");  
map.set("C", "3");  
var test = map.values();  
console.log(test.next().value);  
console.log(test.next().value);  
console.log(test.next().value);
```

ผลลัพธ์

1

2

3

WeakMap

คือออบเจกต์ Map ที่จะถูกลบทิ้งที่ที่ถูกเซตค่าให้เป็น null



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

Weakest

ใช้สำหรับเพิ่มค่าลงในอ็อบเจกต์

ตัวอย่างที่ 1

```
var wm = new WeakMap();  
var obj = {};  
var add = wm.set(obj, "ABCD");  
console.log(add);
```

ผลลัพธ์

WeakMap { Object: "ABCD" }

Weakdelete

ลบข้อมูลออกจาก WeakMap แบบรายตัว

ตัวอย่างที่ 1

```
var wm = new WeakMap();  
wm.set(window, "ABCD");  
wm.delete(window);  
var chk = wm.has(window);  
console.log(chk);
```

ผลลัพธ์

false



RUK-COM
Hosting

ตัวอย่างที่ 2

```
var wm = new WeakMap();
wm.set(window, "ABCD");
var chk = wm.has(window);
console.log(chk);
```

ผลลัพธ์

true

Weakget

ใช้ดึงค่าออกจาก Weakmap

ตัวอย่างที่ 1

```
var wm = new WeakMap();
wm.set(window, "ABCD");
var chk = wm.get(window);
console.log(chk);
```

ผลลัพธ์

ABCD

Weakhas

ใช้ตรวจสอบว่ามีค่าหรือไม่



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ตัวอย่างที่ 1

```
var wm = new WeakMap();  
wm.set(window, "ABCD");  
var chk = wm.has(window);  
console.log(chk);
```

ผลลัพธ์

true

Math

ในการเรียกใช้ เมทอดของ จะต้องประกาศ **Math** ก่อนทุกครั้ง

Abs

ใช้คำนวณหาค่าสัมบูรณ์

ตัวอย่างที่ 1

```
var a = 9;  
var b = 20;  
var c = Math.abs(a - b);  
console.log(c);
```



RUK-COM
Hosting

ผลลัพธ์

11

Acos

หาค่าอาร์คโคไซน์

ตัวอย่างที่ 1

```
var myvar = Math.acos(0.1)
console.log(myvar);
```

ผลลัพธ์

1.4706289056333368

Acosh

ใช้หาค่า อินเวอร์ส ไฮเพอโบลิก โคไซน์

ตัวอย่างที่ 1

```
var myvar = Math.acosh(5)
console.log(myvar);
```

ผลลัพธ์

2.2924316695611777

Asin

หาค่าอาร์คไซน์



ตัวอย่างที่ 1

```
var myvar = Math.asin(0.8)  
console.log(myvar);
```

ผลลัพธ์

0.9272952180016123

Asinh

ใช้หาค่าไฮเพอโบลิก อาร์คไซน์

ตัวอย่างที่ 1

```
var myvar = Math.asinh(0.9);  
console.log(myvar);
```

ผลลัพธ์

0.8088669356527826

Atan

หาค่าอาร์คแทน



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var myvar = Math.atan(0.9);  
console.log(myvar);
```

ผลลัพธ์

0.7328151017865066

Atanh

ใช้หาค่าไฮเพอโบลิก อาร์คแทน

ตัวอย่างที่ 1

```
var myvar = Math.atanh(0.9);  
console.log(myvar);
```

ผลลัพธ์

1.4722194895832204

Cbrt

ใช้ถอดรากสำหรับค่าที่ถูกยกกำลัง 3



ตัวอย่างที่ 1

```
var myvar = Math.cbrt(8);  
console.log(myvar);
```

ผลลัพธ์

2

Ceil

ใช้สำหรับปัดเศษทศนิยมขึ้น

ตัวอย่างที่ 1

```
var myvar = Math.ceil(0.99999998);  
console.log(myvar);
```

ผลลัพธ์

1

Cos

หาค่าโคไซน์



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var myvar = Math.cos(45);
console.log(myvar);
```

ผลลัพธ์

0.5253219888177297

Cosh

ใช้หาค่าไฮเพอโบลิก อาร์คโคไซน์

ตัวอย่างที่ 1

```
var myvar = Math.cosh(1);
console.log(myvar);
```

ผลลัพธ์

1.543080634815244

E

หาค่า ลอการิทึมฐาน e



ตัวอย่างที่ 1

```
var myvar = Math.E;  
console.log(myvar);
```

ผลลัพธ์

2.718281828459045

Floor

ใช้ปัดเศษทศนิยมขึ้นและถ้าปัดขึ้นไม่ได้ก็ลบทศนิยมออกไป

ตัวอย่างที่ 1

```
var myvar = Math.floor(99.994);  
console.log(myvar);
```

ผลลัพธ์

99

log10

ค่า ลอการิทึมฐาน 10



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var myvar = Math.log10(100);
console.log(myvar);
```

ผลลัพธ์

2

log2

หาค่า ลอการิทึมฐาน 2

ตัวอย่างที่ 1

```
var myvar = Math.log2(32);
console.log(myvar);
```

ผลลัพธ์

5

Max

หาค่าที่มากที่สุด



ตัวอย่างที่ 1

```
var myvar = Math.max(1,2,3,4,5,6,7,8,9,20);  
console.log(myvar);
```

ผลลัพธ์

20

Min

หาค่าที่น้อยที่สุด

ตัวอย่างที่ 1

```
var myvar = Math.min(1,2,3,4,5,6,7,8,9,20);  
console.log(myvar);
```

ผลลัพธ์

1

PI

ดึงค่าพายออกมาใช้งาน

ตัวอย่างที่ 1

```
var myvar = Math.PI;  
console.log(myvar);
```

ผลลัพธ์

3.141592653589793



RUK-COM
Hosting

Pow

ใช้สำหรับยกกำลัง

ตัวอย่างที่ 1

```
var myvar = Math.pow(2,3);  
console.log(myvar);
```

ผลลัพธ์

8

Random

ใช้สำหรับสุ่มเลขออกมา

ตัวอย่างที่ 1

```
var myvar = Math.random();  
console.log(myvar);
```

ผลลัพธ์

0.7531884365465699

Round

ลบจุดทศนิยมทิ้งไป



ตัวอย่างที่ 1

```
var myvar = Math.round(25.98);  
console.log(myvar);
```

ผลลัพธ์

26

Sin

ใช้หาค่าไซน์

ตัวอย่างที่ 1

```
var myvar = Math.sin(45);  
console.log(myvar);
```

ผลลัพธ์

0.8509035245341184

Sqrt

ใช้หาค่าสแควรท



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var myvar = Math.sqrt(9);  
console.log(myvar);
```

ผลลัพธ์

3

Tan

ใช้สำหรับหาค่าแทนเจนต์

ตัวอย่างที่ 1

```
var myvar = Math.tan(30);  
console.log(myvar);
```

ผลลัพธ์

-6.405331196646276

Tanh

ใช้สำหรับหาค่าไฮเพอโบลิค แทนเจนต์



ตัวอย่างที่ 1

```
var myvar = Math.tanh(1);  
console.log(myvar);
```

ผลลัพธ์

0.7615941559557649

Miscellaneous

Eval

เปรียบเทียบค่าหรือประโยคให้เป็นตัวแปร

ตัวอย่างที่ 1

```
var a = 3;  
var c = eval(a * 6);  
console.log(c);
```

ผลลัพธ์

18



RUK-COM
Hosting

Number

isFinite

ใช้ตรวจสอบว่าเป็นตัวเลขที่มีค่าสิ้นสุด

ตัวอย่างที่ 1

```
tt = 856.84656;  
var myvar = isFinite(tt);  
console.log(myvar);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
tt = '254Abc';  
var myvar = isFinite(tt);  
console.log(myvar);
```

ผลลัพธ์

false

isInteger

ใช้ตรวจสอบว่าเป็นตัวเลขหรือไม่



ตัวอย่างที่ 1

```
var tt = 12345;  
var myvar = Number.isInteger(tt);  
console.log(myvar);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var tt = "ddd12345";  
var myvar = Number.isInteger(tt);  
console.log(myvar);
```

ผลลัพธ์

false

isNaN

ใช้ตรวจสอบว่าเป็นค่า NaN หรือไม่

ตัวอย่างที่ 1

```
var myvar = Number.isNaN(NaN);  
console.log(myvar);
```



RUK-COM
Hosting

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var myvar = Number.isNaN(123);  
console.log(myvar);
```

ผลลัพธ์

false

parseFloat

ใช้จัดการตัวอักษรให้กลายเป็นจำนวนเต็มหรือจำนวนที่มีจุดทศนิยม

ตัวอย่างที่ 1

```
var tt = '0.99999999';  
var myvar = parseFloat(tt);  
console.log(myvar);
```

ผลลัพธ์

0.99999999



ตัวอย่างที่ 2

```
var tt = 'ABcd';  
var myvar = parseFloat(tt);  
console.log(myvar);
```

ผลลัพธ์

NaN

parseInt

ใช้จัดการตัวอักษรให้กลายเป็นจำนวนเต็ม

ตัวอย่างที่ 1

```
var tt = '123ABcd';  
var myvar = parseInt(tt);  
console.log(myvar);
```

ผลลัพธ์

123



RUK-COM
Hosting

toExponential

ใช้สำหรับหาค่าเอ็กโพเนนเชียล

ตัวอย่างที่ 1

```
var tt = 7789;  
var myvar = tt.toExponential();  
console.log(myvar);
```

ผลลัพธ์

7.789e+3

toFixed

ฟิกจำนวนเลขหลังจุดทศนิยม และปัดเศษ

ตัวอย่างที่ 1

```
var tt = 1.123456789;  
var myvar = tt.toFixed(5);  
console.log(myvar);
```

ผลลัพธ์

1.12346



toLocaleString

แปลงตัวเลขให้เป็นภาษาที่กำหนด

ตัวอย่างที่ 1

```
var tt = 789.123456789;
var myvar = tt.toLocaleString('th-TH', {
    style: 'currency', currency: 'THB'
});
console.log(myvar);
```

ผลลัพธ์

THB789.12

toString

ใช้แปลงค่าเลขเป็นตัวอักษร

ตัวอย่างที่ 1

```
var tt = 789.123456789;
var myvar = tt.toString();
console.log(myvar);
```

ผลลัพธ์

789.123456789

ตัวอย่างที่ 2



RUK-COM
Hosting


```
var myvar = (-0xff).toString(2);
console.log(myvar);
```

ผลลัพธ์

-11111111

Object

Assign

เพิ่มข้อมูลลงไปในอ็อบเจกต์

ตัวอย่างที่ 1

```
var obj1 = { a: 1234 };
var obj2 = Object.assign({}, obj1);
console.log(obj2);
```

ผลลัพธ์

Object { a: 1234 }

Create

สร้างอ็อบเจกต์ใหม่โดยจะไม่นรันโค้ดใน Constructor



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ตัวอย่างที่ 1

```
var Animal = {traits: {},};  
var lion = Object.create(Animal);  
lion.traits.legs = 4;  
var bird = Object.create(Animal);  
bird.traits.legs = 2;  
console.log(lion.traits.legs);
```

ผลลัพธ์

2

defineProperties

ใช้สำหรับกำหนด property ให้อ็อบเจกต์



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var obj = {};
Object.defineProperty(obj, {
    "property1": {
        value: "1234",
        writable: true
    },
    "property2": {
        value: "ABCD",
        writable: false
    }
});
console.log(obj);
```

ผลลัพธ์

```
property1: "1234"
property2: "ABCD"
```

Freeze

ป้องกันการเพิ่ม ลบ แก้ไข กับอ็อบเจกต์



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ตัวอย่างที่ 1

```
var obj = {};  
Object.freeze(obj);  
Object.defineProperty(obj, {  
    "property1": {  
        value: "1234",  
        writable: true  
    },  
    "property2": {  
        value: "ABCD",  
        writable: false  
    }  
});  
console.log(obj);
```

ผลลัพธ์

TypeError: can't define property
"property1": Object is not extensible

getOwnPropertyDescriptor

ดึงข้อมูลของ property



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var obj = {
  val: "1234",
  writable: true,
  enumerable: true,
  configurable: true
};
my_obj = Object.getOwnPropertyDescriptor(obj, 'val');
console.log(my_obj);
```

ผลลัพธ์

```
Object { value: "1234",
writable: true,
enumerable: true,
configurable: true }
```

hasOwnProperty

ใช้ตรวจสอบว่าในอ็อบเจกต์นั้นมี property อยู่หรือไม่



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ตัวอย่างที่ 1

```
o = new Object();  
o.prop = 'ABC1234';  
my_obj = o.hasOwnProperty('prop');  
console.log(my_obj);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
o = new Object();  
o.prop;  
my_obj = o.hasOwnProperty('prop');  
console.log(my_obj);
```

ผลลัพธ์

false

Is

ใช้ตรวจสอบค่าสองตัวว่าเหมือนกันไหม



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var test = Object.is('ABCD', 'ABCD');
console.log(test);
```

ผลลัพธ์

true**isExtensible**

ใช้ตรวจสอบว่าอ็อบเจกต์นั้นผ่านการใช้งานฟังก์ชัน preventExtension, seal, freeze หรือไม่

ตัวอย่างที่ 1

```
var obj = {ABCD: 'ABCD'};
var test = Object.isExtensible(obj);
console.log(test);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var obj = {ABCD: 'ABCD'};
var frozen = Object.freeze(obj);
var test = Object.isExtensible(obj);
console.log(test);
```



ผลลัพธ์

false

isFrozen

ใช้ตรวจสอบว่าอ็อบเจกต์นั้นสามารถเพิ่มเติมหรือแก้ไขค่าได้

ตัวอย่างที่ 1

```
var obj = {ABCD:'ABCD'};  
var chk = Object.isFrozen(obj);  
var test = Object.isExtensible(obj);  
console.log(test);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var obj = {ABCD:'ABCD'};  
var lock = Object.freeze(obj);  
var chk = Object.isFrozen(obj);  
var test = Object.isExtensible(obj);  
console.log(test);
```

ผลลัพธ์

false



RUK-COM
Hosting

isPrototypeOf

ใช้กำหนดอ็อบเจ็กต์ให้สืบทอดมาจากอ็อบเจ็กต์อื่น

ตัวอย่างที่ 1

```
var err = new Error("my name is error");  
var set = Object.getPrototypeOf(err);  
var test = set === Error.prototype;  
console.log(test);
```

ผลลัพธ์

true



isSealed

ใช้สำหรับตรวจสอบว่าผ่านการใช้งานฟังก์ชัน seal มาหรือไม่

ตัวอย่างที่ 1

```
var obj = {};  
var test = Object.isSealed(obj);  
console.log(test);
```

ผลลัพธ์

false

ตัวอย่างที่ 2

```
var obj = {};  
Object.seal(obj);  
var test = Object.isSealed(obj);  
console.log(test);
```

ผลลัพธ์

true



RUK-COM
Hosting

Keys

ใช้สำหรับดึงค่าคีย์ออกมาจากอ็อบเจกต์

ตัวอย่างที่ 1

```
var obj = {a:'x',b:'y',c:'z'};
var test = Object.keys(obj);
console.log(test);
```

ผลลัพธ์

Array ["a", "b", "c"]

preventExtensions

ใช้สำหรับป้องกันการเพิ่ม property เข้าไปในอ็อบเจกต์

ตัวอย่างที่ 1

```
var obj = {};
Object.preventExtensions(obj);
Object.defineProperty(obj, {
  "test": {
    a: 'x',
    b: 'y',
    c: 'z'
  }
});
console.log(obj);
```



ผลลัพธ์

► TypeError: can't define property "test":
Object is not extensible

Seal

ใช้ป้องกันการเพิ่มสมาชิกใหม่และไม่ให้มีการแก้ไขสมาชิกเดิม

ตัวอย่างที่ 1

```
var obj = {};
Object.seal(obj);
Object.defineProperties(obj, {
  "test": {
    a: 'x',
    b: 'y',
    c: 'z'
  },
  "test2": {
    D: '1',
    E: '2',
    F: '3'
  }
});
console.log(obj);
```

ผลลัพธ์

► TypeError: can't define property "test":
Object is not extensible



RUK-COM
Hosting

toString

แปลงค่าในอ็อบเจกต์ให้เป็นสตริงซึ่งจะคืนค่าเป็นชนิดของอ็อบเจกต์นั้นๆ

ตัวอย่างที่ 1

```
var obj = {};
Object.defineProperty(obj, {
  "test": {
    a: 'x',
    b: 'y',
    c: 'z'
  },
  "test2": {
    D: '1',
    E: '2',
    F: '3'
  }
});
var test = obj.toString();
console.log(test);
```

ผลลัพธ์

[object Object]



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

Promise

เป็นออบเจกต์ที่ใช้งานร่วมกับการเขียนแบบ **asynchronous** โดยออกแบบมาให้รับค่าที่ยังไม่เสร็จ แต่สัญญาว่าจะเสร็จในอนาคต

All

รับค่าจาก Promise object หลายๆ ค่า แล้วนำมาแสดงพร้อมกัน

ตัวอย่างที่ 1

```
var p1 = Promise.resolve(3);  
var p2 = 9999;  
var p3 = new Promise((resolve, reject) => {  
  setTimeout(resolve, 100, "xxxx");  
});  
  
Promise.all([p1, p2, p3]).then(values => {  
  console.log(values);  
});
```

ผลลัพธ์

`Array [3, 9999, "xxxx"]`



RUK-COM
Hosting

Catch

ใช้เก็บค่า reject ที่ promise ทำงานผิดพลาด

ตัวอย่างที่ 1

```
var p1 = new Promise(function(resolve, reject) {  
    throw 'my name is Error';  
});  
p1.catch(function(e) {  
    console.log(e);  
});
```

ผลลัพธ์

my name is Error



Race

สร้าง promise อันใหม่ แต่ให้มามีค่าเหมือนกับ promise อันเดิม

ตัวอย่างที่ 1

```
var p1 = new Promise(function(resolve, reject) {
  setTimeout(resolve, 0, 'old value');
});
var p2 = new Promise(function(resolve, reject) { });
var p3 = new Promise(function(resolve, reject) { });
var rac = Promise.race( [p1, p2, p3] );
rac.then(function(res) {
  console.log(res);
});
```

ผลลัพธ์

old value

Reject

สร้าง error สำหรับ Promise

ตัวอย่างที่ 1

```
var p = Promise.reject('my error');
p.catch(function(res) {
  console.log(res);
});
```



RUK-COM
Hosting

ผลลัพธ์

my error

Resolve

กำหนดการทำงานที่เสร็จสิ้น ให้ promise

ตัวอย่างที่ 1

```
var p = Promise.resolve('it is success');
p.then(function(result) {
  console.log(result);
});
```

ผลลัพธ์

it is success

then

ใช้สำหรับกำหนดงานให้ promise

ตัวอย่างที่ 1

```
function timeout(t) {
  return new Promise(function(resolve, reject) {
    setTimeout(resolve, t);
  });
}
timeout(1000).then(() => {
  console.log('success');
});
```



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ผลลัพธ์

success

RegExp

ในการใช้งาน RegExp นั้น ผู้ใช้งานควรจะเข้าใจเกี่ยวกับการสร้าง **pattern** เสียก่อนจึงจะสามารถใช้งานได้อย่างถูกต้อง

Match

ตรวจสอบประโยคว่าตรงกับ regex ที่กำหนดไว้หรือไม่

ตัวอย่างที่ 1

```
var tt = "I am programmer";  
var res = tt.match('mer');  
console.log(res);
```

ผลลัพธ์

Array ["mer"]



RUK-COM
Hosting

Replace

ใช้แปลงค่าที่ตรงกับ regex ที่กำหนด ให้เป็นค่าที่ต้องการ

ตัวอย่างที่ 1

```
var tt = "I am programmer";  
var res = tt.replace("am", "5555");  
console.log(res);
```

ผลลัพธ์

I 5555 programmer

Search

ใช้ค้นหาข้อความที่ตรงกับ regex หรือไม่ และจะคืนค่าเป็นตำแหน่ง

ตัวอย่างที่ 1

```
var tt = "I am programmer";  
var res = tt.search("p");  
console.log(res);
```

ผลลัพธ์

5



Split

แบ่งประโยคออกโดยใช้ regex

ตัวอย่างที่ 1

ใช้ วรรค ในการแบ่งประโยค

```
var tt = "I am programmer";  
var res = tt.split(" ");  
console.log(res);
```

ผลลัพธ์

```
Array [ "I", "am", "programmer" ]
```

lastIndex

ใช้ค้นหาคำหรือคำที่ต้องการโดยจะคืนค่าเป็นตำแหน่ง

ตัวอย่างที่ 1

```
var tt = "ABCD123";  
var res = tt.lastIndexOf('D');  
console.log(res);
```

ผลลัพธ์

```
3
```



RUK-COM
Hosting

Exec

ใช้ค้นหาที่ตรงกับ regex ตัวแรก

ตัวอย่างที่ 1

จากแพทเทอนั้นหมายถึง X ตามด้วยอะไรก็ได้แต่มีปิดท้ายด้วย Z

```
var re = /(x).+?(z)/;
var res = re.exec('a b c defg hi12345 xyz');
console.log(res);
```

ผลลัพธ์

`Array ["xyz", "x", "z"]`

Flags

ใช้สำหรับกำหนดค่าในการค้นหา regex เช่นกำหนด i หมายถึงว่าไม่สนใจตัวเล็กหรือใหญ่ดัง

ตัวอย่าง

ตัวอย่างที่ 1

```
var tt = 'ABCDEFGH0123';
var test = tt.match(/d/i);
console.log(test);
```

ผลลัพธ์

`Array ["D"]`



Global

ใช้ในการกำหนดให้ค้นหาว่า **flags** เป็นชนิด **global** หรือไม่

ตัวอย่างที่ 1

```
var reg = new RegExp('ABCDEFGH0123', 'g');  
console.log(reg.global);
```

ผลลัพธ์

true

ignoreCase

ใช้ในการกำหนดให้ค้นหาว่า **flags** ถูกกำหนดให้เป็นแบบ **Non case sensitive** หรือไม่

ตัวอย่างที่ 1

```
var reg = new RegExp('ABCDEFGH0123', 'i');  
console.log(reg.ignoreCase);
```

ผลลัพธ์

true



RUK-COM
Hosting

Multiline

ใช้ในการกำหนดให้ค้นหาว่า **flags** ถูกกำหนดให้เป็นแบบ ค้นหาจากจุดเริ่มต้นของประโยคหรือไม่(เหมือนกับการใช้ $^$ ใน regex)

ตัวอย่างที่ 1

```
var reg = new RegExp('ABCDEFGH0123', 'm');
console.log(reg.multiline);
```

ผลลัพธ์

true

Source

คืนค่ารูปแบบ pattern ของ RegExp

ตัวอย่างที่ 1

```
var reg = /I am pattern/i;
console.log(reg.source);
```

ผลลัพธ์

I am pattern



Test

ใช้ค้นหา **string** หรือค่าที่กำหนดว่ามีหรือไม่ ถ้ามีจะคืนค่า **true** ถ้าไม่มีจะคืนค่า **false**

ตัวอย่างที่ 1

```
var str = "hello world! I am programmer";  
var fin = /hello/.test(str);  
console.log(fin);
```

ผลลัพธ์

true

toString

แปลงออบเจ็คให้เป็นสตริงธรรมดา

ตัวอย่างที่ 1

```
var reg = new RegExp('ABCDEFGH0123', 'igm');  
var test = reg.toString();  
console.log(test);
```

ผลลัพธ์

/ABCDEFGH0123/gim



RUK-COM
Hosting

Unicode

ค้นหา **flags** ว่าถูกกำหนดให้เป็นแบบ **Unicode** หรือไม่

ตัวอย่างที่ 1

```
var regex = new RegExp('0123Abc hello word', 'u');
var test = regex.unicode;
console.log(test);
```

ผลลัพธ์

true

Set

เป็นออบเจกต์ที่ใช้แทนโครงสร้างของข้อมูล ซึ่งสมาชิกภายในจะมีค่าที่ไม่ซ้ำกัน

Add

เพิ่มข้อมูลลงในออบเจกต์ของ Set

ตัวอย่างที่ 1

```
var myvar = new Set();
myvar.add(5);
myvar.add(99);
myvar.add("hello");
console.log(myvar);
```

ผลลัพธ์

Set [5, 99, "hello"]



Clear

ลบสมาชิกของ Set ออกทั้งหมด

ตัวอย่างที่ 1

```
var myvar = new Set();  
myvar.add(5);  
myvar.add(99);  
myvar.add("hello");  
myvar.clear();  
console.log(myvar);
```

ผลลัพธ์

Set []

Delete

ลบสมาชิกของ Set ออกตามที่กำหนด

ตัวอย่างที่ 1

```
var myvar = new Set();  
myvar.add(5);  
myvar.add(99);  
myvar.add("hello");  
myvar.delete("hello");  
console.log(myvar);
```

ผลลัพธ์

Set [5, 99]



RUK-COM
Hosting

Entries

แปลงอ็อบเจกต์ Set ให้กลายเป็นแบบ Iterator

ตัวอย่างที่ 1

```
var myvar = new Set();
myvar.add(5);
myvar.add(99);
myvar.add("hello");
var ent = myvar.entries();
console.log(ent.next().value);
console.log(ent.next().value);
console.log(ent.next().value);
```

ผลลัพธ์

Array [5, 5]

Array [99, 99]

Array ["hello", "hello"]

forEach

ใช้วนลูปเพื่อนำฟังก์ชันเข้าไปใช้งานสมาชิกของ อ็อบเจกต์ Set

ตัวอย่างที่ 1

```
function test(var1, var2) {
    console.log("I am " + var1 + " = " + var2);
}
new Set(["apple", "man"]).forEach(test);
```



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

ผลลัพธ์

I am apple = apple

I am man = man

Has

ใช้ตรวจสอบว่ามีสมาชิกตามที่ กำหนดหรือไม่

ตัวอย่างที่ 1

```
var myvar = new Set();  
myvar.add(5);  
myvar.add(99);  
myvar.add("hello");  
chk = myvar.has(99);  
  
console.log(chk);
```

ผลลัพธ์

true



RUK-COM
Hosting

values

ดึงคิย์ออกมาจากอ็อบเจกต์ Set

ตัวอย่างที่ 1

```
var myvar = new Set();  
myvar.add(5);  
myvar.add(99);  
myvar.add("hello");  
var ke = myvar.values();  
console.log(ke.next().value);  
console.log(ke.next().value);  
console.log(ke.next().value);
```

ผลลัพธ์

5

99

hello



size

ใช้นับจำนวนสมาชิกของ อ็อบเจกต์ Set

ตัวอย่างที่ 1

```
var myvar = new Set();  
myvar.add(5);  
myvar.add(99);  
myvar.add("hello");  
chk = myvar.size;  
console.log(chk);
```

ผลลัพธ์

3

String

Anchor

ใช้สำหรับสร้างลิงค์(anchor tag ใน HTML)

ตัวอย่างที่ 1

```
var myvar = 'abcd12345';  
var anc = myvar.anchor('i am link');  
console.log(anc);
```



RUK-COM
Hosting

ผลลัพธ์

[i am link](#)abcd12345

charAt

ใช้ดึงตัวอักษรจากประโยคโดยการกำหนดเป็นลำดับของตัวอักษร

ตัวอย่างที่ 1

```
var myvar = 'ABCDEFGF';
var test = myvar.charAt(4);
console.log(test);
```

ผลลัพธ์

E

Concat

ใช้สำหรับเชื่อมคำหรือประโยค

ตัวอย่างที่ 1

```
var myvar = 'ABCDEFGF';
var test = myvar.concat('012','456','789');
console.log(test);
```

ผลลัพธ์

ABCDEFGF012456789



endsWith

ใช้ตรวจสอบว่าประโยคลงท้ายด้วยคำหรือค่าที่กำหนดหรือไม่

ตัวอย่างที่ 1

```
var myvar = 'ABCDEFGG';  
var test = myvar.endsWith('G');  
console.log(test);
```

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var myvar = 'ABCDEFGG';  
var test = myvar.endsWith('G',0);  
console.log(test);
```

ผลลัพธ์

false

Includes

ใช้ตรวจสอบว่ามีคำหรือค่าในประโยค หรือไม่

ตัวอย่างที่ 1

```
var myvar = 'ABC DEF 12345';  
var test = myvar.includes('DEF');  
console.log(test);
```



RUK-COM
Hosting

ผลลัพธ์

true

ตัวอย่างที่ 2

```
var myvar = 'ABC DEF 12345';  
var test = myvar.includes('DEF', 5);  
console.log(test);
```

ผลลัพธ์

false

indexOf

ใช้ดึงตำแหน่งของค่าในประโยค

ตัวอย่างที่ 1

```
var myvar = 'ABC DEF 12345';  
var test = myvar.lastIndexOf('DEF');  
console.log(test);
```

ผลลัพธ์

4



Length

ใช้หาค่าความยาวของสตริง

ตัวอย่างที่ 1

```
var myvar = 'ABC DEF 12345';  
var test = myvar.length;  
console.log(test);
```

ผลลัพธ์

13

Link

ใช้สำหรับสร้างลิงค์(anchor tag ใน HTML)

ตัวอย่างที่ 1

```
var tt = 'link';  
var url = 'www.google.co.th';  
var test = tt.link(url);  
console.log(test);
```

ผลลัพธ์

```
<a href="www.google.co.th">link</a>
```



RUK-COM
Hosting

Match

ใช้จับคู่คำหรือข้อความด้วย regex

ตัวอย่างที่ 1

```
var tt = 'A B CDE FGH IJK123';
var pat = /c/i;
var test = tt.match(pat);
console.log(test);
```

ผลลัพธ์

`Array ["c"]`

Normalize

แปลงอักขระ Unicode ให้อยู่ในรูปแบบปกติ

ตัวอย่างที่ 1

```
var tt = '\u0209\u0128\u0199';
tt.normalize('NFC');
console.log(tt);
```

ผลลัพธ์

`ïïk`



padEnd

สร้างสตริงให้อยู่ในรูปแบบที่กำหนดและสามารถกำหนดจำนวนตัวอักษรได้ แต่ว่าถ้าตัวอักษรมีไม่ครบตามที่กำหนด ก็จะมีตัวอักษรใหม่จนครบ(สตริงที่กำหนดจะอยู่หลังสุด)

ตัวอย่างที่ 1

```
var tt = 'ABCD';
var test = tt.padEnd(20, "0EFG");
console.log(test);
```

ผลลัพธ์

ABCD0EFG0EFG0EFG0EFG

padStart

สร้างสตริงให้อยู่ในรูปแบบที่กำหนดและสามารถกำหนดจำนวนตัวอักษรได้ แต่ว่าถ้าตัวอักษรมีไม่ครบตามที่กำหนด ก็จะมีตัวอักษรใหม่จนครบ(สตริงที่กำหนดจะอยู่หน้าสุด)

ตัวอย่างที่ 1

```
var tt = 'ABCD';
var test = tt.padStart(20, "0123");
console.log(test);
```

ผลลัพธ์

0123012301230123ABCD



RUK-COM
Hosting

Raw

แปลงอักขระทุกอย่างเป็นสตริงและเลี่ยงอักขระพิเศษ

ตัวอย่างที่ 1

```
var test = String.raw`Hi\n${2+3}!u000A`;
console.log(test);
```

ผลลัพธ์

Hi\n5!u000A

repeat

ใช้สำหรับสร้างอักขระที่กำหนดแบบซ้ำๆ ไปโดยการกำหนดรอบ

ตัวอย่างที่ 1

```
var tt = 'ABCD';
var test = tt.repeat(10);
console.log(test);
```

ผลลัพธ์

ABCDABCDABCDABCDABCDABCDABCDABCDABCDABCD



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

Replace

ใช้ regex ค้นหาและแทนที่

ตัวอย่างที่ 1

```
var tt = 'I love you';  
var test = tt.replace(/love/, '****');  
console.log(test);
```

ผลลัพธ์

I **** you

Search

ใช้ค้นหาคำที่กำหนด ถ้าเจอจะคืนค่าเป็นตำแหน่งของตัวอักษร ถ้าไม่เจอ คืนค่า -1

ตัวอย่างที่ 1

```
var tt = 'I love you';  
var test = tt.search('love');  
console.log(test);
```

ผลลัพธ์

2



RUK-COM
Hosting

Slice

ใช้ลบอักขระตามที่กำหนด โดยทำได้ด้วยการกำหนดลำดับของตัวอักษรที่จะลบ

ตัวอย่างที่ 1

```
var tt = 'I love you';
var test = tt.slice(1,-1);
console.log(test);
```

ผลลัพธ์

love yo

Split

ใช้แยกอักขระที่กำหนดออกจากกัน และยังสามารถกำหนดจำนวนได้ด้วย

ตัวอย่างที่ 1

```
var str = 'AB CD EFGHIJK LMN';
var test = str.split(' ', 2);
console.log(test);
```

ผลลัพธ์

Array ["AB", "CD"]



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

startsWith

ตรวจสอบว่าประโยคขึ้นต้นด้วยอักขระหรือคำที่กำหนดหรือไม่

ตัวอย่างที่ 1

```
var str = 'AB CD EFGHIJK LMN';  
var test = str.startsWith('A');  
console.log(test);
```

ผลลัพธ์

true

Substr

ดึงตัวอักษรออกจากประโยค โดยกำหนดจุดเริ่มต้นและสิ้นสุดได้

ตัวอย่างที่ 1

```
var str = 'ABCDEFGHJKLMNOP1234567890';  
var test = str.substr(1,3);  
console.log(test);
```

ผลลัพธ์

BCD



RUK-COM
Hosting

substring

ดึงตัวอักษรออกมาจากประโยค โดยกำหนดจุดเริ่มต้นและสิ้นสุดได้

ตัวอย่างที่ 1

```
var str = 'ABCDEFGHJKLMNOP1234567890';  
var test = str.substring(0,5);  
console.log(test);
```

ผลลัพธ์

ABCDE

toLowerCase

ใช้สำหรับแปลงตัวพิมพ์ใหญ่ให้เป็นตัวพิมพ์เล็ก

ตัวอย่างที่ 1

```
var str = 'ABCDEFGHJKLMNOP1234567890';  
var test = str.toLowerCase();  
console.log(test);
```

ผลลัพธ์

abcdefghijklmnop1234567890



toString

ใช้สำหรับแปลงอ็อบเจกต์ให้ตัวอักษร

ตัวอย่างที่ 1

```
var obj = new String('abcdefghijklmn1234567890');  
var test = obj.toString();  
console.log(test);
```

ผลลัพธ์

abcdefghijklmn1234567890

toUpperCase

ใช้สำหรับแปลงตัวพิมพ์เล็กให้เป็นตัวพิมพ์ใหญ่

ตัวอย่างที่ 1

```
var str = 'abcdefghijklmn1234567890';  
var test = str.toUpperCase();  
console.log(test);
```

ผลลัพธ์

ABCDEFGHIJKLMN1234567890



RUK-COM
Hosting

trim

ใช้ลบช่องว่างออกจากด้านหน้าและหลังของคำหรือประโยค

ตัวอย่างที่ 1

```
var str = '      A B C D E FGHI ';
var test = str.trim();
console.log(test);
```

ผลลัพธ์

A B C D E FGHI

valueOf

ใช้สำหรับดึงค่าของอ็อบเจกต์

ตัวอย่างที่ 1

```
var str = new String('my name is boy');
var test = str.valueOf();
console.log(test);
```

ผลลัพธ์

my name is boy



Proxy

เป็นออบเจกต์ที่ใช้ห่อหุ้มอีกออบเจกต์หนึ่ง เพื่อเรียกใช้งานฟังก์ชันหรือพรอพเพอร์ตี้ของออบเจกต์ที่ถูกห่อหุ้ม โดยไม่เข้าไปเปลี่ยนแปลงค่าในออบเจกต์ที่ห่อหุ้มนั้น

Construct

ใช้สำหรับสร้างการทำงานให้กับ **constructor**

ตัวอย่างที่ 1

```
var p = new Proxy(function() {}, {  
  construct: function(target, argL, newTarget) {  
    console.log("call=> " + argL);  
    return { value: argL[0] * 100 };  
  }  
});  
console.log(new p(1).value);
```

ผลลัพธ์

call=> 1

100



RUK-COM
Hosting

defineProperty

ใช้กำหนดค่า property ให้ method

ตัวอย่างที่ 1

```
var p = new Proxy({}, {
  defineProperty(target, prop, des) {
    console.log(des);
    return Reflect.defineProperty(target, prop, des);
  }
});
Object.defineProperty(p, "name", { value: "proxy" });
```

ผลลัพธ์

```
Object { value: "proxy" }
```

deleteProperty

ใช้ลบค่า property ให้ method ตัวอย่างที่ 1



ได้งานง่ายๆ เพียง 3 ขั้นตอน

1. **คลิกที่รูปนี้** เพื่อสร้างโปรไฟล์กับ GetLinks
2. รอรับข้อเสนองานหลากหลายตำแหน่ง
3. รับโบนัส **10,000** บาท

สมัครเลย

```

var myvar = {  webName: "ABCD"};
var p = new Proxy(myvar, {
  deleteProperty: function(target, prop) {
    console.log(prop);
    Reflect.deleteProperty(target, prop);
    return true;
  }
});
console.log(myvar);
delete p.webName;
console.log(myvar);

```

ผลลัพธ์

```

Object { webName: "ABCD" }
webName
Object {  }

```

Get

เข้าถึงค่า property



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var p = new Proxy({}, {
  get: function(target, prop, receiver) {
    console.log("call: " + prop);
  }
});

console.log(p.a);
```

ผลลัพธ์

call: a

undefined

Apply

ใช้สำหรับสร้างการทำงานให้กับฟังก์ชัน

ตัวอย่างที่ 1

```
var p = new Proxy(function() {}, {
  apply: function(target, arg, argL) {
    console.log(argL);
  }
});

p(1, 2, 3);
```



ผลลัพธ์

Array [1, 2, 3]

getOwnPropertyDescriptor

เข้าถึง index ของ property

ตัวอย่างที่ 1

```
var p = new Proxy({ a: 100 }, {
  getOwnPropertyDescriptor: function(target, prop) {
    console.log("call: " + prop);
    return {value: 123456789, configurable: true, enumerable: true};
  }
});

console.log(Object.getOwnPropertyDescriptor(p, 'a').value);
```

ผลลัพธ์

call: a

123456789

getPrototypeOf

ใช้กำหนดให้สืบทอดคุณสมบัติจากออบเจกต์อื่น



RUK-COM
Hosting

ตัวอย่างที่ 1

```
var obj = {};
var p = new Proxy(obj, {
  getPrototypeOf(target) {
    return Array.prototype;
  }
});
var test = Object.getPrototypeOf(p) === Array.prototype;
console.log(test);
```

ผลลัพธ์

true

Has

ใช้ตรวจสอบค่า ถ้ามีจะคืนค่า true ถ้าไม่มีจะคืนค่า false

ตัวอย่างที่ 1

```
var p = new Proxy({}, {
  has: function(target, prop) {
    console.log("=> " + prop);
    return true;
  }
});

console.log("a" in p);
```



ผลลัพธ์

=> a

true

isExtensible

ใช้ตรวจสอบว่าอ็อบเจกต์นั้นผ่านการใช้งานจากฟังก์ชันอื่นมาหรือไม่ ถ้ายังจะคืนค่า **true** แต่ผ่านการใช้งานจะคืนค่า **false**

ตัวอย่างที่ 1

```
var p = new Proxy({}, {
  isExtensible: function(target) {
    console.log("called");
    return true;
  }
});
console.log(Object.isExtensible(p));
```

ผลลัพธ์

called

true



RUK-COM
Hosting

ownKeys

เข้าถึงค่าของ `method`

ตัวอย่างที่ 1

```
var p = new Proxy({}, {
  ownKeys: function(target) {
    console.log("called");
    return ["a", "b", "c"];
  }
});

console.log(Object.getOwnPropertyNames(p));
```

ผลลัพธ์

`called`

`Array ["a", "b", "c"]`



preventExtensions

ใช้สำหรับป้องกันการเพิ่ม property เข้าไปในเมตอด

ตัวอย่างที่ 1

```
var p = new Proxy({}, {  
  preventExtensions: function(target) {  
    console.log("called");  
    Object.preventExtensions(target);  
    return true;  
  }  
});  
console.log(Object.preventExtensions(p));
```

ผลลัพธ์

called

Object { }



RUK-COM
Hosting

Set

เซตค่าให้กับ method

ตัวอย่างที่ 1

```
var p = new Proxy({}, {
  set: function(target, prop, value, receiver) {
    console.log("called: " + prop + " = " + value);
    return true;
  }
});
p.a = 9999;
```

ผลลัพธ์

called: a = 9999



setPrototypeOf

เซทให้ออบเจ็คสืบทอดคุณสมบัติจากออบเจ็คอื่น ถ้าสำเร็จจะคืนค่า **true** ไม่สำเร็จจะคืนค่า **false**

ตัวอย่างที่ 1

```
var refalse = {  
  setPrototypeOf(target, newProto) {  
    return false;  
  }  
};  
var newProto = {}, target = {};  
var p1 = new Proxy(target, refalse);  
var test = Reflect.setPrototypeOf(p1, newProto);  
console.log(test);
```

ผลลัพธ์

false



RUK-COM
Hosting