Slice & Reverse

Method slice()

ใช้สำหรับคัดลอกสมาชิกจากลำดับเริ่มต้นจนถึงก่อนลำดับสุดท้าย (ถ้าไม่ได้มีการระบุลำดับสุดท้าย จะคัดลอกจนสิ้นสุด array)

Syntax slice()

array.slice(start,end)

- ค่า parameter เป็น int ใส่บอกตำแหน่งที่เราจะหาค่าเริ่มจุดที่เท่าไหร่ และสิ้นสุดจุดที่เท่าไหร่
- Output ที่ได้เป็น array ตัวใหม่ ที่มีข้อมูลเหมือนกับในตัว array ตั้งต้นหากเราไม่ได้กำหนดพารามิเตอร์ หรือหากเรากำหนดค่า parameter ก็จะปริ้นค่าที่อยู่ใน array ออกมาตามลำดับที่เรากำหนดภายใน parameter
- สามารถกำหนดตำแหน่งใน parameter เป็น + หรือ - ได้ เป็น + จะนับจากค่าที่ 0 ไป ยังตำแหน่งต่อไปเรื่อยๆ หาก เริ่มที่ - จะเริ่มจากตำแหน่งสุดท้ายไปยังตำแหน่งแรก

Example

```
//Array ตั้งต้น
let array1 = [1, 2, 3, 4, 5, 6, 7, 8]; //array นี้จะเก็บชนิดข้อมูลเดียวกันทั้งหมด
let array2 = ["i want to sleep",1,{A:'test',B:'test1'},[1, 2, 3, 4, 5],(() => 'test3'),((parameter) =>
parameter) ]; //array นี้จะเก็บชนิดข้อมูลต่างชนิดกัน
//test ของ array1
console.log(`------------------primitive array--------------------`);
console.log(`------------------default--------------------`);
console.log("default : " + array1);
```

Output

```
------------------primitive array--------------------
------------------default--------------------
default : 1,2,3,4,5,6,7,8
```

นี่คือรูปแบบของ Array ตั้งต้น

```
//test ของ array2
console.log(
  `------------------array has string,int,object,array,function(notparameter) and
function(parameter) ----------------------`
);
```

```
console.log(`------------------default----------------------`);
console.log("default : " + array2);
console.log(`------------------slice----------------------`);
```

Output

```
------------------array has string,int,object,array,function(notparamrter) and function(paramrter) ----------------------
------------------default----------------------
default : i want to sleep,1,[object Object],1,2,3,4,5,() => "test3",(parameter) => parameter
------------------slice----------------------
```

นี้คือรูปแบบของ Array ตั้งต้น

Method slice ที่ไม่มีการรับ parameter

```
//function ไว้เก็บ method slice ที่ไม่มีการรับ parameter
const sliceNoPar = (model) => {
  return model.slice();
};
//test array1
//Slice has no parameter
console.log(`------------------slice----------------------`);
console.log(`------------------no parameter ----------------------`);
console.log("array use slice : " + sliceNoPar(array1));
console.log("slice is same array?  ");
console.log(array1 === sliceNoPar(array1));
```

Output

```
------------------no parameter ----------------------
array use slice : 1,2,3,4,5,6,7,8
slice is same array?
false
```

เมื่อเราไม่ได้ใส่ parameter ลงไปใน method slice ดังนั้น method slice output ที่ออกมาจึงเป็น Array ที่มีสมาชิกเหมือนเดิมแต่จะคนละ address กับตัวตั้งต้น เหมือนการโคลน Array มานั้นเอง ซึ่งดูได้จากการ test ข้างล่างที่ดูว่า method slice ใช้ array ตัวเก่าไหม

```
//test array2
//Slice has no parameter
console.log(`------------------no parameter ----------------------`);
console.log("array use slice : " + sliceNoPar(array2));
console.log("slice is same array?  ");
console.log(array2 === sliceNoPar(array2));
```

Output

```
------------------no parameter --------------------
array use slice : i want to sleep,1,[object Object],1,2,3,4,5,() => "test3",(parameter) => parameter
slice is same array?
false
```

เหมือนกับตัว primitive array เลย

Method slice ที่มีการรับ parameter 1 ตัว

```
//function ไว้เก็บ method slice ที่มีการรับ parameter 1 ตัว นั่นคือตัวเริ่ม
const sliceOnePar = (model, start) => {
  return model.slice(start);
};
//test array1
//Slice has 1 parameter
console.log(`------------------1 parameter----------------------`);
console.log("array use slice(start = +) : " + sliceOnePar(array1, 1));
console.log("array use slice(start = -) : " + sliceOnePar(array1, -1));
console.log("array use slice(length) : " + sliceOnePar(array1, array1.length));
```

Output

```
------------------1 parameter----------------------
array use slice(start = +) : 2,3,4,5,6,7,8
array use slice(start = -) : 8
array use slice(length) :
```

จะเห็นได้ว่า method slice สามารถรับ parameter มาตัวเดียวได้โดยไม่ต้องใส่ค่าไป 2 ค่า แต่ค่านั้นจะถูกเก็บไว้เป็นค่า เริ่มต้นที่ตำแหน่ง index 0 แล้วไปยังตัวสุดท้ายใน Array

บรรทัด 1 : เราใส่ค่า start เป็น 1 (เริ่มที่index = 1) จะเห็นได้ว่า output ออกมาเป็นค่าทั้งหมดยกเว้น ตำแหน่งที่ 0

บรรทัด 2 : เราลองใส่ค่าที่เป็นลบ จะเห็นได้ว่ามันจะนั้น เริ่มนั้นจากตัวสุดท้ายไล่มาหาด้านหน้าใน ตัวอย่างเราเลือกเป็นตำแหน่ง -1 ซึ่งถ้าลองไล่ใน arrayแล้วจะเป็นตัวสุดท้ายซึ่งก็คือเลข8นั้นเองจากนั้นก็ไล่ไปยังตัวสุดท้าย output ที่ออกมาเลยเป็น array ที่มีค่าคือ 8 นั่นเอง

บรรทัด 3 : เราลองใส่ค่าแทนที่จะเป็นลบหรือบวกเราใส่ค่า lengthของ array ไป จะเห็นได้ว่ามันจะเริ่มที่ตำแหน่งหลังจากสุดท้ายซึ่งก็คือไม่มีเลยออกมาเป็น array เปล่าๆ

```
//test array2
//Slice has 1 parameter
console.log(`------------------1 parameter----------------------`);
console.log("array use slice(start = +) : " + sliceOnePar(array2, 1));
```

```
console.log("array use slice(start = -) : " + sliceOnePar(array2, -1));
console.log("array use slice(length) : " + sliceOnePar(array2, array2.length));
```

Output

```
------------------1 parameter---------------------
array use slice(start = +) : 1,[object Object],1,2,3,4,5,() => "test3",(parameter) => parameter
array use slice(start = -) : (parameter) => parameter
array use slice(length) :
```

จะเห็นได้ว่า method slice สามารถรับ parameter มาตัวเดียวได้โดยไม่ต้องใส่ค่าไป 2 ค่า แต่ค่านั้นจะถูกเก็บไว้เป็นค่า เริ่มต้นที่ตำแหน่ง index 0 แล้วไปยังตัวสุดท้ายใน Array และ ยังใช้เป็น array ที่เก็บค่าเป็น primitive, object,array,functionทั้งที่มี parameter และ ไม่มี จะเห็นได้ว่าแทบไม่ต่างจากตัวที่เป็น primitive เลย

Method slice ที่มีการรับ parameter 2 ตัว

```
//function ไว้เก็บ method slice ที่มีการรับ parameter 2 ตัว คือ ตัวที่ใช้เริ่ม และตัวที่ใช้จบ
const sliceTwoPar = (model, start, finish) => {
  return model.slice(start, finish);
};
//test array1
//Slice has 2 parameter
console.log(`------------------2 parameter---------------------`);
//แสดงค่าarrayที่เริ่มต้นด้วย + และสิ้นสุดด้วยตำแหน่ง+
console.log(
  "array use slice(start = +, end = +) : " + sliceTwoPar(array1, 1, 2)
);
//แสดงค่าarrayที่เริ่มต้นด้วย + และสิ้นสุดด้วยตำแหน่ง-
console.log(
  "array use slice(start = +,end = -) : " + sliceTwoPar(array1, 1, -2)
);
//แสดงค่าarrayที่เริ่มต้นด้วย - และสิ้นสุดด้วยตำแหน่ง+
console.log(
  "array use slice(start = -, end = +) : " + sliceTwoPar(array1, -4, 5)
);
//แสดงค่าarrayที่เริ่มต้นด้วย - และสิ้นสุดด้วยตำแหน่ง-
console.log(
  "array use slice(start = -,end = -) : " + sliceTwoPar(array1, -4, -3)
);
console.log(
```

```
  "array use slice(length) : " + sliceTwoPar(array1, 1, array1.length)
);
```

Output

```
------------------2 parameter----------------------
array use slice(start = +, end = +) : 2
array use slice(start = +,end = -) : 2,3,4,5,6
array use slice(start = -, end = +) : 5
array use slice(start = -,end = -) : 5
array use slice(length) : 2,3,4,5,6,7,8
---------------------------------------
```

เราได้ลองใส่ parameter ทั้งค่าเริ่มต้นกับค่าสุดท้าย ใน method slice จะเห็นได้ว่าค่าstartจะเป็นการนับindex = 0 ส่วนค่า end นั้นก็ยังจะนั้นโดยเริ่มที่ index = 0 แต่จะไม่ return ค่าถึงตำแหน่งค่าของ end (return แค่ค่าก่อนหน้า 1 ตำแหน่ง)

บรรทัด 1 ; เริ่มที่ตำแหน่งที่ start = 1 (ค่า = 2) ไปยังตำแหน่งที่ end = 2 (ค่า = 3) แต่ไม่เอาค่า ณ ตำแหน่งนั้น จึงทำให้ return ออกมาเป็น array ที่มีค่าข้างในแค่ 2

บรรทัด 2 ; เริ่มที่ตำแหน่งที่ start = 1 (ค่า = 2) ไปยังตำแหน่งที่ end = -2 (นับจากข้างหลังค่าเลย = 7) แต่ไม่เอาค่า ณ ตำแหน่งนั้นจึงทำให้ return ออกมาเป็น array ที่มีค่าข้างในแค่ 2,3,4,5,6

บรรทัด 3 ; เริ่มที่ตำแหน่งที่ start = -4 (ค่า = 5) ไปยังตำแหน่งที่ end = 5 (ค่า = 6) แต่ไม่เอาค่า ณ ตำแหน่งนั้น จึงทำให้ return ออกมาเป็น array ที่มีค่าข้างในแค่ 5

บรรทัด 4 ;เริ่มที่ตำแหน่งที่ start = -4 (ค่า = 5) ไปยังตำแหน่งที่ end = -3 (นับจากข้างหลังค่าเลย = 6) แต่ไม่เอาค่า ณ ตำแหน่งนั้นจึงทำให้ return ออกมาเป็น array ที่มีค่าข้างในแค่ 5

บรรทัด 5 ; เริ่มที่ตำแหน่งที่ start = 1 (ค่า = 2) ไปยังตำแหน่งที่ end = จำนวนlengthของarray1 (ค่า=8) output ที่ออกมาเลยจะค่าตั้งแต่ตำแหน่งที่2 ไล่ไปยังตัวสุดท้าย

```
//test array2
//Slice has 2 parameter
console.log(`------------------2 parameter----------------------`);
console.log(
  "array use slice(start = +, end = +) : " + sliceTwoPar(array2, 1, 2)
);
console.log(
  "array use slice(start = +,end = -) : " + sliceTwoPar(array2, 1, -2)
);
console.log(
  "array use slice(start = -, end = +) : " + sliceTwoPar(array2, -4, 5)
```

```
);
console.log(
  "array use slice(start = -,end = -) : " + sliceTwoPar(array2, -4, -3)
);
console.log(
  "array use slice(length) : " + sliceTwoPar(array2, 1, array2.length)
);
```

Output

```
------------------2 parameter----------------------
array use slice(start = +, end = +) : 1
array use slice(start = +,end = -) : 1,[object Object],1,2,3,4,5
array use slice(start = -, end = +) : [object Object],1,2,3,4,5,() => "test3"
array use slice(start = -,end = -) : [object Object]
array use slice(length) : 1,[object Object],1,2,3,4,5,() => "test3",(parameter) => parameter
```

เราได้ลองใส่ parameter ทั้งค่าเริ่มต้นกับค่าสุดท้าย ใน method slice และ array ที่ใช้นั้นมีทั้ง primitive, object,array,functionทั้งที่มี parameter และ ไม่มี จะเห็นได้ว่าค่าstartจะเป็นการนับindex = 0 ส่วนค่า end นั้นก็ยังจะนั้นโดยเริ่มที่ index = 0 แต่จะไม่ return ค่าถึงตำแหน่งค่าของ end (return แค่ค่าก่อนหน้า 1 ตำแหน่ง) และยังมีoutputคล้ายกับอันข้างบนที่เป็น primitive หมดเลย

Method reverse ()

เป็นการเรียงลำดับค่าใน array แบบย้อนกลับ

Syntax reverse()

array.reverse()

- ไม่มีค่า parameter
- ใช้สำหรับกลับลำดับสมาชิกในอาเรย์ สมาชิกตัวสุดท้ายจะมาอยู่ที่ตำแหน่งแรก สมาชิกตัวรองสุดท้ายจะมาอยู่ตำแหน่งที่สอง ถัดไปเรื่อยๆ จนกระทั่งสมาชิกตัวแรกไปอยู่ที่ตำแหน่งสุดท้ายของอาเรย์

Example

```
//Array ตั้งต้น
let array1 = [1, 2, 3, 4, 5, 6, 7, 8];  //array นี้จะเก็บชนิดข้อมูลเดียวกัน
let array2 = [
        "i want to sleep",
        1,
        { A: "test", B: "test1" },
        [1, 2, 3, 4, 5],
        () => "test3",
        (parameter) => parameter,];  //array นี้จะเก็บชนิดข้อมูลต่างชนิดกัน
//ฟังก์ชันไว้เก็บ method reverse()
const Reversefun = (model) => {
  return model.reverse();
};
//test array1
console.log(`------------------primitive array----------------------`);
console.log(`------------------default----------------------`);
console.log("default : " + array1);
console.log(`------------------Reverse----------------------`);
//แสดง ค่าของarray1 จากตัวท้ายสุดมายังตัวแรกสุด
console.log("array use Reverse :" + Reversefun(array1));
//ให้แสดงการตรวจสอบว่า array1 กับ array1ที่ถูกReverseเป็นarrayเดียวกันหรือไม่
console.log("Reverse is same array?  " );
console.log(array1 === Reversefun(array1));
console.log(`----------------------------------------`);
```

Output

```
------------------primitive array---------------------
------------------default---------------------
default : 1,2,3,4,5,6,7,8
------------------Reverse---------------------
array use Reverse :8,7,6,5,4,3,2,1
Reverse is same array?
true
---------------------------------------
```

จะเห็นได้ว่าเมื่อมีการใช้ method reverse นั้น ผลลัพธ์ที่ออกมาจะแสดงผลย้อนกลับ ซึ่งนั่นก็คือตัวที่อยู่ลำดับสุดท้ายจะมาอยู่ที่ตำแหน่งแรกแทน และต่อไปเรื่อยๆ

ส่วนการเช็คว่าใช้ array เดียวกับอันตั้งต้นไหม สรุปคือใช่เพราะมี address เดียวกันนั้นเอง

```
//test array2
console.log(
  `------------------array has string,int,object,array,function(not parameter) and
function(parameter) ----------------------`
);
console.log(`------------------default----------------------`);
console.log("default : " + array2);
console.log(`------------------Reverse----------------------`);
console.log("array use Reverse :" + array2);
console.log("Reverse is same array?  " );
console.log(array2 === Reversefun(array2));
```

Output

```
------------------array has string,int,object,array,function(notparamrter) and function(paramrter) ----------------------
------------------default--------------------
default : i want to sleep,1,[object Object],1,2,3,4,5,() => 'test3',(parameter) => parameter
------------------Reverse--------------------
array use Reverse :i want to sleep,1,[object Object],1,2,3,4,5,() => 'test3',(parameter) => parameter
Reverse is same array?
true
```

จะเห็นได้ว่าคล้ายกับ ตัวของ primitive array เพราะหลักการการทำงานนั้นเหมือนกัน