Web Application Development for Industrial Engineers

การพัฒนาเวปแอปพลิเคชันสำหรับวิศวกรอุตสาหการ

Arrays

- An array is a single `object` that contains multiple values enclosed in square brackets and separated by commas.

```
let myNameArray = ['Chris', 'Bob', 'Jim'];
let myNumberArray = [10, 15, 40];
```

Finding the length of an array

```
const shopping = ['bread', 'milk', 'cheese', 'hummus', 'noodles'];
console.log(shopping.length); // 5
```

Accessing and modifying array items

```
const shopping = ['bread', 'milk', 'cheese', 'hummus', 'noodles'];
console.log(shopping[0]);
// returns "bread"
```

```
const shopping = ['bread', 'milk', 'cheese', 'hummus', 'noodles'];
shopping[0] = 'tahini';
console.log(shopping);
// shopping will now return [ "tahini", "milk", "cheese", "hummus", "noodles" ]
```

Multi-dimensional array

```
const random = ['tree', 795, [0, 1, 2]];
random[2][2];
```

Finding items in an array

```
const birds = ['Parrot', 'Falcon', 'Owl'];
console.log(birds.indexOf('Owl')); //  2
console.log(birds.indexOf('Rabbit')); // -1
```

Adding items (end)

```
const myArray = ['Manchester', 'Liverpool'];
myArray.push('Cardiff');
console.log(myArray); // [ "Manchester", "Liverpool", "Cardiff" ]
myArray.push('Bradford', 'Brighton');
console.log(myArray); // [ "Manchester", "Liverpool", "Cardiff", "Bradford", "Brighton" ]
```

Adding items (end)

```
const myArray = ['Manchester', 'Liverpool'];
const newLength = myArray.push('Bristol');
console.log(myArray); // [ "Manchester", "Liverpool", "Bristol" ]
console.log(newLength); // 3
```

Adding items (start)

```
const myArray = ['Manchester', 'Liverpool'];
myArray.unshift('Edinburgh');
console.log(myArray); // [ "Edinburgh", "Manchester", "Liverpool" ]
```

Removing items

- To remove the last item from the array, use `pop()`.

```
const myArray = ['Manchester', 'Liverpool'];
myArray.pop();
console.log(myArray); // [ "Manchester" ]
```

```
const myArray = ['Manchester', 'Liverpool'];
const removedItem = myArray.pop();
console.log(removedItem); // "Liverpool"
```

Removing items

- To remove the first item from an array, use `shift()`

```
const myArray = ['Manchester', 'Liverpool'];
myArray.shift();
console.log(myArray); // [ "Liverpool" ]
```

Removing items

- Using `splice()`
  - First argument says where to start removing items
  - Second argument says how many items should be removed.

```
const myArray = ['Manchester', 'Liverpool', 'Edinburgh', 'Carlisle'];
const index = myArray.indexOf('Liverpool');
if (index !== -1) {
  myArray.splice(index, 1);
}
console.log(myArray); // [ "Manchester", "Edinburgh", "Carlisle" ]
```

String-array conversion

```
const myData = 'Manchester,London,Liverpool,Birmingham,Leeds,Carlisle';
```

- Convert to array

```
const myArray = myData.split(',');
myArray;
```

- Convert to string

```
const myNewString = myArray.join(',');
myNewString;
```

Accessing every item

```
const birds = ['Parrot', 'Falcon', 'Owl'];

for (const bird of birds) {
  console.log(bird);
}
```

Objects

- An object is a structure of code that models a real-life object.
- For example, an object that represents a box which contains
  - `Width` / `Length` / `Height`
- An object that represents a person which contains
  - `Name` / `Height` / `Weight` / `Language` / `How to say hello`

Declaring an object

- Declare a blank object

```
const person = {};
```

- Initialize an object

```
let dog = { name: 'Spot', breed: 'Dalmatian' };
```

Retrieve the information

- Retrieve the information stored in the object

```
let dog = { name: 'Spot', breed: 'Dalmatian' };
console.log(dog.name); // 'Spot'
```

More complex object

```
const person = {
  name: ['Bob', 'Smith'],
  age: 32,
  gender: 'male',
  interests: ['music', 'skiing'],
  greeting: function () {
    alert(`Hi! I'm " ${this.name[0]}.`);
  },
};
```

More complex object

```
person.name;
person.name[0];
person.age;
person.interests[1];
person.greeting();
```

Object member

- The value of an object member can be pretty much anything.
  - String
  - Number
  - Arrays
  - Functions.
- The data are referred to as the object's *properties*.
- The function is referred to as the object's *method*.

Dot notation

```
const person = {
  name: {
    first: 'Bob',
    last: 'Smith',
  },
  age: 30,
};
```

```
person.name.first;
person.name.last;
```

Bracket notation

```
person['age'];
person['name']['first'];
```

- Looks very similar to how you access the items in an array/
- It is basically the same thing — instead of using an index number to select an item, you are using the name associated with each member's value.

Set object members

```
person.age = 45;
person['name']['last'] = 'Cratchit';
person['eyes'] = 'hazel'; // New properties
person.farewell = function () {
  alert('Bye everybody!');
};
```

```
let myDataName = 'height';
let myDataValue = '1.75m';
person[myDataName] = myDataValue;
```

What is "this"?

```
greeting: function() {
  alert('Hi! I\'m ' + this.name.first + '.');
}
```

- The this keyword refers to the current object the code is being written inside — so in this case this is equivalent to person.

What is "this"?

```
const person1 = {
  name: 'Chris',
  greeting: function () {
    alert("Hi! I'm " + this.name + '.');
  },
};

const person2 = {
  name: 'Deepti',
  greeting: function () {
    alert("Hi! I'm " + this.name + '.');
  },
};
```

What is "this"?

- Using `this` isn't hugely useful when you are writing out object literals by hand.
- But it really comes into its own when you are dynamically generating objects (for example using constructors).

Calculator App (V2)

- https://ie-software-dev.netlify.app/codes/t08_js/t01_calculator/