JavaScript

Buổi 4: Object. Kiểu dữ liệu thời gian (Date).

Nội dung buổi 4

- Làm quen với Object.
- Cú pháp ES6 (Destructuring, Spread Operator, Rest Pattern, for/of).
- Web API (setTimeOut, setInterval).
- Kiểu dữ liệu thời gian (Date).
- Chuẩn hóa dữ liệu thời gian.
- Chuẩn hóa dữ liệu thời số.

Làm quen với Object

Làm quen với Object

- Object (đối tượng) có thể hiểu đơn giản là một tập hợp các cặp khóa (keys) và giá trị (values) được khai báo chung trong một đơn vị dữ liệu.
- Quy tắc đặt tên key tương tự với quy tắc đặt tên biến, tuy nhiên có dùng số để đặt tên key (không khuyến kích). Giá trị của một key có thể là giá trị bất kỳ (Primitive, mảng, object).

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
  languages: ["TypeScript", "ReactJS", "NodeJS"],
  location: {
    country: "UK",
    city: "London",
  },
  married: true,
};
```

Làm quen với Object

Lấy giá trị từ object:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
};

coolDev.fullName; // "Jonas Smith"
coolDev["age"]; // 34
coolDev.company; // undefined
```

Cập nhật giá trị trong object:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
};

coolDev.fullName = "Jonas Smithmane";
coolDev["age"] = 35;
console.log(coolDev);
// => { fullName: "Jonas Smithmane", age: 35 }
```

Làm quen với Object

Thêm và xóa key - value:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
};

coolDev.married = true;
delete coolDev.age;

console.log(coolDev);
// => { fullName: "Jonas Smith", married: true }
```

Lấy ra các keys, values từ object:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
};

Object.keys(coolDev);
// => ["fullName", "age"]
Object.values(coolDev);
// => ["Jonas Smith", 34]
```

Bài thực hành số 1

Bài thực hành số 1

Bài 1: Tạo một biến lưu thông tin về một con vật nuôi trong nhà là một object gồm các key sau: type (chó, mèo, chim...), name, breed (giống), hairColor, favoriteFood (là một mảng gồm các giá trị chuỗi).

Bài 2: In ra cửa sổ console giá trị của một key bất kỳ trong object tạo ra từ bài 1.

Bài 3: Thêm vào object ở bài 1 key “yearOld” để bổ sung số tuổi cho con vật, sử dụng giá trị số. In ra cửa sổ console dữ liệu của object sau khi đã bổ sung key - value mới.

Bài 4: Viết một hàm nhận tham số là một object giống object của bài 1. Từ các key và giá trị của object đó, hàm trả về một chuỗi có nội dung như sau: “Nhà tôi có một con mèo tên là Tom, nó thích ăn cá, bánh xèo, bít tết.”

Bài 5: Tạo một biến có giá trị là một mảng chứa 3 object giống bài 1. Thông tin về các con vật không được trùng nhau quá nhiều và tuổi của chúng phải khác nhau.

Bài 6: Viết một hàm nhận tham số là một mảng giống như mảng của bài 5. Hàm tìm ra con vật có tuổi lớn nhất và trả về object thông tin về con vật đó.

Cú pháp ES6

ES6 - Destructuring Arrays, Objects

Destructuring array:

```
const [a, b, c] = [1, 2, 3, 4];
console.log(a); // 1
console.log(b); // 2
console.log(c); // 3
```

Destructuring object:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
};
const { fullName, age } = coolDev;
console.log(fullName); // "Jonas Smith"
console.log(age); // 34
```

ES6 - Destructuring Arrays, Objects

Áp dụng với tham số trong hàm:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
  married: true,
};

function introduce({ fullName, age }) {
  const yearBirth = 2022 - age;
  return `My name is ${fullName}, I was born in ${yearBirth}`;
}

console.log(introduce(coolDev));
// => "My name is Jonas Smith, I was born in 1988"
```

```
const date = [8, 10, 2022];

function createDateString([day, month, year]) {
  const outputDay = day.padStart(2, 0);
  const outputMonth = month.padStart(2, 0);

  return outputDay + "/" + outputMonth + "/" + year;
}

console.log(createDateString(date));
// => "08/10/2022"
```

ES6 - Destructuring Arrays, Objects

Đổi tên tham số trong destructuring (Có thể áp dụng với cả khai báo biến):

```
const company = {
  fullName: "Techmaster",
};

const coolDev = {
  fullName: "Jonas Smith",
  position: "Front End Developer",
};

function introduce({ fullName, position }, { fullName: companyName }) {
  return `My name is ${fullName}, I'm a ${position} at ${companyName}`;
}

function introduce2({ fullName, position }, company) {
  return `My name is ${fullName}, I'm a ${position} at ${company.fullName}`;
}

introduce(coolDev, company);
introduce2(coolDev, company);
```

Bài thực hành số 2

Bài thực hành số 2

Sử dụng cú pháp ES6 Destructuring để làm các bài tập sau:

Bài 1: Sử dụng object tạo ra trong Bài 1 của Bài thực hành số 1. Sử dụng kỹ thuật destructuring để lấy ra key name và favoriteFood. In ra cửa sổ console giá trị của các key đó.

Bài 2: Sử dụng mảng tạo ra trong Bài 5 của Bài thực hành số 1. Sử dụng kỹ thuật destructuring để lấy ra con vật ở vị trí thứ cuối cùng trong mảng. In ra cửa sổ console tên của con vật đó.

Bài 3: Làm lại Bài 4 trong Bài thực hành số 1 bằng kỹ thuật destructuring (áp dụng với tham số).

ES6 - Rest Pattern

Áp dụng chung với cách viết destructuring và tham số, sử dụng để nhóm các thành phần còn lại, chưa được khai báo chung thành một mảng hoặc object.

Áp dụng với mảng:

```
const languages = ["JavaScript", "PHP", "Java", "Ruby"];
const [js, ...others] = languages;
console.log(js); // "JavaScript"
console.log(others); // ["PHP", "Java", "Ruby"]
```

Áp dụng với object:

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
  married: true,
};
const { fullName, ...others } = coolDev;

console.log(fullName); // "Jonas Smith"
console.log(others);
// => { age: 34, married: true, }
```

Bài thực hành số 3

Bài thực hành số 3

Sử dụng cú pháp Rest Pattern để làm các bài tập sau:

Bài 1: Viết một hàm nhận tham số là một mảng, hàm trả về một mảng có tất cả phần tử của mảng tham số truyền vào trừ phần tử đầu tiên.

Bài 2: Viết một hàm nhận tham số là một object giống Bài 1 của Bài thực hành số 1. Hàm trả về một mảng mới giống với tham số truyền vào nhưng bỏ đi 2 key là “favoriteFood” và “breed”. Viết theo 2 cách, khai báo biến trong hàm và áp dụng với tham số.

ES6 - Spread Operator

Khi áp dụng với giá trị (vế bên phải của dấu “=“ trong phép gán) có thể sử dụng cú pháp “...” để phá bỏ cặp ngoặc [ ] hoặc { }.

```
const coolDev = {
  fullName: "Jonas Smith",
  age: 34,
};

const copyCoolDev = { ...coolDev };
// => { fullName: "Jonas Smith", age: 34 }

const anotherCoolDev = { ...coolDev, age: 27, title: "Developer" };
// => { fullName: "Jonas Smith", age: 27, title: "Developer" }

const notSoCoolDev = { fullName: "Mike Jordy", age: 25, ...coolDev };
// => { fullName: "Jonas Smith", age: 34 }
```

```
const classA = ["Mai", "Lan", "Trung"];

const classB = [...classA];
// => ["Mai", "Lan", "Trung"]

const classC = ["Phong", ...classA, "Như"];
// => ["Phong", "Mai", "Lan", "Trung", "Như"]
```

Bài thực hành số 4

Bài thực hành số 4

Sử dụng cú pháp Spread Operator để làm các bài tập sau:

Bài 1: Viết một hàm nhận tham số là một mảng, hàm trả về một mảng có tất cả phần tử của mảng tham số truyền vào trừ phần tử đầu tiên.

Bài 2: Viết một hàm nhận tham số là một mảng, hàm trả về một mảng gồm 2 phần tử cuối cùng của mảng tham số truyền vào.

Bài 3: Viết một hàm nhận tham số là một object giống Bài 1 của Bài thực hành số 1. Hàm trả về một mảng mới giống với tham số truyền vào nhưng bỏ đi 2 key là “favoriteFood” và “breed”. Viết theo 2 cách, khai báo biến trong hàm và áp dụng với tham số.

ES6 - for/of

Bài toán: Tìm số chẵn.

```
const numbersArr = [1, 2, 3, 4, 5, 6];

function findEvenNumbers(numbers) {
  const result = [];

  for (const num of numbers) {
    if (num % 2 === 0) {
      result.push(num);
    }
  }

  return result;
}
console.log(findEvenNumbers(numbersArr));
// => [2, 4, 6]
```

Bài thực hành số 5

Bài thực hành số 5

Sử dụng vòng lặp for/of để làm các bài tập sau:

Bài 1: Viết một hàm nhận tham số là một mảng gồm các chuỗi, hàm kiểm tra xem trong mảng tham số truyền vào có các phần tử nào có chữ “n”. Hàm trả về mảng có các phần tử đó.

Bài 2: Viết một hàm nhận tham số là một mảng gồm các chữ số, hàm trả về giá trị tổng của các phần tử có trong mảng tham số truyền vào.

Bài 3: Viết một hàm nhận 2 tham số, tham số thứ nhất là một mảng gồm các giá trị số hoặc chuỗi, tham số thứ hai là một giá trị số hoặc chuỗi, hàm kiểm tra xem trong mảng tham số có phần tử nào có giá trị bằng với tham số thứ 2 hay không. Nếu có trả về true, không có trả về false.

Web API.
setTimeOut, setInterval

Web API là gì?

- API là viết tắt của cụm từ Application Programming Interface tiếng Việt là Giao diện lập trình ứng dụng.
- Web API là các phương thức có sẵn cung cấp bởi trình duyệt để các ứng dụng web có thể trao đổi thông tin dữ liệu với môi trường bên ngoài và nhiều tiện ích khác như:
    - Kết nối với Back End Server để trao đổi thông tin.
    - Gửi thông báo đẩy (push notification) cho người dùng.
    - Gửi tọa độ vị trí hiện tại.
    - Hẹn giờ thực thi code.
    - Lưu trữ dữ liệu trên máy người dùng (Local storage, Cookie, Session, IndexedDB)
    - Nhiều tiện ích khác (DOM, console, fullscreen, upload file...)

setTimeOut

- **setTimeout** nhận 2 tham số, tham số thứ nhất là một hàm, tham số thứ hai là giá trị số đại diện cho thời gian tính theo mili giây (1000 mili giây bằng 1 giây).
- **setTimeout** sẽ gọi hàm ở tham số thứ nhất sau một khoảng thời gian quy định ở tham số thứ hai.

```
setTimeout(() => {
  console.log("Show after 3 seconds");
}, 3000);

// // Sau 3 giây cửa sổ console in ra giá trị:
// // "Show after 3 seconds"
```

setInterval

- **setInterval** nhận 2 tham số, tham số thứ nhất là một hàm, tham số thứ hai là giá trị số đại diện cho thời gian tính theo mili giây (1000 mili giây bằng 1 giây).
- **setInterval** sẽ liên tục gọi hàm ở tham số thứ nhất sau mỗi khoảng thời gian bằng giá trị quy định ở tham số thứ hai.

```
let seconds = 1;

setInterval(() => {
  console.log(`${seconds} ${seconds === 1 ? "second" : "seconds"} past`);
  seconds++;
}, 1000);

// Sau mỗi giây cửa sổ console in lần lượt các giá trị:
// "1 second past"
// "2 seconds past"
// "3 seconds past"
// ...
```

Clear setTimeout, clearInterval

Bài toán tương tự ví dụ trước. Tuy nhiên, ở ví dụ này cửa sổ console sẽ chỉ in đến giây thứ 5.

```
let seconds = 1;
const interval = setInterval(() => {
  if (seconds === 5) {
    clearInterval(interval);
  }

  console.log(`${seconds} ${seconds === 1 ? "second" : "seconds"} past`);
  seconds++;
}, 1000);
```

Hàm **clearInterval** nhận tham số là phần setInterval cần bỏ

Clear setTimeout làm tương tự như trên với hàm **clearTimeout**, trường hợp cần clear setTimeout thường áp dụng trong lập trình bất đồng bộ, phổ biến là kỹ thuật debouncing.

Bài thực hành số 6

Bài thực hành số 6

Bài 1: Viết một hàm nhận 2 tham số, tham số 1 là một chuỗi và tham số 2 là một số. Sau khi gọi hàm một khoảng thời gian với số giây bằng với tham số thứ 2, hàm in ra cửa sổ console với nội dung chuỗi của tham số 1.

Bài 2: Viết một hàm nhận tham số là một mảng gồm các giá trị thuộc kiểu chữ hoặc số. Sau 1 giây hàm in ra cửa sổ console lần lượt từng giá trị trong mảng, sau khi in xong giá trị cuối cùng thì kết thúc.

Bài 3: Viết một hàm nhận tham số là một số (tối thiểu là 5). Sau mỗi giây, hàm làm việc sau:

- Tạo ra một số nguyên ngẫu nhiên trong khoảng từ 1 cho đến giá trị của tham số.
- In ra cửa sổ console giá trị hiện tại của số ngẫu nhiên theo dạng: “Lần 1: 2”, “Lần 2: 5”...
- Khi in đến giá trị ngẫu nhiên bằng với tham số truyền vào thì dừng lại.

Bài 4 (Nâng cao): Tạo đồng hồ đếm ngược. Viết một hàm nhận giá trị là số đại diện cho số giây (Không quá 3600). Mỗi giây hàm in ra cửa sổ console một chuỗi theo dạng “12:05” đại diện cho số phút và giây. Đếm ngược cho đến khi hết thời gian và in ra cửa sổ console dòng chữ “KẾT THÚC”.

Kiểu dữ liệu thời gian (Date)

Làm quen với Date

Khai báo giá trị Date:

```
const present = new Date(); // Trả về giá trị Date ở hiện tại.
// present === "Fri Oct 21 2022 16:58:01 GMT+0700 (Giờ Đông Dương)"

const future = new Date(2050, 5, 19, 15, 23); // Trả về giá trị Date cố định
// future === "Sun Jun 19 2050 15:23:00 GMT+0700 (Giờ Đông Dương)"
```

Class Date nhận các tham số là:

- Tham số 1: Số năm.
- Tham số 2: Số tháng (0 - 11).
- Tham số 3: Số ngày (1 - 31).
- Tham số 4: Số giờ (0 - 23).
- Tham số 5: Số phút (0 - 59).
- Tham số 6: Số giây (0 - 59).

Chú ý: *Nếu tham số nhập vào vượt quá khoảng trên giá trị thời gian sẽ cộng thêm khoảng thời gian chênh lệch bị vượt quá.*

Làm quen với Date

Lấy dữ liệu từ giá trị Date:

```
const date = new Date(2022, 11, 24, 18, 30, 56);
date; // "Sat Dec 24 2022 18:30:56 GMT+0700 (Giờ Đông Dương)"
date.getFullYear(); // 2022
date.getMonth(); // 11 (Tháng 12)
date.getDate(); // 24
date.getDay(); // 6 (Thứ Bảy)
date.getHours(); // 18
date.getMinutes(); // 30
date.getSeconds(); // 30
date.toISOString(); // "2022-12-24T11:30:56.000Z"
date.getTime(); // 1671881456000 - Số mili giây trôi qua kể từ Unix Time (01/01/1970)
Date.now(); // 1666350266686 - Số mili giây trôi qua kể từ Unix Time đến nay
```

Bài thực hành số 7

Bài thực hành số 8

Bài 1: Viết một hàm nhận tham số là một object gồm các key: day, month, year. Với thông tin thời gian nhận được từ tham số, hàm trả về kết quả ngày đó là ngày nào trong tuần (Monday, Tuesday...).

Bài 2: Viết một hàm nhận tham số là một chuỗi có vai trò làm ký tự ngăn cách, ví dụ “-”, “/”... Hàm trả về thời gian hiện tại (ngày, tháng năm) với ký tự ngăn cách là tham số truyền vào. Ví dụ: “24/12/2022”, “24-12-2022”...

Bài 3: Viết một hàm tính thời điểm hiện tại đến giáng sinh năm 2022 là bao nhiêu ngày?

Bài 4: Viết một hàm nhận tham số tương tự bài 1 là ngày tháng năm sinh của một người, hàm trả về số tuổi hiện tại của người đó (Đúng theo tuổi khai sinh).

Bài 5 (Nâng cao): Viết một hàm nhận kết quả là một số lớn hơn 2022. Hàm kiểm tra từ hiện tại cho đến năm có giá trị bằng tham số truyền vào có những năm nhuận nào, trả về một mảng gồm các năm đó.

Chuẩn hóa dữ liệu thời gian, giá trị số

Chuẩn hóa dữ liệu thời gian

```
const date = new Date(2022, 11, 24, 8, 30);

new Intl.DateTimeFormat("vi-VN").format(date);
// => "24/12/2022"
new Intl.DateTimeFormat("vi-VN", { dateStyle: "full" }).format(date);
// => "Thứ Bảy, 24 tháng 12, 2022"
new Intl.DateTimeFormat("vi-VN", { timeStyle: "short" }).format(date);
// => "08:30"
new Intl.DateTimeFormat("vi-VN", { dayPeriod: "long" }).format(date);
// => "sáng"
new Intl.DateTimeFormat("vi-VN", {
  weekday: "narrow",
  day: "2-digit",
  month: "2-digit",
  year: "2-digit",
}).format(date);
// => "T7, 24/12/22"
```

Tài liệu đầu đủ: *https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Intl/DateTimeFormat/DateTimeFormat*

Chuẩn hóa giá trị số

```
new Intl.NumberFormat("vi-VN").format(123456.789);
// => "123.456,789"
new Intl.NumberFormat("vi-VN", {
  style: "currency",
  currency: "VND",
}).format(123456.789);
// => "123.457 ₫"
new Intl.NumberFormat("vi-VN", {
  style: "unit",
  unit: "celsius",
}).format(37);
// => "37°C"
new Intl.NumberFormat("vi-VN", {
  style: "percent",
  unit: "celsius",
}).format(0.5);
// => "50%"
```

Tài liệu đầu đủ: *https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Intl/NumberFormat/NumberFormat*

Unit list: *https://tc39.es/proposal-unified-intl-numberformat/section6/locales-currencies-tz_proposed_out.html#sec-issanctionedsimpleunitidentifier*

Hoàn thành JavaScript – Buổi 4

Good job!