TW-05 TEAM LEAD VERSION

TEAMWORK

CLARUSWAY
WAY TO REINVENT YOURSELF

Meeting Agenda

▶ Icebreaking

▶ Questions

▶ Interview Questions

▶ Coffee Break

▶ Coding Challenge

▶ Video of the week

▶ Retro meeting

▶ Case study / project

Teamwork Schedule

Ice-breaking   **10m**

- Personal Questions (Stay at home & Corona, Study Environment, Kids etc.)
- Any challenges (Classes, Coding, studying, etc.)
- Ask how they're studying, give personal advice.
- Remind that practice makes perfect.

Workshop Activities (Tuesday)   **90m**

1. Capital Letters

- The purpose of this coding challenge is to capitalize the first letter of each word in the given long text.

Sample Input :

```
There are many variations of passages of Lorem Ipsum available, but the majority
have suffered alteration in some form, by injected humour, or randomised words
which don't look even slightly believable. If you are going to use a passage of
Lorem Ipsum, you need to be sure there isn't anything embarrassing hidden in the
middle of text. All the Lorem Ipsum generators on the Internet tend to repeat
predefined chunks as necessary, making this the first true generator on the
Internet. It uses a dictionary of over 200 Latin words, combined with a handful of
model sentence structures, to generate Lorem Ipsum which looks reasonable. The
generated Lorem Ipsum is therefore always free from repetition, injected humour, or
non-characteristic words etc.
```

Expected Output :

```
There Are Many Variations Of Passages Of Lorem Ipsum Available, But The Majority
Have Suffered Alteration In Some Form, By Injected Humour, Or Randomised Words
which Don't Look Even Slightly Believable. If You Are Going To Use a Passage Of
Lorem Ipsum, You Need To Be Sure There Isn't Anything Embarrassing Hidden In The
Middle Of Text. All The Lorem Ipsum Generators On The Internet Tend To Repeat
Predefined Chunks As Necessary, Making This The First True Generator On The
Internet. It Uses A Dictionary Of Over 200 Latin Words, Combined With A Handful Of
Model Sentence Structures, To Generate Lorem Ipsum Which Looks Reasonable. The
Generated Lorem Ipsum Is Therefore Always Free From Repetition, Injected Humour, Or
non-characteristic Words Etc.
```

> Solution :

```javascript
function capitalizeWords(text) {
  // Split the text into words based on spaces
  const words = text.split(' ');

  // Capitalize the first letter of each word
  const capitalizedWords = words.map(word => {
    if (word.length > 0) {
      return word[0].toUpperCase() + word.slice(1);
    } else {
      return ''; // Handle empty words (e.g., multiple spaces)
    }
  });

  // Join the capitalized words back into a single string
  const capitalizedText = capitalizedWords.join(' ');

  return capitalizedText;
}

const inputText = `There are many variations of passages of Lorem Ipsum available,
but the majority have suffered alteration in some form, by injected humour, or
randomised words which don't look even slightly believable. If you are going to use
a passage of Lorem Ipsum, you need to be sure there isn't anything embarrassing
hidden in the middle of text. All the Lorem Ipsum generators on the Internet tend
to repeat predefined chunks as necessary, making this the first true generator on
the Internet. It uses a dictionary of over 200 Latin words, combined with a handful
of model sentence structures, to generate Lorem Ipsum which looks reasonable. The
generated Lorem Ipsum is therefore always free from repetition, injected humour, or
non-characteristic words etc.`;

const result = capitalizeWords(inputText);
console.log(result);
```

2. Filtering and Transforming Data

- You have an array of objects representing books, and each object has properties like "title," "author," and "rating." Your task is to create a JavaScript function that filters the books based on a specific rating threshold and then transforms the filtered books into a new array containing only the "title" and "author" properties of the selected books.

For Example :

```javascript
const books = [
  { title: 'The Catcher in the Rye', author: 'J.D. Salinger', rating: 4.2 },
  { title: 'To Kill a Mockingbird', author: 'Harper Lee', rating: 4.6 },
  { title: '1984', author: 'George Orwell', rating: 4.0 },
  { title: 'The Great Gatsby', author: 'F. Scott Fitzgerald', rating: 4.7 },
```

```javascript
  { title: 'Pride and Prejudice', author: 'Jane Austen', rating: 4.5 },
];

const minRating = 4.5;

const filteredAndTransformedBooks = filterAndTransformBooks(books, minRating);

console.log(filteredAndTransformedBooks);


//Expected Output
[
  { title: 'To Kill a Mockingbird', author: 'Harper Lee' },
  { title: 'The Great Gatsby', author: 'F. Scott Fitzgerald' },
  { title: 'Pride and Prejudice', author: 'Jane Austen' }
]
```

- You can use this sample array:

```javascript
const books = [
  { title: 'The Catcher in the Rye', author: 'J.D. Salinger', rating: 4.2 },
  { title: 'To Kill a Mockingbird', author: 'Harper Lee', rating: 4.6 },
  { title: '1984', author: 'George Orwell', rating: 4.0 },
  { title: 'The Great Gatsby', author: 'F. Scott Fitzgerald', rating: 4.7 },
  { title: 'Pride and Prejudice', author: 'Jane Austen', rating: 4.5 },
  { title: 'Moby-Dick', author: 'Herman Melville', rating: 4.1 },
  { title: 'The Lord of the Rings', author: 'J.R.R. Tolkien', rating: 4.8 },
  { title: 'The Hobbit', author: 'J.R.R. Tolkien', rating: 4.3 },
  { title: 'Harry Potter and the Sorcerer\'s Stone', author: 'J.K. Rowling',
rating: 4.7 },
  { title: 'Brave New World', author: 'Aldous Huxley', rating: 4.0 },
  { title: 'The Shining', author: 'Stephen King', rating: 4.2 },
  { title: 'The Da Vinci Code', author: 'Dan Brown', rating: 3.9 },
  { title: 'The Alchemist', author: 'Paulo Coelho', rating: 4.4 },
  { title: 'War and Peace', author: 'Leo Tolstoy', rating: 4.9 },
  { title: 'Crime and Punishment', author: 'Fyodor Dostoevsky', rating: 4.5 },
  { title: 'The Odyssey', author: 'Homer', rating: 4.6 },
  { title: 'Frankenstein', author: 'Mary Shelley', rating: 4.2 },
  { title: 'The Hunger Games', author: 'Suzanne Collins', rating: 4.1 },
  { title: 'The Road', author: 'Cormac McCarthy', rating: 4.3 },
  { title: 'The Hitchhiker\'s Guide to the Galaxy', author: 'Douglas Adams',
rating: 4.7 },
  { title: 'The Count of Monte Cristo', author: 'Alexandre Dumas', rating: 4.8 },
];
```

Solution :

```javascript
function filterBooksByRating(books, minRating) {
    return books
        .filter(book => book.rating >= minRating)
```

```
      .map(book => ({ title: book.title, author: book.author }));
  }

  const books = [
    { title: 'The Catcher in the Rye', author: 'J.D. Salinger', rating: 4.2 },
    { title: 'To Kill a Mockingbird', author: 'Harper Lee', rating: 4.6 },
    { title: '1984', author: 'George Orwell', rating: 4.0 },
    { title: 'The Great Gatsby', author: 'F. Scott Fitzgerald', rating: 4.7 },
    { title: 'Pride and Prejudice', author: 'Jane Austen', rating: 4.5 },
  ];

  const minRating = 4.5;
  const filteredBooks = filterBooksByRating(books, minRating);

  console.log(filteredBooks);
```

Team Work Activities (Friday)   90m

Ask Questions   20m

1. How do you access a property of an object in JavaScript?

A. By using square brackets
B. By using the dot notation
C. By using parentheses
D. A and B correct

Answer: D

2. How do you check if a property exists in an object in JavaScript?

A. By using the exist keyword
B. By using the contains keyword
C. By using the hasOwnProperty method
D. By using the isProperty method

Answer: C

```
const person = {
  name: "Bob"
};

console.log(person.hasOwnProperty("name")); // true - Using hasOwnProperty
console.log("age" in person); // false - Using the 'in' operator
```

3. How do you delete a property from an object in JavaScript

A. By using the `delete` keyword

B. By using the `remove` keyword

C. By setting the property value to null

D. By assigning an empty string to the property

Answer: A

```
const person = {
 name: "John",
 age: 30
};

delete person.age; // Removing the 'age' property
```

4. How do you add a new property to an existing object in JavaScript

A. By using the `add` keyword

B. By using the `insert` keyword

C. By using the `update` keyword

D. By assigning a value to a new key

Answer: D

5. What is an object in JavaScript?

A. A function

B. A data tool

C. A data structure

D. An array

Answer: C

6. How can you create an empty object in JavaScript?

A. `emptyObject = {};`

B. `emptyObject = new Empty();`

C. `emptyObject = Object.empty();`

D. `emptyObject = new Object;`

Answer: A

7. How do you clone an object in JavaScript?

A. Use the Object.clone() method
B. Use the Object.assign() method or the spread operator (...)
C. Use the Object.copy() method
D. Use the Object.duplicate() method

Answer: B

By Object.assign() :

```javascript
const originalObject = { name: "John", age: 30 };

// Clone the original object using Object.assign()
const clonedObject = Object.assign({}, originalObject);

// Now, 'clonedObject' is a separate copy of 'originalObject'
console.log(clonedObject); // { name: 'John', age: 30 }
```

By spread operator :

```javascript
const originalObject = { name: "John", age: 30 };

// Clone the original object using the spread operator
const clonedObject = { ...originalObject };

// Now, 'clonedObject' is a separate copy of 'originalObject'
console.log(clonedObject); // { name: 'John', age: 30 }
```

8. What is object destructuring in JavaScript?

A. A way to create objects from strings
B. A way to concatenate objects
C. A way to merge objects
D. A way to extract properties from an object and assign them to variables

Answer: D

9. How do you swap the values of two variables without using a temporary variable using array destructuring?

A. const a = b; const b = a
B. const [a, b] = [a, b];
C. const [a, b] = [b, a];
D. const [b; a] = [a; b];

Answer: C

10. What happens if you try to destructure an array with more variables than there are elements in the array?

A. Extra variables are assigned undefined
B. An error is thrown
C. The array is automatically resized
D. Only the first few variables are assigned values

Answer: A

11. What does the rest element (...) do in array destructuring?

A. It spreads elements into multiple arrays
B. It gathers remaining elements into an array
C. It removes elements from the array
D. It reverses the order of elements in the array

Answer: B

```javascript
// Example 1: Collecting remaining elements
const numbers = [1, 2, 3, 4, 5];

// Using rest element to collect the remaining elements
const [first, second, ...rest] = numbers;

console.log(first); // 1
console.log(second); // 2
console.log(rest); // [3, 4, 5]
```

12. What is JSON (JavaScript Object Notation)?

A. A lightweight data interchange format
B. A JavaScript method for creating objects
C. A way to define variables in JavaScript
D. A JavaScript library for animations

Answer: A : JSON is a lightweight data interchange format that is often used to transmit data between a server and a web application. It is based on a subset of JavaScript object literal notation.

```json
{
  "name": "John Doe",
  "age": 30,
  "city": "New York",
  "isStudent": false,
  "hobbies": ["reading", "hiking", "cooking"],
  "address": {
    "street": "123 Main St",
```

```
      "zipcode": "10001"
  }
}
```

13. Write a code for get *sum of every positive element* in given array

```javascript
const input = [1, -4, 12, 0, -3, 30, 42, -150];

input.filter(function (num) {
    return num > 0;
  }).reduce(function (accumulator, currentValue) {
    return accumulator + currentValue;
  }, 0);

// or written with Arrow function
input
  .filter((num) => num > 0)
  .reduce((accumulator, currentValue) => accumulator + currentValue, 0);

//output: 85
```

14. Write a code for abbreviate the given name and return the name initials.

```javascript
const input = "John Ronald Reuel Tolkien"

input.split(" ").map(function (word) {
    return word[0];
  }).join("");

// or written with Arrow function
input
  .split(" ")
  .map((word) => word[0])
  .join("");


//output: JRRT
```

15. If you want to square each element of an array and return a new array with the squared values, which method would you use?

A. reduce
B. filter
C. map
D. All of the above

Answer: C

```
const numbers = [1, 2, 3, 4, 5];
const squaredNumbers = numbers.map((num) => num ** 2);
```

16. If you want to find the sum of all even numbers in an array, which method would you use?

A. map
B. reduce
C. filter
D. forEach

Answer: B

```
const numbers = [1, 2, 3, 4, 5];
const sumOfEvens = numbers.reduce((acc, num) => (num % 2 === 0 ? acc + num : acc),
0);
```

17. Write a code get each array elements length to a new array with map() method.

```
const names = ["Alice", "Bob", "Charlie"];
const nameLengths = names.map((name) => name.length);

console.log(nameLengths) // [5, 3, 7]
```

18. Write a code get each array elements capitalized with map() method.

```
const words = ["apple", "banana", "cherry"];
const capitalizedWords = words.map((word) => word.toUpperCase());

console.log(capitalizedWords) //['APPLE', 'BANANA', 'CHERRY']
```

19. Which class indicates red colored border in bootsrap?

A. border warning
B. border border-danger
C. border border-warning
D. border-danger

Answer: B

20. Which class indicates lowercased text in bootsrap?

A. lowercase
B. text-lowercased
C. text-lowercase
D. txt-lowercased

Answer: C

21. Which class adds zebra-stripes to a table?

A. .table-zebra
B. .table-bordered
C. .table-striped
D. .even and .odd

Answer: C

22. What is Bootstrap?

A. A popular JavaScript framework
B. A server-side programming language
C. An advanced database management system
D. A responsive front-end web development framework

Answer: D

Interview Questions   15m

1. What is the difference between Object.keys(), Object.values(), and Object.entries()?

Answer : Object.keys() returns property names, Object.values() returns property values, and Object.entries() returns key-value pairs

2. What is the Object.freeze() method used for?

Answer :To make an object immutable, preventing changes to its properties

3. What is constructor functions in JavaScript?

Answer :In JavaScript, constructor functions are special functions used to create and initialize objects. They are used as templates or blueprints for creating multiple objects of the same type, each with its own unique property values but sharing common methods. Constructor functions are typically invoked with the new keyword.

Here's how you define and use a constructor function:

```javascript
// Constructor function for creating Person objects
function Person(name, age) {
  this.name = name;
  this.age = age;
}

// Creating instances (objects) using the constructor function
const person1 = new Person("Alice", 25);
const person2 = new Person("Bob", 30);

console.log(person1); // { name: 'Alice', age: 25 }
console.log(person2); // { name: 'Bob', age: 30 }
```

In the example above:

- We define a constructor function `Person` that takes two parameters, `name` and `age`. Inside the constructor function, `this` refers to the newly created object.
- We create two instances (objects) of the `Person` constructor using the `new` keyword, passing specific values for the `name` and `age` properties.
- Each instance, `person1` and `person2`, has its own set of `name` and `age` properties, initialized with the values provided during object creation. Constructor functions are useful for creating multiple objects that share a common structure and behavior. They allow you to encapsulate the object's initialization logic and methods within a reusable function, promoting code organization and reducing redundancy. Additionally, constructor functions can be used in conjunction with prototypes to share methods across all instances, which can help conserve memory and optimize performance when dealing with many objects of the same type.

4. Explain `reduce()` method in Javascript

Answer : .reduce() runs a callback for every array element just like .map() does. The only difference is that reduce() passes the result of this accumulator from one array element to the other. Some built-in reduce() functions are: Array.prototype.reduce(), and the reduceRight() method which are used to apply functions against accumulators from 1. Left to right and 2. Right to left respectively.

The accumulator either contains the initial value or the return value from the previous call. The accumulator could be any string, integer, object, etc. It is the net result of the function. The present value of the accumulator

is simply the element that is being worked against.

Accumulators must be passed when .reduce() is being called.

Coding Challenge   **15m**

1. High Priced Product Categories

- You are given an array of objects representing a collection of products, each with a name, price, and category. Your task is to use map, filter, and reduce to calculate the average price of products in each category, and then return an array of objects containing only the categories that have an average price above 50.

- Sample input :

```js
const products = [
  { name: "Product 1", price: 20, category: "Electronics" },
  { name: "Product 2", price: 30, category: "Clothes" },
  { name: "Product 3", price: 40, category: "Electronics" },
  { name: "Product 4", price: 50, category: "Clothes" },
  { name: "Product 5", price: 60, category: "Clothes" },
  { name: "Product 6", price: 70, category: "Electronics" },
  { name: "Product 7", price: 80, category: "Clothes" },
  { name: "Product 8", price: 90, category: "Electronics" },
];
```

- Expected outcome :

```js
[
  { category: 'Clothes', average: 55 },
  { category: 'Electronics', average: 55 }
]
```

> Solution :

```js
const products = [
  { name: "Product 1", price: 20, category: "Electronics" },
  { name: "Product 2", price: 30, category: "Clothes" },
  { name: "Product 3", price: 40, category: "Electronics" },
  { name: "Product 4", price: 50, category: "Clothes" },
  { name: "Product 5", price: 60, category: "Clothes" },
```

```
    { name: "Product 6", price: 70, category: "Electronics" },
    { name: "Product 7", price: 80, category: "Clothes" },
    { name: "Product 8", price: 90, category: "Electronics" },
];

/* Use map to create an object with category as the key
and an array of products as the value */
const productsByCategory = products.reduce((acc, product) => {
  const category = product.category;
  if (!acc[category]) {
    acc[category] = [];
  }
  acc[category].push(product);
  return acc;
}, {});

// Use map to calculate the average price for each category
const avgPriceByCategory = Object.keys(productsByCategory).map(category => {
  const sum = productsByCategory[category].reduce((acc, product) => acc +
product.price, 0);
  return { category: category, average: sum / productsByCategory[category].length
};
});

// Use filter to only select categories with an average above a certain threshold
const highPricedCategories = avgPriceByCategory.filter(category => category.average
> 50);
  console.log(highPricedCategories)
```

2. HR VS IT Department

- **Task :** You are given an array of objects representing a collection of employees, each with a name, salary, and department. Your task is to use map, filter, and reduce to calculate the average salary for each department and then return an array of objects containing only the departments that have an average salary above 65000.

- Sample input :

```
const employees = [
  { name: "John", salary: 50000, department: "IT" },
  { name: "Jane", salary: 60000, department: "HR" },
  { name: "Bob", salary: 55000, department: "IT" },
  { name: "Sophie", salary: 75000, department: "HR" },
  { name: "Mike", salary: 65000, department: "IT" },
  { name: "Emily", salary: 80000, department: "HR" },
  { name: "David", salary: 70000, department: "IT" },
];
```

- Expected outcome :

```
[
  { department: 'HR', average: 71666 }
]
```

Solution :

```javascript
const employees = [
  { name: "John", salary: 50000, department: "IT" },
  { name: "Jane", salary: 60000, department: "HR" },
  { name: "Bob", salary: 55000, department: "IT" },
  { name: "Sophie", salary: 75000, department: "HR" },
  { name: "Mike", salary: 65000, department: "IT" },
  { name: "Emily", salary: 80000, department: "HR" },
  { name: "David", salary: 70000, department: "IT" },
];



/* Use reduce to create an object with department as the key
and an array of employee objects as the value */
const employeesByDepartment = employees.reduce((acc, employee) => {
  const department = employee.department;
  if (!acc[department]) {
    acc[department] = [];
  }
  acc[department].push(employee);
  return acc;
}, {});

// Use map to calculate the average salary for each department
const avgSalaryByDepartment = Object.keys(employeesByDepartment).map(department =>
{
  const sum = employeesByDepartment[department].reduce((acc, employee) => acc +
employee.salary, 0);
  return { department: department, average: sum /
employeesByDepartment[department].length };
});

// Use filter to only select departments with an average above a certain threshold
const highPaidDepartments = avgSalaryByDepartment.filter(department =>
department.average > 65000);
console.log(highPaidDepartments)
```

☕

Coffee Break   **10m**

☕

Video of the Week   10m

- [Object Destructuring in Javascript](#)

Retro Meeting on a personal and team level   10m

Ask the questions below:

- What went well?
- What could be improved?
- What will we commit to do better in the next week?

Closing   5m

- Next week's plan

- QA Session