Collections

HyperionDev

Muhammad Zahir Junejo

Lecture - Housekeeping

- ❑ The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly.
    - ❑ Please review Code of Conduct (in Student Undertaking Agreement) if unsure
- ❑ No question is daft or silly - **ask them!**
- ❑ Q&A session at the end of the lesson, should you wish to ask any follow-up questions.
- ❑ Should you have any questions after the lecture, please schedule a mentor session.
- ❑ For all non-academic questions, please submit a query: www.hyperiondev.com/support

Lecture Objectives

1. What collections are and their significance in programming.
2. The concept of arrays as ordered collections.
3. How to manipulate arrays, access their elements, and use negative indices.
4. Treating strings as collections of characters.

What are Collections?

- ❏ Collections are data structures used to store and organize multiple items of data.
- ❏ They enable us to work with multiple values as a single unit.
- ❏ Collections allow for efficient data manipulation and retrieval.

Arrays

- Arrays are ordered collections of elements.
- Elements can be of different data types, such as numbers, strings, objects, or even other arrays.
- Each element in an array is assigned a numeric index (starting from 0).
- Example:

```
const colors = ['red', 'green', 'blue'];

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

Manipulating Arrays

- Adding Elements:
  - Use array_name.push(“zahir”) to add elements to the end of an array.
  - Use array_name.unshift(“zahir”) to add elements to the beginning.
- Removing Elements:
  - Use array_name.pop() to remove the last element.
  - Use array_name.shift() to remove the first element.

```
colors.push('yellow');

numbers.pop();
```

Accessing Array Elements

- Use the index to access individual elements in an array.
- Arrays are zero-indexed, so the first element is at index 0.
- Use negative indices to count from the end of the array.
- Example:

```
const firstColor = colors[0];

const lastColor = colors[colors.length - 1];
```

Maps

- Maps are collections that store key-value pairs.
- Keys can be of any data type (strings, numbers, objects).
- Maps maintain the order of insertion.
- Example:

```
const userMap = new Map();

userMap.set('name', 'Alice');

userMap.set('age', 25);
```

Working with Maps

- Adding Data:
  - Use the set() method to add key-value pairs.
- Retrieving Data:
  - Use the get() method to retrieve values based on keys.
- Deleting Data:
  - Use the delete() method to remove key-value pairs.

```
userMap.set('email', 'alice@example.com');

const userName = userMap.get('name');

userMap.delete('age');
```

Strings as Collections

❑ Strings can be treated as collections of characters.

❑ You can access individual characters using indexing.

❑ Iterate through characters using loops or array methods.

❑ Example:

```
const message = 'HelLo, WoLlD!';

const firstChar = message[0];
```

References

- ❑ https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Indexed_collections
- ❑ https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Keyed_collections
- ❑ https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

Questions and Answers

Thank You!