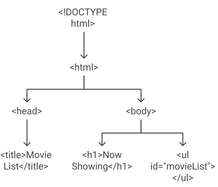
Understanding the DOM in JavaScript

The Document Object Model (DOM) is a programming interface for web documents. It represents the structure of a document as a tree of objects, allowing developers to manipulate the content, structure, and style of a webpage dynamically. In this document, we will explore the DOM using the example of a movie listing application like BookMyShow, focusing on how to render a list of movies.

Basic Structure of the DOM

Consider a simple HTML structure:

```
<!DOCTYPE html>
<html>
<head>
    <title>Movie List</title>
</head>
<body>
    <h1>Now Showing</h1>
    <ul id="movieList"></ul>
</body>
</html>
```



In this example, the DOM tree would look like this:

- Document
  - html
    - head
      - title
    - body
      - h1
      - ul [id="movieList"]

Rendering the Movie List

Let's say we have a list of movies that we want to display on our webpage. We can use JavaScript to manipulate the DOM and render this list dynamically.

Step 1: Sample Movie Data

First, we need some sample movie data. This could come from an API or a static array for simplicity:

```
const movies = [
    { title: "Movie 1", year: 2023 },
    { title: "Movie 2", year: 2022 },
    { title: "Movie 3", year: 2021 },
];
```

Step 2: Accessing the DOM

Next, we will access the **ul** element where we want to render our movie list. We can do this using **document.getElementById()**:

```
const movieListElement = document.getElementById('movieList');
```

Step 3: Rendering the Movies

Now, we can loop through our movie data and create **li** elements for each movie, appending them to the **ul**:

```
movies.forEach(movie => {
    const li = document.createElement('li'); // Create a new list item
    li.textContent = `${movie.title} (${movie.year})`; // Set the text content
    movieListElement.appendChild(li); // Append the list item to the movie list
});
```

Complete Example

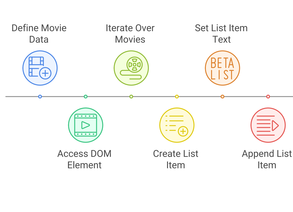
Here's how the complete JavaScript code would look:

```
<!DOCTYPE html>
<html>
<head>
    <title>Movie List</title>
</head>
<body>
    <h1>Now Showing</h1>
    <ul id="movieList"></ul>

    <script>
        const movies = [
            { title: "Movie 1", year: 2023 },
            { title: "Movie 2", year: 2022 },
            { title: "Movie 3", year: 2021 },
        ];

        const movieListElement = document.getElementById('movieList');

        movies.forEach(movie => {
            const li = document.createElement('li');
            li.textContent = `${movie.title} (${movie.year})`;
            movieListElement.appendChild(li);
        });
    </script>
</body>
</html>
```



Explanation of the Code

1. **HTML Structure**: We have a simple HTML structure with a heading and an empty unordered list (**ul**) where the movies will be displayed.
2. **JavaScript Logic**:
   - We define an array of movie objects.
   - We access the **ul** element using its ID.
   - We loop through the array of movies, creating a new **li** element for each movie, setting its text content, and appending it to the **ul**.

Conclusion

The DOM allows us to dynamically manipulate the content of a webpage using JavaScript. In our example, we demonstrated how to render a list of movies in a simple movie listing application similar to BookMyShow. By understanding the DOM and how to interact with it, you can create interactive and dynamic web applications.