JavaScript Basics

A Hello World App in JavaScript

Creating a "Hello World" app is a traditional first step when learning a new programming language. In JavaScript, this involves writing a simple script that outputs "Hello World" to the user.

Example:
```html
<!DOCTYPE html>
<html>
<head>
  <title>Hello World</title>
</head>
<body>
  <script>
    document.write("Hello World!");
  </script>
</body>
</html>
```
In this example, the `document.write` method writes "Hello World!" to the HTML document.

Communicating with End Users from JavaScript

JavaScript provides several methods to communicate with users, such as `alert`, `prompt`, and `confirm`.

- **alert**: Displays a message to the user.
  ```javascript
  alert("Hello, World!");
  ```

- **prompt**: Asks the user to input some information.
  ```javascript
  let name = prompt("What is your name?");
  alert("Hello, " + name + "!");
  ```

- **confirm**: Asks the user to confirm something.
  ```javascript
  ```

```
let isSure = confirm("Are you sure?");
if (isSure) {
   alert("You are sure!");
} else {
   alert("You are not sure.");
}
```

Separating HTML and JavaScript Sources

It's a good practice to separate HTML and JavaScript for better maintainability. JavaScript can be placed in an external file and linked to the HTML file.

Example:
index.html
```html
<!DOCTYPE html>
<html>
<head>
   <title>Hello World</title>
   <script src="script.js"></script>
</head>
<body>
</body>
</html>
```

script.js
```javascript
document.write("Hello World!");
```

Accessing the DOM from JavaScript

The Document Object Model (DOM) represents the structure of a web page. JavaScript can be used to access and manipulate the DOM.

Example:
```html
<!DOCTYPE html>
<html>
<head>
   <title>DOM Example</title>
```

```
</head>
<body>
  <div id="myDiv">Hello</div>
  <script>
    let myDiv = document.getElementById("myDiv");
    myDiv.innerHTML = "Hello, World!";
  </script>
</body>
</html>
```
In this example, `document.getElementById` is used to access the `div` element, and `innerHTML` is used to change its content.

The Use of Strict Mode

Strict mode is a way to opt into a restricted variant of JavaScript, which can help catch common coding errors and "unsafe" actions.

Example:
```javascript
"use strict";
x = 3.14; // This will cause an error because x is not declared
```

Variable Declarations: var, let and const

JavaScript provides three ways to declare variables: `var`, `let`, and `const`.

- **var**: Function-scoped variable.
  ```javascript
  var x = 10;
  ```

- **let**: Block-scoped variable.
  ```javascript
  let y = 20;
  ```

- **const**: Block-scoped constant (cannot be reassigned).
  ```javascript
  const z = 30;
  ```

Empty Values in JavaScript: undefined and null

JavaScript has two types of empty values: `undefined` and `null`.

- **undefined**: A variable that has been declared but not assigned a value.
 ```javascript
 let a;
 console.log(a); // undefined
 ```

- **null**: An assignment value that represents no value.
 ```javascript
 let b = null;
 console.log(b); // null
 ```

User Interactions Using alert, prompt, and confirm

JavaScript allows interaction with the user through `alert`, `prompt`, and `confirm` functions.

- **alert**: Displays a message.
 ```javascript
 alert("This is an alert!");
 ```

- **prompt**: Asks the user for input.
 ```javascript
 let userInput = prompt("Please enter something:");
 console.log(userInput);
 ```

- **confirm**: Asks for user confirmation.
 ```javascript
 let userConfirmed = confirm("Do you agree?");
 console.log(userConfirmed); // true if OK, false if Cancel
 ```

These methods are useful for simple interactions and debugging but are less commonly used in modern web applications, which often use more sophisticated methods for user interaction.