What is Strict Mode?

Strict Mode is a feature in JavaScript that helps catch common coding mistakes and prevent the use of unsafe actions.

Enabling Strict Mode

Strict Mode can be enabled by adding `"use strict";` at the beginning of a script or function.

1. Enabling Strict Mode for an Entire Script

```
"use strict";
console.log("Strict mode is enabled");
```

2. Enabling Strict Mode for a Function

```
function myFunction() {
    "use strict";
    let x = 10;
    console.log(x);
}
myFunction();
```

Benefits of Strict Mode

1. **Prevents accidental global variables:**

    ```
    "use strict";
    x = 10; // Error: x is not defined
    ```

    Without Strict Mode, `x` would be created as a global variable.

2. **Disallows duplicate parameter names:**

    ```
    "use strict";
    function sum(a, a) { // Error: Duplicate parameter name not allowed
        return a + a;
    }
    ```

3. **Prevents assigning values to read-only properties:**

```
"use strict";
Object.defineProperty(this, "PI", { value: 3.14, writable: false });
PI = 3.1415; // Error: Cannot assign to read-only property
```

4. **Throws an error for deleting non-deletable properties:**

```
"use strict";
let obj = Object.freeze({ name: "Alice" });
delete obj.name; // Error: Cannot delete property
```

5. **Makes `this` in functions `undefined` instead of the global object:**

```
"use strict";
function showThis() {
    console.log(this); // undefined
}
showThis();
```

6. **Prohibits Octal Syntax:**

```
"use strict";
let num = 010; // Error: Octal literals are not allowed
```

7. **Prevents using reserved keywords for future JavaScript versions:**

```
"use strict";
let public = 10; // Error: Unexpected strict mode reserved word
```

When to Use Strict Mode?

- When writing new JavaScript code, it is recommended to use strict mode to catch potential errors early.
- When working with older code, be careful when enabling strict mode, as it may break some parts of the code.

Checking if Strict Mode is Enabled

```
(function() {
    "use strict";
    console.log(this === undefined); // true
})();
```

By using `"use strict";` , developers can write safer and more reliable JavaScript code.