Complete HTML Guide

1. Let and Const

ES6 introduced `let` and `const` for block-scoped variable declarations.

- `let`: allows you to declare variables that can be reassigned.
- `const`: allows you to declare constants (cannot be reassigned).

Example:

```
let x = 10;
x = 20;

const y = 30;
// y = 40; // Error: Assignment to constant variable.
```

2. Arrow Functions

Arrow functions provide a shorter syntax for writing functions and do not bind their own `this`.

Example:

```
const add = (a, b) => a + b;

console.log(add(2, 3)); // 5
```

3. Template Literals

Template literals allow embedded expressions and multi-line strings using backticks (``).

Example:

```
const name = "John";
console.log(`Hello, ${name}!`);
```

4. Default Parameters

Function parameters can have default values.

Example:

```
function greet(name = "Guest") {
  console.log("Hello, " + name);
}
greet(); // Hello, Guest
```

5. Destructuring Assignment

Allows unpacking values from arrays or properties from objects.

Array Destructuring:

```
const [a, b] = [1, 2];
```

Object Destructuring:

```
const {name, age} = {name: "Alice", age: 25};
```

6. Spread and Rest Operators

Spread (...) expands an array/object. Rest (...) collects values into an array.

Spread Example:

```
const arr1 = [1, 2];
const arr2 = [...arr1, 3, 4];
```

Rest Example:

```
function sum(...numbers) {
  return numbers.reduce((a, b) => a + b);
}
```

7. Object Literals

ES6 allows shorthand syntax for object properties and methods.

Example:

```
const name = "Bob";
const user = {
  name,
  greet() {
    console.log("Hi " + this.name);
  }
};
```

8. Promises

Promises represent the eventual result of an asynchronous operation.

Example:

```
const promise = new Promise((resolve, reject) => {
  setTimeout(() => resolve("Success"), 1000);
});

promise.then(result => console.log(result));
```

9. Classes

ES6 introduced classes as syntactic sugar over prototypes.

Example:

```
class Person {
  constructor(name) {
    this.name = name;
  }
  greet() {
    console.log("Hello " + this.name);
  }
```

```
}

const p = new Person("Tom");
p.greet(); // Hello Tom
```

10. Modules (import/export)

ES6 modules allow code to be split across multiple files.

Export from module:

```
export const PI = 3.14;
```

Import into another file:

```
import { PI } from './math.js';
```

11. Enhanced Loops (for...of)

for...of allows iteration over iterable objects like arrays, strings, etc.

Example:

```
for (let value of [1, 2, 3]) {
  console.log(value);
}
```

12. Map and Set

Map stores key-value pairs. Set stores unique values.

Example:

```
const map = new Map();
map.set('a', 1);

const set = new Set([1, 2, 2, 3]); // set: 1, 2, 3
```

13. Symbols

Symbols are unique and immutable primitive values.

Example:

```
const sym1 = Symbol("id");
const sym2 = Symbol("id");
console.log(sym1 === sym2); // false
```

14. Iterators and Generators

Generators allow defining iterable sequences using function* syntax.

Example:

```
function* count() {
  yield 1;
  yield 2;
  yield 3;
}
```