ChatGPT - ES6 JavaScript Concepts

Here are detailed notes with code examples for ES6 JavaScript concepts:

---

1. Template Strings

Template strings allow embedding expressions and multiline strings using backticks (``).

Features:

- String interpolation using `${expression}`
- Multiline strings without needing escape characters

javascript

```javascript
// String interpolation
const name = "John";
console.log(`Hello, ${name}!`); // Output: Hello, John!

// Multiline strings
const message = `
Hello,
This is a multiline string.
`;
console.log(message);
```

---

2. `Map`

`Map` is a built-in object for key-value pairs. Keys can be any type.

Features:

- Maintains key insertion order
- Provides methods like `set`, `get`, `has`, and `delete`

javascript

```
const map = new Map();
map.set('name', 'Alice');
map.set(42, 'The answer');
console.log(map.get('name')); // Output: Alice
console.log(map.has(42)); // Output: true
```

---

3. `Filter`

Creates a new array containing elements that pass a provided test.

javascript

```
const numbers = [1, 2, 3, 4, 5];
const evenNumbers = numbers.filter(num => num % 2 === 0);
console.log(evenNumbers); // Output: [2, 4]
```

---

4. `Find`

Returns the first element that satisfies a condition, or `undefined` if none do.

javascript

```
const users = [{ id: 1, name: "Alice" }, { id: 2, name: "Bob" }];
const user = users.find(u => u.id === 2);
console.log(user); // Output: { id: 2, name: "Bob" }
```

---

5. `Every` and `Some`

- `Every`: Tests if **all** elements pass a condition.
- `Some`: Tests if **any** element passes a condition.

javascript

```javascript
const nums = [2, 4, 6];

// Every
console.log(nums.every(num => num % 2 === 0)); // Output: true

// Some
console.log(nums.some(num => num > 5)); // Output: true
```

---

6. `Reduce`

Applies a reducer function to each array element, accumulating a single value.

javascript

```javascript
const nums = [1, 2, 3, 4];
const sum = nums.reduce((acc, curr) => acc + curr, 0);
console.log(sum); // Output: 10
```

---

7. Fat Arrow Function

A concise way to write functions. They don't bind their own `this`.

javascript

```javascript
// Basic syntax
const add = (a, b) => a + b;
console.log(add(2, 3)); // Output: 5

// Multiline function
const multiply = (a, b) => {
  const result = a * b;
  return result;
};
console.log(multiply(3, 4)); // Output: 12
```

8. Object Literals

Shorthand for defining objects with properties and methods.

```javascript
const name = "John";
const age = 30;

const person = {
  name,
  age,
  greet() {
    return `Hello, my name is ${this.name}.`;
  },
};
console.log(person.greet()); // Output: Hello, my name is John.
```

---

9. Rest and Spread

Rest: Collects arguments into an array.

```javascript
function sum(...numbers) {
  return numbers.reduce((a, b) => a + b, 0);
}
console.log(sum(1, 2, 3, 4)); // Output: 10
```

Spread: Expands an array or object.

```javascript
const arr = [1, 2, 3];
const newArr = [...arr, 4, 5];
console.log(newArr); // Output: [1, 2, 3, 4, 5]

const obj = { a: 1, b: 2 };
```

```javascript
const newObj = { ...obj, c: 3 };
console.log(newObj); // Output: { a: 1, b: 2, c: 3 }
```

---

10. Classes

ES6 introduced classes for object-oriented programming.

javascript

```javascript
class Person {
  constructor(name, age) {
    this.name = name;
    this.age = age;
  }
  greet() {
    return `Hi, I'm ${this.name}.`;
  }
}

const john = new Person("John", 30);
console.log(john.greet()); // Output: Hi, I'm John.
```

---

11. Destructuring

Allows unpacking values from arrays or objects into variables.

javascript

```javascript
// Array destructuring
const [a, b] = [1, 2];
console.log(a, b); // Output: 1 2

// Object destructuring
const { name, age } = { name: "Alice", age: 25 };
console.log(name, age); // Output: Alice 25
```

12. Modules

Enable importing/exporting functions, objects, or values between files.

Example:

math.js:

```javascript
export const add = (a, b) => a + b;
export const subtract = (a, b) => a - b;
```

app.js:

```javascript
import { add, subtract } from './math.js';
console.log(add(2, 3)); // Output: 5
console.log(subtract(5, 2)); // Output: 3
```

These notes provide a comprehensive guide to ES6 features with examples for better understanding and practical use!