Here's a summary of the ES6+ features you mentioned:

1. **Arrow Functions**:
   - **Syntax**: `(parameters) => { statements }`
   - **Example**:

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

   - **Behavior**: Arrow functions do not have their own `this`, `arguments`, or `super`. This makes them useful in callbacks where you want to preserve the lexical scope of `this`.

2. **Destructuring**:
   - **Object Destructuring**:

```javascript
const person = { name: 'Alice', age: 30 };
const { name, age } = person;
```

   - **Array Destructuring**:

```javascript
const [first, second] = [1, 2, 3];
```

   - **Usage**: Destructuring simplifies the extraction of values from arrays and objects.

3. **Template Literals**:
   - **Syntax**: Use backticks (`` ` ``) instead of quotes.
   - **Example**:

```javascript
const name = 'Bob';
const greeting = `Hello, ${name}!`;
```

   - **Features**: Supports multi-line strings and embedding expressions inside the template.

4. **Classes**:
   - **Syntax**:

```javascript
class Person {
  constructor(name) {
    this.name = name;
  }
  greet() {
    return `Hello, ${this.name}`;
  }
}
```

   - **Usage**: Provides syntactic sugar for creating objects with methods, supporting inheritance and a more intuitive way of working with prototypes.

5. **Modules**:
   - **Exporting**:

```javascript
export const pi = 3.14;
export function add(x, y) { return x + y; }
```

   - **Importing**:

```javascript
import { pi, add } from './math';
```

   - **Usage**: Modules help organize code into reusable, self-contained units, supporting better code structure and reusability.

These features provide a more concise and powerful way to write modern JavaScript.