Comparison operators (==, ===, !=, !==, <, >, <=, >=)

Sure! Let's dive deep into **Comparison Operators** in JavaScript.

🔍 What Are Comparison Operators?

Comparison operators are used to **compare two values**. They return a **Boolean** result: `true` or `false`.

🔷 List of Comparison Operators

Operator	Name	Description	Example
`==`	Equal to	Compares values (ignores type)	`5 == "5"` → `true`
`===`	Strict equal to	Compares both value **and** type	`5 === "5"` → `false`
`!=`	Not equal to	Checks if values are **not equal** (ignores type)	`5 != "5"` → `false`
`!==`	Strict not equal to	Checks if values or types are **not equal**	`5 !== "5"` → `true`
`>`	Greater than	Checks if left operand is greater than right	`6 > 4` → `true`
`<`	Less than	Checks if left operand is less than right	`3 < 5` → `true`
`>=`	Greater than or equal to	Checks if left operand is greater than or equal to right	`5 >= 5` → `true`
`<=`	Less than or equal to	Checks if left operand is less than or equal to right	`4 <= 5` → `true`

✅ Detailed Examples

🟡 `==` (Loose Equality)

Compares values **after type coercion**.

```
console.log(5 == "5"); // true (type is converted)
console.log(null == undefined); // true
```

🟢 `===` (Strict Equality)

Compares value **and** type — no type coercion.

```
console.log(5 === "5"); // false
console.log(5 === 5);   // true
```

✅ **Always prefer** `===` to avoid unexpected bugs from type coercion.

🔴 `!=` (Loose Inequality)

```
console.log(5 != "5"); // false (because value is same)
```

🔵 `!==` (Strict Inequality)

```
console.log(5 !== "5"); // true (different type)
```

🔺 `>` (Greater Than)

```
console.log(10 > 5);  // true
console.log(2 > 3);   // false
```

🔻 `<` (Less Than)

```
console.log(3 < 7);  // true
```

🔼 `>=` (Greater Than or Equal To)

```
console.log(6 >= 6); // true
console.log(7 >= 5); // true
```

🔽 `<=` (Less Than or Equal To)

```
console.log(3 <= 3); // true
```

📌 Special Cases to Know

Comparing Strings

Lexicographical order (alphabetical):

```
console.log("apple" < "banana"); // true
console.log("2" < "10");         // false (string comparison)
```

Comparing `null`, `undefined`, `NaN`

```
console.log(null == undefined); // true
console.log(null === undefined); // false

console.log(NaN == NaN); // false
console.log(Number.isNaN(NaN)); // true
```

✅ Summary

- Use `===` and `!==` for **safe comparisons**.
- Avoid `==` and `!=` unless you **understand type coercion**.
- Comparisons return `true` or `false`.

Let me know if you'd like a quick **cheat sheet** or want to continue with the next topic: `Logical Operators`.