Reactのコードを読んだら JavaScriptの仕様が見えてきた話

React Tokyo Meetup #17

セイ

React のソース、読んだことありますか？

面白いコードを1つ見つけたので、今日はみんなで読み解きます。

クイズ

このコードの処理、わかりますか？

```
function is(x, y) {
  return (
    (x === y && (x !== 0 || 1 / x === 1 / y)) || (x !== x && y !== y)
  );
}
```

facebook/react · packages/shared/objectIs.js (L14–18)

ヒント：`===` だと比較できない値がある

解説①: `+0` と `-0`

```
(x === y && (x !== 0 || 1 / x === 1 / y))
```

- `+0 === -0` は `true`（区別できない）
- なので `x === y` だけでは足りない
- `x` が `0` のときだけ `1 / x === 1 / y` で判定
- `1 / +0` → `Infinity`、`1 / -0` → `-Infinity`

+0 と -0 は `===` では区別できないが、Infinity と -Infinity は区別できる

```
const x = -0;
const y = 0;
console.log("x === y", x === y);
console.log("x !== 0", x !== 0);
console.log("1 / x === 1 / y", 1 / x === 1 / y);
console.log("1 / x", 1 / x);
console.log("1 / y", 1 / y);
```

```
x === y true
x !== 0 false
1 / x === 1 / y false
1 / x -Infinity
1 / y Infinity
```

解説②: NaN

```
(x !== x && y !== y)
```

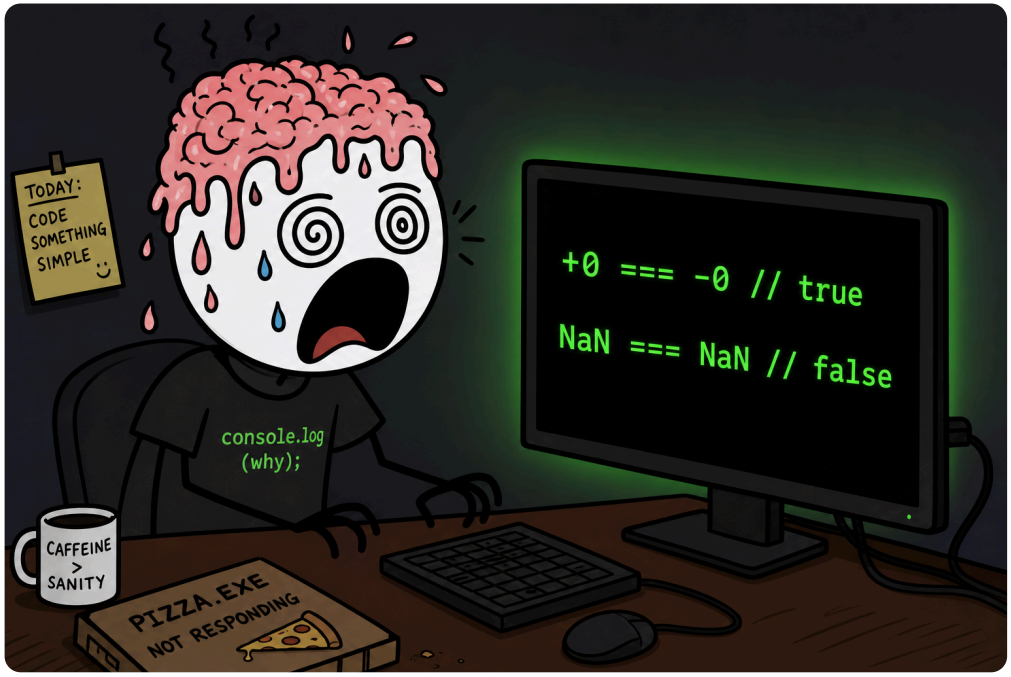
- `NaN === NaN` は `false`
- `NaN` だけ「自分と等しくない」→ `x !== x` が `true`
- `x` と `y` 両方で確認すると、`NaN` 同士を同値扱いできる

それ以外は最初の `x === y` で判定できる

```
const x = NaN;
const y = NaN;
console.log("NaN === NaN", NaN === NaN);
console.log("x !== x", x !== x);
console.log("y !== y", y !== y);
console.log("x !== x && y !== y", x !== x && y !== y);
```

```
NaN === NaN false
x !== x true
y !== y true
x !== x && y !== y true
```

知ってたはずだったのに……



`1 == "1"` が `true`、

`1 + "1"` が `"11"` みたいなのは知ってた

`+0` / `-0` / `NaN` までは知らなかった

こんなめんどい比較、しなきゃいけないの……？

React では `Object.is` を使う

でも、こういう比較のために `Object.is` がある

さっきの `is` は、そのフォールバック実装

```
const objectIs =
  typeof Object.is === "function" ? Object.is : is;
```

facebook/react · objectIs.js (L20–22)

Hooks が「前回と同じか」を見るとき（state 更新のスキップ、effect の deps 比較など）に使われる

まとめ

- `+0` と `-0` は `===` では区別できない
- `NaN` と `NaN` も `===` では `true` にならない（同値とみなせない）
- 厳密に比べるなら `Object.is`
- React のコードを読むと、JS の言語仕様が見えてくる

X @Sei_engineer

ありがとうございました

GitHub