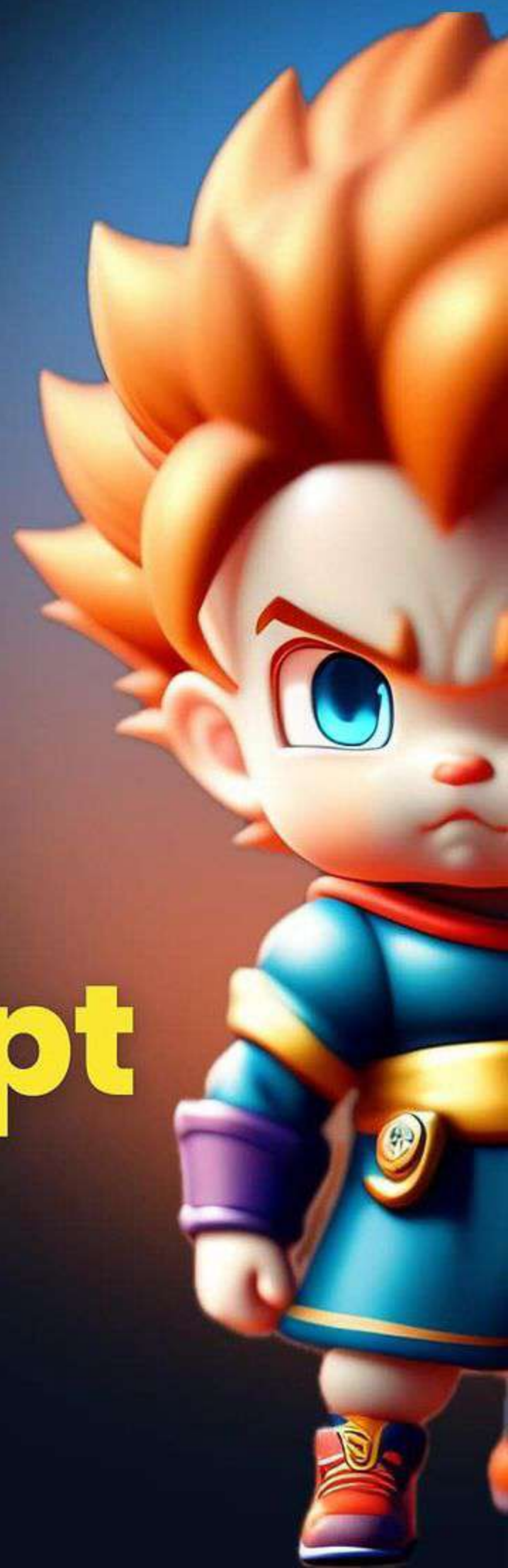
Memory Leaks in JavaScript

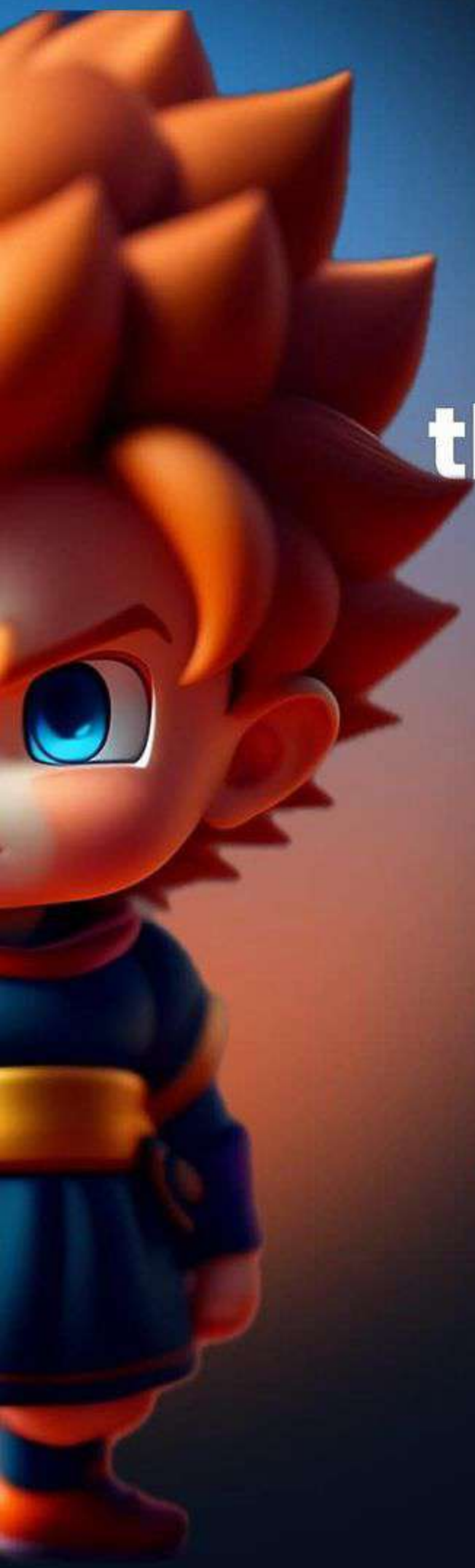


Code For Real
@codeforreal

Code For Real
@codeforreal

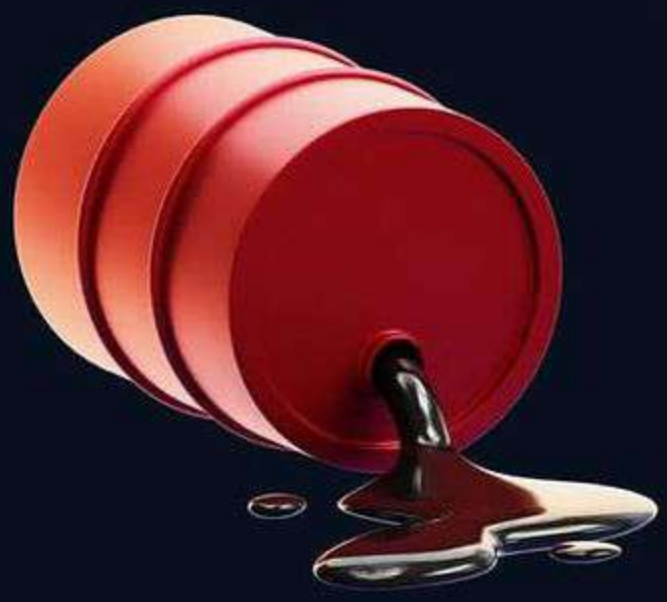
Give a follow

Swipe to see the code practices that are prone to memory leaks in JavaScript.



Dont' forget to save it

Save



A **memory leak** in JavaScript occurs when a program unintentionally retains references to objects or data in memory that are no longer needed, preventing the JavaScript engine's garbage collector from reclaiming that memory. Over time, if memory leaks are not addressed, they can lead to increased memory consumption, decreased performance, and eventually, application crashes.

Let’s go over several examples of memory leaks in Javascript:

Code For Real
@codeforreal

Save

Unreleased Event Listeners:

```
// Memory leak example - Unreleased event listener
const button = document.getElementById('myButton');

function handleClick() {
  // ...
}

button.addEventListener('click', handleClick);

// Missing removal of event listener
// button.removeEventListener('click', handleClick);
```

In this case, if you forget to remove the event listener when it's no longer needed, the associated DOM element (button) and the function (handleClick) will not be garbage collected.

Code For Real
@codeforreal

Save

Interval/Timers without Cleanup:

```
// Memory leak example - Timers without cleanup
function startInterval() {
  const intervalId = setInterval(() => {
    // ...
  }, 1000);
}

startInterval();

// Missing clearInterval(intervalId) when the interval is no
longer needed
```

If you start intervals or timers and forget to clear them when they are no longer needed, they will continue running indefinitely, causing a memory leak.

Code For Real
@codeforreal

Save

Closures and Out-of-Scope Variables:

```
// Memory leak example - Closures and out-of-scope variables
function createClosure() {
  const data = 'Sensitive data';
  return function() {
    console.log(data);
  };
}

const leakedFunction = createClosure();

// 'leakedFunction' retains a reference to 'data' even when it's
no longer needed
```

In this example, the closure retains a reference to the data variable, preventing it from being garbage collected when leakedFunction exists.

Code For Real
@codeforreal

Save

Circular References:

```
// Memory leak example - Circular references
const objA = {};
const objB = {};

objA.ref = objB;
objB.ref = objA;

// Neither 'objA' nor 'objB' can be garbage collected due to the
circular reference
```

Circular references occur when objects reference each other, forming a cycle. JavaScript engines need to detect and handle these situations carefully to avoid memory leaks.

Code For Real
@codeforreal

Save

Unclosed WebSockets or Network Requests:

```
// Memory leak example - Unclosed WebSockets or network requests
function connectWebSocket() {
  const socket = new WebSocket('ws://example.com');
  socket.onmessage = (event) => {
    // ...
  };
}

connectWebSocket();

// Missing socket.close() when the WebSocket is no longer needed
```

Unclosed network connections, such as WebSockets or open HTTP requests, can lead to memory leaks if they are not closed when no longer in use.

Code For Real
@codeforreal

Save

DOM Node Removal without Event Cleanup:

```
// Memory leak example - DOM node removal without event cleanup
const button = document.getElementById('myButton');

function handleClick() {
  // ...
}

button.addEventListener('click', handleClick);

// Removing the button from the DOM, but the event listener remains
button.parentNode.removeChild(button);
```

Removing a DOM node without also removing associated event listeners can cause memory leaks because the event listeners still reference the removed element.

Detecting and resolving memory leaks can be challenging, but it's essential for maintaining the performance and reliability of JavaScript applications. Advanced tools like memory profilers and heap snapshots in browser developer tools can help identify and diagnose memory leaks in more complex scenarios.

Code For Real
@codeforreal

Save

Code For Real

@codeforreal

Did You Like This Post?

Let me know in the comments 👇

Save it