Mastering Throttling for Performance Optimization in JavaScript

SUMANTH M

Frontend developer

What is Throttling?

- Throttling is a programming technique that ensures a function is executed at most once in a specified period.
- Unlike debouncing, which delays execution until after a pause, throttling enforces a limit on how often a function can run.
- Commonly used to optimize performance in high-frequency events like scrolling, resizing, or keypresses.

SUMANTH M
Frontend developer

Why is Throttling Important?

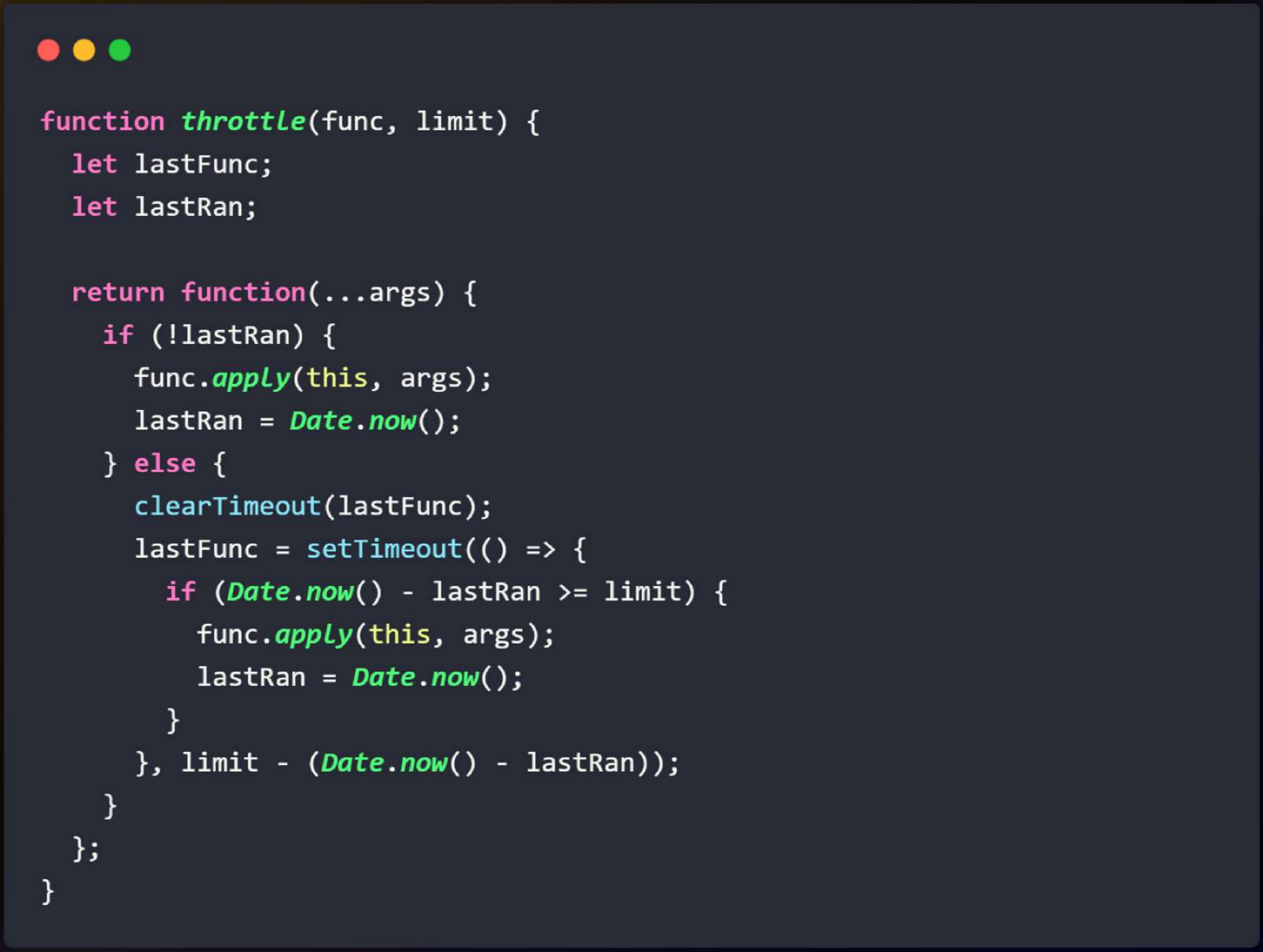
- Performance Improvement: Reduces the frequency of function calls, minimizing the load on the browser.
- Resource Management: Helps manage CPU and memory usage during high-frequency events.
- User Experience: Ensures smooth interaction without overwhelming the system with too many function calls.

SUMANTH M
Frontend developer

3

React Js

How Does Throttling Work?



```
function throttle(func, limit) {
  let lastFunc;
  let lastRan;

  return function(...args) {
    if (!lastRan) {
      func.apply(this, args);
      lastRan = Date.now();
    } else {
      clearTimeout(lastFunc);
      lastFunc = setTimeout(() => {
        if (Date.now() - lastRan >= limit) {
          func.apply(this, args);
          lastRan = Date.now();
        }
      }, limit - (Date.now() - lastRan));
    }
  };
}
```

SUMANTH M
Frontend developer

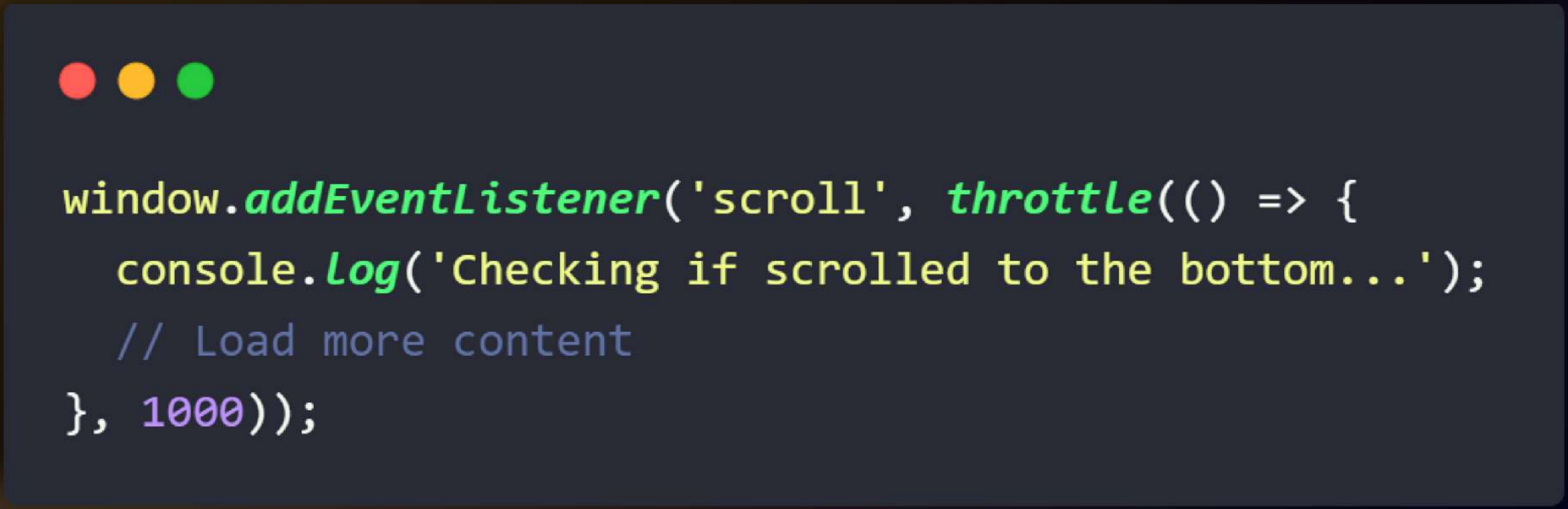
Key Point: This function ensures that the specified function executes at most once every limit milliseconds.

Throttling limits the number of times a function can be executed over time.
A function is allowed to execute at defined intervals, regardless of how many times the event is triggered.
Example of a simple throttle function

SUMANTH M
Frontend developer

4

Real-World Example: Scrolling for Infinite Scroll Feature



```
window.addEventListener('scroll', throttle(() => {
  console.log('Checking if scrolled to the bottom...');
  // Load more content
}, 1000));
```

Use throttling to limit function calls while checking if the user has scrolled to the bottom of the page.

Key Point: The scroll event handler executes at most once every second, reducing load on the browser.

SUMANTH M
Frontend developer

Performance Benefits of Throttling

- Reduced Function Calls: Limits the number of times a function runs, which is crucial for high-frequency events.
- Lower Resource Usage: Decreases CPU and memory consumption by managing event handling more effectively.
- Smoother User Experience: Prevents lag and ensures a responsive interface during user interactions.
- Efficient Network Requests: Helps to minimize unnecessary API calls during frequent events like typing.

SUMANTH M
Frontend developer

When to Use Throttling?

- Scroll Events: To limit how often the scroll handler runs, especially for infinite scrolling.
- Resize Events: To handle window resizing events efficiently without overwhelming the system.
- Button Clicks: To prevent multiple clicks on buttons in rapid succession, especially for form submissions.
- API Requests: To manage the frequency of API calls, preventing server overload.

SUMANTH M
Frontend developer

7 Common Pitfalls of Throttling

- Too Long a Throttle Time: A long throttle time may lead to laggy interactions, making the app feel unresponsive.
- Overusing: Not every function needs throttling; use it judiciously where it's truly beneficial.
- Misunderstanding: Confusing throttling with debouncing; while both are for limiting function execution, they serve different purposes.

SUMANTH M
Frontend developer

8

React Js

Debouncing vs Throttling

- Debouncing: Delays execution until after the event has stopped firing for a specified time.
- Throttling: Ensures a function is called at most once in a specified interval, regardless of how many times the event is triggered.

SUMANTH M
Frontend developer

React Js

Wrap-Up

Throttling is a powerful technique for optimizing performance in JavaScript applications.
It enhances resource management, smooths user interactions, and reduces unnecessary function calls.
Implement throttling in scenarios involving high-frequency events to improve overall application performance.

SUMANTH M
Frontend developer