SALESFORCELWC.COM

CHEATSHEET

> *Stop using @track for everything in LWC. Over-tracking complex SObject record objects is silently killing your Lightning page load speeds. Here is how to fix it.*

LWC Performance Tuning: Rendering, Data Access, and Caching Patterns

Eliminate silent re-renders, optimize reactive state, and deliver lightning-fast Salesforce components.

5 slides inside

Swipe → Save →

2 Master LWC Reactivity Triggers

Before tuning, you must know what actually forces LWC to schedule a component re-render. Modern LWC handles primitives automatically, so stop over-tracking.

- Tracked properties changing or mutated objects/arrays trigger updates.
- A parent component passing a new @api value triggers updates.
- Wire adapters returning new data packages trigger updates.
- Primitives do NOT need @track in modern LWC.
- Mutating an un-tracked object property will NOT trigger a re-render.
- Calling component methods does not schedule rendering.

3 Step 1: Eliminate Over-Tracking

Avoid pulling complete, nested SObject records into reactive properties. Over-tracking causes massive, unnecessary paint loops when unrelated fields update.

- Prefer flat, primitive-typed reactive properties.
- Extract only the specific fields your HTML template renders.
- Assign new object references instead of deeply mutating nested arrays.
- Use @track only when you must mutate objects/arrays in place.

Instead of tracking a whole record, bind individual reactive primitives to isolate rendering triggers.

JAVASCRIPT

```
// Avoid: entire record object is reactive
@track record = {};

// Prefer: only the fields you render are reactive
@track accountName = '';
@track accountPhone = '';
```

5 Step 2: Debounce Heavy State Updates

Search inputs and filter inputs that update a reactive state on every keystroke run complex rendering calculations continuously.

- Implement a simple 300ms debounce pattern for search boxes.
- Delay writing to the reactive state until user input pauses.
- Saves significant browser CPU overhead and UI layout thrashing.

THAT'S A WRAP

Build Snappy Salesforce Components

Ready to eliminate layout lag, master LWC caching patterns, and structure high-performance Lightning layouts? Get the complete architectural guide.

- Read the full technical breakdown at salesforcelwc.com
- Master rendering, caching, and data access optimization patterns.

→ salesforcelwc.com

6 slides · CHEATSHEET