Batch DOM Changes:

Instead of making multiple changes one by one, batch them together. This reduces the number of reflows and repaints.

JavaScript

```javascript
// Inefficient way
for (let i = 0; i < 100; i++) {
    document.body.appendChild(document.createElement('div'));
}

// Efficient way
const fragment = document.createDocumentFragment();
for (let i = 0; i < 100; i++) {
    fragment.appendChild(document.createElement('div'));
}
document.body.appendChild(fragment);
```

Use CSS Classes:

Apply changes by toggling CSS classes rather than modifying styles directly. This helps in minimizing reflows.

JavaScript

```javascript
// Inefficient way
element.style.width = '100px';
element.style.height = '100px';
element.style.backgroundColor = 'red';

// Efficient way
element.classList.add('new-style');
```


CSS

```css
.new-style {
    width: 100px;
    height: 100px;
    background-color: red;
}
```

Avoid Complex CSS Selectors: Simplify your CSS selectors to reduce the time the browser takes to match them.

CSS

```css
/* Inefficient */
div > p:first-child span {
    color: red;
}

/* Efficient */
.highlight {
    color: red;
}
```

Minimize DOM Depth: *Keep the DOM tree shallow. Changes at one level can cause reflows at all levels*.

HTML

```
<!-- Inefficient -->
<div>
    <div>
        <div>
            <div>
                <p>Text</p>
            </div>
        </div>
    </div>
</div>

<!-- Efficient -->
<div>
    <p>Text</p>
</div>
```

Optimize Animations: *Use requestAnimationFrame for animations and consider using will-change to inform the browser about upcoming changes*

JavaScript

```javascript
// Using requestAnimationFrame
function animate() {
    // Animation code here
    requestAnimationFrame(animate);
}
requestAnimationFrame(animate);

// Using will-change
element.style.willChange = 'transform';
```

Modify Hidden Elements: If you need to make multiple changes to an element, hide it first (e.g., using display: none), make the changes, and then show it again.

```
// Inefficient way
element.style.display = 'block';
element.style.width = '100px';
element.style.height = '100px';

// Efficient way
element.style.display = 'none';
element.style.width = '100px';
element.style.height = '100px';
element.style.display = 'block';
```

Use Positioning Wisely: Use position: absolute or position: fixed for elements that change frequently to remove them from the document flow

```css
/* Inefficient for frequently changing elements */
.moving-element {
    position: relative;
}

/* Efficient */
.moving-element {
    position: absolute;
}
```