

# Solutions to Common Problems

- [Tracking Element Visibility](#)

# Tracking Element Visibility

This could be used for lazy loading (perhaps on the Node Tree View)

The react docs have an article on [useEffect](#) has an example in the **Examples of connecting to an external system section** that has a code sample of how to detect if an element is on screen. I imagine being able to use this if we needed roll our own lazy loading solution for that would not require that we have a specific height, or if we wanted to have lists nested in other lists (like in the tree view for example where each node has it's own children that we want to load in batches).

**Examples of connecting to an external system**

Connecting to a chat server 2. Listening to a global browser event 3. Triggering an animation 4. Controlling a modal dialog 5. Tracking element visibility

### Example 5 of 5: Tracking element visibility

In this example, the external system is again the browser DOM. The `App` component displays a long list, then a `Box` component, and then another long list. Scroll the list down. Notice that when all of the `Box` component is fully visible in the viewport, the background color changes to black. To implement this, the `Box` component uses an `Effect` to manage an `IntersectionObserver`. This browser API notifies you when the DOM element is visible in the viewport.

```
App.js  Box.js  Reset  Fork
```

```
1 import { useRef, useEffect } from 'react';
2
3 export default function Box() {
4   const ref = useRef(null);
5
6   useEffect(() => {
7     const div = ref.current;
8     const observer = new IntersectionObserver(entries => {
9       const entry = entries[0];
10      if (entry.isIntersecting) {
11        document.body.style.backgroundColor = 'black';
12        document.body.style.color = 'white';
13      } else {
14        document.body.style.backgroundColor = 'white';
15        document.body.style.color = 'black';
16      }
17    });
18    observer.observe(div);
19  });
20 }
```

- Item #0 (keep scrolling)
- Item #1 (keep scrolling)
- Item #2 (keep scrolling)
- Item #3 (keep scrolling)
- Item #4 (keep scrolling)
- Item #5 (keep scrolling)
- Item #6 (keep scrolling)
- Item #7 (keep scrolling)
- Item #8 (keep scrolling)
- Item #9 (keep scrolling)
- Item #10 (keep scrolling)
- Item #11 (keep scrolling)
- Item #12 (keep scrolling)
- Item #13 (keep scrolling)
- Item #14 (keep scrolling)
- Item #15 (keep scrolling)
- Item #16 (keep scrolling)
- Item #17 (keep scrolling)
- Item #18 (keep scrolling)
- Item #19 (keep scrolling)

▼ Show more