How do you use the Resize Observer API in Svelte?

Written by Seth Corker on Benevolent Bytes

The Resize Observer API is a useful browser API that allows you to watch the size of elements and receive updates whenever the size changes. I've wrapped this API to use in React with a custom hook as it can be useful to apply different styles or even swap out components depending on the size the element is taking up. So how would we go about using this API in Svelte?

Get to the code!

There's not much to it so take a look at the code and let's walk through it.

Get a reference to the element

In React, whenever you want to interact with the DOM directly you have to create a reference with the `useRef` hook and then access `current` on the ref as it changes. Svelte's API is a bit easier to do this.

```
<script>
  let boxEl;
</script>

<div class="box" bind:this={boxEl}></div>
```

We simply declare a variable then use the `bind:this` attribute on the element. Done. Now we can access the DOM on `boxEl`.

Adding the Resize Observer

We add the observer on mount with the aptly named `onMount` lifecycle hook in Svelte.

```
onMount(() => {
    const resizeObserver = new ResizeObserver(entries => {
      // We're only watching one element
      const entry = entries.at(0);

      //Get the block size
      boxWidth = entry.contentBoxSize[0].blockSize;
    });

    resizeObserver.observe(boxEl);

    // This callback cleans up the observer
    return () => resizeObserver.unobserve(boxEl);
  });
```

The key things to *observer* is that we create a new `ResizeObserver` with a callback that will be triggered whenever the element changes. You can observer multiple elements but for our use case we only care about one, that's why I'm ignoring the other entries and just pulling out the first one - `const entry = entries.at(0);`.

I want to see the size of the element so I'm setting `boxWidth = entry.contentBoxSize[0].blockSize;` and Svelte will see the assignment to the variable `boxWidth` and infer this is a reactive variable so the UI will update.

Once it's setup, we observe the element with `resizeObserver.observe(boxEl);`. The last piece of the puzzle is providing a cleanup callback so we return `resizeObserver.unobserve(boxEl);` so that if the component were to ever be unmounted, the observer will stop watching for updates.

Wrapping up

There you have it, the Resize Observer API is a powerful web API that can come in handy. It's easy to use regardless of component framework you decide to go with.

Resources

- bind:this on Svelte Docs

- onMount