Portals: New ways to view the web

(Web) Portals as a technology are not new. Java has used portlets (as defined in JSR 362 for the latest edition) in containers for a long time. Teams have used other languages to use Portals.

Chrome has (re)introduced the concept of portals in a different context. They are meant to navigate from page to page and give you a preview of the content before you navigate to it. According to the specification:

> This specification extends HTML to define a new kind of top-level browsing context, which can be embedded in another document, and a mechanism for replacing the contents of another top-level browsing context with the previously embedded context.

The feature and it's acompanying specification are still under development and have limited browser support. As of this writing (Late November 2019) the feature is only available in Chrome (stable and canary) with the `chrome://flags/#enable-portals` flag enabled.

We have multiple options for creaating portals in a page.

The first option is to use the `portal` tag to setup the portals that we want to use. The HTML code looks like this:

```html
<portal src="https://css-tricks.com/">
<h2>Your browser doesn't support Portals</h2>

<p>Here's the link:
  <a href="https://css-tricks.com/">
    CSS Tricks</a>
</portal>
```

We can also create portal elements programmatically. To create an equivalent example to the HTML element programmatically we would do something like this:

```
const style = document.createElement('style');
style.innerHTML = `
  portal {
    width: 40%;
    height: 600px;
    opacity: 0;
    box-shadow: 0 0 20px 10px #999;
    transform: scale(0.4);
    transform-origin: bottom left;
    bottom: 20px;
    left: 20px;
    animation-name: fade-in;
    animation-duration: 1s;
    animation-delay: 2s;
    animation-fill-mode: forwards;
  }
  .portal-transition {
    transition: transform 0.4s;
  }
  @media (prefers-reduced-motion: reduce) {
    .portal-transition {
      transition: transform 0.001s;
    }
  }
  .portal-reveal {
    transform: scale(1.0) translateX(-20px) translateY(20px);
  }
  @keyframes fade-in {
    0%   { opacity: 0; }
    100% { opacity: 1; }
  }
`;
const portal = document.createElement('portal');
'portal'// page we want to navigate tosrc = 'https://css-tricks.com';
// Add a'https://css-tricks.com'// Add a class that defines the transitio
portal.addEventListener('click', evt => {
  // Animate the portal once user 'click'// Animate the portal once user
  portal.classList.add('portal-reveal');
});
portal.addEventListener('transitionend', evt => {
  if (evt.propertyName == 'transform') {
    portal.activate();
  }
});
document.body.append(style, portal);
```

We could create individual portals using the code above but it gets tedious to repeat code over and over so we wrap it around a function to make reuse easier.

In this case we also move the CSS code outside the Javascript to make sure that we can reuse for all the portals we create. We'll have to play with the CSS based on both the size of the portals we want to create and how we want to animate them when they open.

```
function createPortal(url) {
  const portal = document.createElement('portal');
  portal.src = url;
  portal.classList.add('portal-transition');
  portal.addEventListener('click', evt => {
    portal.classList.add('portal-reveal');
  });
  portal.addEventListener('transitionend', evt => {
    if (evt.propertyName == 'transform') {
      portal.activate();
    }
  });
  document.body.append(portal);
}
```

Why would we use portals?

While it takes longer for portals to appear when you render the page it gives you something that links, on their own, cannot. A live preview of the target page

I put together a quick demo of using portals to create a porfolio so you can see both how portals work and the difference that it has with building the same type of page with Flexbox.

What’s left?

As awesome as they are Portals are not a complete solution (or I haven’t fully figured how to make it one yet).

The first, and most important, thing is that portal navigation is broken in Chrome right now. Once you click on a portal the history and back nagivation are lost.

According to KenjiBaheux:

> The team is working on making back navigation work (it’s currently broken, as in it doesn’t do anything at the moment).
>
> From: Twitter

The second issue for me is that I can’t reliably style portals that I create programmatically. I don’t want to duplicate the style inside evevery portal element and I don’t want to be forced into using the HTML version where creating them programmatically is more appropriate.

This may be the use case for customizing the styles when embedded as a portal and I’m not understanding the examples I’ve seen or the specification well enough to implement it.

There is no way, to my knowledge, to polyfill the feature which means that you’ve got to come up with an alternative for browsers that don’t support the feature.

For an early stage API, portals offer an intriguing posibility for navigating across pages, either same domain or cross domains.