CSS Containment and container queries

It looks like CSS containment will finally bring container queries into browsers without the need for a polifill.

This post will discuss the current implementation in Chromium browsers.

Note:

The specification for container queries hasn't been finalized. It is possible but unlikely that the specification and the corresponding CSS will change.

Don't use `@container` in production until the feature is finalized.

What problem do container queries solve?

Container queries give developers finer control over the layout of components. Rather than using media queries that provide responsiveness based on the viewport state, they give responsiveness based on the parent container or the next ancestor that has containment applied to it.

This means that we can have both large elements and layouts that are use media queries and smaller components that use container queries to provide finer-controlled layouts based on the status of elements on the page, not on the page overall.

How they work?

```
<div class="card-container">
  <div class="card">
    <figu<div class="card"><!-- image and caption go here -->igure>
```

```
    <div>
      <header>
        <!-- Hea<div><!-- Header and related information -->er>
      <div class="notes">
        <!-- content <div class="notes"><!-- content related data -->k me
    </div>
  </div>

  <!-- additional cards -->
</div></button><!-- additional cards -->
</div>
```

Becfore we can use use `@container`, we need to create a parent element that has containment by setting `contain: layout inline-size`.

`contain: layout inline-size` creates a new containing block and new block formatting context, letting the browser separate it from the rest of the layout.

```
.card-container {
  contain: style layout inline-size;
  width: 100%;
}
```

Then we can use the `@container` pseudo element to change the layout of our elements based on the width of the constrained parent.

The syntax is similar to media queries and the `@media` pseudo element but with different results.

The example below uses four container queries to change the way content will look based on the parent element's width.

```
@container (max-width: 850px) {
  /*
  if the width of the parent is less than 850px
  */
```

```
}

@container (max-width: 650px) {
  /*
  if the width of the parent is less than 650px
  */
}

@container (max-width: 460px) {
  /*
  if the width of the parent is less than 460px
  */
}

@container (max-width: 300px) {
  /*
  if the width of the parent is less than 300px
  */
}
```

Alternatives for browsers that don't support `@container`

I'm still researching the way to get container queries to work in older browsers.

To make sure we don't break things we use a feature query to wrap the `@container` style selectors to make sure that it will only run in browsers that support containment.

```
@supports (contain: inline-size) {
  @container (max-width: 850px) {}

  @container (max-width: 650px) {}
}
```

Once we have the native container queries handled, we can use third party

libraries like eqio to provide equivalent functionality for browsers that don't support it natively.

First we load the script. This example uses the unpkg CDN.

```
<script src="https://unpkg.com/eqio/umd/eqio.min.js"></script>
```

Once the script is loaded you can use code like the one below to make sure you initialize all the elements you want eqio with.

We wrap this on a feature detection for Intersection observer, and would load a polyfill if the feature is not supported natively.

```
function supportsIntersectionObserver() {
  return ('IntersectionObserver' in window &&
    'IntersectionObserverEntry' in window &&
    'intersectionRatio' in window'IntersectionObserverEntry'prototype)
}

if (!supportsIntersectionObserver) {
  console.log('loading Intersection Observer polyfill');
  'loading Intersection Observer polyfill'// Load Intersection Observer po
} else {
  console.log('browser supports Intersection observer. Keep going');
  // if the browser supports
  // Intersection Observer then do nothing
}

// run eqio codeentsByClassName('eqio'));

const eqios = [];

els.forEach((el) => {
  eqios.push(new Eqio(el));
});

'eqio'
```

Then add the following attribute to the HTML elements that you want to use container queries:

- The `eqio` class to the element in addition to any other class you need
- A `data-eqio-sizes` attribute whose value is a JSON-serializable array of sizes that you want to match on
- A `data-eqio-prefix` attribute to tell eqio the prefix for your class names.

```
<div
  class="media-object eqio"
  data-eqio-sizes='["<400", ">700"]'
  data-eqio-prefix="media-object"
>
  …
</div>
```

The example component will:

- be customised when its width is 400 or smaller ("<" is a synonym for `max-width`, not "less than")
- be customised when its width is 700 or greater (">" is a synonym for `min-width`, not "greater than").
- apply the following classes `media-object-eqio-<400` and `media-object-eqio->700` as appropriate

The final step is to create the CSS for each matching condition. The query is a combination of the `data-eqio-prefix` HTML attribute and the value of the query we want to match, <400 or >700.

< and > are special characters in CSS so they need to be escaped as `\<` and `\>`.

```
@supports not (contain: inline-size) {
  .media-object-eqio-\<400 {
    /* less than or equal to 400px */
  }

  .media-object-eqio-\>700 {
    /* greater than or equal to 700px */
  }
```

```
}
```

Both solutions should be functionally identical but, as with everything on the web, please test it with your oown project to make sure it works as intended in all browsers.

Links and resources

- An introduction to CSS Containment
- The new responsive: Web design in a component-driven world
- Next Gen CSS: @container
- CSS Container Queries — MDN
- CSS Podcast, Episode 43: Containment
- CSS Containment Module Level 1
- Can I use: CSS containment
- CSS Triggers — What gets triggered by mutating a given property
- CSS Containment in Chrome 52
- Avoid Large, Complex Layouts and Layout Thrashing
- CSS Contain