Supporting more colours on the web with P3

Written by Seth Corker on Benevolent Bytes

What is DCI-P3?

P3 is is a colour gamut defined by Digital Cinema Initiatives, hence the DCI prefix. Essentially, it defined the possible colours that can be represented. On the web, you're probably quite familiar with colours being defined in hexadecimal, RGB or HSL. These all sit within the sRGB colour gamut.

Why is it useful?

P3 is a larger colour space than sRGB so, if your display allows, it can display a wider range of colours. The difference between the sRGB and P3 is that the latter can support around 25% more colours, specifically it allows for more greens and reds.

If you want to learn more about what this looks like in practice, Dean Jackson has a good in-depth post with comparisons between the two colour spaces.

What supports it?

One thing that's important to note is that hardware and software need to support the colour gamut for it to work. Apple and a few other manufacturers produce hardware that supports the P3 colour space and with native applications like Photoshop and video editing software but the web has been left out until recently. CSS Color Module Level 4 defines additional colour spaces for use in CSS, this means as web developers - we can leverage different colour spaces with an easy to use syntax. WebKit supports P3 so if you're using a modern apple device and Safari, you can leverage it today.

How can I use it?

The simplest way is like this:

```
h1 {
  background-color: color(display-p3 0.925 0.093 0.351)
}
```

Notice the `color` function allows for the gamut followed by three decimal values where `color(display-p3 1 1 1)` is white. You can also add a fourth value delimited by a slash to support transparency like so: `color(display-p3 1 1 1 / 0.5)`.

Checkout the WebKit announcement post by Nikita Vasilyev's post, Wide Gamut Color in CSS with Display-P3 for some features added to Safari dev tools to make P3 easier to work with and some helpful tips.

Detecting support

Supporting devices that don't yet support functionality is important on the web and it can be done easily with graceful degradation or detecting support.

The cascade helps us add support easily because if the browser doesn't understand a rule then it will simply ignore it and fallback to tho last one instead. Here, if the browser doesn't support P3, it will use the RGB rule instead.

```
nav {
    background-color: rgb(0, 255, 0);
    background-color: color(display-p3 0 1 0);
}
```

If you want to do it a different way, through progressive enhancement, here's how you'd do that. Media queries allow you to explicitly detect if P3 is supported:

```
@media (color-gamut: p3) {
  header {
    background-color: color(display-p3 0.925 0.093 0.351)
  }
}
```

In JavaScript you can use `window.matchMedia` and supply the media query there too:

```
if (window.matchMedia("(color-gamut: p3)").matches) {
   el.style.backgroundColor = "color(display-p3 0.925 0.093 0.351)"
}
```

And finally, you can even use P3 on the canvas by using `getContext` and supplying a `colourSpace` option:

```
const myCanvas = document.getElementById("canvas");
const myContext = canvas.getContext("2d", { colorSpace: "display-p3" });
```

Where to go from here?

There's a lot of work going on to support more recent advances like HDR and colour spaces on the web. The links I added are a great place to learn some more about the technology and how to use it.