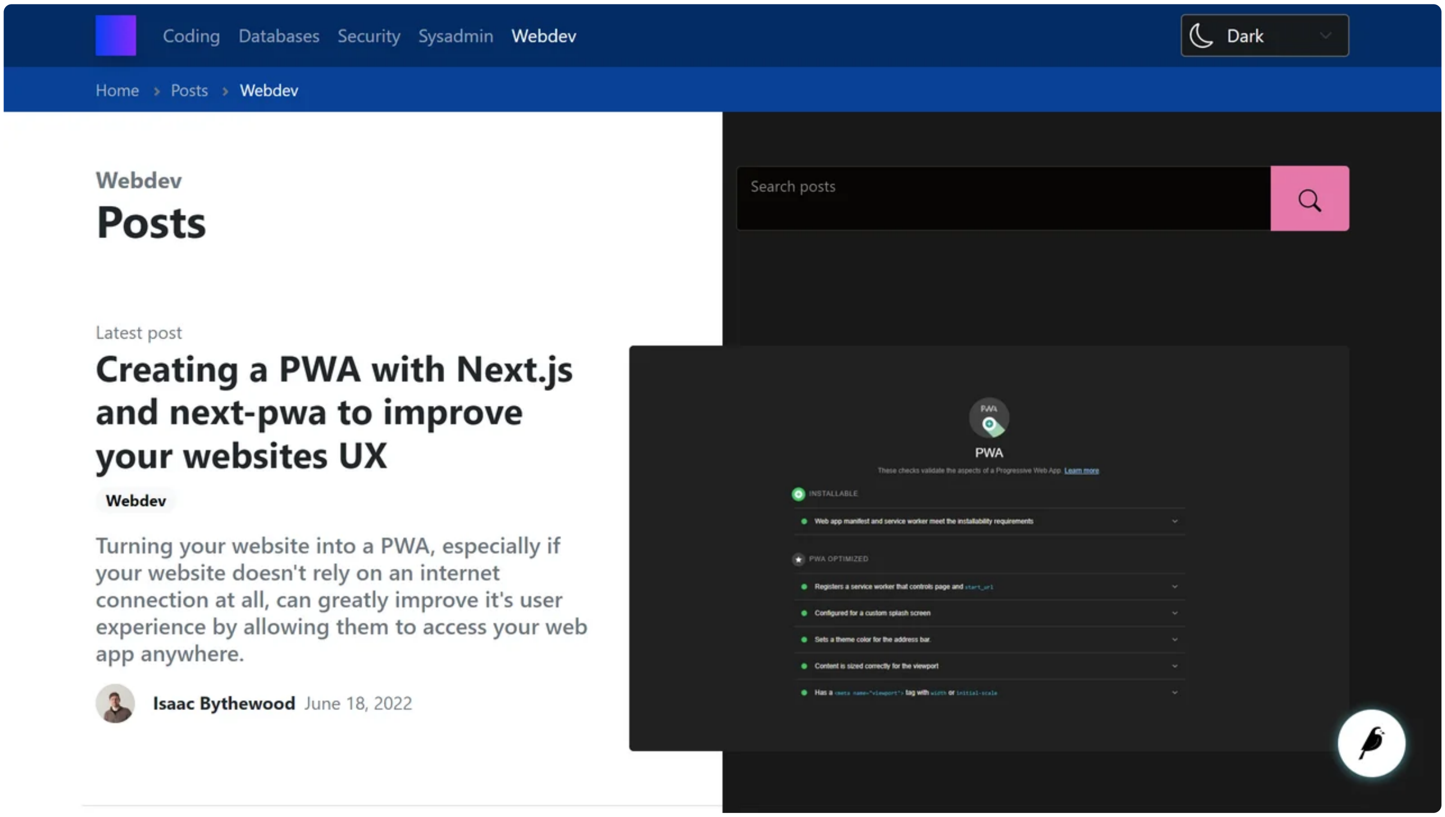
Webdev

⏱ 5 min read

Adding dark mode with automatic system preference selection

Creating a dark, or light, version of your website may seem like a daunting task if you think you need an entirely new color pallet. It's 2022 though and we have the widely supported invert CSS filter.



Webdev

Posts

Latest post

Creating a PWA with Next.js and next-pwa to improve your websites UX

Webdev

Turning your website into a PWA, especially if your website doesn't rely on an internet connection at all, can greatly improve it's user experience by allowing them to access your web app anywhere.

Isaac Bythewood June 18, 2022

PWA

These checks validate the aspects of a Progressive Web App. Learn more

INSTALLABLE

- Web app manifest and service worker meet the installability requirements

PWA OPTIMIZED

- Registers a service worker that controls page and `start_url`
- Configured for a custom splash screen
- Sets a theme color for the address bar.
- Content is sized correctly for the viewport
- Has a `<meta name="viewport">` tag with `width` or `initial-scale`

Author

Isaac Bythewood

July 02, 2022

Luckily all modern browsers since 2015 have had a CSS filter to help with creating dark and light mode color schemes based on your current theme, invert. With roughly ~95% global support for this filter you can safely create a dark mode version of your site with it. I'm doing it on this site right now!

Here's the dark theme for this website:

```css
/* dark.css */

main.dark {
  background: #e7e7e7;
  filter: invert(1);
}

main.dark img {
  filter: invert(1);
}

main.dark .block-code {
  filter: invert(1);
}

main.dark .reverse-invert {
  filter: invert(1);
}
```

One of the first things you'll want to fix when doing a global invert is images. The invert filter will invert literally everything, including pictures, which you probably don't want inverted. I made a global `img` option to reverse all image inverts by inverting it again. I've seen no noticeable performance loss or image issues from doing this thus far. My blocks of code are already "dark mode" so there's no point in inverting them. There is also a helper class `.reverse-invert` that can be used on the fly when I think it's needed. I also don't invert the entire page and I just invert my content since my navbar and footer work in both dark and light mode. You could easily change this to be `body.dark` to invert everything.

You now have a dark mode theme for your site. Spot check over things and add `.reverse-invert` when you think it's needed and maybe run a lighthouse check for color accessibility doing slight adjustments where required.

The next step is to make our color mode swapper, I add a little bit more CSS for this but mostly relied on Bootstrap classes.

```css
/* dark.css cont. */

#prefers-color-scheme {
  width: 150px;
  background-color: #171a1d;
  border-color: #6b6b6b;
  color: white;
  padding-left: 40px;
}

.prefers-color-scheme-icon {
  position: absolute;
  color: white;
  margin: 8px;
}
```

Then for the HTML element I use a simple select field with some inline SVG icons that I got from [Bootstrap's icon project](#).

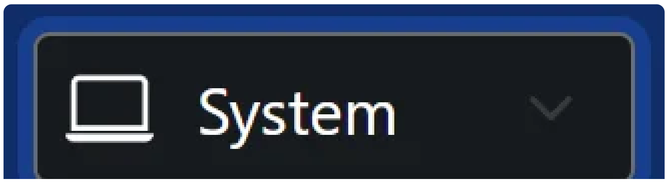
```html
<div id="color-scheme-selector">
  <svg xmlns="http://www.w3.org/2000/svg" width="22" height="22" fill="
    <path d="M13.5 3a.5.5 0 0 1 .5.5V11H2V3.5a.5.5 0 0 1 .5-.5h11zm-11-
  </svg>
  <svg xmlns="http://www.w3.org/2000/svg" width="22" height="22" fill="
    <path d="M8 11a3 3 0 1 1 0-6 3 3 0 0 1 0 6zm0 1a4 4 0 1 0 0-8 4 4 0
  </svg>
  <svg xmlns="http://www.w3.org/2000/svg" width="22" height="22" fill="
    <path d="M6 .278a.768.768 0 0 1 .08.858 7.208 7.208 0 0 0-.878 3.46
  </svg>
  <select class="form-select" id="prefers-color-scheme" aria-label="Sele
```

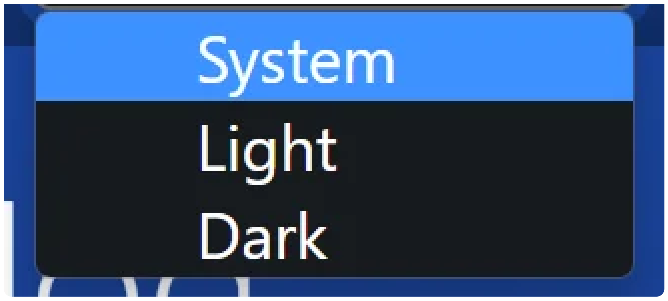
```
    <option value="system" selected>
      System
    </option>
    <option value="light">
      Light
    </option>
    <option value="dark">
      Dark
    </option>
  </select>
</div>
```

And finally the JavaScript portion that makes the selector work. This entire system checks for your computer's preferred color scheme above all else and will use that unless you manually change it.

```
/**
 * dark.js
 *
 * Detects the systems current preference for dark or light mode but al
 * overriding the system preference.
 */

const main = document.querySelector("main");

const getSystemPreference = () => {
  if (window.matchMedia("(prefers-color-scheme: dark)").matches) {
    return "dark";
  } else {
    return "light";
  }
};

const setIcon = (preference) => {
  const icons = document.querySelectorAll(".prefers-color-scheme-icon")
  icons.forEach((icon) => {
    icon.classList.add("d-none");
  });
  icons.forEach((icon) => {
    if (icon.classList.contains(preference)) {
      icon.classList.remove("d-none");
    }
  });
};

const setPreference = (preference) => {
  setIcon(preference);
  if (preference === "system") {
    preference = getSystemPreference();
  }
  if (preference === "dark") {
    main.classList.add("dark");
  } else {
    main.classList.remove("dark");
  }
};

const select = document.querySelector("#prefers-color-scheme");
select.addEventListener("change", () => {
  localStorage.setItem("darkMode", select.value);
  setPreference(select.value);
});

const storedPreference = localStorage.getItem("darkMode");
if (storedPreference) {
  select.value = storedPreference;
  setPreference(storedPreference);
} else {
  setPreference("system");
}
```

Once all added you'll have a selector that looks something like this that you can then swap between modes with.





You now have a fully working dark and light mode theme selector with an easy to use invert system to make future maintenance and additions easy.

Finally, **an invert word of warning**, I've found that some elements, like "position: fixed" and "position: sticky" sometimes don't preform as expected when being inverted unless you invert the "body" or "html" element of your website so always test. You may have to change how you invert based on your site's design.