Precache Files

If you want your web app to work offline or there are assets you know can be cached for a long time, precaching is the best approach.

Precaching a file will ensure that a file is downloaded and cached before a service worker is installed, meaning that if your service worker is installed, your files will be cached.

Workbox provides an easy way to precache files, ensuring that as your service worker changes, the precached files are maintained efficiently, only downloading updated files and cleaning up after the service worker is made redundant.

Precaching files with Workbox can be done like so:

```
workbox.precaching.precacheAndRoute([
    '/styles/index.0c9a31.css',
    '/scripts/main.0d5770.js',
    { url: '/index.html', revision: '383676' },
]);
```

The above snippet will download the files `/styles/index.0c9a31.css`, `/scripts/main.0d5770.js` and `/index.html` during the service worker install event and create a route that serves these files directly from the cache.

This list of files is normally generated using a tool that manages the versioning of files.

Generating a Precache Manifest

Most users of Workbox will generate the list of files to precache by using one of the following Workbox tools, although you can use any tools that outputs files with revisioning. Below is a list of tools that can generate this list of files.

Note: If you're not sure what option is best for you, [start with the CLI](#) as it's easy to setup and will give you a better understanding of how the other tools work.

Workbox Command Line Interface (CLI)

Ideal for developers who are **unfamiliar with Node** or **have simple needs**.

LEARN HOW TO USE THE CLI

workbox Build

Perfect for developers wanting to **programmatically build the list in Node** or are **using Gulp** for their build process.

LEARN HOW TO USE WORKBOX-BUILD

Workbox Webpack Plugin

Ideal for **developers using webpack** to build their project.

LEARN HOW TO USE THE WEBPACK PLUGIN