Best practices

I posted this as an answer to this question in Quora and I thought I would post it here and expand on it a little bit with things I thought about after I wrote the answer.

This is not an exhaustive list of performance best practices. It's what I use and how I use them. You may have others and some of these may not apply to you. I'd love to hear what works for you... you can contact me via Twitter (https://twitter.com/elrond25).

The question:

What are the best practices for optimizing resources (JavaScript, CSS, images) used by an HTML page?

- In general
  - Use Lighthouse (available in Chrome as part of the DevTools audits menu or as an extension)
    - The performance score is a good sign of how your app/site is doing
    - There are other audits you can run separately or at the same time
  - Always try to serve content via HTTP2.
    - HTTP2 solves a lot of the performance issues in older versions of HTTP. See this article for the nerdy details
    - If you want to take the time to test it, http2 push may also help increase your site's performance. **It is imperative that you test this because, if poorly implemented, you can wreck performance with push**
  - Use a CDN like Akamai or Cloudflare to host and serve your static assets. Even basic services are good enough based on my experience
  - Consider using a service worker even if you're not creating a PWA.
    - Service workers will improve performance on second and subsequent visits because of the browser will fetch content from your local computer
    - A service worker is the entry point to advanced features like web push notifications, background sync and background fetch among others

        - You can configure different caching strategies based on the needs of your site or app
        - If the browser doesn't support service workers then it won't get the performance boost and you will loose access to the advanced features but it will still display content for your users
    - Consider preloading resources
- For your images
    - Use responsive images rather than create a single version of the image. Using a single small image means that it'll look like crap in retina displays for desktop and higher-end mobile devices
    - Create responsive images as part of your build process. I use Gulp and gulp-responsive
    - Serve WebP images to browsers that support them.
        - You can incorporate WebP support in your responsive images
        - WebP is significantly smaller than JPG or PNG but not all browsers support the format
    - If you can't or don't want to use responsive images you can compress the images with Imagemin. I also do this as part of my build process with Gulp and gulp-imagemin
    - Use an image CDN like Cloudinary or Photon if you use Wordpress to host your assets. They'll do all the work for you
- For your scripts and stylesheets
    - Consider minimizing your scripts. I use gulp and uglify-es
    - If you will use a lot of Javascript consider using a Bundler like Webpack or Rollup
        - I'm one of the few developers who doesn't think you need to bundle all your assets (CSS, Javascript and images) when building your site or app
        - Test if a bundler improves performance for the content you're using it for before you decide to adopt it
    - Consider minimizing your stylesheets. I use sass/scss and normally create a compressed version either from CLI or using gulp-sass during the build process
    - I don't concatenate them because I cache on the client using service workers and they work better as separate items
- HTML
    - I don't normally minimize my HTML until the size hits 75K or so. ***I'm old school and a lot of what I learned when I first started working on the web was by looking at other people's code, duplicating it locally and then tweaking it***

to see what happened. I think it's still useful to learn that way.

- With the performance optimizations for scripts, images and stylesheets I think I've made up for not removing whitespace from my HTML content