Front-end Test

Page Object Module:

**23 front-end performance engineering rules for web app**

**Fully utilize cookies and cache.**

**9. Use cookie-free domains**

When the browser requests a static image and sends cookies with the request, the server ignores the cookies. These cookies result in unnecessary network traffic. To work around this problem, make sure that static components are requested with cookie-free requests by creating a subdomain and hosting them there.

**10. Reduce cookie size**

HTTP cookies are used for authentication, personalization, and other purposes. Cookie information is exchanged in the HTTP headers between web servers and the browser, so keeping the cookie size small minimizes the impact on response time. Also consider the use of HTML5 instead of cookies for retaining application state information.

**1. Add long-term headers expiration dates**

Near-future headers expiration dates prevent effective caching and cause a repeat visit to your site from the device to be slower than necessary.

**2. Make fewer HTTP requests**

Latency has a substantial impact on mobile application performance. Reducing the number of unique objects on the page will help reduce sensitivity to latency.

*[ Understand what your team needs to know to*[*take advantage of test automation*](https://content.microfocus.com/l/software-test-automation-tb?utm_source=techbeacon&utm_medium=techbeacon&utm_campaign=00134846&_ga=2.27348920.504322932.1594330559-468876911.1593762255)*with TechBeacon's Guide. Plus: Get the*[*Buyer's Guide For Software Test Automation Tools*](https://content.microfocus.com/software-test-automation-tb/buyers-guide-software-test-automation?lx=wYdl7c&utm_source=techbeacon&utm_medium=techbeacon&utm_campaign=00134846&_ga=2.27348920.504322932.1594330559-468876911.1593762255)*]*

**3. Avoid URL redirect**

Redirects can cause significant delays for mobile user web access. Try to reduce the number of redirects.

**19. Reduce DNS lookups**

The Domain Name System (DNS) maps hostnames to IP addresses, just like phonebooks map people's names to their phone numbers. When you type a URL into the browser, the browser contacts a DNS resolver that returns the server's IP address. DNS has a larger cost on mobile networks than on traditional wired networks; typically, it takes 120 to over 400 milliseconds for a mobile lookup of the IP address associated with a hostname. The browser can't download anything from the host until the lookup completes.

**22. Use HTML5**

HTML5 offers new features that can reduce the number of bytes going from your server to the client. These features include local storage, manifest, and databases.

**6. Make AJAX cacheable**

One of AJAX's benefits is that it provides instantaneous feedback to the user by requesting information asynchronously from the back-end web server. However, using AJAX doesn't guarantee that the user will wait for the asynchronous [JavaScript and XML responses](http://techbeacon.com/17-best-practices-java-performance-engineering) to return. Optimizing AJAX responses is important to improve performance, and making the responses cacheable is the best way to optimize them.

**7. Avoid HTTP 404 (Not Found) error**

Making an HTTP request and receiving a 404 (Not Found) error is expensive and degrades the user experience. Some sites have helpful 404 messages (for example, "Did you mean...?"), which may assist the user, but server resources are still wasted.

**8. Remove unsupported components**

Not all devices support all types of components (such as Flash).

**11. Compress components with gzip**

Gzip compression reduces the HTTP response size and thus the response time. Response size is typically reduced by about 70 percent.

**12. Minify CSS and JavaScript**

Minification removes unnecessary characters from a file to reduce its size, thereby improving load times. When a file is minified, comments and unnecessary white space characters (space, newline, and tab) are removed. This [improves response time](http://techbeacon.com/java-performance-tools-teams-survey-who-doing-what) since the size of the download is reduced.

**13. Avoid CSS expressions**

CSS expressions are a powerful—and dangerous—way to dynamically set CSS properties. These expressions are evaluated frequently: when the page is rendered and resized, when the page is scrolled, and even when the user manipulates the page via touch events. These frequent evaluations can degrade the user experience.

**14. Remove unnecessary CSS rules**

Any rule in the CSS that isn't necessary for the current page shouldn't be downloaded. Consider evaluating your CSS with an automated tool that identifies unused selectors and analyzes CSS coverage.

**15. Reduce the number of DOM elements**

A complex page means more bytes to download, and it also means slower DOM access in JavaScript. [Httparchive.org](http://httparchive.org/trends.php) reports an average of 1,300 DOM elements per page. Reduce the number of DOM elements on the page to improve performance.

**16. Don't scale images in HTML**

Web page designers sometimes set image dimensions by using the width and height attributes of the HTML image element. Avoid doing this, because it can result in images being larger than necessary. For example, if your page requires image myimg.jpg, which has dimensions 240 x 720 but displays it with dimensions 120 x 360 using the width and height attributes, the browser will download an image that is larger than necessary.

**17. Reference images in the HTML**

Images that are only referenced in the CSS won't be downloaded until the CSS is downloaded. Consider referencing the images in the main HTML page to speed up mobile page rendering time.

**18. Use a content delivery network (CDN)**

The proximity of the mobile carrier's Internet gateway to web servers can impact response times. Using a CDN brings content to servers that are closer to the user, reducing latency and improving performance.

**20. Use GET for AJAX requests**

When using the XMLHttpRequest object, the browser implements POST in two steps: first it sends the headers, and then it sends the data. It's better to use GET instead of POST, because GET sends the headers and the data together (unless there are many cookies). However, Internet Explorer's maximum URL length is 2 KB, so if you're sending more than this amount of data, you may not be able to use GET.

**21. Put Javascript at the bottom**

The problem caused by scripts in the HEAD section is that they block parallel downloads. The iPhone actually downloads components in the order it sees fit, so scripts and stylesheets are always downloaded first. As long as your scripts are above the stylesheets and not in the HEAD, they won't block anything and will always be among the first things downloaded, so there's no need to put all of them at the bottom.

**23. Resize images**

Devices have limited screen sizes. There's normally no reason to have images larger than the screen size.

These rules are a great place to start, but applying them requires considerable knowledge and skill. Stay tuned for future articles where I will explore these rules in greater detail.