Replacing Images with WebP equivalents on the server

WebP provides smaller files and bettter quality than equivalent JPG or PNG files. The problem is that not all browser support the WebP format, only Chromium-based browsers like Chrome, Opera, plus Edge and Firefox (according to caniuse.com. There is no easy way to support browsers that support WebP and those that don't in the same page without modifying the HTML on the pages.

Another option is to let the server replace images with their WebP versions for those browsers that support it. By configuring the server to replace the images where supported without having to change the HTML on your pages you gain the benefits of the WebP format

How to do this will depend on what server you're using. For this we'll explore configurations for Nginx and Apache.

Nginx

If you're using Nginx the following code will serve WebP images if the browser supports the format and there is an image available on the server.

Note that this example is also set up to experiment with

The configuration, will do the following:

1. Check if the accept header include webp
2. Check if there's a WebP image on the server
3. If there is a file on the server then add the Vary Accept header
4. If the browser supports WebP images then replace the image with the WebP equivalent

```
location / {
  if ($http_accept ~* "webp")
  {
```

```
    set $webp_accept "true";
  }

  if (-f $request_filename.w"true"
    set $webp_local  "true";
  }

  if ($webp_local = "true") {
    add_header Vary Accept;
  }

  if ($webp_accept = "true") {
    rewr"true"*) $1.webp break;
  }

  if ($http_user_agent ~* "(?i)(MSIE)") {
    proxy_hide_header Vary;
    add_header Cache-Control private;
  }

  "(?i)(MSIE)"# Rest of configuration goes here
}
```

Apache

The Apache configuration is a set of re-write rules that will do the same thing for Apache that the configuration for Nginx did.

The example below can be used in the global server context (httpd.conf), virtualhost context (<VirtualHost> blocks), or directory context (.htaccess files and <Directory> blocks).

The process is almost the same as the one for Nginx:

1. If the rewrite module is installed and active
    1. Activates the rewrite engine and sets the base for the follow on steps
    2. Checks if the user agent matches Chrome or Opera

        3. Checks if the browser sent the Accept header
        4. Check if the WebP file exists on the server
        5. Replace the images with the WebP equivalent
            1. It uses case insensitive matching
            2. It forces the mime type of the result to be `image/webp`
            3. It sets the environment variable webp
            4. It stops the matching. This is the last step in the matching chain
    2. If the headers module is installed
        1. Set the Vary Accept header
    3. Add the WebP mime type and associate it with the `.webp` extension

```
&lt;IfModule mod_rewrite.c&gt; #1
  RewriteEngine On #i
  RewriteCond %{HTTP_USER_AGENT} Chrome [OR] #ii
  RewriteCond %{HTTP_USER_AGENT} Opera [OR] #ii

  RewriteCond %{HTTP_ACCEPT} image/webp [OR] #iii

  RewriteCond %{DOCUMENT_ROOT}/$1\.webp -f #iv

  RewriteRule (.+)\.(?:jpe?g|png)$ $1.webp [NC,T=image/webp,E=webp,L] #v
&lt;/IfModule&gt;

&lt;IfModule mod_headers.c&gt; #2
  Header append Vary Accept env=REDIRECT_accept #i
&lt;/IfModule&gt;

AddType image/webp .webp #3
```

Conclusion

It is possible to swap JPG and PNG images for an equivalent WebP images without modifying the HTML documents. This is particularly important for older content that is unlikely to be updated.

This technique is not a replacement for client-side responsive images but a complement for when updating existing content with the code for responsive

images is not feasible because of time or cost.