.htaccess & Nginx redirects cheat sheet

301 redirects for Apache and Nginx, put live safely: back up, 302, check, 301.

One broken line in .htaccess gives every page a 500 error, and browsers keep a 301 even after you remove it. Go live with 302s first. Undo, Apache: `cp .htaccess.bak .htaccess` · Nginx: copy the backup back, then `sudo nginx -t && sudo nginx -s reload`

The safe order

1. Back up the current config

```
cp .htaccess .htaccess.bak   # Apache, in the site root
cp /etc/nginx/sites-available/example.com \
  ~/example.com.nginx.bak   # Nginx
```

2. Go live as temporary 302s

Apache `[R=302,L]` and `[R=302,NE,L]`, Nginx `return 302` and `rewrite … redirect`. Nginx: test, then reload:

```
sudo nginx -t && sudo nginx -s reload
```

3. Check every old URL: one hop, right page

```
curl -sIL 'http://example.com/old-page'
```

Or paste the list into boga.io/tools/bulk-redirect-checker/

4. Make them permanent (301)

The same rules with 301: Apache `[R=301,L]` and `[R=301,NE,L]`, Nginx `return 301` and `permanent`; reload Nginx again. Temporary on purpose? Keep the 302s.

Apache (.htaccess)

Top of the file, above `# BEGIN WordPress`, after one `RewriteEngine On`. For www, swap the two names.

Force HTTPS and non-www, in one hop

```
RewriteCond %{HTTPS} off [OR]
RewriteCond %{HTTP_HOST} !^example\.com$ [NC]
RewriteRule ^(.*)$ https://example.com/$1 [R=301,L]
```

Remove the trailing slash

```
RewriteCond %{REQUEST_FILENAME} !-d
RewriteCond %{REQUEST_URI} ^(.+)/$
RewriteRule ^ %1 [R=301,L]
```

Add a trailing slash

```
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_URI} !\.[^/]+$
RewriteRule ^(.+[^/])$ /$1/ [R=301,L]
```

Redirect one page to another

```
RewriteRule ^old-page/?$ /new-page [R=301,NE,L]
```

Redirect a whole folder

```
RewriteRule ^blog(?:/(.*))?$ /news/$1 [R=301,L]
```

Move a whole domain to a new one

```
RewriteCond %{HTTP_HOST} ^(www\.)?old\.example$ [NC]
RewriteRule ^(.*)$ https://new.example%{REQUEST_URI} \
  [R=301,L]
```

Nginx

`server` blocks go next to your site's own, with your certificate where they listen on 443. The other lines go inside your site's `server` block.

Force HTTPS and non-www, in one hop

```
server {
    listen 80;
    listen [::]:80;
    server_name example.com;
    return 301 https://$host$request_uri;
}
server {
    listen 80;
    listen [::]:80;
    listen 443 ssl;
    listen [::]:443 ssl;
    server_name www.example.com;
    return 301 https://example.com$request_uri;
}
```

Remove the trailing slash

```
if (!-d $request_filename) {
    rewrite "^/(.+)/$" /$1 permanent;
}
```

Add a trailing slash

```
location ~ "^([^.]+[^/])$" { return 301
  $1/$is_args$args; }
```

Redirect one page to another

```
rewrite "^/old-page/?$" /new-page permanent;
```

Redirect a whole folder

```
rewrite "^/blog(?:/(.*))?$" /news/$1 permanent;
```

Move a whole domain to a new one

```
server {
    listen 80;
    listen [::]:80;
    listen 443 ssl;
    listen [::]:443 ssl;
    server_name old.example www.old.example;
    return 301 https://new.example$request_uri;
}
```

Not working?

- **Ignored:** an old 301 in your browser cache (try a private window), rules below the WordPress block, `AllowOverride None`, or the site runs Nginx, which never reads `.htaccess`.
- **Too many redirects** behind Cloudflare: use the proxy version (`X-Forwarded-Proto`) or set SSL/TLS to Full.
- **A query string** is never part of the pattern. Test it in a condition above the rule: `RewriteCond %{QUERY_STRING} ^id=5$`