MaintenancePage

Recipes for Implementing a Maintenance Page

It is relatively common for Apache httpd users to wish to serve a standard page while underlying software deployments are occurring, or even if the site is down temporarily. Below are details on some of the methods by which this can be achieved. All recipes share in common the idea that you redirect to a page served with a 503 return code to prevent search engines indexing the maintenance page.

Touch a File Method

This recipe is good because it can be simply programmed in a shell script or similar. Maintenance is enabled by 'touching' a single file on the file system and disabled by removing that file. In this example a generic piece of Apache configuration allows either individual virtual hosts to be put into maintenance mode, or the whole server. In practice, especially with a large number of virtual host, the maintenance configuration could be split out into an Include file, or moved to the global context, although in this case you may need to set `RewriteOptions InheritBefore` in each virtual host.

What is not included below is the Directory configuration to enable the serving of the URI `/maintenance/index.html` itself as a page.

Also please note, if a virtual host is accessed using any ServerAlias then you need to add `UseCanonicalName On` to the configuration, otherwise only people using the actual ServerName for a particular virtual host will get the maintenance page. If, for whatever reason, you cannot use this directive, then an appropriate filename can be substituted for `%{SERVER_NAME}` in the relevant RewriteCond in each virtual host.

```
<VirtualHost *:8080>
  ServerName myfirstdomain.com
  DocumentRoot "/var/www/myfirstdomain/htdocs"

  # Redirect all request to a 503 return code when in maintenance mode
  ErrorDocument 503 /maintenance/index.html
  RewriteEngine on
  RewriteCond /var/www/maintenance/ALL -f [OR]
  RewriteCond /var/www/maintenance/%{SERVER_NAME} -f
  RewriteCond %{REQUEST_URI} !=/maintenance/index.html
  RewriteRule ^ - [R=503,L]

  # Redirect away from the maintenance page if not in maintenance mode
  RewriteCond /var/www/maintenance/ALL !-f
  RewriteCond /var/www/maintenance/%{SERVER_NAME} !-f
  RewriteRule ^/maintenance/index.html$ / [R,L]
</VirtualHost>

<VirtualHost *:8080>
  ServerName myseconddomain.com
  DocumentRoot "/var/www/myseconddomain/htdocs"

  # Redirect all request to a 503 return code when in maintenance mode
  ErrorDocument 503 /maintenance/index.html
  RewriteEngine on
  RewriteCond /var/www/maintenance/ALL -f [OR]
  RewriteCond /var/www/maintenance/%{SERVER_NAME} -f
  RewriteCond %{REQUEST_URI} !=/maintenance/index.html
  RewriteRule ^ - [R=503,L]

  # Redirect away from the maintenance page if not in maintenance mode
  RewriteCond /var/www/maintenance/ALL !-f
  RewriteCond /var/www/maintenance/%{SERVER_NAME} !-f
  RewriteRule ^/maintenance/index.html$ / [R,L]
</VirtualHost>
```

The downside to this recipe is that it incurs file system checks for all requests hitting your web server, so careful performance testing is required. With this configuration you may then perform the following operations.

```
# Enable maintenance mode for all virtual hosts
touch /var/www/maintenance/ALL
# Enable maintenance mode for a single virtual host
touch /var/www/maintenance/myseconddomain.com
```

IfDefine Method

This recipe requires that you modify `apachectl` to cope with 'enabling/disabling' maintenance. The upside is that there are no file system checks, the downside is that it requires a full stop/start since you cannot pass a name that is interpreted by IfDefine with a restart.

```
<VirtualHost *:8080>
  ServerName myfirstdomain.com
  DocumentRoot "/var/www/myfirstdomain/htdocs"

  <IfDefine Maintenance>
    ErrorDocument 503 /maintenance/index.html
    RewriteEngine on
    RewriteCond %{REQUEST_URI} !=/maintenance/index.html
    RewriteRule ^ - [R=503,L]
  </IfDefine>

  <IfDefine !Maintenance>
    RewriteEngine on
    RewriteRule ^/maintenance/index.html$ / [R,L]
  </IfDefine>
</VirtualHost>

<VirtualHost *:8080>
  ServerName myseconddomain.com
  DocumentRoot "/var/www/myseconddomain/htdocs"

  <IfDefine Maintenance>
    ErrorDocument 503 /maintenance/index.html
    RewriteEngine on
    RewriteCond %{REQUEST_URI} !=/maintenance/index.html
    RewriteRule ^ - [R=503,L]
  </IfDefine>

  <IfDefine !Maintenance>
    RewriteEngine on
    RewriteRule ^/maintenance/index.html$ / [R,L]
  </IfDefine>
</VirtualHost>
```

And then the following addition to the `case` statement in apachectl.

```
maintenance)
    $HTTPD -k graceful-stop
    sleep 3
    $HTTPD -D Maintenance -k start
    ;;
```

Maintenance with Exceptions

Sometimes it is advantageous to configure the server so that a few or even many IP address may bypass the maintenance page. Using the first example above as a template, we can create a RewriteMap that allows individual IP addresses or even IP ranges to ignore the maintenance setting. This recipe can easily be extended to allows class A or B ranges as well.

```
<VirtualHost *:8080>
  ServerName myfirstdomain.com
  DocumentRoot "/var/www/htdocs"

  UseCanonicalName On
  ErrorDocument 503 /maintenance/index.html
  RewriteEngine on
  RewriteMap exceptions /var/www/maintenance/exceptions.map

  # Allow Individual IP addresses past maintenance page
  RewriteCond ${exceptions:%{REMOTE_ADDR}} =OK
  RewriteRule ^ - [L]

  # Allow Class C ranges past maintenance page
  RewriteCond %{REMOTE_ADDR} ^(\d+)\.(\d+)\.(\d+)\.
  RewriteCond ${exceptions:%1.%2.%3} =OK
  RewriteRule ^ - [L]

  # Redirect all request to a 503 return code when in maintenance mode
  RewriteCond /var/www/maintenance/ALL -f [OR]
  RewriteCond /var/www/maintenance/%{SERVER_NAME} -f
  RewriteCond %{REQUEST_URI} !=/maintenance/index.html
  RewriteRule ^ - [R=503,L]

  # Redirect away from the maintenance page if not in maintenance mode
  RewriteCond /var/www/maintenance/ALL !-f
  RewriteCond /var/www/maintenance/%{SERVER_NAME} !-f
  RewriteRule ^/maintenance/index.html$ / [R,L]
</VirtualHost>
```

Sample `exceptions.map` file

```
# Allow a single IP address through
192.168.0.1 OK
# Allow a whole Class C through
172.20.0 OK
```