RewriteContext

Introduction

The Apache HTTPD Server deals with requests in discrete phases. While this is usually transparent to the user and administrator it does have an effect on the behaviour of mod_rewrite when rulesets are placed in different contexts. To oversimplify a little, when rules are placed in VirtualHost blocks (or in the main server context) they get evaluated before the server has yet mapped the requested URI to a filesystem path. Conversely, when rules are placed in .htaccess files, or in Directory blocks in the main server config, they are evaluated after this phase has occured.

This timing issue affects two areas of rewrite rule construction, the base URI as seen by RewriteRule and the values of certain environment variables. See the note entitled "Per-directory Rewrites" here for further details.

Base URI

The most visible change to look out for here is that in VirtualHost (per-server) context the request URI as seen by RewriteRule will start with a / (slash). Conversely in .htaccess or Directory (per-directory) the directory containing the rules, plus a trailing slash, are stripped before the comparison to the pattern.

As an example, consider this basic rule to map the request for 'bar' to 'baz' for a request of /foo/bar

```
# in /var/www/foo/.htaccess
RewriteBase /foo
RewriteEngine On
RewriteRule ^bar baz
```

```
# in VirtualHost
RewriteEngine On
RewriteRule ^/foo/bar$ %{DOCUMENT_ROOT}/foo/baz
```

Note that RewriteBase is used only to re-assemble relative subsitutions, not to strip the per-directory prefix which is always the context of the rules themselves.

Environment Variables

As mentioned, in VirtualHost (per-server) context, rewrite rules run before the request has been mapped to a filesystem path. Resulting from this, certain of the variables available to mod_rewrite concerned with filesystem paths are incomplete, including SCRIPT_FILENAME (and its mirror, REQUEST_FILENAME). During VirtualHost (per-server) context, these variables contain a copy of REQUEST_URI, as opposed to their ultimate filesystem path values.

Conceptually, since *_FILENAME haven't been decided in per-server context, you should probably avoid using them in this context and just use REQUEST_URI.

As an example, the following two rulesets attempt to see if the request uri corresponds to an existing file and take action if not.

```
# in .htaccess, SCRIPT_FILENAME is a complete _filesystem_ path
RewriteEngine On
RewriteCond %{SCRIPT_FILENAME} !-f
RewriteRule (.*) script.cgi
```

```
# in VirtualHost, SCRIPT_FILENAME is incomplete, so this ruleset SHOULD just use REQUEST_URI
# to see where this URI _may_ have been mapped in the filesystem
RewriteEngine On
RewriteCond %{DOCUMENT_ROOT}/%{SCRIPT_FILENAME} !-f
RewriteRule (.*) /script.cgi
```