

FIREFOX HACKS™

*Tips & Tools for Next-Generation
Web Browsing*



O'REILLY®

Nigel McFarlane

HACK
#27

Fix Web Servers to Support Firefox Content

Provide support for the bonus content types that Mozilla technology supports.

Firefox supports more varied content than just that of web pages. In addition to the big world of XUL-based applications, there are a number of configuration and security content types that can be delivered to the browser. This Hack explains how to set up Apache so that all these types are delivered with the right Content-Type. It also explains what to do about broken versions of Internet Explorer.

Configuring Firefox Content Types in Apache

The following MIME types are the ones most commonly used by Firefox. Update your Apache *httpd.conf* file to include these directives, and then restart httpd or signal it to reread the config files.

Support for HTML content types:

```
AddType application/xhtml+xml      .xhtml
AddType application/http-index-format .hti
AddType text/html                    .html
AddType text/css                     .css
AddType text/xsl                     .xslt
AddType application/x-javascript     .js
```

Support for other presentable XML content:

```
AddType application/xml             .xml /* or .. */
AddType text/xml                    .xml
AddType image/svg+xml               .svg
```

Mozilla does not use the MIME type required for pure MathML; such content should be delivered inside a compound document identified as XHTML or XML.

Support for remote XUL and RDF:

```
AddType application/vnd.mozilla.xul+xml .xul
AddType application/rdf+xml            .rdf
```

Support for delivery of extensions, patches and configuration items:

```
AddType application/x-xpinstall      .xpi
AddType application/x-javascript-config .cfg .jsc
```

Support for server delivery of certificates, either for certificate-database install or for immediate use as data:

```
AddType application/x-x509-ca-cert   .x509ca
AddType application/x-x509-server-cert .x509sv
AddType application/x-x509-crl        .x509_crl # OSCP
AddType application/x-x509-user-cert  .x509ua
```

Fix Web Servers to Support Firefox Content

```

AddType application/x-x509-email-cert .x509em
AddType application/x-pkcs7-crl .pk7_crl
AddType application/pkix-crl .pk_crl
AddType application/x-pkcs7-mime .pk7m_crl
AddType application/pkcs7-signature .pk7_sig
AddType application/pkix-crl .pk_crl
AddType application/ocsp-response .ocsp

```

Filename extensions are merely suggestive in this case; most content of these types will likely be generated by a server program rather than delivered directly from flat files.

Supporting Both Standards and Internet Explorer

All modern browsers, except for all Internet Explorer (IE) versions, can handle XHTML content delivered with the correct `application/xhtml+xml` MIME type. Generally speaking, IE can't handle this case. Something has to be done to accommodate that. In the past, all XHTML content was delivered as `text/html` for IE's benefit. Now that many standards-oriented browsers are gaining in market share, it's time to reduce IE to a special case and deliver the correct content types by default.

The XHTML standard *does* allow XHTML 1.0 to be delivered as `text/html`. This is a stopgap measure that works for all browsers. The XHTML standard *does not* allow this trick for XHTML 1.1 content or higher. That means that there is no standards-based fix available for most IE versions for XHTML 1.1 or higher. The only available solution is to hack the Windows Registry so that content with `application/xhtml+xml` is treated as `text/html`. See <http://juicystudio.com/xhtml-registry-hack.asp> for details. To make the server lie when required, try the following code snippets.

PHP content rewriter. These tests pick up all the weird variants of IE, including Pocket PC versions, and replace the true content type with a lie for IE:

```

<?
$agent = $_SERVER['HTTP_USER_AGENT'];
$broken = FALSE;
$broken = $broken || (strpos($agent, "msie") && !strpos($agent, "opera"));
$broken = $broken || strpos($agent, "microsoft internet explorer");
$broken = $broken || strpos($agent, "mspie");
$broken = $broken || strpos($agent, "pocket");

if ( $broken ) {
    header("Content-Type: text/html");
}
else {
    header("Content-Type: application/xhtml+xml");
}

```

```
}
?>
```

For more complete browser-sniffing code, visit <http://www.apptools.com/phptools/browser/source.php>.

Perl CGI content rewriter. Similarly, here's the rewriter in Perl:

```
use CGI;
my ($query,$broken) = (0, false);
$query = new CGI;
$query->use_named_parameters(1);

$broken = $query->http("HTTP_USER_AGENT") =~ /pocket/i;
$broken = ... # as for PHP

print "Content-type: application/xhtml+xml\n\n" if not $broken;
print "Content-type: text/html\n\n" if $broken;
# ... and so on.
```

Native Apache rewriter. For static files, use Apache's `mod_rewrite` module, which must be configured and enabled before it is usable. The obscure code required for the IE lie is:

```
RewriteEngine on
RewriteBase /
RewriteCond %{HTTP_USER_AGENT} pocket [nocase,ornext]
RewriteCond %{HTTP_USER_AGENT} mspie [nocase,ornext]
RewriteCond %{HTTP_USER_AGENT} pocket [nocase,ornext]
RewriteCond %{HTTP_USER_AGENT} msie [nocase]
RewriteCond %{HTTP_USER_AGENT} !msie.*opera [nocase]
RewriteCond %{HTTP_USER_AGENT} !opera.*msie [nocase]
RewriteCond %{REQUEST_URI} \.html$
RewriteCond %{THE_REQUEST} HTTP/1\.[^0]
RewriteRule .* - [type=text/html]
```