Level: Advanced
Works with: Notes/Domino
Updated: 02-Jan-2003

by
Chuck
Dumont

In his *LDD Today* article "**Building Web applications in Domino 6: Browser caching and response header rules**," John Chamberlain describes new features in Domino 6 that enable Domino administrators to specify caching rules for Domino Web server responses based on URL path. In this article, we discuss additional enhancements for browser-side caching in Domino 6 that will be available in the Domino 6.0.1 maintenance release. These enhancements include the use of HTTP/1.1 Entity Tags (ETags) by the Domino server and the @SetHTTPHeader function used by Web applications to further control browser caching behavior. In addition, we describe how to diagnose your cache validation issues.

This article is intended for experienced Domino Web server administrators and Web application developers. It assumes familiarity with HTTP headers and the Lotus Formula Language.

@SetHTTPHeader

Domino 6 introduced the @SetHTTPHeader function, which allows applications to set HTTP response headers. Application developers can use this function to set headers that control browser caching behavior. These headers include Expires and Cache-Control for HTTP/1.1 responses or Pragma: no-cache for HTTP/1.0 responses. As an example of how you could use these headers, suppose that you have a frequently visited Web page, customized for each user, that is updated on a daily basis. Because the page is personalized for each user, you do not want the response cached in public caches (proxy servers and so on), but you do want the response cached by browsers. Furthermore, you want the cached response to expire at midnight, so that the browser will not attempt to validate the cached response if the page is visited again during the same day. One way that you could do this is to create a hidden field that is computed-for-display and contains the following formula:

```
cacheable := @If(SERVER_PROTOCOL = "HTTP/1.0"; 0; 1);
@If(cacheable; @SetHTTPHeader("Cache-Control"; "private"); @SetHTTPHeader("Pragma"; "no-cache"));
@If(cacheable;0;@Return(0));
now := @Now;
midnight := @Adjust(now;0;0;1;-@Hour(now);-@Minute(now);-@Second(now));
@SetHTTPHeader("Expires"; midnight);
```

Because the response is only suitable for private caches (i.e. browsers), we require that the HTTP version of the request be at least 1.1. The poorly named SERVER_PROTOCOL field is a CGI environment variable which contains the protocol string from the request. If the request version is 1.0, then the requester does not recognize the Cache-Control header, so we set the HTTP/1.0 Pragma:no-cache header value and return. Otherwise, we set the Cache-Control:private header value and then set an Expires header with a value of midnight of the current day. This tells the browser to cache this response, but not to bother validating the cached response with the server until the following day.

Note: Using @SetHTTPHeader to set the Expires or Cache-Control headers in Domino 6.0 can result in duplicate and/or conflicting headers. This problem is fixed in Domino 6.0.1. If the application sets one or more Expires, Cache-Control, or Pragma:no-cache headers with @SetHTTPHeader, then the Domino 6.0.1 server will not set any of these headers. Also, using @SetHTTPHeader to set the Last-Modified header overrides the server's setting of this header. See the **Domino Designer 6 Help** for a list of supported CGI environment variables.

Browser cache validation

The previous @SetHTTPHeader example informs the browser how to cache a response and tells it how long the cached response should be considered valid. If the browser needs to display a cached response after the valid period has expired, then the browser needs to validate the cached response with the server. Cache validation is accomplished by sending a conditional GET request to the server. A conditional GET request is the same as a normal GET request, except that the request includes an If-Modified-Since header with a last modified date and/or an If-None-Match header with an ETag. Both the last modified date and the ETag are received from the server in a Last-Modified header and an ETag header, respectively, with the response that was cached. The last modified date and ETag are saved by the browser along with the cached response and are used by the server to validate cached responses in conditional GETs. If the server can determine from the Last-Modified date or the ETag that is returned to it in a conditional GET request that the cached response is still valid, then it returns a status 304 - Not Modified response with no data. If the browser's cached response is no longer valid, then the server returns a status 200 - OK response with the updated page results.

The Domino Web server uses a number of factors to determine whether or not to validate cached responses. These can include the database design Last-Modified date, the database data Last-Modified date, and the document Last-Modified date. If the response contains any off-database data (that is, any data from a source other than the database specified in the URL) or time variant information, then the server cannot validate the response because the server cannot make a determination regarding the validity of the cached response. However, the server cannot always tell how the various sources of data that go into forming a response are used. For example, the @Now function used to determine the Expires time of the response in the previous example does not make the response data time variant. Nevertheless, it flags the response as a time variant response and consequently, the server does not validate any conditional GETs for this resource. To give application developers more control over cache validation, the server recognizes the X-Domino-CacheValidationDependsOn response header as a server directive which specifies how cache validation should be handled by the server. Applications can set this header with the @SetHTTPHeader function. Note that because this header is recognized by the server as a server directive, it is not output to the response stream unless the Domino environment variable DominoTraceCacheValidation=1 is set.

X-Domino-CacheValidationDependsOn: [Document | Database | <TimeDate>]
Setting this header with a value of Document (or Database) tells the server that the document (or database data) Last-Modified time should be used for validating any cached responses that use this form. Note that for both Document and Database the cached response is also assumed to be dependant upon the design Last-Modified time of the database, and the time specified in the If-Modified-Since header of the response being validated is assumed to be the later of the document Last-Modified time (or database data last modified-time) and the design Last-Modified time at the time that the response was sent. If a TimeDate value is specified, then cache validation is accomplished by comparing this value directly with the time specified in the request's If-Modified-Since header. This implies that when using a TimeDate value, the response must be fully evaluated before cache validation can be attempted, so cached responses are not validated using ETags.

Note: The X-Domino-CacheValidationDependsOn header is supported in Domino version 6.0.1 and later.

The **Domino Designer 6 Help** contains information about @SetHTTPHeader as well as @GetHTTPHeader, which is a read-only counterpart to @SetHTTPHeader.

Entity tags

Domino 6.0.1 introduces the use of Entity tags (ETags) for cache validation. ETags, which were introduced in HTTP/1.1, are server-defined tokens that uniquely identify multiple instances or versions of the same resource. ETags can significantly improve server performance by increasing the effectiveness of browser-side caching.

Prior to Domino 6.0.1, Domino used only the Last-Modified header for cache validation; however, there are two main problems with the Last-Modified header only model of cache validation. The first problem is that the Last-Modified date, by itself, often does not provide enough information for the server to validate the response without first determining what the sources of the data are. In practice, this means that the server must generally evaluate the response to the point where the output has been generated and is ready to send to the browser before the server can make a determination regarding cache validation. So the server must do all of the work of generating the response, with the exception of actually sending it, to validate a cached response. The second problem has to do with validating user variant responses. Consider the following scenario:

- User A logs in and accesses user specific information from a Domino Web site.
- User A logs out and closes the browser.
- User B logs in on the same machine (without changing the user profile).
- User B accesses the same Web page as User A and sees User A's cached response.

Using a Last-Modified date only for cache validation, the server cannot validate any cached responses that contain any type of user variant information because the server has no way of knowing if the cached response is for the same user as the currently authenticated user.

The use of ETags for cache validation resolves both of these problems. ETags are used in much the same manner as Last-Modified times, except that the contents of the ETag header are defined by the server, instead of by the protocol.

By placing information in the entity tag that identifies which data sources the response depends on (database, document, design) and the Last-Modified times of those sources, the server can now make a determination up front, and with minimal overhead, as to whether or not the cached response is still valid. This eliminates the need for fully evaluating the response before being able to validate a cached response. Also, by including the user name of the authenticated user of a user variant response in the entity tag, the server can now validate user variant cached responses.

Entity tags are only supported by HTTP/1.1. The Domino server sends an entity tag, which is a base-64 encoded strings of characters, in an ETag header for all cacheable responses because the server uses HTTP/1.1 for all responses. If the client is an HTTP/1.0 browser, then the ETag header is ignored and conditional GET's from that browser contain only the Last-Modified date. You can disable ETags for diagnostic purposes. Set the Domino environment variable DominoDisableETags=1 in the server Notes.ini file or with the server console command:

```
set config DominoDisableETags=1
```

Diagnosing cache validation issues

If you are trying to understand whether or not cache validation for a particular resource is behaving as expected, you can configure the Web server to output response headers which contain detailed information about cache validation results. These headers are enabled by the environment variable DominoTraceCacheValidation=1. This environment variable can be set in the Notes.ini file or with the server console command:

```
set config DominoTraceCacheValidation=1
```

Note: The DominoTraceCacheValidation environment setting is supported in Domino version 6.0.1 and later.

When cache validation is attempted using an ETag request header, the result is specified in the X-Domino-CacheValidationWithETagResult header. This header can have one of the following values:

- Failed using ETag
- Succeeded using ETag

If cache validation using an ETag fails, then the X-Domino-CacheValidationWithETagReason header describing the reason for the failure is added to the response. This header can have one of the following values:

- Invalid ETag
- The user specified in the ETag is not the authenticated user
- Design Last Modified time
- Database Last Modified time
- Document Last Modified time
- Cache Strategy
- No If-None-Match header
- Server Config

When cache validation is attempted using an If-Modified-Since request header, then the result is specified in the X-Domino-CacheValidationWithLastModifiedResult header. This header can have one of the following values:

- Failed using Last-Modified
- Succeeded using Last-Modified

If cache validation with a Last-Modified time fails, then the X-Domino-CacheValidationWithLastModifiedReason header describing the reason for the failure is added to the response. This header can have one of the following values:

- Design Last Modified time
- Database Last Modified time
- Document Last Modified time
- Application specified Last Modified time
- Cache Strategy
- No If-Modified-Since header
- Server Config
- The response is user specific

The following table describes the values (that is, the reasons for failure) for the X-Domino-CacheValidationWithETagReason and X-Domino-CacheValidationWithLastModifiedReason headers.

Value	**Description**
Invalid ETag	The ETag specified in the If-None-Match header of the request was not generated by the Domino server, or its version is higher than that understood by the server.
The user specified in the ETag is not the authenticated user	For user specific responses, the name of the user that the response was generated for is encoded in the ETag. If this user is not the same as the authenticated user when validating a cached response with an ETag in an If-None-Match header, then the validation fails.
Design Last Modified time	The database design has changed, and the response is dependent on the design modification time.
Database Last Modified time	The database data has changed, and the response is dependant on the database modification time.
Document Last Modified time	The document data has changed and the response is dependent on the document modification time.
Application specified Last Modified time	The application specified Last-Modified time (specified with the X-Domino-CacheValidationDependsOn server directive header) is later than the If-Modified-Since time in the request.
Cache Strategy	The response could not be validated because of the caching strategy. This can cause cache validation to fail for ETags when validation must be done with the If-Modified-Since time (for example, with an application specified Last-Modified time).
No If-Modified-Since header	This causes cache validation using Last-Modified to fail.
No If-None-Match header	This causes cache validation using ETag to fail.
Server Config	This can be caused by DominoDisableETags=1 for cache validation using ETags, or DominoDebugDisableNotModified=1 for all cache validation.
The response is user specific	This happens when an attempt is made to validate a user variant cached response with an If-Modified-Since header. User variant responses can only be validated using ETags.

In addition to these two headers, the following headers—all of which are introduced in Domino 6.0.1—may be included depending upon the request type and the cache validation results.

Header	**Description**
X-Domino-UserName	The canonical user name of the authenticated user
X-Domino-DesignLastModified	The database design Last-Modified time
X-Domino-DatabaseLastModified	The database Last-Modified time
X-Domino-DocumentLastModified	The document Last-Modified time

X-Domino-AppSpecifiedLastMod	The time specified in the X-Domino-CacheValidationDependOn response header added by the application by @SetHTTPHeader
X-Domino-CacheStrategyFlags	The cache strategy flags for the current response
X-Domino-ComputeEvalInfoFlags	The computeEvalInfo flags for the current response
X-Domino-EvalInfoFlags	The evalInfo flags for the current response
X-Domino-ETag-Decoded	The Base64 decoded ETag header value
X-Domino-ETag-Version	The ETag version number
X-Domino-ETag-StrategyFlags	The ETag cache strategy flags (These are the same as the flags that were specified in the X-Domino-CacheStrategyFlags header for the response that generated the ETag.)
X-Domino-ETag-DesignLastModified	The database design Last-Modified time encoded in the ETag
X-Domino-ETag-DatabaseLastModified	The database Last-Modified time encoded in the ETag
X-Domino-ETag-DocumentLastModified	The document Last-Modified time encoded in the ETag
X-Domino-ETag-DocumentUNID	The document UNID encoded in the ETag
X-Domino-ETag-UserName	The canonical user name encoded in the ETag

The following output is excerpted from debug session logs for a single request/response cycle with DominoTraceCacheValidation=1. It shows the HTTP headers for the request and response. You can see that for this request, cache validation was attempted using the ETag specified in the If-None-Match header in the request; however, the validation failed because the design of the database had been modified since the ETag was generated for the cached response. Note that the information in the X-Domino-ETag-... headers relates to the ETag specified in the request, not the ETag returned in the response. Note also that because cache validation failed using the ETag, no attempt at cache validation is performed using the If-Modified-Since date. If the request had not contained an If-None-Match header, then cache validation would have been attempted using the If-Modified-Since date.

```
GET /mail/cdumont.nsf HTTP/1.1
Accept: */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
If-Modified-Since: Fri, 13 Sep 2002 17:33:33 GMT
If-None-Match: W/"MTAtODEwRC04NTI1NkMzMzAwNjA3NDg0LTg1MjU2QzMyMDAyMTI3
REYtMC1DTj1DaHVjayBEdW1vbnQvTz1DZXJ0"
User-Agent: Mozilla/4.0 (compatible; MSIE 5.5; Windows NT 5.0; .NET CLR 1.0.3705)
Host: chuck-dumont3
Connection: Keep-Alive

....

HTTP/1.1 200 OK
Server: Lotus-Domino
Date: Mon, 09 Dec 2002 16:53:19 GMT
Connection: close
Last-Modified: Mon, 09 Dec 2002 16:51:51 GMT
Content-Type: text/html; charset=US-ASCII
Content-Length: 860
X-Domino-CacheValidationWithETagResult: Failed using ETag
X-Domino-CacheValidationWithETagReason: Design Last Modified time
X-Domino-DesignLastModified: 12/09/2002 11:51:50 AM
X-Domino-ETag-Decoded: 10-810D-85256C3300607484-85256C32002127DF-0-CN=Chuck Dumont/O=Cert
X-Domino-ETag-Version: 1.0
```

```
X-Domino-ETag-StrategyFlags: DbDesign DbData UserSpecific
X-Domino-ETag-DesignLastModified: 09/13/2002 01:33:32 PM
X-Domino-ETag-DatabaseLastModified: 09/12/2002 02:02:08 AM
X-Domino-ETag-UserName: CN=Chuck Dumont/O=Cert
X-Domino-CacheStrategyFlags: DbDesign DbData UserSpecific
X-Domino-ComputeEvalInfoFlags: UserVariant DbData
X-Domino-EvalInfoFlags: UsedBrowserInfo
Cache-control: private
ETag: W/"MTAtODEwRC04NTI1NkM4QTAwNUNBMzBGLTg1MjU2QzMyMDAy
MTI3REYtMC1DTj1DaHVjayBEdW1vbnQvTz1DZXJ0"
```

Conclusion

Effective use of browser caching can have a significant impact on server performance by reducing the server workload. Domino 6 adds new features that enable Domino administrators and application developers to better exploit browser caching. The use of ETags allows the server to validate cached responses with greater efficiency and enables caching of user variant responses on browsers. Diagnostic features enable application developers to monitor and understand cache validation results.

ABOUT THE AUTHOR

Chuck Dumont has been a developer on the Domino Web Server team for the past five years and is now working on Lotus' next generation of WebSphere-based collaboration products.