

Ctrl+F5 Won't Save You

Strategies for Static Asset Caching



Christopher Burge

Senior Technical Account Manager, Acquia
(Since April 2022)

First Drupal website
Drupal 6
Spring 2011

First contribution credit
Organic Groups Menu
December 26, 2014

First contrib module
Block Form Alter
July 11, 2019

Scenario

- A nonprofit org posts in annual report to its website and sends out a blast email to its members with a link.
- Almost immediately, staff find and fix an error with the financials in the report.
- Hours later, they notice the URL in the email is still loading the original report with the error.

*Nonprofit's website is running on Apache + Varnish

Scenario

- Original filename: 2026_annual_report.pdf
- Replacement filename: 2026_annual_report_0.pdf
- Varnish is caching original file for 1 year

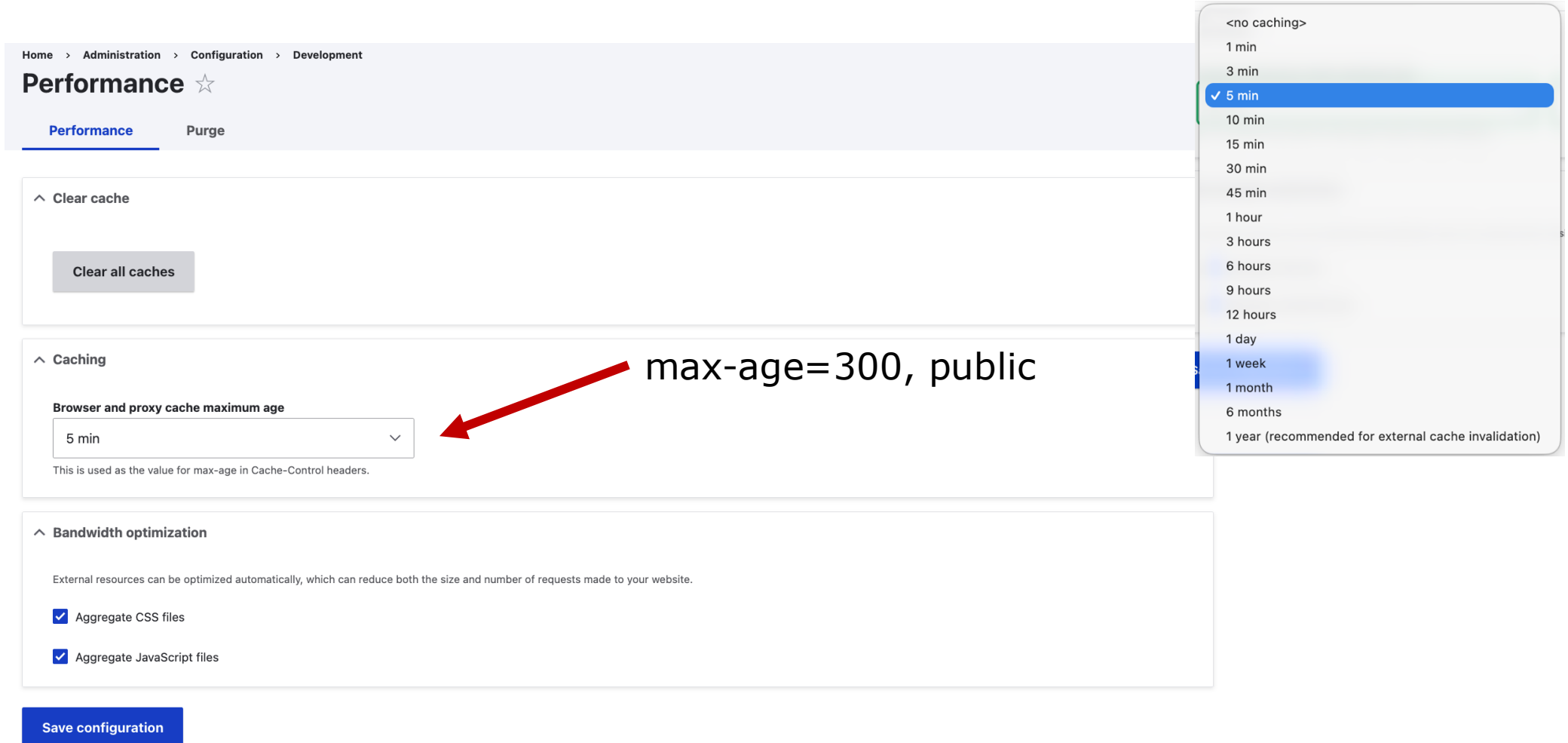
Cache-Control Header

- Header instructs external caches how to behave
 - External caches: Varnish, CDN, browser, etc.
- Examples:
 - max-age=300, public
 - must-revalidate, no-cache, private

Learn more:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/Cache-Control>

Cache-Control (Dynamic Content)



The screenshot shows the 'Performance' configuration page in a web application. The breadcrumb trail is 'Home > Administration > Configuration > Development'. The page has two tabs: 'Performance' (active) and 'Purge'. The 'Clear cache' section contains a 'Clear all caches' button. The 'Caching' section has a dropdown menu for 'Browser and proxy cache maximum age' currently set to '5 min'. A red arrow points from the text 'max-age=300, public' to this dropdown. The dropdown menu is open, showing a list of options from '<no caching>' to '1 year (recommended for external cache invalidation)'. The '5 min' option is selected. The 'Bandwidth optimization' section has two checked checkboxes: 'Aggregate CSS files' and 'Aggregate JavaScript files'. A 'Save configuration' button is at the bottom.

Home > Administration > Configuration > Development

Performance ☆

Performance | Purge

^ Clear cache

Clear all caches

^ Caching

Browser and proxy cache maximum age

5 min

This is used as the value for max-age in Cache-Control headers.

^ Bandwidth optimization

External resources can be optimized automatically, which can reduce both the size and number of requests made to your website.

☒ Aggregate CSS files

☒ Aggregate JavaScript files

Save configuration

max-age=300, public

- <no caching>
- 1 min
- 3 min
- ✓ 5 min
- 10 min
- 15 min
- 30 min
- 45 min
- 1 hour
- 3 hours
- 6 hours
- 9 hours
- 12 hours
- 1 day
- 1 week
- 1 month
- 6 months
- 1 year (recommended for external cache invalidation)

/admin/config/development/performance

Cache-Control (Static Assets)

Requires mod_expires to be enabled.

<IfModule mod_expires.c>

Enable expirations.

ExpiresActive On

Cache all files for 1 year after access.

ExpiresDefault "access plus 1 year"

max-age=31536000



<FilesMatch \.php\$>

Do not allow PHP scripts to be cached unless they explicitly send cache
headers themselves. Otherwise all scripts would have to overwrite the
headers set by mod_expires if they want another caching behavior. This may
fail if an error occurs early in the bootstrap process, and it may cause
problems if a non-Drupal PHP file is installed in a subdirectory.

ExpiresActive Off

</FilesMatch>

</IfModule>

[docroot]/.htaccess

Detour into Philosophy

- Should user-uploaded file names be reusable?
- E.g. 2026_annual_report.pdf

Reusing File Names

- Caching Considerations
 - How can all layers of external cache be invalidated?
- Drupal Core Behavior
 - By default, user-uploaded file paths cannot be reused
 - Orphaned managed files are never deleted*
 - See <https://www.drupal.org/node/2891902> (since D8.4.0)

*2026_annual_report.pdf will continue to be available to anyone with the URL

Replacing Media Entity Files



Media Entity File Replace

Add Issues for Media Entity File Replace
to dashboard +

View

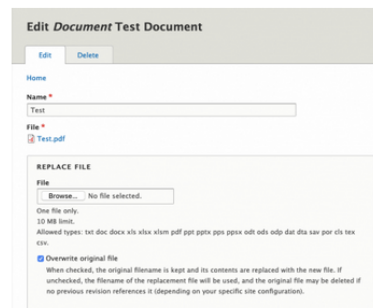
Version control

View history

This module allows content editors to easily replace the source files associated with file-based media types (like "Document"). The replacement file *overwrites* the existing file, keeping the same filename and path, which is usually what content editors want to do when performing a file replacement.

This module operates on the *media entity* level, meaning the file replacement is done by accessing the edit form for a media entity that has a file source field on it. If you're using Drupal's media module for document management, then this is what you want.

Without this module, content editors would typically try and replace a file by editing the media entity, using the "remove" button on the file widget, and uploading a new replacement file with the same filename. However, this does not work as intended, as Drupal will append a number to the end of the replacement file, like "_1" or "_2", instead of actually overwriting the original. The only workaround is to delete the media entity, hope the site is configured to automatically delete unused files on cron runs, wait for that deletion to occur and then upload a new document with the same file



Star

84



Followed

Maintainers



bkosborne



joevagyok

Issues for Media Entity File Replace

To avoid duplicates, please search

https://www.drupal.org/project/media_entity_file_replace

External Caching

- Need to invalidate Varnish and browser cache

External Caching

Purge

[View](#) [Version control](#) [View history](#)

The modular external cache invalidation framework.

The `purge` module facilitates cleaning **external caching systems, reverse proxies** and **CDNs** as content actually changes. This allows external caching layers to keep unchanged content cached infinitely, making content delivery more efficient, resilient and better guarded against traffic spikes.

The `8.x-3.x` versions enable invalidation of content from external systems leveraging Drupal's brand new cache architecture. The technology-agnostic plugin architecture allows for different server configurations and use cases. Last but not least, it enforces a separation of concerns and should be seen as a **middleware** solution (see [README.md](#)).

Getting started

For most simple configurations, start with:

- `drush en purge purge_ui`
- `drush en purge_drush purge_queueer_coretags purge_processor_cron`
- Head over to `admin/config/development/performance/purge`.
- Now you need to install - and probably configure - a third-party module that provides a **purger**. If no module supports invalidation of your cache layer and doing so works over HTTP, then use the generic [purge_purger_http](#).

Third-party integration

This project aims to get all modules dealing with proxies and CDNs on board and to integrate with Purge. As known to date, these modules are or are being integrated:

- [purge_purger_http](#) for generic HTTP-based invalidation, e.g. `nginx`, `squid`, etc.
- [purge_queueer_url](#) for legacy platforms not supporting cache tags. This is a **poor** solution when you regularly import content, it can lead to unsustainable big queues!
- [acquia_purge](#)
- [akamai](#)
- [cloudflare](#)

<https://www.drupal.org/project/purge>

★ Star 75

✉ Followed

Maintainers



japerry



djbobbydra
ke



nielsvm

Issues for Purge

To avoid duplicates, please search before submitting a new issue.

Search

[Advanced search](#)

All issues
88 open, 6 RTBC, 331 total

Bug report
38 open, 3 RTBC, 148 total

Statistics

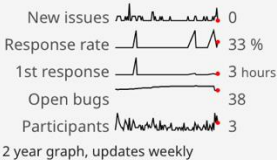
New issues

Response rate

1st response

Open bugs

Participants



0

33 %

3 hours

38

3

2 year graph, updates weekly

Varnish purger

[dashboard +](#)

[View](#) [Version control](#) [View history](#)

What is this?

This is the Varnish purger for the [Purge module](#).

Project name and module name

Because of some bad naming in the start - the project name is `varnish_purge`, but the module itself is called `varnish_purger`. Sorry for that, but it's hard to move namespace after you have registered one.

Varnish configuration

To work with cache tags you need to have a Varnish vcl-file that implements is, [here is a suggestion for default.vcl for Varnish 4 and using BAN](#).

Supported Varnish versions

This module has nothing to do with which version of version you are using, as long as Varnish supports BANs, this module should work - so use Varnish 3 and up with this module.

Image bans

Since release of 8.x-1.7 we now have basic support for image bans with the sub module `varnish_image_purge`, using a VCL setting like the following:

```
sub vcl_recv {
  if (req.method == "URIBAN") {
    ban("req.http.host == " + req.http.host + " && req.url == " +
      req.url);
    # Throw a synthetic page so the request won't go to the backend.
    return (synth(200, "Ban added."));
  }
}
```

https://www.drupal.org/project/varnish_purge

★ Star 36

✉ Followed

Maintainers



deadbeef



javivf



littletough
ts



misc



dieterholvo
et

Issues for Varnish purger

To avoid duplicates, please search before submitting a new issue.

Search

[Advanced search](#)

All issues
25 open, 2 RTBC, 103 total

Bug report
10 open, 2 RTBC, 41 total

Statistics

External Caching

Purge File

[View](#)[Version control](#)[View history](#)

This module purges the URLs of files through the Purge module. The purge is done when the files are either updated or deleted. This functionality is useful when your site allows replacing files maintaining the same URL and is using an external cache for anonymous users like Varnish or Acquia purge.

How to use

First, ensure that your site:

- Have enabled the purge module.
- Have at least one purger enabled that supports URLs.
- Have at least one purge processor enabled.

Then, go to 'Configuration > Development > Performance > Purge file' and set the purge processor that you want to be used, depending on your site requirements:

Purge file ☆

[Performance](#) [Purge](#) [Purge file](#)

[Home](#) » [Administration](#) » [Configuration](#) » [Development](#) » [Performance](#)

Processor *

Late runtime processor ▼

The purge processor that will be used to purge the updated / deleted files.

Save configuration

[★ Unstar](#)

17

[✉ Followed](#)

Maintainers



bryan
toapanta



omarlopesi
no



lpeidro



tunic

Issues for Purge File

To avoid duplicates, please search before submitting a new issue.

https://www.drupal.org/project/purge_file

External Caching

- Varnish invalidation is solved, but what about browsers?
- Browser cache can't be invalidated
- BUT Cache-Control header can instruct to check first

Max-age

- max-age value is relative to the server, NOT the browser (or external cache)
 - A max-age value is evaluated based on the first time Apache served the file.
 - The first time your browser received the asset is irrelevant.
 - Once max-age has been exceeded, it will behave like no-cache (check upstream before serving from cache).
- E.g. If Apache serves an asset with max-age=3600, at 13:00 UTC, all external caches will serve cached copies until 14:00 UTC. After 14:00 UTC, all external caches will behave as if no-cache is set.

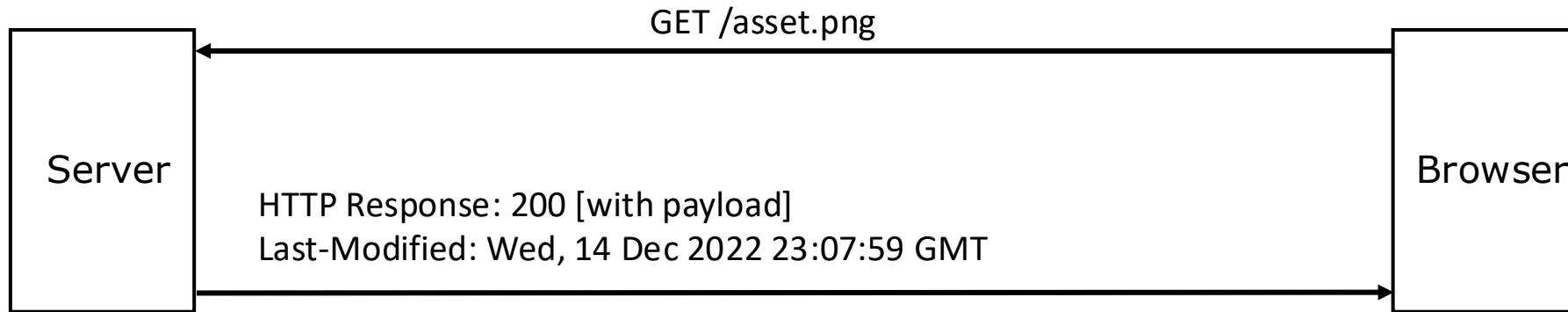
No-cache

- no-cache
 - Response *can* be cached
 - Response *must* be revalidated with origin before used
 - E.g. You can use - but check first

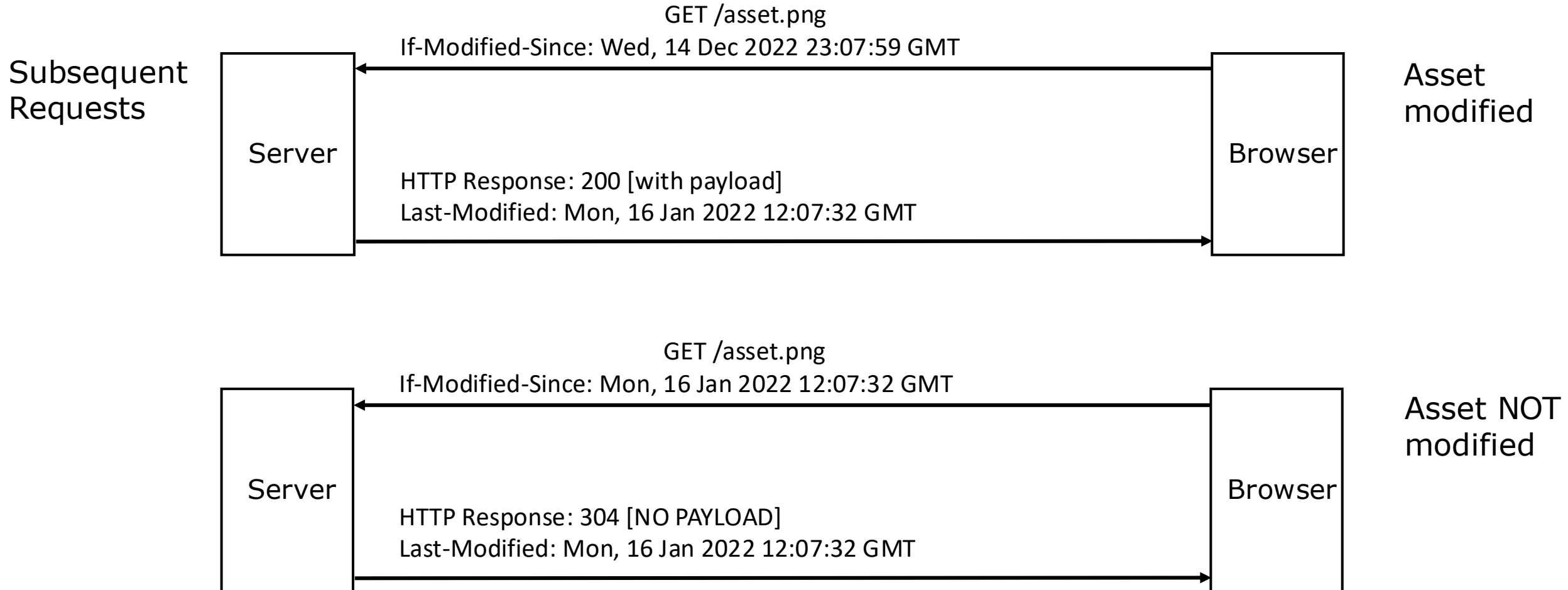
Once max-age has been reached, then behavior is no-cache

Cache Revalidation

Initial
Request



Cache Revalidation



Last-Modified vs ETag

	ETag	Last-Modified
Value representation	Hash of content, modified timestamp, size, etc	Date/time resource was modified on server
Data type	An opaque string E.g. "675af34563dc-tr34"	An HTTP-date E.g. "Fri, 12 Jun 2026 18:40:51 GMT"
Resolution	High precision	Limited to 1-second granularity
Priority	Takes precedence over Last-Modified.	Used as a fallback if the ETag is missing.

One more trick: s-maxage

- s-maxage is max-age but exclusively for shared cache
- Shared cache = Varnish, CDN, etc.
- Shared cache $\neq$ Browser (e.g. private cache)

E.g. Cache-control: max-age=60, s-maxage=3600, public

- Shared cache max-age is 1 hour
- Private cache max-age is 1 minute

Cache-Control for Static Assets

```
<IfModule mod_headers.c>
```

```
  <FilesMatch "\.(ico|jpg|jpeg|png|gif|webp|avif|txt|pdf)$">
```

```
    Header set Cache-Control "max-age=0, s-maxage=3600, public"
```

```
  </FilesMatch>
```

```
</IfModule>
```

Scenario

- File is replaced with identical file name, 2026_annual_report.pdf
- Varnish is purged (Purge File)
- Browsers revalidate requests (Cache-Control header)
- Users instantly see corrected Annual Report PDF.

What about JS and CSS Files?

- JS/CSS files should be defined and attached via Drupal's Libraries system
 - <https://www.drupal.org/node/2274843>
- Core appends a cache-busting string

What about JS and CSS Files?

my_module/css/red-text.css:

```
.text-content {  
  color: red !important;  
}
```

my_module/my_module.libraries.yml:

```
global-styling:  
  css:  
    theme:  
      css/red-text.css: {}
```

my_module/my_module.module:

```
function libraries_test_preprocess_page(&$variables) {  
  $variables['#attached']['library'][] = 'my_module/global-styling';  
}
```

What about JS and CSS Files?

- Cache-busting string is automatically appended:

200	GET	drupal-ca...	contextual.icons.theme.css?tgzj8j	stylesheet	css	1.20 kB	3.40...
200	GET	drupal-ca...	red-text.css?tgzj8j	stylesheet	css	450 B	42 B
200	GET	drupal-ca...	shortcut.theme.css?tgzj8j	stylesheet	css	910 B	1.30 ...

- Cache-busting string is regenerated on cache-rebuild

What about JS and CSS Files?

- D10.2.0 added “asset.query_string” service
 - <https://www.drupal.org/node/3358337>

core/lib/Drupal/Core/Asset/AssetQueryString.php:

```
public function reset(): void {  
    // The timestamp is converted to base 36 in order to make it more compact.  
    $this->state->set(self::STATE_KEY, base_convert(strval($this->time->getRequestTime()), 10, 36));  
}
```

What about JS and CSS Files?

- When aggregation is enabled, no cache-busting string is appended
- E.g. https://drupal-camp-asheville26.ddev.site/sites/default/files/js/js_e8kUYOFpEQBFV1fZnr-5m0heHHPX34zpG3X59RrICMM.js?scope=header&delta=0&language=en&theme=olivero&include=eJxdjmsKwzAMgy-Uxxl2kuImojXz4uC67Y6_DFoG-yWhTwhRfXGbXFVmsnxpcgPSqgeMm6N5mHmZOnfk24Sig7x9J8nV9k6Sfkm8dkbJcGMqzgeicHsGIW-FN8wayssaTK8L_B2yFOh7ffx8BNEBZ

What about JS and CSS Files?

- Aggregated URLs are generated based on assets
 - Don't need cache-busting query string
 - Behavior hinges on library version declaration

global-styling:

version: 1.2.3

css:

theme:

css/red-text.css: {}

What about JS and CSS Files?

- If no library version is declared, then by hashed `file_get_contents()` value
- If library version is declared, then by version
 - Don't forget to increment library versions if declaring!
 - Or maybe don't declare a library version to begin with?
- Beginning on D11.4.0, library version will be disregarded in favor of a hashed `file_get_contents()` value

What about JS and CSS Files?

Always use the file contents to determine asset aggregate filename hashes

[View](#) [Edit](#) [View history](#)

Problem/Motivation

When building asset aggregate URLs, we use a combination of library versions for libraries that have them, and file contents for libraries that don't - this is to save hashing every file in an aggregate.

After [#3505776: Retain asset aggregates for 45 days by default to reduce generation of identical files](#) it is a bit more awkward if you update a file without updating a version for a library, because drush cr doesn't automatically clear the aggregate files. There are ways to disable the behaviour in local dev etc. but it can be a gotcha.

Thinking about whether to open this issue at all, I suddenly realised that all of core's own libraries are defined with `VERSION` which changes every patch release, even though the individual files often don't change for months or years.

This means that for core libraries, the logic is doing the opposite of what we want it to - results in more changes to aggregate filenames rather than less.

We can remove `VERSION` from all core libraries, but given it would also make things easier for contrib/custom themes and modules, I think always using the file contents is probably easier. Can profile it to see what the impact is.

Closed (fixed)	
Project:	Drupal core
Version:	11.4.x-dev
Component:	asset library system
Priority:	Normal
Category:	Task
Assigned:	Unassigned
Reporter:	catch
Created:	8 May 2026 at 12:17 CDT
Updated:	14 Jun 2026 at 21:55 CDT

[★ Follow](#) 6 followers

Jump to comment: [Most recent](#), [Add new](#)

<https://www.drupal.org/project/drupal/issues/3589208>

Static Asset Cache Buster

Static Asset Cache Buster

Buster to dashboard +

[View](#)[Edit](#)[Version control](#)[View history](#)[Maintainers](#)

When static assets, such as images and PDFs, are served by the web server, they are cached by external caches (e.g. Varnish, CDN, browsers). This becomes problematic when a given asset is replaced on the web server.

For example, if <https://example.com/sites/default/files/2022-annual-report.pdf> is updated with a newer file with the same file name, the external caches will continue to serve the old version. This issue exists when using modules like [Media Entity File Replace](#) and [Media: Acquia DAM](#).

This module seeks to address the issue by appending a query string to asset URLs, which acts as a cache buster. The value of the 'cb' query parameter is a hashed/truncated value from the file entity's 'changed' timestamp.

For example, *2022-annual-report.pdf* becomes *2022-annual-report.pdf?cb=64227fd1*. When the file is updated, the value of the 'cb' query string parameter will change.

This module doesn't provide any configuration.

★ Star 7

✉ All issues

Maintainers



chris burge

Issues for Static Asset Cache Buster

https://www.drupal.org/project/static_asset_cache_buster

Static Asset Cache Buster

- Cache Buster Query String:

```
function _static_asset_cache_buster_get_cache_buster_query(string $timestamp): array {  
    return [  
        'cb' => substr(hash('md5', $timestamp), 0, 8),  
    ];  
}
```

- Result e.g.: 2026_annual_report.pdf?cb=9e94c91c

Static Asset Cache Buster

- Use either a "Path wildcard" or a "URL wildcard" invalidation type for Purge configuration
- By default, 2026_annual_report.pdf will be purged
 - 2026_annual_report.pdf?cb=9e94c91c will not be purged
- Using wildcard will purge 2026_annual_report.pdf*

Practical Considerations

- Always perform end-to-end testing with caching changes.
 - Specifically test cache invalidation.
- Major browsers appear to cache JS, CSS, fonts, and images
- Firefox and Edge cache but validate PDFs
- Safari doesn't cache PDFs
- Chrome??
 - "Your organization has blocked the use of DevTools on this page"

Baseline Recommendations

- It depends...
- If filenames *cannot* be reused
 - Then you probably don't need to do anything
- If filenames *can* be reused
 - Media Entity File Replace + Purge File + Cache-Control [OR]
 - Media Entity File Replace + Purge File + Static Asset Cache Buster

Related Media Modules

- https://www.drupal.org/project/media_download
- https://www.drupal.org/project/media_entity_download
- https://www.drupal.org/project/media_link_enhancements
- https://www.drupal.org/project/media_entity_link
- https://www.drupal.org/project/orphaned_files

Questions? Comments?

Thank an organizer!