BROWSER
CACHE
BOOSTER

User Manual

Version 1.4.0

Digital Venture
Unternehmensberatung GmbH
Römerstr. 16
82131 Gauting
Germany
Tel: +49 89 548466-0
E-Mail: info@browsercachebooster.com

Table of Contents

Benefits

The DVU Browser Cache Booster (BCB) is an extension for the Adobe® Experience Manager (AEM) that helps to avoid redundant web browser requests for static resources while simultaneously preserving the possibility for content authors to change their resources at any time.

This is achieved by automatically adding a “tag” (a fingerprint, represented by a hash value) to the file name of each static resource that is delivered. Additionally, basic HTML optimizations such as whitespace compression, comment removal or the compression of inline JavaScript and CSS code are applied.

Furthermore, the BCB maintains an internal resource database that tracks the requested resources as well as resource dependencies derived from the returned HTML and CSS code. This allows the AEM dispatcher cache to be flushed selectively rather than on a subtree basis. Thus, the number of resources that can be served directly from the AEM dispatcher cache is maximized and a significant amount of load is taken from the AEM publish instance that otherwise would be required to re-render unchanged resources.

The BCB does not interfere with AEM's internal processing. It is designed as an HTTP preprocessor and postprocessor that only reads a few details from AEM in order to optimally fulfill its task. This design makes the BCB invisible to AEM, ensuring maximum compatibility with AEM.

By storing as many static resources as possible in the web browser cache and maximizing the files that can be served directly from the AEM dispatcher cache, the BCB contributes to

- minimize the number of requests that are sent across the network
- reduce the data transfer volume for returning users
- lower the server load
- facilitate the web server management
- **maximize the user experience**

Setup

The DVU Browser Cache Booster has been carefully engineered and pre-configured to meet the requirements of most AEM installations. Nonetheless, it is best practice to

1) evaluate and test the BCB and its configuration in a development environment with you AEM application
2) validate the proper function and behavior of the BCB in a production-related integration environment
3) always create a full backup of the AEM instance before you install the BCB.

Failure to comply with these best practices can result in loss of data, business interruption, or other commercial damage.

Pre-Installation Steps

Forwarded-Header

In typical AEM installations, the rendered HTML and CSS code references resources just by their request path (without protocol, host and port). If your AEM installation also just references resources by their request path, you can skip this section and continue with URL Rewriting.

To distinguish between internal and external resources, the BCB needs to know the original protocol, host and port that was used by the client web browser. This information is available when an HTTP request is received from the Internet but may get lost on its journey through the corporate network.

To preserve this information, it is required that the first recipient of the incoming HTTP request adds the original protocol, host and port to the request before the request is passed on to the next recipient in the chain.

If content delivery networks (CDNs), reverse proxy servers or load balancers are part of your IT infrastructure, you must either add the

- `Forwarded` (RFC 7239) request header
  (for example: `Forwarded: proto=https; host=www.myhost.com:4711`)

or the legacy de-facto standard request headers

- `X-Forwarded-Proto`
  The name of the originally used schema
  (for example: `X-Forwarded-Proto: https`)
- `X-Forwarded-Host`
  The name of the originally specified host and port
  (for example: `X-Forwarded-Host: www.myhost.com:4711`)

to each incoming HTTP request.

The `Forwarded` request header takes precedence over the legacy request headers.

If the above request headers cannot be provided by your IT infrastructure, you will have to configure all possible resource origins manually (see Resource Origins for details).

- See Appendix 1 for a sample Apache web server configuration file that creates the `Forwarded` and `X-Forwarded-*` HTTP request headers.
- In case you filter the HTTP request headers that are passed on to the AEM instance, do not forget to adjust the `/clientheaders` section of your AEM dispatcher `dispatcher.any` configuration file accordingly.

URL Rewriting

In typical AEM installations, only the request paths of `html` resources are shortened (e.g. by removing the `/content/<my-app>` request path prefix) to make them easier to read and remember. If your AEM installation also just rewrites the request paths of `html` resources, you can skip this section and continue with AEM OSGi Settings.

If the incoming request path of a non-`html` resource is rewritten (e.g. via Apache's `mod_rewrite`) before it is passed on to the AEM instance, it will be necessary to set two additional custom HTTP headers that allow the BCB to match the incoming request path seen by the AEM instance with the resource path that was used in the outgoing HTML or CSS code:

- `X-Request-URI`
  The value of the unmodified Apache web server `REQUEST_URI` variable
  (for example: `X-Request-URI: /us/flag.png`)
- `X-Query-String`
  The value of the unmodified Apache web server `QUERY_STRING` variable
  (for example: `X-Query-String: foo=bar&bar=foo`)

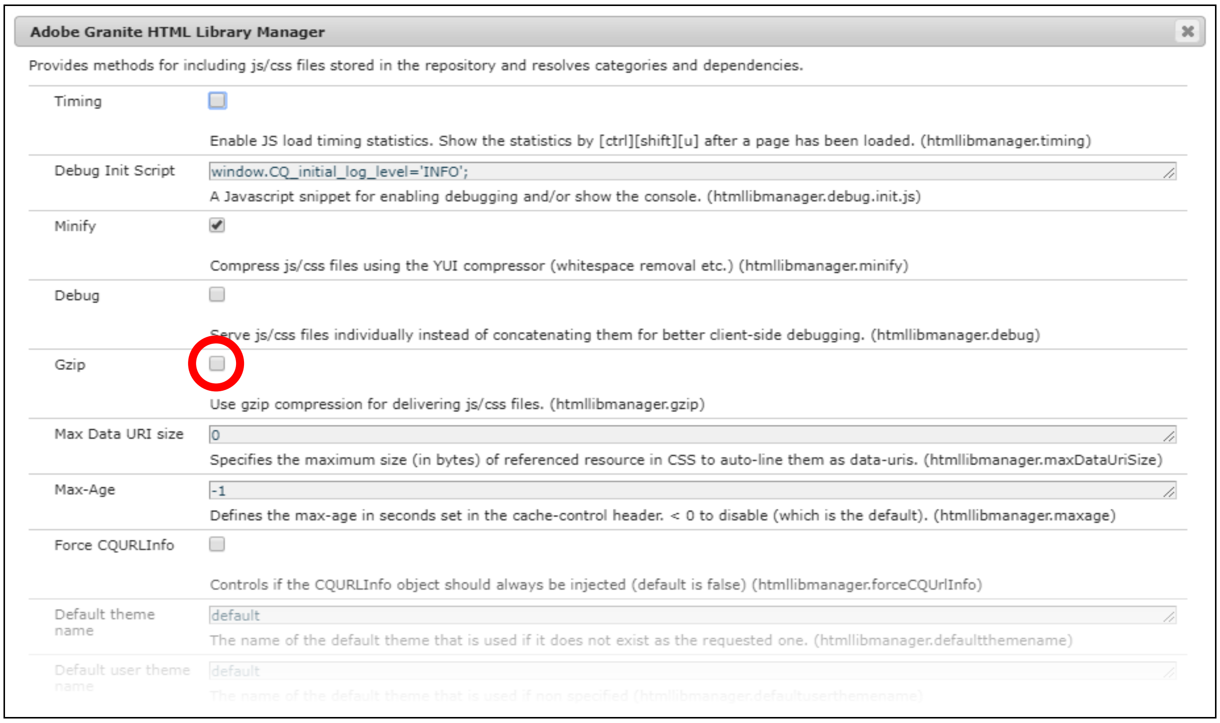
In case you filter the HTTP request headers that are passed on to the AEM instance, do not forget to adjust the `/clientheaders` section of your AEM dispatcher `dispatcher.any` configuration file accordingly.

AEM OSGi Settings

Even though the BCB can handle both compressed and uncompressed resources, for performance reasons it is advisable to disable `Gzip` (GNU zip) compression in the `Adobe Granite HTML Library Manager` OSGi configuration. This eliminates the need for the BCB to decompress resources before they can be processed. The `Gzip` compression will be performed later either by the BCB (see Compression Method) or the AEM dispatcher when the resource processing is complete.

To disable `Gzip` compression in the `Adobe Granite HTML Library Manager` configuration, please execute the following steps:

- Log in to the AEM instance as user `admin` (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on `Adobe Granite HTML Library Manager`
- Uncheck `Gzip`
- Click the `[Save]` button



Adobe Granite HTML Library Manager

Provides methods for including js/css files stored in the repository and resolves categories and dependencies.

Timing	☐ Enable JS load timing statistics. Show the statistics by [ctrl][shift][u] after a page has been loaded. (htmllibmanager.timing)
Debug Init Script	window.CQ_initial_log_level='INFO'; A Javascript snippet for enabling debugging and/or show the console. (htmllibmanager.debug.init.js)
Minify	✓ Compress js/css files using the YUI compressor (whitespace removal etc.) (htmllibmanager.minify)
Debug	☐ Serve js/css files individually instead of concatenating them for better client-side debugging. (htmllibmanager.debug)
Gzip	☐ Use gzip compression for delivering js/css files. (htmllibmanager.gzip)
Max Data URI size	0 Specifies the maximum size (in bytes) of referenced resource in CSS to auto-line them as data-uris. (htmllibmanager.maxDataUriSize)
Max-Age	-1 Defines the max-age in seconds set in the cache-control header. < 0 to disable (which is the default). (htmllibmanager.maxage)
Force CQURLInfo	☐ Controls if the CQURLInfo object should always be injected (default is false) (htmllibmanager.forceCQUrlInfo)
Default theme name	default The name of the default theme that is used if it does not exist as the requested one. (htmllibmanager.defaultthemename)
Default user theme name	default The name of the default theme that is used if non specified (htmllibmanager.defaultuserthemename)

Installation

The BCB is provided as a standard AEM package that can be installed via the *AEM Package Manager*.

- If you have purchased a BCB license, please **install the BCB license package first.** The BCB license package is identical for all AEM versions.
- Install the BCB package. **Never install a BCB package that was created for an AEM version that is higher than the AEM version you run.** Ideally, install the BCB package that matches your AEM version. The Syntax for the BCB package name is

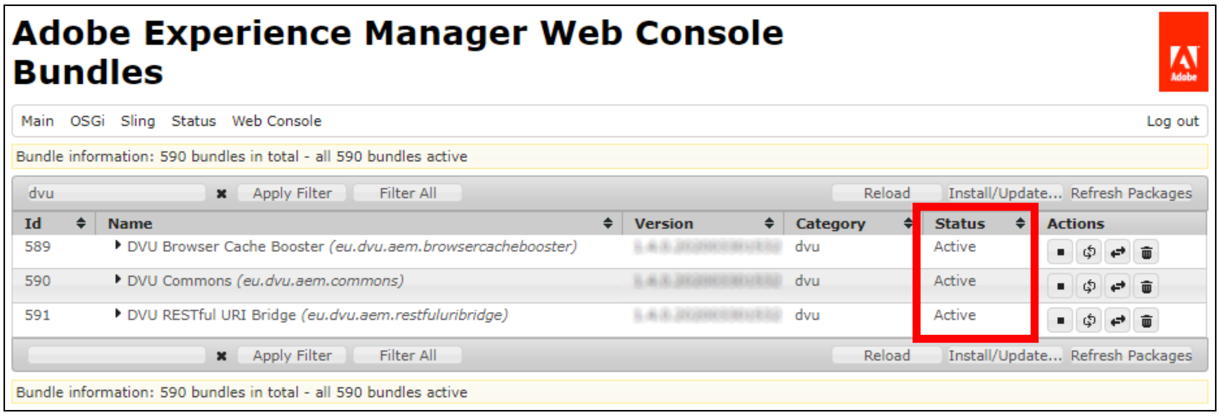
  `browsercachebooster-<BCB version>-aem-<AEM version>.zip`

To install an AEM package, please execute the following steps:

- Surf to `<protocol>://<host>:<port>/crx/packmgr/index.jsp`
- Log in as user `admin`
- Click the `[Upload]` button
- Click the `[Browse]` button, select the package file and click on the `Open` button
- Click the `[OK]` button
- When the package file was successfully uploaded and appears at the top of the package list, click on `Install`

The BCB is active as soon as the installation is complete. To verify the success of the installation, please follow the steps below:

- Surf to *<protocol>*://*<host>*:*<port>*/system/console/bundles
- In the filter input field, enter the filter term `DVU`
- Click the `[Apply Filter]` button
- Verify that the status of all listed bundles is `Active`



Adobe Experience Manager Web Console Bundles

Main OSGi Sling Status Web Console — Log out

Bundle information: 590 bundles in total - all 590 bundles active

dvu ✖ Apply Filter Filter All — Reload Install/Update... Refresh Packages

Id	Name	Version	Category	Status	Actions
589	DVU Browser Cache Booster *(eu.dvu.aem.browsercachebooster)*	[redacted]	dvu	Active	
590	DVU Commons *(eu.dvu.aem.commons)*	[redacted]	dvu	Active	
591	DVU RESTful URI Bridge *(eu.dvu.aem.restfuluribridge)*	[redacted]	dvu	Active	

✖ Apply Filter Filter All — Reload Install/Update... Refresh Packages

Bundle information: 590 bundles in total - all 590 bundles active

Post-Installation Steps

Dispatcher Configuration

To allow the response headers generated by the BCB to be delivered to the client web browser, response header caching must be activated in the AEM dispatcher web server module configuration.

Response header caching is supported from AEM dispatcher web server module version 4.1.11 onwards.

- If you manually configured Apache directives to set the `Cache-Control` response header, do not forget to disable these directives now.
- It is strongly recommended that you always use the latest AEM dispatcher web server module. The latest AEM dispatcher web server module can be downloaded from `https://docs.adobe.com/content/help/en/experience-manager-dispatcher/using/getting-started/release-notes.html`

On the web server machine, locate the configuration file the AEM dispatcher web server module is using (the configuration file is typically called `dispatcher.any` but the file name may vary depending on the system administrator's preferences; to locate the dispatcher configuration file in use, recursively search all active Apache configuration files for the file path specified with the `DispatcherConfig` directive).

- Open the dispatcher configuration file in your favorite text editor
- Locate the `/cache` section
- Underneath the `/cache` section, create the following configuration directives

```
/headers
  {
  "Cache-Control"
  "Content-Disposition"
  "Content-Type"
  "Last-Modified"
  "X-Content-Type-Options"
  }
```

- Save the changes
- Restart the web server
- **Manually delete all existing content from the AEM dispatcher cache directory**

Flush Agent Configuration

The BCB ships with preconfigured `Browser Cache Booster Flush` agents for AEM author and AEM publish instances that both support resource-only flushing.

- Resource-only flushing is a highly recommended but optional feature.
- If you choose to keep the standard AEM dispatcher flush agent, the BCB will continue to delete all affected resources from the AEM dispatcher cache and AEM will additionally invalidate the entire directory subtree.

- An AEM instance can be connected to multiple AEM dispatchers, but in the opposite direction, the AEM dispatchers can only be connected to that exact AEM instance.
- The dispatcher cache of an AEM publish instance can still be flushed from an AEM author instance by means of a standard AEM dispatcher flush agent, provided the BCB on the AEM author instance is configured not to use this flush agent for its own purposes (see DVU Resource Flusher Configuration for details).
- Using the `Browser Cache Booster Flush` agent without enabled BCB results in outdated files being served from the AEM dispatcher cache.

The following steps describe the procedure how to enable resource-only flushing on an AEM instance:

- Open your preferred web browser
  - For AEM author instances, surf to `<protocol>://<host>:<port>/etc/replication/agents.author.html`
  - For AEM publish instances, surf to `<protocol>://<host>:<port>/etc/replication/agents.publish.html`
- On the `Agents on author/publish` page, search for and click on your currently enabled flush agent (e.g. `Dispatcher Flush`)
- On the flush agent page, click the `[Edit]` button.
- In the tabbed dialog, under the `Transport` tab, copy the entire value of the `URI` field to the clipboard (`CTRL-C`)
- In the tabbed dialog, under the `Settings` tab, uncheck the `Enabled` check box
- Click the `[OK]` button
- Click the web browser's `[Back]` button to return to the `Agents on author/publish` page
- Click the web browser's `[Reload]` button to refresh the `Agents on author/publish` page
- Search for and click on the `Browser Cache Booster Flush` agent
- On the `Browser Cache Booster Flush` agent page, click the `[Edit]` button.
- In the tabbed dialog, under the `Transport` tab, replace the entire value of the `URI` field with the content of the clipboard (`CTRL-V`)
- In the tabbed dialog, under the `Settings` tab, check the `Enabled` check box
- Click the `[OK]` button
- On the `Browser Cache Booster Flush` agent page, click on the `Test connection` link to make sure that the `Browser Cache Booster Flush` agent is operational

Customization

DVU Browser Cache Booster Configuration

The DVU Browser Cache Booster Configuration defines all configurable settings of the BCB.

Changing the OSGi configuration of the BCB may require you to **manually delete** all existing content from the AEM dispatcher cache directory.

- Log in to the AEM instance as user `admin` (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on `DVU Browser Cache Booster Configuration`

Enable Tagger

If checked, static resource URLs will be tagged and/or cached. If unchecked, static resource URLs will neither be tagged nor cached, but previously tagged static resource URLs will be untagged before they are passed on to avoid 404 (not found) responses.

The default value is checked.

Resource Origins

The resource origins define, from a web browser perspective, the protocol, host and port number of resources that reside on this AEM instance. A resource origin is specified as a URL without path, query and fragment, but must at least contain protocol and host (e.g. `http://localhost` or `http://localhost:4503`). If references to local resources are consistently specified without an origin (that is, they only consist of path, query and fragment), it is not necessary to specify resource origins.

There is no default value.

Excluded Resources

The excluded resources define regular expressions for resources that should be ignored with respect to tagging and caching. The regular expressions must match the request path and query of the respective resources and **are evaluated in the order in which they are specified. The first matching regular expression wins.**

Resources that are solely used for maintenance or operational purposes and do not affect website user content can be excluded.

If a servlet produces line-oriented output (e.g. progress information), the request path of the servlet must be excluded so that the output lines are displayed immediately.

The default values are:

Regular Expression	*Description*
`/(bin|crx|system|admin)([./?].*)?`	Excludes all servlet-based resources
`/etc/(replication|workflow|taskmanagement)([./?].*)?`	Excludes all AEM-internal resources
`/libs/granite/ui/content([./?].*)?`	Excludes all AEM maintenance resources

Compress HTML Whitespaces
If checked, whitespaces in HTML resources will be compressed.

Whitespace compression will not remove all whitespaces but rather replace multiple subsequent whitespaces with a single whitespace because there are cases in which whitespaces are syntactically required (e.g. the trailing new line after the "`<pre>`" tag).

The default value is checked.

Compress JavaScript Whitespaces
If checked, whitespaces in **inline JavaScript** embedded in HTML resources will be compressed.

This setting only affects HTML resources and has no impact on external JavaScript resources (to compress those resources as well, check `Minify` in the `Adobe Granite HTML Library Manager` OSGi configuration)

The default value is checked.

Compress CSS Whitespaces
If checked, whitespaces in **inline CSS** embedded in HTML resources will be compressed.

This setting only affects HTML resources and has no impact on external CSS resources (to compress those resources as well, check `Minify` in the `Adobe Granite HTML Library Manager` OSGi configuration)

The default value is checked.

Remove HTML Comments
If checked, comments in HTML resources will be removed.

Server Side Include (SSI) directives as well as legacy Internet Explorer "Conditional Comments" will not be removed.

The default value is checked.

Rewrite HTML Hyperlinks
If checked, hyperlinks in HTML resources will be rewritten if the link target is taggable.

This configuration setting only affects downloadable resources (e.g. PDF files) that are referenced with an `<a href="… ">` tag.

The default value is unchecked.

Rewritable `data-`Attributes

The rewritable `data-` attributes specify the names of custom HMTL `data-` attributes that contain rewritable resource URLs. If you specify a resource URL statically in a custom HTML `data-` attribute the resource URL can be rewritten and linked to the page on which it is used even though the resource itself continues to load dynamically via JavaScript.

There are no default values.

Tag Prefix

The tag prefix is prepended to the hash code of a resource to make the resulting tagged resource URL distinctive. If the tag prefix is empty, only the hash code of the resource will be used for the tagged resource URL (not recommended since ambiguous tags might lead to unpredictable results).

The default value is "`~`".

Hash Function

The hash function is used to create a hash code (fingerprint) of the resource data. The hash code is subsequently used to determine whether the resource data has changed.

In information technology, a hash function is any function that can be used to map input data of arbitrary length to output data of fixed length, the so-called "hash code". Hash codes are typically used to speed up data lookup (e.g. hash tables) or to ensure data integrity (e.g. checksums). If different input data results in identical hash codes, this is called a "hash collision". Hash collisions are inevitable and the risk of being affected by a hash collision can only be reduced by increasing the length of the hash code. Hash functions are typically most efficient if the hash code length equals the length of a processor register (which is normally 32, 64 or 128 bits long, depending on the CPU type). As a rule of thumb: the longer the hash code, the less hash collisions, but the more computational effort.

The following hash functions are available:

Hash Function	*Description*
`CRC-32C (32-bit)`	`https://en.wikipedia.org/wiki/Cyclic_redundancy_check`
`xxHash-64 (64-bit)`	`https://github.com/Cyan4973/xxHash`
`MurmurHash3 (128-bit)`	`https://en.wikipedia.org/wiki/MurmurHash`

The default value is "`xxHash-64 (64-bit)`".

Default `Cache-Control` Directives

The default `Cache-Control` directives are applied whenever a resource cannot be tagged (because it is not referenced in the HTML or CSS code and can therefore not be rewritten). The authoritative description of relevant Cache-Control directives can be found in RFC 7234, section 5.2.1. The default Cache-Control directives can be overridden by a resource rule.

The default value is "" (empty).

Tagged `Cache-Control` Directives
The tagged `Cache-Control` directives are applied to all tagged resources.

The following `Cache-Control` directives can be configured:

Directive	*Description*
`max-age=31536000`	The resource can be cached for one year.
`immutable`	The resource is `immutable` (most web browsers do not yet support this directive; FireFox supports this directive only if the HTTPS protocol is used).
`max-age=31536000, immutable`	The blend of both, in case a web browser does not yet support the `immutable` directive (FireFox also supports the blended directives, however, only if the HTTPS protocol is used).

Regardless of which directive you choose, a resource will anyway be evicted from the web browser cache if it is not requested on a regular basis.

The default value is "`max-age=31536000`".

Resource Rules
The resource rules define how specific resources are handled. A resource rule must consist of a regular expression for the request path and query that must end with a semicolon and be followed by a semicolon-separated list of instructions that will be applied to every resource that matches the regular expression. An instruction must consist of an instruction name, a colon and a value. Allowed instruction names are `taggable`, `cacheable` and `dispatcher`. The `taggable` and `dispatcher` instructions only support Boolean values (`false`, `true`) while the `cacheable` instruction additionally supports the exact `Cache-Control` directives to be applied to a matching resource. Resources without a file extension and HTML resources are untaggable by default but are cacheable. If an instruction name is omitted in a rule, its value will be assumed to be `true`. **The resource rules are evaluated in the order in which they are specified. The first matching rule wins.**

- If `cacheable` is set to `false`, no `Cache-Control` directives will be added to the response and the web browser will decide on the caching itself.
- `https://regex101.com/` is a handy online tool for developing and testing regular expressions.

The default values are:

Resource Rule	*Description*
.*/contexthub(\.kernel\.js)?; taggable: false; cacheable: private, max-age=3600; dispatcher: false	Matches resources whose request path ends with “`/contexthub`” or “`/contexthub.kernel.js`”; these resources cannot be tagged and must not be cached by the dispatcher but are served from the web browser cache for at most 3600 seconds (one hour). **This rule should be removed if AEM’s ContextHub framework is not used.**
.*/[^/]*segment[^/]*\.js; taggable: false; cacheable: private, max-age=3600; dispatcher: false	Matches “`.js`” resources whose file name contains the word “`segment`”; these resources cannot be tagged and must not be cached by the dispatcher but are served from the web browser cache for at most 3600 seconds (one hour). **This rule should be removed if AEM’s ContextHub framework is not used.**
`.*?(/[^.]*\.?|\?.*); taggable: false; cacheable: false; dispatcher: false`	Matches all resources without a file extension or with a query; these resources cannot be tagged and must not be cached by the dispatcher. The web browser decides on the caching itself.
`.*\.json; taggable: false; cacheable: false; dispatcher: false`	Matches “`.json`” resources; these resources cannot be tagged and must not be cached by the dispatcher. The web browser decides on the caching itself.
`.*\.html; taggable: false; cacheable: false`	Matches “`.html`” resources; these resources cannot be tagged but can be cached by the dispatcher. The web browser decides on the caching itself.
`.*\.(js|css); taggable: true; cacheable: max-age=3600`	Matches “`.js`” and “`.css`” resources; these resources can be tagged, cached by the dispatcher and served from the web browser cache for at most 3600 seconds (one hour) while they are not tagged.

`.*\.(jpe?g|webp|png|gif|svg|woff2?|ttf|ico); taggable: true; cacheable: max-age=3600`	Matches "`.jpeg`", "`.jpg`", "`.webp`", "`.png`", "`.gif`", "`.svg`", "`.woff`", ."`.woff2`", "`.ttf`" and "`.ico`" resources; these resources can be tagged, cached by the dispatcher and served from the web browser cache for at most 3600 seconds (one hour) while they are not tagged.
`.*\.(pdf|flv|qt|m(ov|p(4|3))); taggable: true; cacheable: max-age=3600`	Matches "`.pdf`", "`.flv`" , "`.qt`", "`.mov`", "`.mp4`", and "`.mp3`" resources; these resources can be tagged, cached by the dispatcher and served from the web browser cache for at most 3600 seconds (one hour) while they are not tagged.
`.*\.conf; taggable: false; cacheable: false; dispatcher: false`	Matches "`.conf`" resources; these resources cannot be tagged and must not be cached by the dispatcher. The web browser decides on the caching itself.

Default `Content-Disposition`

The default `Content-Disposition` specifies the proposed web browser action if a user wants to open an individual resource embedded in a web page (e.g. an image).

The default value is "`inline`".

Content-Disposition	*Description*
`inline`	Expresses the preference to open the selected resource in the web browser (provided the MIME type of the resource can be displayed in the web browser, otherwise it will be offered for download).
`attachment`	Expresses the preference to always offer the selected resource for download.

Enable International File Names

If checked, international (non-US-ASCII) file name support will be enabled when a resource is downloaded.

The default value is unchecked.

MIME Type Mappings

The MIME type (media type) mappings allow to replace deprecated MIME types with the latest IANA-assigned MIME types. MIME type mappings are solely applied to the `Content-Type` of a response. A MIME type mapping must consist of the old MIME type, an equals sign, and the new MIME type.

A new file type (e.g. a new image format) requires new MIME type. Web browsers usually quickly support a new file type, but the standardization process of the corresponding

MIME type takes much longer. To bridge this gap, web browser manufacturers define a proprietary MIME type that may or may not equal the final IANA-assigned MIME type.

The default values are:

`text/javascript=application/javascript`
`application/x-javascript=application/javascript`
`application/x-font-woff2=font/woff2`
`application/font-woff2=font/woff2`
`application/x-font-woff=font/woff`
`application/font-woff=font/woff`
`application/x-font-opentype=font/otf`
`application/x-font-otf=font/otf`
`application/font-otf=font/otf`
`application/x-font-truetype=font/ttf`
`application/x-font-ttf=font/ttf`
`application/font-ttf=font/ttf`

Compressible MIME Types

The compressible MIME types specify the MIME type of (typically text-based) resources that can be compressed efficiently.

In general, all resources are compressible, but for some resources the cost-benefit ratio is better than for others.

MIME types (in the meantime renamed to the broader term “Media Types” because MIME originally stands for “Multipurpose Internet Mail Extensions”) are **standardized** and can be looked up at `https://www.iana.org/assignments/media-types/media-types.xhtml`.

The default values are:

MIME Type (Media Type)	*Description*
`text/html`	e.g. “`.html`”, “Hypertext Markup Language”
`text/css`	e.g. “`.css`”, “Cascading Style Sheet”
`application/javascript`	e.g. “`.js`”, “JavaScript”
`text/plain`	e.g. “`.txt`”, plain text
`application/json`	e.g. “`.json`”, “JavaScript Object Notation”
`text/xml`	e.g. “`.xml`”, “Extensible Markup Language”
`application/xml`	e.g. “`.xml`”, “Extensible Markup Language”
`application/rss+xml`	e.g. “`.rss`”, “Rich Site Summary”
`image/svg+xml`	e.g. “`.svg`”, “Scalable Vector Graphics”

Compression Method

The compression method defines how compressible MIME types are compressed. If the selected compression method is not supported by the requesting web browser, the requested resource will not be compressed.

Currently only the `Gzip` method (`https://en.wikipedia.org/wiki/Gzip`) and the `Deflate` method (`https://en.wikipedia.org/wiki/DEFLATE`) are supported where the `Gzip` method is commonly considered to be superior to the `Deflate` method.

Ideally, compression should only be performed by a system that ultimately delivers content to the website user (typically the AEM dispatcher). Compression is a trade-off of CPU time for bandwidth. In the LAN environment of a data center, there is typically no shortage of bandwidth.

The default value is “No compression”.

Buffer Size

The buffer size specifies how much response data will be kept in main memory when input/output operations are performed. The greater the buffer, the faster the response data can be read or written. The determining factors are the number of simultaneous requests and the available main memory.

The default value is "`24 KB`".

Temporary File Directory

The temporary file directory specifies the directory that will be used whenever a temporary file must be created. If a temporary file directory is specified, it will be created automatically in case it does not already exist. If no temporary file directory is specified, the default Java temporary file directory (`system.io.tmpdir`) will be used.

The default value is "" (empty).

Save Period

The save period defines after how much time the resource state will periodically be saved to file. Supported units are "`seconds`", "`minutes`", "`hours`" and "`days`".

The save period must not be shorter than 5 minutes (300 seconds).

The default value is "`1 hour`".

DVU Browser Cache Booster Resource Flusher

The `DVU Browser Cache Booster Resource Flusher` is used by the BCB to send `delete` requests to the dispatcher.

- Log in to the AEM instance as user `admin`
  (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to
  `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on
  `DVU Browser Cache Booster Resource Flusher`

Default Flush Mode

The default flush mode specifies how resources are flushed.

- `Asynchronous` means that flushes are performed via the replication queue of a flush agent.
- `Synchronous` means that flushes are performed directly bypassing the flush agent's replication queue. For safety reasons, synchronous flush jobs, however, are also stored in the JCR.

⚠

- Setting the default flush mode to `Synchronous` may result in a high system load if many files need to be deleted from the AEM dispatcher cache at the same time or if an AEM dispatcher is unreachable.
- To limit the maximum system load, the `Maximum Parallel Jobs`, `Maximum Retries` and `Retry Delay` properties of the `DVU Browser Cache Booster Flush Job Queue` OSGi configuration must be customized according to your specific requirements.

The default value is `Asynchronous`.

DVU Browser Cache Booster Resource Flusher Configuration

The DVU Browser Cache Booster Resource Flusher Configuration defines the configuration for a specific flush agent. It can be used to control flush operations initiated by the BCB on resource level.

ℹ

- If no resource flusher configurations are defined, all enabled flush agents will be triggered.
- If, however, resource flusher configurations are defined, only the configured and enabled flush agents will be triggered.
- The `Triggers` settings of a flush agent are disregarded.

- Log in to the AEM instance as user `admin`
  (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to
  `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on
  `DVU Browser Cache Booster Resource Flusher Configuration`
- Click on the `[+]` button to add a new configuration

Flush Agent

The flush agent specifies the exact **node name** (not the title, e.g. `browsercachebooster-flush`) of the flush agent to trigger. Please note that from the moment a resource flusher configuration is defined, **only** configured flush agents will be triggered.

Request Path Patterns

The optional request path patterns allow to define regular expressions that are applied to the request path of a resource. A pattern can be negated by a leading exclamation mark. Please note that from the moment request path patterns are defined, **only** request paths that match one of the specified patterns will be flushed. The first matching pattern wins.

DVU Browser Cache Booster Flush Scheduler

The DVU Browser Cache Booster Flush Scheduler can be used to periodically flush resources from the AEM dispatcher cache. This allows dynamic resources that do not change with every request to be delivered from the AEM dispatcher cache (e.g. daily statistics). Please remind that only resources that have a file extension and no query can be stored in the AEM dispatcher cache.

DVU Browser Cache Booster Flush Scheduler Configuration

The DVU Browser Cache Booster Flush Scheduler Configuration allows to define the schedule and the request path patterns of a flush task.

- Log in to the AEM instance as user `admin`
  (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to
  `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on
  `DVU Browser Cache Booster Flush Scheduler Configuration`
- Click on the `[+]` button to add a new configuration

Cron Expression

The cron expression specifies when to delete the respective resources from the AEM dispatcher cache. The syntax of a cron expression is explained on the Quartz Scheduler website.

An online cron expression generator can be found at

`https://www.freeformatter.com/cron-expression-generator-quartz.html`

Request Path Patterns

The request path patterns allow to define regular expressions that are applied to the request path of a resource. All resources that match one the specified request path patterns will be deleted from the AEM dispatcher cache. The first matching pattern wins.

If a matching resource is statically embedded by other resources, those other resources will be deleted from the AEM dispatcher cache as well.

DVU Browser Cache Booster Resource Manager

The DVU Browser Cache Booster Resource Manager resolves, monitors and administers resources.

Resource resolution is a relatively elaborate process because the JCR must be accessed to retrieve node properties and access permissions. Therefore, this information is cached to prevent redundant resource resolutions.

The resource resolution cache does not need to be very large. After the dispatcher has cached all static resources, the BCB only needs to resolve repetitively requested dynamic resources that are loaded via JavaScript.

- Log in to the AEM instance as user `admin`
  (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to
  `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on
  `DVU Browser Cache Booster Resource Manager`

Resource Resolution Cache Capacity

Maximum number of entries that can be stored in the resource resolution cache. The minimum capacity is 5 entries.

The default capacity is 255 entries.

DVU Browser Cache Booster Resource Tagger

The DVU Browser Cache Booster Resource Tagger filters HTTP requests and orchestrates the corresponding HTTP responses.

This servlet filter normally does not need to be customized. In case that other custom servlet filters are registered with the HTTP Whiteboard, it may be necessary to adjust the service ranking of the DVU Browser Cache Booster Resource Tagger servlet filter.

- Log in to the AEM instance as user `admin`
  (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to
  `<protocol>://<host>:<port>/system/console/status-httpservice`
- Locate the `eu.dvu.aem.browsercachebooster.tagger.impl.ResourceTaggerImpl` class and determine its current position in the filter chain (it should be the **second** filter in the filter chain, directly below the DVU RESTful URI Bridge)
- Surf to
  `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on
  `DVU Browser Cache Booster Resource Tagger`
- Adjust the service ranking

Service Ranking

An integer value specifying where to register the filter in the filter chain. Higher rankings will be placed first in the chain, that is, filter chains are sorted in descending order.

The default value is `131071`.

DVU RESTful URI Bridge

The DVU RESTful URI Bridge allows Sling servlets to retrieve any sub-path underneath its registered Sling servlet path by means of the "`eu.dvu.aem.restfuluribridge.RestfulUriBridge.PATH_INFO`" request attribute.

This servlet filter normally does not need to be customized. In case that other custom servlet filters are registered with the HTTP Whiteboard, it may be necessary to adjust the service ranking of the DVU RESTful URI Bridge servlet filter.

- Log in to the AEM instance as user `admin` (e.g. via `<protocol>://<host>:<port>/crx/de/index.jsp`)
- Surf to `<protocol>://<host>:<port>/system/console/status-httpservice`
- Locate the `eu.dvu.aem.restfuluribridge.impl.RestfulUriBridgeImpl` class and determine its current position in the filter chain (it should be the **first** filter in the filter chain, directly above the DVU Browser Cache Booster Resource Tagger)
- Surf to `<protocol>://<host>:<port>/system/console/configMgr`
- Search for and click on `DVU RESTful URI Bridge`
- Adjust the service ranking

Service Ranking

An integer value specifying where to register the filter in the filter chain. Higher rankings will be placed first in the chain, that is, filter chains are sorted in descending order.

The default value is `131072`.

Maintenance

DVU Browser Cache Booster REST interface

status command

The `status` command can be used to query the BCB database. Optionally, a request path can be added to query only a specific resource.

If logged in as user `admin`, the `status` command can be invoked with the following URL:

```
<protocol>://<host>:<port/bin/dvu/browsercachebooster/status[request path]
```

reset command

The `reset` command allows to reset the BCB database. All known resources (resources that exist in the BCB database) are deleted from BCB database and the AEM dispatcher cache. Optionally, a request path can be added to reset only a specific resource.

If logged in as user `admin`, the `reset` command can be invoked with the following URL:

```
<protocol>://<host>:<port/bin/dvu/browsercachebooster/reset[request path]
```

zap command

The `zap` command is like the `reset` command, but no resources are deleted from the AEM dispatcher cache. Zapped resources must therefore be deleted manually. Optionally, a request path can be added to zap only a specific resource.

If logged in as user `admin`, the `zap` command can be invoked with the following URL:

```
<protocol>://<host>:<port/bin/dvu/browsercachebooster/zap[request path]
```

flush command

The `flush` command can be used to delete all known resources (resources that exist in the BCB database) from the AEM dispatcher cache. Optionally, a request path can be added to delete only a specific resource. The BCB database is not changed.

If logged in as user `admin`, the `wipe` command can be invoked with the following URL:

```
<protocol>://<host>:<port/bin/dvu/browsercachebooster/flush[request path]
```

wipe command

The `wipe` command is like the flush command, but it allows to specify a base path for recursive flush operations. If a base path is specified, all resources below the base path will be deleted from the AEM dispatcher cache. The BCB database is not changed.

If logged in as user `admin`, the `wipe` command can be invoked with the following URL:

```
<protocol>://<host>:<port/bin/dvu/browsercachebooster/wipe[base path]
```

Appendix

Appendix 1: Sample Apache Web Server forwarded.conf

```
# forwarded.conf: Forwarded settings
# Requires mod_rewrite and mod_headers to be loaded

#LogLevel debug rewrite:trace7

# Use "Host" request header values for SERVER_NAME and SERVER_PORT
UseCanonicalName off

# If SSL is active, set standard environment variables in SSL context
<IfModule mod_ssl.c>
    SSLOptions +StdEnvVars
</IfModule>

RewriteEngine on

# Get local address and port
RewriteRule ^ - [NS,ENV=THIS_HOST:%{SERVER_ADDR}:%{SERVER_PORT}]

# Reset flag variables
RewriteRule ^ - [NS,ENV=!APPEND_FORWARDED_HEADER,ENV=!SET_FORWARDED_HEADER]

# Get standard "Forwarded" request header
RewriteRule ^ - [NS,ENV=FORWARDED_HEADER:%{HTTP:Forwarded}]

# Check if the "Forwarded" header is present
RewriteCond %{ENV:FORWARDED_HEADER} ^.+$
RewriteRule ^ - [NS,ENV=APPEND_FORWARDED_HEADER:1]

# Check if the "Forwarded" header is absent
RewriteCond %{ENV:FORWARDED_HEADER} !^.+$
RewriteRule ^ - [NS,ENV=SET_FORWARDED_HEADER:1]

# Initialize client
RewriteRule ^ - [NS,ENV=X_FORWARDED_FOR:%{REMOTE_ADDR}:%{REMOTE_PORT}]

# Overwrite client with value of legacy "X-Forwarded-For" request header
# if present
RewriteCond %{HTTP:X-Forwarded-For} ^(.+)$
RewriteRule ^ - [NS,ENV=X_FORWARDED_FOR:%1]

# Overwrite client with value of standard "Forwarded" request header
# if present
RewriteCond %{ENV:FORWARDED_HEADER} \bfor="?([^\]]+?\]?(?::\d+)?)(?:[";,]|$) [NC]
RewriteRule ^ - [NS,ENV=X_FORWARDED_FOR:%1]

# Initialize protocol
RewriteRule ^ - [NS,ENV=X_FORWARDED_PROTO:http]
RewriteCond %{HTTPS} ^on$ [NC]
RewriteRule ^ - [NS,ENV=X_FORWARDED_PROTO:https]
```

```
# Overwrite protocol with value of legacy "X-Forwarded-SSL" request header
# if present
RewriteCond %{HTTP:X-Forwarded-SSL} ^on$ [NC]
RewriteRule ^ - [NS,ENV=X_FORWARDED_PROTO:https]

# Overwrite protocol with value of legacy "X-Forwarded-Proto" request
# header if present
RewriteCond %{HTTP:X-Forwarded-Proto} ^(.+)$
RewriteRule ^ - [NS,ENV=X_FORWARDED_PROTO:%1]

# Overwrite protocol with value of standard "Forwarded" request header
# if present
RewriteCond %{ENV:FORWARDED_HEADER} \bproto="?(https?)(?:[";,]|$) [NC]
RewriteRule ^ - [NS,ENV=X_FORWARDED_PROTO:%1]

# Initialize host
RewriteRule ^ - [NS,ENV=X_FORWARDED_HOST:%{HTTP:Host}]

# Overwrite host with value of legacy "X-Forwarded-Host" request header
# if present
RewriteCond %{HTTP:X-Forwarded-Host} ^(.+)$
RewriteRule ^ - [NS,ENV=X_FORWARDED_HOST:%1]

# Overwrite host with value of standard "Forwarded" request header
# if present
RewriteCond %{ENV:FORWARDED_HEADER} \bhost="?([^\]]+?\]?(?::\d+)?)(?:[";,]|$) [NC]
RewriteRule ^ - [NS,ENV=X_FORWARDED_HOST:%1]

# Strip port number from host if default HTTP port
RewriteCond %{ENV:X_FORWARDED_PROTO} ^http$
RewriteCond %{ENV:X_FORWARDED_HOST} ^(.*):80$
RewriteRule ^ - [NS,ENV=X_FORWARDED_HOST:%1]

# Strip port number from host if default HTTPS port
RewriteCond %{ENV:X_FORWARDED_PROTO} ^https$
RewriteCond %{ENV:X_FORWARDED_HOST} ^(.*):443$
RewriteRule ^ - [NS,ENV=X_FORWARDED_HOST:%1]

# Initialize request URI
RewriteRule ^ - [NS,ENV=X_REQUEST_URI:%{REQUEST_URI}]

# Overwrite request URI with value of custom "X-Request-URI" request
# header if present
RewriteCond %{HTTP:X-Request-URI} ^(.+)$
RewriteRule ^ - [NS,ENV=X_REQUEST_URI:%1]

# Initialize query string
RewriteRule ^ - [NS,ENV=!X_QUERY_STRING]
RewriteCond %{QUERY_STRING} ^(.+)$
RewriteRule ^ - [NS,ENV=X_QUERY_STRING:%1]

# Overwrite query string with value of custom "X-Query-String" request
# header if present
RewriteCond %{HTTP:X-Query-String} ^(.+)$
RewriteRule ^ - [NS,ENV=X_QUERY_STRING:%1]
```

```
# Create missing "Forwarded" header
RequestHeader set Forwarded
"by=\"%{THIS_HOST}e\";for=\"%{X_FORWARDED_FOR}e\";proto=%{X_FORWARDED_PROTO}e;host=
\"%{X_FORWARDED_HOST}e\"" ENV=SET_FORWARDED_HEADER

# Update existing "Forwarded" header
RequestHeader append Forwarded "by=\"%{THIS_HOST}e\"" ENV=APPEND_FORWARDED_HEADER

# Set legacy "X-Forwarded-*" headers
RequestHeader set X-Forwarded-For "%{X_FORWARDED_FOR}e" ENV=X_FORWARDED_FOR
RequestHeader set X-Forwarded-Proto "%{X_FORWARDED_PROTO}e" ENV=X_FORWARDED_PROTO
RequestHeader set X-Forwarded-Host "%{X_FORWARDED_HOST}e" ENV=X_FORWARDED_HOST

# Set custom headers
# Uncomment if the external request path of a resource differs from the
# request path that is visible to the AEM instance.
#RequestHeader set X-Request-URI "%{X_REQUEST_URI}e" ENV=X_REQUEST_URI
#RequestHeader set X-Query-String "%{X_QUERY_STRING}e" ENV=X_QUERY_STRING
```