darkweak feat(dependen...	3c8632e · 5 months ago	
.github/workflows	feat(dependencies): ...	5 months ago
examples	feat(dependencies): ...	3 years ago
fixtures/cache-tests	Merge pull request #...	4 years ago
.gitignore	Fix review from @du...	4 years ago
Caddyfile	feat(dependencies): ...	2 years ago
LICENSE	feat(repository): add ...	2 years ago
README.md	feat(dependencies): ...	last year
admin.go	feat(dependencies): ...	5 months ago
app.go	feat(dependencies): ...	5 months ago
cleaner.go	feat(dependencies): ...	5 months ago
configuration.go	feat(dependencies): ...	5 months ago
configuration.json	feat(dependencies): ...	last year
dispatch.go	feat(dependencies): ...	last year
go.mod	feat(dependencies): ...	5 months ago
go.sum	feat(dependencies): ...	5 months ago
httpcache.go	feat(dependencies): ...	5 months ago
httpcache_test.go	feat(dependencies): ...	5 months ago

About

Distributed HTTP caching module for Caddy

Readme

Apache-2.0 license

Activity

Custom properties

360 stars

12 watching

26 forks

Report repository

Releases 16

v0.16.0 Latest
on Jul 9

+ 15 releases

Packages

No packages published

Contributors 9

Languages

Go 100.0%

README Apache-2.0 license

Caddy Module: http.handlers.cache

This is a distributed HTTP cache module for Caddy based on Souin cache.

> ⚠ Warning
>
> Since `v1.7.0` Souin (the development repository that cache-handler is based on) implements only one storage. If you need a specific storage you have to take it from the storages repository and add it either in your code, during the build otherwise.
> (e.g. with otter using caddy) You have to build your caddy module with the desired storage `xcaddy build --with github.com/caddyserver/cache-handler --with github.com/darkweak/storages/otter/caddy` and configure otter in your Caddyfile/JSON configuration file.
> See the documentation about the storages.

Features

- RFC 7234 compliant HTTP Cache.
- Sets the `Cache-Status` HTTP Response Header
- REST API to purge the cache and list stored resources.
- ESI tags processing (using the go-esi package).
- Builtin support for distributed cache.

Minimal Configuration

Using the minimal configuration the responses will be cached for `120s`

```
{
    cache
}

example.com {
    cache
    reverse_proxy your-app:8080
}
```

Global Option Syntax

Here are all the available options for the global options

```
{
    log {
        level debug
    }
    cache {
        allowed_http_verbs GET POST PATCH
        api {
            basepath /some-basepath
            prometheus
            souin {
                basepath /souin-changed-endpoi
            }
        }
        badger {
            path the_path_to_a_file.json
        }
        cache_keys {
            .*\.css {
                disable_body
                disable_host
                disable_method
                disable_query
                headers X-Token Authorization
                hide
            }
        }
        cache_name Another
        cdn {
            api_key XXXX
            dynamic
            email darkweak@protonmail.com
            hostname domain.com
            network your_network
            provider fastly
            strategy soft
            service_id 123456_id
            zone_id anywhere_zone
        }
        etcd {
            configuration {
                # Your etcd configuration here
            }
        }
        key {
            disable_body
            disable_host
            disable_method
            headers Content-Type Authorization
        }
        log_level debug
        mode bypass
        nuts {
            path /path/to/the/storage
        }
        olric {
            url url_to_your_cluster:3320
            path the_path_to_a_file.yaml
            configuration {
                # Your olric configuration here
            }
        }
        regex {
            exclude /test2.*
        }
        stale 200s
        ttl 1000s
        default_cache_control no-store
    }
}

:4443
respond "Hello World!"
```

Cache directive Syntax

Here are all the available options for the directive options

```
@match path /path

handle @match {
    cache {
        cache_name ChangeName
        cache_keys {
            (host1|host2).*\.css {
                disable_body
                disable_host
                disable_method
                disable_query
                headers X-Token Authorization
            }
        }
        cdn {
            api_key XXXX
            dynamic
            email darkweak@protonmail.com
            hostname domain.com
            network your_network
            provider fastly
            strategy soft
            service_id 123456_id
            zone_id anywhere_zone
        }
        key {
            disable_body
            disable_host
            disable_method
            disable_query
            headers Content-Type Authorization
        }
        log_level debug
        regex {
            exclude /test2.*
        }
        stale 200s
        ttl 1000s
        default_cache_control no-store
    }
}
```

Provider Syntax

Badger

The badger provider must have either the path or the configuration directive.

```
badger-path.com {
    cache {
        badger {
            path /tmp/badger/first-match
        }
    }
}
```

```
badger-configuration.com {
    cache {
        badger {
            configuration {
                # Required value
                ValueDir <string>

                # Optional
                SyncWrites <bool>
                NumVersionsToKeep <int>
                ReadOnly <bool>
                Compression <int>
                InMemory <bool>
                MetricsEnabled <bool>
                MemTableSize <int>
                BaseTableSize <int>
                BaseLevelSize <int>
                LevelSizeMultiplier <int>
                TableSizeMultiplier <int>
                MaxLevels <int>
                VLogPercentile <float>
                ValueThreshold <int>
                NumMemtables <int>
                BlockSize <int>
                BloomFalsePositive <float>
                BlockCacheSize <int>
                IndexCacheSize <int>
                NumLevelZeroTables <int>
                NumLevelZeroTablesStall <int>
                ValueLogFileSize <int>
                ValueLogMaxEntries <int>
                NumCompactors <int>
                CompactL0OnClose <bool>
                LmaxCompaction <bool>
                ZSTDCompressionLevel <int>
                VerifyValueChecksum <bool>
                EncryptionKey <string>
                EncryptionKeyRotationDuration
                BypassLockGuard <bool>
                ChecksumVerificationMode <int>
                DetectConflicts <bool>
                NamespaceOffset <int>
            }
        }
    }
}
```

Etcd

The etcd provider must have the configuration directive.

```
etcd-configuration.com {
    cache {
        etcd {
            configuration {
                Endpoints etcd1:2379 etcd2:237
                AutoSyncInterval 1s
                DialTimeout 1s
                DialKeepAliveTime 1s
                DialKeepAliveTimeout 1s
                MaxCallSendMsgSize 10000000
                MaxCallRecvMsgSize 10000000
                Username john
                Password doe
                RejectOldCluster false
                PermitWithoutStream false
            }
        }
    }
}
```

NutsDB

The nutsdb provider must have either the path or the configuration directive.

```
nuts-path.com {
    cache {
        nuts {
            path /tmp/nuts-path
        }
    }
}
```

```
nuts-configuration.com {
    cache {
        nuts {
            configuration {
                Dir /tmp/nuts-configuration
                EntryIdxMode 1
                RWMode 0
                SegmentSize 1024
                NodeNum 42
                SyncEnable true
                StartFileLoadingMode 1
            }
        }
    }
}
```

Olric

The olric provider must have either the url directive to work as client mode.

```
olric-url.com {
    cache {
        olric {
            url olric:3320
        }
    }
}
```

The olric provider must have either the path or the configuration directive to work as embedded mode.

```
olric-path.com {
    cache {
        olric {
            path /path/to/olricd.yml
        }
    }
}
```

```
olric-configuration.com {
    cache {
        nuts {
            configuration {
                Dir /tmp/nuts-configuration
                EntryIdxMode 1
                RWMode 0
                SegmentSize 1024
                NodeNum 42
                SyncEnable true
                StartFileLoadingMode 1
            }
        }
    }
}
```

Redis

The redis provider must have either the URL or the configuration directive.

```
redis-url.com {
    cache {
        redis {
            url 127.0.0.1:6379
        }
    }
}
```

You can also use the configuration. Refer to the Souin docs or rueidis client options to define your config as key value.

What does these directives mean?

Key	Description	Value e
`allowed_http_verbs`	The HTTP verbs allowed to be cached	`GET PO` `(defau`
`api`	The cache-handler API cache management	
`api.basepath`	BasePath for all APIs to avoid conflicts	`/your-` `(defau`
`api.prometheus`	Enable the Prometheus metrics	
`api.souin.basepath`	Souin API basepath	`/anoth` `(defau`
`badger`	Configure the Badger cache storage	
`badger.path`	Configure Badger with a file	`/anywh`
`badger.configuration`	Configure Badger directly in the Caddyfile or your JSON caddy configuration	See the options
`cache_name`	Override the cache name to use in the Cache-Status response header	`Anothe` `Souin`
`cache_keys`	Define the key generation rules for each URI matching the key regexp	
`cache_keys.{your regexp}`	Regexp that the URI should match to override the key generation	`.+\.cs`
`cache_keys.{your regexp}`	Regexp that the URI should match to override the key generation	`.+\.cs`
`cache_keys.{your regexp}.disable_body`	Disable the body part in the key matching the regexp (GraphQL context)	`true` `(defau`
`cache_keys.{your regexp}.disable_host`	Disable the host part in the key matching the regexp	`true` `(defau`
`cache_keys.{your regexp}.disable_method`	Disable the method part in the key matching the regexp	`true` `(defau`
`cache_keys.{your regexp}.disable_query`	Disable the query string part in the key matching the regexp	`true` `(defau`
`cache_keys.{your regexp}.headers`	Add headers to the key matching the regexp	`Author` `Additio`
`cache_keys.{your regexp}.hide`	Prevent the key from being exposed in the `Cache-Status` HTTP response header	`true` `(defau`
`cdn`	The CDN management, if you use any cdn to proxy your requests Souin will handle that	

`cdn.provider`	The provider placed before Souin	`akamai`
		`fastly`
		`souin`

`cdn.api_key`	The api key used to access to the provider	`XXXX`
`cdn.dynamic`	Enable the dynamic keys returned by your backend application	`(defau`
`cdn.email`	The api key used to access to the provider if required, depending the provider	`XXXX`
`cdn.hostname`	The hostname if required, depending the provider	`domain`
`cdn.network`	The network if required, depending the provider	`your_n`
`cdn.strategy`	The strategy to use to purge the cdn cache, soft will keep the content as a stale resource	`hard` `(defau`
`cdn.service_id`	The service id if required, depending the provider	`123456_`
`cdn.zone_id`	The zone id if required, depending the provider	`anywhe`
`default_cache_control`	Set the default value of `Cache-Control` response header if not set by upstream (Souin treats empty `Cache-Control` as `public` if omitted)	`no-sto`
`key`	Override the key generation with the ability to disable unecessary parts	
`key.disable_body`	Disable the body part in the key (GraphQL context)	`true` `(defau`
`key.disable_host`	Disable the host part in the key	`true` `(defau`
`key.disable_method`	Disable the method part in the key	`true` `(defau`
`key.disable_query`	Disable the query string part in the key	`true` `(defau`
`key.disable_scheme`	Disable the scheme string part in the key	`true` `(defau`
`key.hash`	Hash the key before store it in the storage to get smaller keys	`true` `(defau`
`key.headers`	Add headers to the key matching the regexp	`Author` `Additio`

`key.hide`	Prevent the key from being exposed in the `Cache-Status` HTTP response header	`true` `(defaul`
`key.template`	Use caddy templates to create the key (when this option is enabled, disable_* directives are skipped)	`KEY-{ht` `{http.r`
`max_cacheable_body_bytes`	Set the maximum size (in bytes) for a response body to be cached (unlimited if omited)	`104857`

`mode`	Bypass the RFC respect	One of
		`bypass_`
		`strict`

`nuts`	Configure the Nuts cache storage	
`nuts.path`	Set the Nuts file path storage	`/anywh`
`nuts.configuration`	Configure Nuts directly in the Caddyfile or your JSON caddy configuration	See the options
`etcd`	Configure the Etcd cache storage	
`etcd.configuration`	Configure Etcd directly in the Caddyfile or your JSON caddy configuration	See the options
`olric`	Configure the Olric cache storage	
`olric.path`	Configure Olric with a file	`/anywh`
`olric.configuration`	Configure Olric directly in the Caddyfile or your JSON caddy configuration	See the options
`otter`	Configure the Otter cache storage	
`otter.configuration`	Configure Otter directly in the Caddyfile or your JSON caddy configuration	
`otter.configuration.size`	Set the size of the pool in Otter	`999999`
`redis`	Configure the Redis cache storage	
`redis.url`	Set the Redis url storage	`localh`
`redis.configuration`	Configure Redis directly in the Caddyfile or your JSON caddy configuration	See the options
`regex.exclude`	The regex used to prevent paths being cached	`^[A-z]-`
`stale`	The stale duration	`25m`
`storers`	Storers chain to fallback if a previous one is unreachable or don't have the resource	`otter r`
`timeout`	The timeout configuration	
`timeout.backend`	The timeout duration to consider the backend as unreachable	`10s`
`timeout.cache`	The timeout duration to consider the cache provider as unreachable	`10ms`
`ttl`	The TTL duration	`120s`

`log_level`	The log level	One of
		`DPANIC,`
		`insensi`

Other resources

You can find an example for the Caddyfile or the JSON file.
See the Souin configuration for the full configuration, and its associated Caddyfile

Development and Stable Versions

The **Souin** repository serves as the development version, where new features are introduced and tested. Once these features have been thoroughly stabilized, they are integrated into the **cache-handler** repository through a dependency update. This ensures that **cache-handler** remains the stable and reliable version for production use.