Cache by status code

- Caching limits
- Set cache TTL by response status via the Cloudflare dashboard
- Set cache TTL by response status via the Cloudflare API
  - Syntax
- Set cache TTL by response status via a Cloudflare Worker
- TTL handling for 304 and 200 status codes

import { Render } from "~/components"

Customers can set cache time-to-live (TTL) based on the response status from the origin web server. Cache TTL refers to the duration of a resource in the Cloudflare network before being marked as `STALE` or discarded from cache. Status codes are returned by a resource's origin.

Setting cache TTL based on response status overrides the default cache behavior (standard caching) for static files and overrides cache instructions sent by the origin web server. To cache non-static assets, set a Cache Level of Cache Everything using a Cache Rule. Setting `no-store` **Cache-Control** or a low TTL (using `max-age`/`s-maxage`) increases requests to origin web servers and decreases performance.

Caching limits

The maximum caching limit for Free, Pro, and Business customers is 512 MB per file, and the maximum caching limit for Enterprise customers is 5 GB per file. If you need to raise the limits, contact your account team.

<Render file="edge-ttl" product="cache" params={{ conditional: "dedicated-cache-tile" }} />

Set cache TTL by response status via the Cloudflare dashboard

To set cache TTL by response status, create a Cache Rule for **Cache TTL by status code**.

Set cache TTL by response status via the Cloudflare API

```
curl --request PUT \
"https://api.cloudflare.com/client/v4/zones/{zone_id}/rulesets/{ruleset_id}" \
--header "Authorization: Bearer <API_TOKEN>" \
--header "Content-Type: application/json" \
--data '{
  "rules": [
    {
      "expression": "(http.host eq \"www.example.com\")",
      "description": "set cache TTL by response status",
      "action": "set_cache_settings",
      "action_parameters": {
        "cache": true,
        "edge_ttl": {
          "status_code_ttl": [
            {
              "status_code_range": {
                "to": 299
              },
              "value": 86400
            },
            {
              "status_code_range": {
                "from": 300,
                "to": 499
              },
```

```
            "value": 0  // no-cache
          },
          {
            "status_code_range": {
              "from": 500
            },
            "value": -1  // no-store
          }
        ],
        "mode": "respect_origin"
      }
    }
  }
]
}'
```

Syntax

Provide a JSON object containing status codes and their corresponding TTLs. Each key-value pair in the cache TTL by status cache rule has the following syntax:

- `status_code`: An integer value such as 200 or 500. `status_code` matches the exact status code from the origin web server. Valid status codes are between 100-999.
- `status_code_range`: Integer values for `from` and `to`. `status_code_range` matches any status code from the origin web server within the specified range.
- `value`: An integer value that defines the duration an asset is valid in seconds or one of the following strings: `no-store` (equivalent to `-1`), `no-cache` (equivalent to `0`).

Set cache TTL by response status via a Cloudflare Worker

The **cacheTtlByStatus** option is a version of the **cacheTtl** feature that designates a cache TTL for a request's response status code (for example, `{ "200-299": 86400, 404: 1, "500-599": 0 }`).

TTL handling for 304 and 200 status codes

1. If a TTL is not explicitly set for status code `304`, we automatically set it to match the TTL of status code `200` (if the user has defined one for `200`).
2. If a user explicitly sets a different TTL for `304` than for `200`, the following behavior will occur:

- When a `200` response is received, the asset is cached with the TTL specified for status `200`.
- Once the asset expires and we revalidate with the origin, if the origin returns a `304`, the cache TTL is updated to the value set for `304`.

For example, if a user specifies a TTL of one hour for status `200` and 0 seconds (cache and always revalidate) for status `304`, the asset will be cached for 1 hour. After it expires, we revalidate with the origin. If the origin returns a `304`, each subsequent request will trigger revalidation. If the origin continues to return `304`, this cycle will persist.

This behavior is likely undesirable unless the user has a specific use case. Therefore, users should ensure that the TTL for `304` matches the TTL for `200` unless they intentionally require this behavior.

□□□ #2
□ □□□ □□□ 2026-11-01 00:14:55 UTC
□ □□□ □□□ 2026-11-01 00:14:57 UTC