Access-Control-Allow-Headers header

- Syntax
- Directives
- Examples
  - Implementing a custom header
  - Supporting multiple headers
  - Bypassing additional restrictions on CORS-safelisted headers
  - Handling preflight requests
    - Request
    - Response
- Specifications
- Browser compatibility
- See also

The HTTP `Access-Control-Allow-Headers` response header is used in response to a preflight request to indicate the HTTP headers that can be used during the actual request. This header is required if the preflight request contains Access-Control-Request-Headers.

> **NOTE:** The CORS-safelisted request headers are always allowed and usually aren't listed in `Access-Control-Allow-Headers` unless there is a need to circumvent the additional safelist restrictions.

Header type	[Response header](https://developer.mozilla.org/en-US/docs/Glossary/Response%20header)

Syntax

```
Access-Control-Allow-Headers: <header-name>
Access-Control-Allow-Headers: <header-name>, <header-name>
Access-Control-Allow-Headers: *
```

Directives

- `<header-name>`
  - : The name of a supported request header. The header may list any number of headers, separated by commas.
- `*` (wildcard)
  - : Any header. The value `*` only counts as a special wildcard value for requests without credentials (requests without HTTP cookies or HTTP authentication information). In requests with credentials, it is treated as the literal header name `*` without special semantics. The Authorization header doesn't accept wildcard and always needs to be listed explicitly.

Examples

Implementing a custom header

Below is an example of an `Access-Control-Allow-Headers` header. It indicates that a custom header named `X-Custom-Header` is supported by CORS requests to the server, in addition to the CORS-safelisted request headers.

```
Access-Control-Allow-Headers: X-Custom-Header
```

Supporting multiple headers

This example shows `Access-Control-Allow-Headers` when it specifies support for multiple headers.

```
Access-Control-Allow-Headers: X-Custom-Header, Upgrade-Insecure-Requests
```

Bypassing additional restrictions on CORS-safelisted headers

Although CORS-safelisted request headers are always allowed and don't usually need to be listed in `Access-Control-Allow-Headers`, listing them anyway will circumvent the additional restrictions that apply.

```
Access-Control-Allow-Headers: Accept
```

Handling preflight requests

Let's look at an example of a preflight request involving `Access-Control-Allow-Headers`.

Request

First, the preflight request is an OPTIONS request that includes some combination of the three preflight request headers: Access-Control-Request-Method, Access-Control-Request-Headers, and Origin.

The preflight request below tells the server that we want to send a CORS `GET` request with the headers listed in Access-Control-Request-Headers (Content-Type and `X-Requested-With`).

```
OPTIONS /resource/foo
Access-Control-Request-Method: GET
Access-Control-Request-Headers: content-type,x-requested-with
Origin: https://www.example.com
```

Response

If the CORS request indicated by the preflight request is authorized, the server will respond to the preflight request with a message that indicates the allowed origin, methods, and headers. Below, we see that `Access-Control-Allow-Headers` includes the headers that were requested.

```
HTTP/1.1 200 OK
Content-Length: 0
Connection: keep-alive
Access-Control-Allow-Origin: https://www.example.com
Access-Control-Allow-Methods: POST, GET, OPTIONS, DELETE
```

```
Access-Control-Allow-Headers: Content-Type, x-requested-with
Access-Control-Max-Age: 86400
```

If the requested method isn't supported, the server will respond with an error.

Specifications

Browser compatibility

See also

- Access-Control-Allow-Origin
- Access-Control-Expose-Headers
- Access-Control-Allow-Methods
- Access-Control-Request-Headers

□□□ #3
□ □□□ □□□ 2026-10-31 03:27:53 UTC
□ cocobook-sync □□□ 2026-11-04 17:35:35 UTC