# fresh

[NPM Version](https://npmjs.org/package/fresh) [NPM Downloads](https://npmjs.org/package/fresh) [Node.js Version](https://nodejs.org/en/) [Build Status](https://travis-ci.org/jshttp/fresh) [Test Coverage](https://coveralls.io/r/jshttp/fresh?branch=master)

HTTP response freshness testing

## Installation

This is a [Node.js](https://nodejs.org/en/) module available through the [npm registry](https://www.npmjs.com/). Installation is done using the [npm install command](https://docs.npmjs.com/getting-started/installing-npm-packages-locally):

$ npm install fresh

## API

var fresh = require('fresh')

### fresh(reqHeaders, resHeaders)

Check freshness of the response using request and response headers.

When the response is still "fresh" in the client's cache true is returned, otherwise false is returned to indicate that the client cache is now stale and the full response should be sent.

When a client sends the Cache-Control: no-cache request header to indicate an end-to-end reload request, this module will return false to make handling these requests transparent.

## Known Issues

This module is designed to only follow the HTTP specifications, not to work-around all kinda of client bugs (especially since this module typically does not recieve enough information to understand what the client actually is).

There is a known issue that in certain versions of Safari, Safari will incorrectly make a request that allows this module to validate freshness of the resource even when Safari does not have a representation of the resource in the cache. The module [jumanji](https://www.npmjs.com/package/jumanji) can be used in an Express application to work-around this issue and also provides links to further reading on this Safari bug.

## Example

### API usage

var reqHeaders = { 'if-none-match': '"foo"' }

var resHeaders = { 'etag': '"bar"' }

fresh(reqHeaders, resHeaders)

// => false

var reqHeaders = { 'if-none-match': '"foo"' }

var resHeaders = { 'etag': '"foo"' }

fresh(reqHeaders, resHeaders)

// => true

### Using with Node.js http server

var fresh = require('fresh')

var http = require('http')

var server = http.createServer(function (req, res) {

// perform server logic

// ... including adding ETag / Last-Modified response headers

if (isFresh(req, res)) {

// client has a fresh copy of resource

res.statusCode = 304

res.end()

return

}

// send the resource

res.statusCode = 200

res.end('hello, world!')

})

function isFresh (req, res) {

return fresh(req.headers, {

'etag': res.getHeader('ETag'),

'last-modified': res.getHeader('Last-Modified')

})

}

server.listen(3000)

## License

[MIT](http://license)