# cookie

[NPM Version](https://npmjs.org/package/cookie) [NPM Downloads](https://npmjs.org/package/cookie) [Node.js Version](https://nodejs.org/en/download) [Build Status](https://github.com/jshttp/cookie/actions/workflows/ci.yml) [Coverage Status](https://coveralls.io/r/jshttp/cookie?branch=master)

Basic HTTP cookie parser and serializer for HTTP servers.

## 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 cookie

## API

var cookie = require('cookie');

### cookie.parse(str, options)

Parse an HTTP Cookie header string and returning an object of all cookie name-value pairs. The str argument is the string representing a Cookie header value and options is an optional object containing additional parsing options.

var cookies = cookie.parse('foo=bar; equation=E%3Dmc%5E2');

// { foo: 'bar', equation: 'E=mc^2' }

#### Options

cookie.parse accepts these properties in the options object.

##### decode

Specifies a function that will be used to decode a cookie's value. Since the value of a cookie has a limited character set (and must be a simple string), this function can be used to decode a previously-encoded cookie value into a JavaScript string or other object.

The default function is the global decodeURIComponent, which will decode any URL-encoded sequences into their byte representations.

**note** if an error is thrown from this function, the original, non-decoded cookie value will be returned as the cookie's value.

### cookie.serialize(name, value, options)

Serialize a cookie name-value pair into a Set-Cookie header string. The name argument is the name for the cookie, the value argument is the value to set the cookie to, and the options argument is an optional object containing additional serialization options.

var setCookie = cookie.serialize('foo', 'bar');

// foo=bar

#### Options

cookie.serialize accepts these properties in the options object.

##### domain

Specifies the value for the [Domain Set-Cookie attribute](https://tools.ietf.org/html/rfc6265#section-5.2.3). By default, no domain is set, and most clients will consider the cookie to apply to only the current domain.

##### encode

Specifies a function that will be used to encode a cookie's value. Since value of a cookie has a limited character set (and must be a simple string), this function can be used to encode a value into a string suited for a cookie's value.

The default function is the global encodeURIComponent, which will encode a JavaScript string into UTF-8 byte sequences and then URL-encode any that fall outside of the cookie range.

##### expires

Specifies the Date object to be the value for the [Expires Set-Cookie attribute](https://tools.ietf.org/html/rfc6265#section-5.2.1). By default, no expiration is set, and most clients will consider this a "non-persistent cookie" and will delete it on a condition like exiting a web browser application.

**note** the [cookie storage model specification](https://tools.ietf.org/html/rfc6265#section-5.3) states that if both expires and maxAge are set, then maxAge takes precedence, but it is possible not all clients by obey this, so if both are set, they should point to the same date and time.

##### httpOnly

Specifies the boolean value for the [HttpOnly Set-Cookie attribute](https://tools.ietf.org/html/rfc6265#section-5.2.6). When truthy, the HttpOnly attribute is set, otherwise it is not. By default, the HttpOnly attribute is not set.

**note** be careful when setting this to true, as compliant clients will not allow client-side JavaScript to see the cookie in document.cookie.

##### maxAge

Specifies the number (in seconds) to be the value for the [Max-Age Set-Cookie attribute](https://tools.ietf.org/html/rfc6265#section-5.2.2). The given number will be converted to an integer by rounding down. By default, no maximum age is set.

**note** the [cookie storage model specification](https://tools.ietf.org/html/rfc6265#section-5.3) states that if both expires and maxAge are set, then maxAge takes precedence, but it is possible not all clients by obey this, so if both are set, they should point to the same date and time.

##### partitioned

Specifies the boolean value for the [Partitioned Set-Cookie](http://rfc-cutler-httpbis-partitioned-cookies) attribute. When truthy, the Partitioned attribute is set, otherwise it is not. By default, the Partitioned attribute is not set.

**note** This is an attribute that has not yet been fully standardized, and may change in the future. This also means many clients may ignore this attribute until they understand it.

More information about can be found in [the proposal](https://github.com/privacycg/CHIPS).

##### path

Specifies the value for the [Path Set-Cookie attribute](https://tools.ietf.org/html/rfc6265#section-5.2.4). By default, the path is considered the ["default path"](https://tools.ietf.org/html/rfc6265#section-5.1.4).

##### priority

Specifies the string to be the value for the [Priority Set-Cookie attribute](https://tools.ietf.org/html/draft-west-cookie-priority-00#section-4.1).

* 'low' will set the Priority attribute to Low.
* 'medium' will set the Priority attribute to Medium, the default priority when not set.
* 'high' will set the Priority attribute to High.

More information about the different priority levels can be found in [the specification](https://tools.ietf.org/html/draft-west-cookie-priority-00#section-4.1).

**note** This is an attribute that has not yet been fully standardized, and may change in the future. This also means many clients may ignore this attribute until they understand it.

##### sameSite

Specifies the boolean or string to be the value for the [SameSite Set-Cookie attribute](https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-09#section-5.4.7).

* true will set the SameSite attribute to Strict for strict same site enforcement.
* false will not set the SameSite attribute.
* 'lax' will set the SameSite attribute to Lax for lax same site enforcement.
* 'none' will set the SameSite attribute to None for an explicit cross-site cookie.
* 'strict' will set the SameSite attribute to Strict for strict same site enforcement.

More information about the different enforcement levels can be found in [the specification](https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-09#section-5.4.7).

**note** This is an attribute that has not yet been fully standardized, and may change in the future. This also means many clients may ignore this attribute until they understand it.

##### secure

Specifies the boolean value for the [Secure Set-Cookie attribute](https://tools.ietf.org/html/rfc6265#section-5.2.5). When truthy, the Secure attribute is set, otherwise it is not. By default, the Secure attribute is not set.

**note** be careful when setting this to true, as compliant clients will not send the cookie back to the server in the future if the browser does not have an HTTPS connection.

## Example

The following example uses this module in conjunction with the Node.js core HTTP server to prompt a user for their name and display it back on future visits.

var cookie = require('cookie');

var escapeHtml = require('escape-html');

var http = require('http');

var url = require('url');

function onRequest(req, res) {

// Parse the query string

var query = url.parse(req.url, true, true).query;

if (query && query.name) {

// Set a new cookie with the name

res.setHeader('Set-Cookie', cookie.serialize('name', String(query.name), {

httpOnly: true,

maxAge: 60 \* 60 \* 24 \* 7 // 1 week

}));

// Redirect back after setting cookie

res.statusCode = 302;

res.setHeader('Location', req.headers.referer || '/');

res.end();

return;

}

// Parse the cookies on the request

var cookies = cookie.parse(req.headers.cookie || '');

// Get the visitor name set in the cookie

var name = cookies.name;

res.setHeader('Content-Type', 'text/html; charset=UTF-8');

if (name) {

res.write('<p>Welcome back, <b>' + escapeHtml(name) + '</b>!</p>');

} else {

res.write('<p>Hello, new visitor!</p>');

}

res.write('<form method="GET">');

res.write('<input placeholder="enter your name" name="name"> <input type="submit" value="Set Name">');

res.end('</form>');

}

http.createServer(onRequest).listen(3000);

## Testing

$ npm test

## Benchmark

$ npm run bench

> cookie@0.5.0 bench

> node benchmark/index.js

node@18.18.2

acorn@8.10.0

ada@2.6.0

ares@1.19.1

brotli@1.0.9

cldr@43.1

icu@73.2

llhttp@6.0.11

modules@108

napi@9

nghttp2@1.57.0

nghttp3@0.7.0

ngtcp2@0.8.1

openssl@3.0.10+quic

simdutf@3.2.14

tz@2023c

undici@5.26.3

unicode@15.0

uv@1.44.2

uvwasi@0.0.18

v8@10.2.154.26-node.26

zlib@1.2.13.1-motley

> node benchmark/parse-top.js

cookie.parse - top sites

14 tests completed.

parse accounts.google.com x 2,588,913 ops/sec ±0.74% (186 runs sampled)

parse apple.com x 2,370,002 ops/sec ±0.69% (186 runs sampled)

parse cloudflare.com x 2,213,102 ops/sec ±0.88% (188 runs sampled)

parse docs.google.com x 2,194,157 ops/sec ±1.03% (184 runs sampled)

parse drive.google.com x 2,265,084 ops/sec ±0.79% (187 runs sampled)

parse en.wikipedia.org x 457,099 ops/sec ±0.81% (186 runs sampled)

parse linkedin.com x 504,407 ops/sec ±0.89% (186 runs sampled)

parse maps.google.com x 1,230,959 ops/sec ±0.98% (186 runs sampled)

parse microsoft.com x 926,294 ops/sec ±0.88% (184 runs sampled)

parse play.google.com x 2,311,338 ops/sec ±0.83% (185 runs sampled)

parse support.google.com x 1,508,850 ops/sec ±0.86% (186 runs sampled)

parse www.google.com x 1,022,582 ops/sec ±1.32% (182 runs sampled)

parse youtu.be x 332,136 ops/sec ±1.02% (185 runs sampled)

parse youtube.com x 323,833 ops/sec ±0.77% (183 runs sampled)

> node benchmark/parse.js

cookie.parse - generic

6 tests completed.

simple x 3,214,032 ops/sec ±1.61% (183 runs sampled)

decode x 587,237 ops/sec ±1.16% (187 runs sampled)

unquote x 2,954,618 ops/sec ±1.35% (183 runs sampled)

duplicates x 857,008 ops/sec ±0.89% (187 runs sampled)

10 cookies x 292,133 ops/sec ±0.89% (187 runs sampled)

100 cookies x 22,610 ops/sec ±0.68% (187 runs sampled)

## References

* [RFC 6265: HTTP State Management Mechanism](https://tools.ietf.org/html/rfc6265)
* [Same-site Cookies](https://tools.ietf.org/html/draft-ietf-httpbis-rfc6265bis-09#section-5.4.7)

## License

[MIT](http://license)