# etag

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

Create simple HTTP ETags

This module generates HTTP ETags (as defined in RFC 7232) for use in HTTP responses.

## 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 etag

## API

var etag = require('etag')

### etag(entity, [options])

Generate a strong ETag for the given entity. This should be the complete body of the entity. Strings, Buffers, and fs.Stats are accepted. By default, a strong ETag is generated except for fs.Stats, which will generate a weak ETag (this can be overwritten by options.weak).

res.setHeader('ETag', etag(body))

#### Options

etag accepts these properties in the options object.

##### weak

Specifies if the generated ETag will include the weak validator mark (that is, the leading W/). The actual entity tag is the same. The default value is false, unless the entity is fs.Stats, in which case it is true.

## Testing

$ npm test

## Benchmark

$ npm run-script bench

> etag@1.8.1 bench nodejs-etag

> node benchmark/index.js

http\_parser@2.7.0

node@6.11.1

v8@5.1.281.103

uv@1.11.0

zlib@1.2.11

ares@1.10.1-DEV

icu@58.2

modules@48

openssl@1.0.2k

> node benchmark/body0-100b.js

100B body

4 tests completed.

buffer - strong x 258,647 ops/sec ±1.07% (180 runs sampled)

buffer - weak x 263,812 ops/sec ±0.61% (184 runs sampled)

string - strong x 259,955 ops/sec ±1.19% (185 runs sampled)

string - weak x 264,356 ops/sec ±1.09% (184 runs sampled)

> node benchmark/body1-1kb.js

1KB body

4 tests completed.

buffer - strong x 189,018 ops/sec ±1.12% (182 runs sampled)

buffer - weak x 190,586 ops/sec ±0.81% (186 runs sampled)

string - strong x 144,272 ops/sec ±0.96% (188 runs sampled)

string - weak x 145,380 ops/sec ±1.43% (187 runs sampled)

> node benchmark/body2-5kb.js

5KB body

4 tests completed.

buffer - strong x 92,435 ops/sec ±0.42% (188 runs sampled)

buffer - weak x 92,373 ops/sec ±0.58% (189 runs sampled)

string - strong x 48,850 ops/sec ±0.56% (186 runs sampled)

string - weak x 49,380 ops/sec ±0.56% (190 runs sampled)

> node benchmark/body3-10kb.js

10KB body

4 tests completed.

buffer - strong x 55,989 ops/sec ±0.93% (188 runs sampled)

buffer - weak x 56,148 ops/sec ±0.55% (190 runs sampled)

string - strong x 27,345 ops/sec ±0.43% (188 runs sampled)

string - weak x 27,496 ops/sec ±0.45% (190 runs sampled)

> node benchmark/body4-100kb.js

100KB body

4 tests completed.

buffer - strong x 7,083 ops/sec ±0.22% (190 runs sampled)

buffer - weak x 7,115 ops/sec ±0.26% (191 runs sampled)

string - strong x 3,068 ops/sec ±0.34% (190 runs sampled)

string - weak x 3,096 ops/sec ±0.35% (190 runs sampled)

> node benchmark/stats.js

stat

4 tests completed.

real - strong x 871,642 ops/sec ±0.34% (189 runs sampled)

real - weak x 867,613 ops/sec ±0.39% (190 runs sampled)

fake - strong x 401,051 ops/sec ±0.40% (189 runs sampled)

fake - weak x 400,100 ops/sec ±0.47% (188 runs sampled)

## License

[MIT](http://license)