# negotiator

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

An HTTP content negotiator for Node.js

## Installation

$ npm install negotiator

## API

var Negotiator = require('negotiator')

### Accept Negotiation

availableMediaTypes = ['text/html', 'text/plain', 'application/json']

// The negotiator constructor receives a request object

negotiator = new Negotiator(request)

// Let's say Accept header is 'text/html, application/\*;q=0.2, image/jpeg;q=0.8'

negotiator.mediaTypes()

// -> ['text/html', 'image/jpeg', 'application/\*']

negotiator.mediaTypes(availableMediaTypes)

// -> ['text/html', 'application/json']

negotiator.mediaType(availableMediaTypes)

// -> 'text/html'

You can check a working example at examples/accept.js.

#### Methods

##### mediaType()

Returns the most preferred media type from the client.

##### mediaType(availableMediaType)

Returns the most preferred media type from a list of available media types.

##### mediaTypes()

Returns an array of preferred media types ordered by the client preference.

##### mediaTypes(availableMediaTypes)

Returns an array of preferred media types ordered by priority from a list of available media types.

### Accept-Language Negotiation

negotiator = new Negotiator(request)

availableLanguages = ['en', 'es', 'fr']

// Let's say Accept-Language header is 'en;q=0.8, es, pt'

negotiator.languages()

// -> ['es', 'pt', 'en']

negotiator.languages(availableLanguages)

// -> ['es', 'en']

language = negotiator.language(availableLanguages)

// -> 'es'

You can check a working example at examples/language.js.

#### Methods

##### language()

Returns the most preferred language from the client.

##### language(availableLanguages)

Returns the most preferred language from a list of available languages.

##### languages()

Returns an array of preferred languages ordered by the client preference.

##### languages(availableLanguages)

Returns an array of preferred languages ordered by priority from a list of available languages.

### Accept-Charset Negotiation

availableCharsets = ['utf-8', 'iso-8859-1', 'iso-8859-5']

negotiator = new Negotiator(request)

// Let's say Accept-Charset header is 'utf-8, iso-8859-1;q=0.8, utf-7;q=0.2'

negotiator.charsets()

// -> ['utf-8', 'iso-8859-1', 'utf-7']

negotiator.charsets(availableCharsets)

// -> ['utf-8', 'iso-8859-1']

negotiator.charset(availableCharsets)

// -> 'utf-8'

You can check a working example at examples/charset.js.

#### Methods

##### charset()

Returns the most preferred charset from the client.

##### charset(availableCharsets)

Returns the most preferred charset from a list of available charsets.

##### charsets()

Returns an array of preferred charsets ordered by the client preference.

##### charsets(availableCharsets)

Returns an array of preferred charsets ordered by priority from a list of available charsets.

### Accept-Encoding Negotiation

availableEncodings = ['identity', 'gzip']

negotiator = new Negotiator(request)

// Let's say Accept-Encoding header is 'gzip, compress;q=0.2, identity;q=0.5'

negotiator.encodings()

// -> ['gzip', 'identity', 'compress']

negotiator.encodings(availableEncodings)

// -> ['gzip', 'identity']

negotiator.encoding(availableEncodings)

// -> 'gzip'

You can check a working example at examples/encoding.js.

#### Methods

##### encoding()

Returns the most preferred encoding from the client.

##### encoding(availableEncodings)

Returns the most preferred encoding from a list of available encodings.

##### encodings()

Returns an array of preferred encodings ordered by the client preference.

##### encodings(availableEncodings)

Returns an array of preferred encodings ordered by priority from a list of available encodings.

## See Also

The [accepts](https://npmjs.org/package/accepts#readme) module builds on this module and provides an alternative interface, mime type validation, and more.

## License

[MIT](http://license)