# Neo-Async



      

Neo-Async is thought to be used as a drop-in replacement for [Async](https://github.com/caolan/async), it almost fully covers its functionality and runs [faster](#benchmark). Benchmark is [here](#benchmark)! Bluebird's benchmark is [here](https://github.com/suguru03/bluebird/tree/aigle/benchmark)! ## Code Coverage ![coverage](https://raw.githubusercontent.com/wiki/suguru03/neo-async/images/coverage.png) ## Installation ### In a browser ```html ``` ### In an AMD loader ```js require(['async'], function(async) {}); ``` ### Promise and async/await I recommend to use [`Aigle`](https://github.com/suguru03/aigle). It is optimized for Promise handling and has almost the same functionality as `neo-async`. ### Node.js #### standard ```bash $ npm install neo-async ``` ```js var async = require('neo-async'); ``` #### replacement ```bash $ npm install neo-async $ ln -s ./node\_modules/neo-async ./node\_modules/async ``` ```js var async = require('async'); ``` ### Bower ```bash bower install neo-async ``` ## Feature [JSDoc](http://suguru03.github.io/neo-async/doc/async.html) \\* not in Async ### Collections - [`each`](http://suguru03.github.io/neo-async/doc/async.each.html) - [`eachSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) - [`eachLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) - [`forEach`](http://suguru03.github.io/neo-async/doc/async.each.html) -> [`each`](http://suguru03.github.io/neo-async/doc/async.each.html) - [`forEachSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) -> [`eachSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) - [`forEachLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) -> [`eachLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) - [`eachOf`](http://suguru03.github.io/neo-async/doc/async.each.html) -> [`each`](http://suguru03.github.io/neo-async/doc/async.each.html) - [`eachOfSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) -> [`eachSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) - [`eachOfLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) -> [`eachLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) - [`forEachOf`](http://suguru03.github.io/neo-async/doc/async.each.html) -> [`each`](http://suguru03.github.io/neo-async/doc/async.each.html) - [`forEachOfSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) -> [`eachSeries`](http://suguru03.github.io/neo-async/doc/async.eachSeries.html) - [`eachOfLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) -> [`forEachLimit`](http://suguru03.github.io/neo-async/doc/async.eachLimit.html) - [`map`](http://suguru03.github.io/neo-async/doc/async.map.html) - [`mapSeries`](http://suguru03.github.io/neo-async/doc/async.mapSeries.html) - [`mapLimit`](http://suguru03.github.io/neo-async/doc/async.mapLimit.html) - [`mapValues`](http://suguru03.github.io/neo-async/doc/async.mapValues.html) - [`mapValuesSeries`](http://suguru03.github.io/neo-async/doc/async.mapValuesSeries.html) - [`mapValuesLimit`](http://suguru03.github.io/neo-async/doc/async.mapValuesLimit.html) - [`filter`](http://suguru03.github.io/neo-async/doc/async.filter.html) - [`filterSeries`](http://suguru03.github.io/neo-async/doc/async.filterSeries.html) - [`filterLimit`](http://suguru03.github.io/neo-async/doc/async.filterLimit.html) - [`select`](http://suguru03.github.io/neo-async/doc/async.filter.html) -> [`filter`](http://suguru03.github.io/neo-async/doc/async.filter.html) - [`selectSeries`](http://suguru03.github.io/neo-async/doc/async.filterSeries.html) -> [`filterSeries`](http://suguru03.github.io/neo-async/doc/async.filterSeries.html) - [`selectLimit`](http://suguru03.github.io/neo-async/doc/async.filterLimit.html) -> [`filterLimit`](http://suguru03.github.io/neo-async/doc/async.filterLimit.html) - [`reject`](http://suguru03.github.io/neo-async/doc/async.reject.html) - [`rejectSeries`](http://suguru03.github.io/neo-async/doc/async.rejectSeries.html) - [`rejectLimit`](http://suguru03.github.io/neo-async/doc/async.rejectLimit.html) - [`detect`](http://suguru03.github.io/neo-async/doc/async.detect.html) - [`detectSeries`](http://suguru03.github.io/neo-async/doc/async.detectSeries.html) - [`detectLimit`](http://suguru03.github.io/neo-async/doc/async.detectLimit.html) - [`find`](http://suguru03.github.io/neo-async/doc/async.detect.html) -> [`detect`](http://suguru03.github.io/neo-async/doc/async.detect.html) - [`findSeries`](http://suguru03.github.io/neo-async/doc/async.detectSeries.html) -> [`detectSeries`](http://suguru03.github.io/neo-async/doc/async.detectSeries.html) - [`findLimit`](http://suguru03.github.io/neo-async/doc/async.detectLimit.html) -> [`detectLimit`](http://suguru03.github.io/neo-async/doc/async.detectLimit.html) - [`pick`](http://suguru03.github.io/neo-async/doc/async.pick.html) \* - [`pickSeries`](http://suguru03.github.io/neo-async/doc/async.pickSeries.html) \* - [`pickLimit`](http://suguru03.github.io/neo-async/doc/async.pickLimit.html) \* - [`omit`](http://suguru03.github.io/neo-async/doc/async.omit.html) \* - [`omitSeries`](http://suguru03.github.io/neo-async/doc/async.omitSeries.html) \* - [`omitLimit`](http://suguru03.github.io/neo-async/doc/async.omitLimit.html) \* - [`reduce`](http://suguru03.github.io/neo-async/doc/async.reduce.html) - [`inject`](http://suguru03.github.io/neo-async/doc/async.reduce.html) -> [`reduce`](http://suguru03.github.io/neo-async/doc/async.reduce.html) - [`foldl`](http://suguru03.github.io/neo-async/doc/async.reduce.html) -> [`reduce`](http://suguru03.github.io/neo-async/doc/async.reduce.html) - [`reduceRight`](http://suguru03.github.io/neo-async/doc/async.reduceRight.html) - [`foldr`](http://suguru03.github.io/neo-async/doc/async.reduceRight.html) -> [`reduceRight`](http://suguru03.github.io/neo-async/doc/async.reduceRight.html) - [`transform`](http://suguru03.github.io/neo-async/doc/async.transform.html) - [`transformSeries`](http://suguru03.github.io/neo-async/doc/async.transformSeries.html) \* - [`transformLimit`](http://suguru03.github.io/neo-async/doc/async.transformLimit.html) \* - [`sortBy`](http://suguru03.github.io/neo-async/doc/async.sortBy.html) - [`sortBySeries`](http://suguru03.github.io/neo-async/doc/async.sortBySeries.html) \* - [`sortByLimit`](http://suguru03.github.io/neo-async/doc/async.sortByLimit.html) \* - [`some`](http://suguru03.github.io/neo-async/doc/async.some.html) - [`someSeries`](http://suguru03.github.io/neo-async/doc/async.someSeries.html) - [`someLimit`](http://suguru03.github.io/neo-async/doc/async.someLimit.html) - [`any`](http://suguru03.github.io/neo-async/doc/async.some.html) -> [`some`](http://suguru03.github.io/neo-async/doc/async.some.html) - [`anySeries`](http://suguru03.github.io/neo-async/doc/async.someSeries.html) -> [`someSeries`](http://suguru03.github.io/neo-async/doc/async.someSeries.html) - [`anyLimit`](http://suguru03.github.io/neo-async/doc/async.someLimit.html) -> [`someLimit`](http://suguru03.github.io/neo-async/doc/async.someLimit.html) - [`every`](http://suguru03.github.io/neo-async/doc/async.every.html) - [`everySeries`](http://suguru03.github.io/neo-async/doc/async.everySeries.html) - [`everyLimit`](http://suguru03.github.io/neo-async/doc/async.everyLimit.html) - [`all`](http://suguru03.github.io/neo-async/doc/async.every.html) -> [`every`](http://suguru03.github.io/neo-async/doc/async.every.html) - [`allSeries`](http://suguru03.github.io/neo-async/doc/async.everySeries.html) -> [`every`](http://suguru03.github.io/neo-async/doc/async.everySeries.html) - [`allLimit`](http://suguru03.github.io/neo-async/doc/async.everyLimit.html) -> [`every`](http://suguru03.github.io/neo-async/doc/async.everyLimit.html) - [`concat`](http://suguru03.github.io/neo-async/doc/async.concat.html) - [`concatSeries`](http://suguru03.github.io/neo-async/doc/async.concatSeries.html) - [`concatLimit`](http://suguru03.github.io/neo-async/doc/async.concatLimit.html) \* ### Control Flow - [`parallel`](http://suguru03.github.io/neo-async/doc/async.parallel.html) - [`series`](http://suguru03.github.io/neo-async/doc/async.series.html) - [`parallelLimit`](http://suguru03.github.io/neo-async/doc/async.series.html) - [`tryEach`](http://suguru03.github.io/neo-async/doc/async.tryEach.html) - [`waterfall`](http://suguru03.github.io/neo-async/doc/async.waterfall.html) - [`angelFall`](http://suguru03.github.io/neo-async/doc/async.angelFall.html) \* - [`angelfall`](http://suguru03.github.io/neo-async/doc/async.angelFall.html) -> [`angelFall`](http://suguru03.github.io/neo-async/doc/async.angelFall.html) \* - [`whilst`](#whilst) - [`doWhilst`](#doWhilst) - [`until`](#until) - [`doUntil`](#doUntil) - [`during`](#during) - [`doDuring`](#doDuring) - [`forever`](#forever) - [`compose`](#compose) - [`seq`](#seq) - [`applyEach`](#applyEach) - [`applyEachSeries`](#applyEachSeries) - [`queue`](#queue) - [`priorityQueue`](#priorityQueue) - [`cargo`](#cargo) - [`auto`](#auto) - [`autoInject`](#autoInject) - [`retry`](#retry) - [`retryable`](#retryable) - [`iterator`](#iterator) - [`times`](http://suguru03.github.io/neo-async/doc/async.times.html) - [`timesSeries`](http://suguru03.github.io/neo-async/doc/async.timesSeries.html) - [`timesLimit`](http://suguru03.github.io/neo-async/doc/async.timesLimit.html) - [`race`](#race) ### Utils - [`apply`](#apply) - [`setImmediate`](#setImmediate) - [`nextTick`](#nextTick) - [`memoize`](#memoize) - [`unmemoize`](#unmemoize) - [`ensureAsync`](#ensureAsync) - [`constant`](#constant) - [`asyncify`](#asyncify) - [`wrapSync`](#asyncify) -> [`asyncify`](#asyncify) - [`log`](#log) - [`dir`](#dir) - [`timeout`](http://suguru03.github.io/neo-async/doc/async.timeout.html) - [`reflect`](#reflect) - [`reflectAll`](#reflectAll) - [`createLogger`](#createLogger) ## Mode - [`safe`](#safe) \* - [`fast`](#fast) \* ## Benchmark [Benchmark: Async vs Neo-Async](http://suguru03.hatenablog.com/entry/2016/06/10/135559) ### How to check ```bash $ node perf ``` ### Environment \* Darwin 17.3.0 x64 \* Node.js v8.9.4 \* async v2.6.0 \* neo-async v2.5.0 \* benchmark v2.1.4 ### Result The value is the ratio (Neo-Async/Async) of the average speed. #### Collections |function|benchmark| |---|--:| |each/forEach|2.43| |eachSeries/forEachSeries|1.75| |eachLimit/forEachLimit|1.68| |eachOf|3.29| |eachOfSeries|1.50| |eachOfLimit|1.59| |map|3.95| |mapSeries|1.81| |mapLimit|1.27| |mapValues|2.73| |mapValuesSeries|1.59| |mapValuesLimit|1.23| |filter|3.00| |filterSeries|1.74| |filterLimit|1.17| |reject|4.59| |rejectSeries|2.31| |rejectLimit|1.58| |detect|4.30| |detectSeries|1.86| |detectLimit|1.32| |reduce|1.82| |transform|2.46| |sortBy|4.08| |some|2.19| |someSeries|1.83| |someLimit|1.32| |every|2.09| |everySeries|1.84| |everyLimit|1.35| |concat|3.79| |concatSeries|4.45| #### Control Flow |funciton|benchmark| |---|--:| |parallel|2.93| |series|1.96| |waterfall|1.29| |whilst|1.00| |doWhilst|1.12| |until|1.12| |doUntil|1.12| |during|1.18| |doDuring|2.42| |times|4.25| |auto|1.97| ## License [![FOSSA Status](https://app.fossa.io/api/projects/git%2Bgithub.com%2Fsuguru03%2Fneo-async.svg?type=large)](https://app.fossa.io/projects/git%2Bgithub.com%2Fsuguru03%2Fneo-async?ref=badge\_large)