# v3.2.5

* Ensure Error objects such as AggregateError are propagated without modification (#1920)

# v3.2.4

* Fix a bug in priorityQueue where it didn't wait for the result. (#1725)
* Fix a bug where unshiftAsync was included in priorityQueue. (#1790)

# v3.2.3

* Fix bugs in comment parsing in autoInject. (#1767, #1780)

# v3.2.2

* Fix potential prototype pollution exploit

# v3.2.1

* Use queueMicrotask if available to the environment (#1761)
* Minor perf improvement in priorityQueue (#1727)
* More examples in documentation (#1726)
* Various doc fixes (#1708, #1712, #1717, #1740, #1739, #1749, #1756)
* Improved test coverage (#1754)

# v3.2.0

* Fix a bug in Safari related to overwriting func.name
* Remove built-in browserify configuration (#1653)
* Varios doc fixes (#1688, #1703, #1704)

# v3.1.1

* Allow redefining name property on wrapped functions.

# v3.1.0

* Added q.pushAsync and q.unshiftAsync, analagous to q.push and q.unshift, except they always do not accept a callback, and reject if processing the task errors. (#1659)
* Promises returned from q.push and q.unshift when a callback is not passed now resolve even if an error ocurred. (#1659)
* Fixed a parsing bug in autoInject with complicated function bodies (#1663)
* Added ES6+ configuration for Browserify bundlers (#1653)
* Various doc fixes (#1664, #1658, #1665, #1652)

# v3.0.1

## Bug fixes

* Fixed a regression where arrays passed to queue and cargo would be completely flattened. (#1645)
* Clarified Async's browser support (#1643)

# v3.0.0

The async/await release!

There are a lot of new features and subtle breaking changes in this major version, but the biggest feature is that most Async methods return a Promise if you omit the callback, meaning you can await them from within an async function.

const results = await async.mapLimit(urls, 5, async url => {

const resp = await fetch(url)

return resp.body

})

## Breaking Changes

* Most Async methods return a Promise when the final callback is omitted, making them await-able! (#1572)
* We are now making heavy use of ES2015 features, this means we have dropped out-of-the-box support for Node 4 and earlier, and many old versions of browsers. (#1541, #1553)
* In queue, priorityQueue, cargo and cargoQueue, the "event"-style methods, like q.drain and q.saturated are now methods that register a callback, rather than properties you assign a callback to. They are now of the form q.drain(callback). If you do not pass a callback a Promise will be returned for the next occurrence of the event, making them await-able, e.g. await q.drain(). (#1586, #1641)
* Calling callback(false) will cancel an async method, preventing further iteration and callback calls. This is useful for preventing memory leaks when you break out of an async flow by calling an outer callback. (#1064, #1542)
* during and doDuring have been removed, and instead whilst, doWhilst, until and doUntil now have asynchronous test functions. (#850, #1557)
* limits of less than 1 now cause an error to be thrown in queues and collection methods. (#1249, #1552)
* memoize no longer memoizes errors (#1465, #1466)
* applyEach/applyEachSeries have a simpler interface, to make them more easily type-able. It always returns a function that takes in a single callback argument. If that callback is omitted, a promise is returned, making it awaitable. (#1228, #1640)

## New Features

* Async generators are now supported in all the Collection methods. (#1560)
* Added cargoQueue, a queue with both concurrency and payload size parameters. (#1567)
* Queue objects returned from queue now have a Symbol.iterator method, meaning they can be iterated over to inspect the current list of items in the queue. (#1459, #1556)
* A ESM-flavored async.mjs is included in the async package. This is described in the package.json "module" field, meaning it should be automatically used by Webpack and other compatible bundlers.

## Bug fixes

* Better handle arbitrary error objects in asyncify (#1568, #1569)

## Other

* Removed Lodash as a dependency (#1283, #1528)
* Miscellaneous docs fixes (#1393, #1501, #1540, #1543, #1558, #1563, #1564, #1579, #1581)
* Miscellaneous test fixes (#1538)

# v2.6.1

* Updated lodash to prevent npm audit warnings. (#1532, #1533)
* Made async-es more optimized for webpack users (#1517)
* Fixed a stack overflow with large collections and a synchronous iterator (#1514)
* Various small fixes/chores (#1505, #1511, #1527, #1530)

# v2.6.0

* Added missing aliases for many methods. Previously, you could not (e.g.) require('async/find') or use async.anyLimit. (#1483)
* Improved queue performance. (#1448, #1454)
* Add missing sourcemap (#1452, #1453)
* Various doc updates (#1448, #1471, #1483)

# v2.5.0

* Added concatLimit, the Limit equivalent of [concat](https://caolan.github.io/async/docs.html#concat) ([#1426](https://github.com/caolan/async/issues/1426), [#1430](https://github.com/caolan/async/pull/1430))
* concat improvements: it now preserves order, handles falsy values and the iteratee callback takes a variable number of arguments ([#1437](https://github.com/caolan/async/issues/1437), [#1436](https://github.com/caolan/async/pull/1436))
* Fixed an issue in queue where there was a size discrepancy between workersList().length and running() ([#1428](https://github.com/caolan/async/issues/1428), [#1429](https://github.com/caolan/async/pull/1429))
* Various doc fixes ([#1422](https://github.com/caolan/async/issues/1422), [#1424](https://github.com/caolan/async/pull/1424))

# v2.4.1

* Fixed a bug preventing functions wrapped with timeout() from being re-used. ([#1418](https://github.com/caolan/async/issues/1418), [#1419](https://github.com/caolan/async/issues/1419))

# v2.4.0

* Added tryEach, for running async functions in parallel, where you only expect one to succeed. ([#1365](https://github.com/caolan/async/issues/1365), [#687](https://github.com/caolan/async/issues/687))
* Improved performance, most notably in parallel and waterfall ([#1395](https://github.com/caolan/async/issues/1395))
* Added queue.remove(), for removing items in a queue ([#1397](https://github.com/caolan/async/issues/1397), [#1391](https://github.com/caolan/async/issues/1391))
* Fixed using eval, preventing Async from running in pages with Content Security Policy ([#1404](https://github.com/caolan/async/issues/1404), [#1403](https://github.com/caolan/async/issues/1403))
* Fixed errors thrown in an asyncifyed function's callback being caught by the underlying Promise ([#1408](https://github.com/caolan/async/issues/1408))
* Fixed timing of queue.empty() ([#1367](https://github.com/caolan/async/issues/1367))
* Various doc fixes ([#1314](https://github.com/caolan/async/issues/1314), [#1394](https://github.com/caolan/async/issues/1394), [#1412](https://github.com/caolan/async/issues/1412))

# v2.3.0

* Added support for ES2017 async functions. Wherever you can pass a Node-style/CPS function that uses a callback, you can also pass an async function. Previously, you had to wrap async functions with asyncify. The caveat is that it will only work if async functions are supported natively in your environment, transpiled implementations can't be detected. ([#1386](https://github.com/caolan/async/issues/1386), [#1390](https://github.com/caolan/async/issues/1390))
* Small doc fix ([#1392](https://github.com/caolan/async/issues/1392))

# v2.2.0

* Added groupBy, and the Series/Limit equivalents, analogous to [\_.groupBy](http://lodash.com/docs#groupBy) ([#1364](https://github.com/caolan/async/issues/1364))
* Fixed transform bug when callback was not passed ([#1381](https://github.com/caolan/async/issues/1381))
* Added note about reflect to parallel docs ([#1385](https://github.com/caolan/async/issues/1385))

# v2.1.5

* Fix auto bug when function names collided with Array.prototype ([#1358](https://github.com/caolan/async/issues/1358))
* Improve some error messages ([#1349](https://github.com/caolan/async/issues/1349))
* Avoid stack overflow case in queue
* Fixed an issue in some, every and find where processing would continue after the result was determined.
* Cleanup implementations of some, every and find

# v2.1.3

* Make bundle size smaller
* Create optimized hotpath for filter in array case.

# v2.1.2

* Fixed a stackoverflow bug with detect, some, every on large inputs ([#1293](https://github.com/caolan/async/issues/1293)).

# v2.1.0

* retry and retryable now support an optional errorFilter function that determines if the task should retry on the error ([#1256](https://github.com/caolan/async/issues/1256), [#1261](https://github.com/caolan/async/issues/1261))
* Optimized array iteration in race, cargo, queue, and priorityQueue ([#1253](https://github.com/caolan/async/issues/1253))
* Added alias documentation to doc site ([#1251](https://github.com/caolan/async/issues/1251), [#1254](https://github.com/caolan/async/issues/1254))
* Added [BootStrap scrollspy](http://getbootstrap.com/javascript/#scrollspy) to docs to highlight in the sidebar the current method being viewed ([#1289](https://github.com/caolan/async/issues/1289), [#1300](https://github.com/caolan/async/issues/1300))
* Various minor doc fixes ([#1263](https://github.com/caolan/async/issues/1263), [#1264](https://github.com/caolan/async/issues/1264), [#1271](https://github.com/caolan/async/issues/1271), [#1278](https://github.com/caolan/async/issues/1278), [#1280](https://github.com/caolan/async/issues/1280), [#1282](https://github.com/caolan/async/issues/1282), [#1302](https://github.com/caolan/async/issues/1302))

# v2.0.1

* Significantly optimized all iteration based collection methods such as each, map, filter, etc ([#1245](https://github.com/caolan/async/issues/1245), [#1246](https://github.com/caolan/async/issues/1246), [#1247](https://github.com/caolan/async/issues/1247)).

# v2.0.0

Lots of changes here!

First and foremost, we have a slick new [site for docs](https://caolan.github.io/async/). Special thanks to [**@hargasinski**](https://github.com/hargasinski) for his work converting our old docs to jsdoc format and implementing the new website. Also huge ups to [**@ivanseidel**](https://github.com/ivanseidel) for designing our new logo. It was a long process for both of these tasks, but I think these changes turned out extraordinary well.

The biggest feature is modularization. You can now require("async/series") to only require the series function. Every Async library function is available this way. You still can require("async") to require the entire library, like you could do before.

We also provide Async as a collection of ES2015 modules. You can now import {each} from 'async-es' or import waterfall from 'async-es/waterfall'. If you are using only a few Async functions, and are using a ES bundler such as Rollup, this can significantly lower your build size.

Major thanks to [**@Kikobeats**](http://github.com/Kikobeats), [**@aearly**](http://github.com/aearly) and [**@megawac**](http://github.com/megawac) for doing the majority of the modularization work, as well as [**@jdalton**](http://github.com/jdalton) and [**@Rich-Harris**](http://github.com/Rich-Harris) for advisory work on the general modularization strategy.

Another one of the general themes of the 2.0 release is standardization of what an "async" function is. We are now more strictly following the node-style continuation passing style. That is, an async function is a function that:

1. Takes a variable number of arguments
2. The last argument is always a callback
3. The callback can accept any number of arguments
4. The first argument passed to the callback will be treated as an error result, if the argument is truthy
5. Any number of result arguments can be passed after the "error" argument
6. The callback is called once and exactly once, either on the same tick or later tick of the JavaScript event loop.

There were several cases where Async accepted some functions that did not strictly have these properties, most notably auto, every, some, filter, reject and detect.

Another theme is performance. We have eliminated internal deferrals in all cases where they make sense. For example, in waterfall and auto, there was a setImmediate between each task -- these deferrals have been removed. A setImmediate call can add up to 1ms of delay. This might not seem like a lot, but it can add up if you are using many Async functions in the course of processing a HTTP request, for example. Nearly all asynchronous functions that do I/O already have some sort of deferral built in, so the extra deferral is unnecessary. The trade-off of this change is removing our built-in stack-overflow defense. Many synchronous callback calls in series can quickly overflow the JS call stack. If you do have a function that is sometimes synchronous (calling its callback on the same tick), and are running into stack overflows, wrap it with async.ensureAsync().

Another big performance win has been re-implementing queue, cargo, and priorityQueue with [doubly linked lists](https://en.wikipedia.org/wiki/Doubly_linked_list) instead of arrays. This has lead to queues being an order of [magnitude faster on large sets of tasks](https://github.com/caolan/async/pull/1205).

## New Features

* Async is now modularized. Individual functions can be require()d from the main package. (require('async/auto')) ([#984](https://github.com/caolan/async/issues/984), [#996](https://github.com/caolan/async/issues/996))
* Async is also available as a collection of ES2015 modules in the new async-es package. (import {forEachSeries} from 'async-es') ([#984](https://github.com/caolan/async/issues/984), [#996](https://github.com/caolan/async/issues/996))
* Added race, analogous to Promise.race(). It will run an array of async tasks in parallel and will call its callback with the result of the first task to respond. ([#568](https://github.com/caolan/async/issues/568), [#1038](https://github.com/caolan/async/issues/1038))
* Collection methods now accept ES2015 iterators. Maps, Sets, and anything that implements the iterator spec can now be passed directly to each, map, parallel, etc.. ([#579](https://github.com/caolan/async/issues/579), [#839](https://github.com/caolan/async/issues/839), [#1074](https://github.com/caolan/async/issues/1074))
* Added mapValues, for mapping over the properties of an object and returning an object with the same keys. ([#1157](https://github.com/caolan/async/issues/1157), [#1177](https://github.com/caolan/async/issues/1177))
* Added timeout, a wrapper for an async function that will make the task time-out after the specified time. ([#1007](https://github.com/caolan/async/issues/1007), [#1027](https://github.com/caolan/async/issues/1027))
* Added reflect and reflectAll, analagous to [Promise.reflect()](http://bluebirdjs.com/docs/api/reflect.html), a wrapper for async tasks that always succeeds, by gathering results and errors into an object. ([#942](https://github.com/caolan/async/issues/942), [#1012](https://github.com/caolan/async/issues/1012), [#1095](https://github.com/caolan/async/issues/1095))
* constant supports dynamic arguments -- it will now always use its last argument as the callback. ([#1016](https://github.com/caolan/async/issues/1016), [#1052](https://github.com/caolan/async/issues/1052))
* setImmediate and nextTick now support arguments to partially apply to the deferred function, like the node-native versions do. ([#940](https://github.com/caolan/async/issues/940), [#1053](https://github.com/caolan/async/issues/1053))
* auto now supports resolving cyclic dependencies using [Kahn's algorithm](https://en.wikipedia.org/wiki/Topological_sorting#Kahn.27s_algorithm) ([#1140](https://github.com/caolan/async/issues/1140)).
* Added autoInject, a relative of auto that automatically spreads a task's dependencies as arguments to the task function. ([#608](https://github.com/caolan/async/issues/608), [#1055](https://github.com/caolan/async/issues/1055), [#1099](https://github.com/caolan/async/issues/1099), [#1100](https://github.com/caolan/async/issues/1100))
* You can now limit the concurrency of auto tasks. ([#635](https://github.com/caolan/async/issues/635), [#637](https://github.com/caolan/async/issues/637))
* Added retryable, a relative of retry that wraps an async function, making it retry when called. ([#1058](https://github.com/caolan/async/issues/1058))
* retry now supports specifying a function that determines the next time interval, useful for exponential backoff, logging and other retry strategies. ([#1161](https://github.com/caolan/async/issues/1161))
* retry will now pass all of the arguments the task function was resolved with to the callback ([#1231](https://github.com/caolan/async/issues/1231)).
* Added q.unsaturated -- callback called when a queue's number of running workers falls below a threshold. ([#868](https://github.com/caolan/async/issues/868), [#1030](https://github.com/caolan/async/issues/1030), [#1033](https://github.com/caolan/async/issues/1033), [#1034](https://github.com/caolan/async/issues/1034))
* Added q.error -- a callback called whenever a queue task calls its callback with an error. ([#1170](https://github.com/caolan/async/issues/1170))
* applyEach and applyEachSeries now pass results to the final callback. ([#1088](https://github.com/caolan/async/issues/1088))

## Breaking changes

* Calling a callback more than once is considered an error, and an error will be thrown. This had an explicit breaking change in waterfall. If you were relying on this behavior, you should more accurately represent your control flow as an event emitter or stream. ([#814](https://github.com/caolan/async/issues/814), [#815](https://github.com/caolan/async/issues/815), [#1048](https://github.com/caolan/async/issues/1048), [#1050](https://github.com/caolan/async/issues/1050))
* auto task functions now always take the callback as the last argument. If a task has dependencies, the results object will be passed as the first argument. To migrate old task functions, wrap them with [\_.flip](https://lodash.com/docs#flip) ([#1036](https://github.com/caolan/async/issues/1036), [#1042](https://github.com/caolan/async/issues/1042))
* Internal setImmediate calls have been refactored away. This may make existing flows vulnerable to stack overflows if you use many synchronous functions in series. Use ensureAsync to work around this. ([#696](https://github.com/caolan/async/issues/696), [#704](https://github.com/caolan/async/issues/704), [#1049](https://github.com/caolan/async/issues/1049), [#1050](https://github.com/caolan/async/issues/1050))
* map used to return an object when iterating over an object. map now always returns an array, like in other libraries. The previous object behavior has been split out into mapValues. ([#1157](https://github.com/caolan/async/issues/1157), [#1177](https://github.com/caolan/async/issues/1177))
* filter, reject, some, every, detect and their families like {METHOD}Series and {METHOD}Limit now expect an error as the first callback argument, rather than just a simple boolean. Pass null as the first argument, or use fs.access instead of fs.exists. ([#118](https://github.com/caolan/async/issues/118), [#774](https://github.com/caolan/async/issues/774), [#1028](https://github.com/caolan/async/issues/1028), [#1041](https://github.com/caolan/async/issues/1041))
* {METHOD} and {METHOD}Series are now implemented in terms of {METHOD}Limit. This is a major internal simplification, and is not expected to cause many problems, but it does subtly affect how functions execute internally. ([#778](https://github.com/caolan/async/issues/778), [#847](https://github.com/caolan/async/issues/847))
* retry's callback is now optional. Previously, omitting the callback would partially apply the function, meaning it could be passed directly as a task to series or auto. The partially applied "control-flow" behavior has been separated out into retryable. ([#1054](https://github.com/caolan/async/issues/1054), [#1058](https://github.com/caolan/async/issues/1058))
* The test function for whilst, until, and during used to be passed non-error args from the iteratee function's callback, but this led to weirdness where the first call of the test function would be passed no args. We have made it so the test function is never passed extra arguments, and only the doWhilst, doUntil, and doDuring functions pass iteratee callback arguments to the test function ([#1217](https://github.com/caolan/async/issues/1217), [#1224](https://github.com/caolan/async/issues/1224))
* The q.tasks array has been renamed q.\_tasks and is now implemented as a doubly linked list (DLL). Any code that used to interact with this array will need to be updated to either use the provided helpers or support DLLs ([#1205](https://github.com/caolan/async/issues/1205)).
* The timing of the q.saturated() callback in a queue has been modified to better reflect when tasks pushed to the queue will start queueing. ([#724](https://github.com/caolan/async/issues/724), [#1078](https://github.com/caolan/async/issues/1078))
* Removed iterator method in favour of [ES2015 iterator protocol](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Iterators_and_Generators) which natively supports arrays ([#1237](https://github.com/caolan/async/issues/1237))
* Dropped support for Component, Jam, SPM, and Volo ([#1175](https://github.com/caolan/async/issues/1175), #[#176](https://github.com/caolan/async/issues/176))

## Bug Fixes

* Improved handling of no dependency cases in auto & autoInject ([#1147](https://github.com/caolan/async/issues/1147)).
* Fixed a bug where the callback generated by asyncify with Promises could resolve twice ([#1197](https://github.com/caolan/async/issues/1197)).
* Fixed several documented optional callbacks not actually being optional ([#1223](https://github.com/caolan/async/issues/1223)).

## Other

* Added someSeries and everySeries for symmetry, as well as a complete set of any/anyLimit/anySeries and all//allLmit/allSeries aliases.
* Added find as an alias for detect. (as well as findLimitandfindSeries`).
* Various doc fixes ([#1005](https://github.com/caolan/async/issues/1005), [#1008](https://github.com/caolan/async/issues/1008), [#1010](https://github.com/caolan/async/issues/1010), [#1015](https://github.com/caolan/async/issues/1015), [#1021](https://github.com/caolan/async/issues/1021), [#1037](https://github.com/caolan/async/issues/1037), [#1039](https://github.com/caolan/async/issues/1039), [#1051](https://github.com/caolan/async/issues/1051), [#1102](https://github.com/caolan/async/issues/1102), [#1107](https://github.com/caolan/async/issues/1107), [#1121](https://github.com/caolan/async/issues/1121), [#1123](https://github.com/caolan/async/issues/1123), [#1129](https://github.com/caolan/async/issues/1129), [#1135](https://github.com/caolan/async/issues/1135), [#1138](https://github.com/caolan/async/issues/1138), [#1141](https://github.com/caolan/async/issues/1141), [#1153](https://github.com/caolan/async/issues/1153), [#1216](https://github.com/caolan/async/issues/1216), [#1217](https://github.com/caolan/async/issues/1217), [#1232](https://github.com/caolan/async/issues/1232), [#1233](https://github.com/caolan/async/issues/1233), [#1236](https://github.com/caolan/async/issues/1236), [#1238](https://github.com/caolan/async/issues/1238))

Thank you [**@aearly**](http://github.com/aearly) and [**@megawac**](http://github.com/megawac) for taking the lead on version 2 of async.

# v1.5.2

* Allow using "constructor" as an argument in memoize ([#998](https://github.com/caolan/async/issues/998))
* Give a better error messsage when auto dependency checking fails ([#994](https://github.com/caolan/async/issues/994))
* Various doc updates ([#936](https://github.com/caolan/async/issues/936), [#956](https://github.com/caolan/async/issues/956), [#979](https://github.com/caolan/async/issues/979), [#1002](https://github.com/caolan/async/issues/1002))

# v1.5.1

* Fix issue with pause in queue with concurrency enabled ([#946](https://github.com/caolan/async/issues/946))
* while and until now pass the final result to callback ([#963](https://github.com/caolan/async/issues/963))
* auto will properly handle concurrency when there is no callback ([#966](https://github.com/caolan/async/issues/966))
* auto will no. properly stop execution when an error occurs ([#988](https://github.com/caolan/async/issues/988), [#993](https://github.com/caolan/async/issues/993))
* Various doc fixes ([#971](https://github.com/caolan/async/issues/971), [#980](https://github.com/caolan/async/issues/980))

# v1.5.0

* Added transform, analogous to [\_.transform](http://lodash.com/docs#transform) ([#892](https://github.com/caolan/async/issues/892))
* map now returns an object when an object is passed in, rather than array with non-numeric keys. map will begin always returning an array with numeric indexes in the next major release. ([#873](https://github.com/caolan/async/issues/873))
* auto now accepts an optional concurrency argument to limit the number o. running tasks ([#637](https://github.com/caolan/async/issues/637))
* Added queue#workersList(), to retrieve the lis. of currently running tasks. ([#891](https://github.com/caolan/async/issues/891))
* Various code simplifications ([#896](https://github.com/caolan/async/issues/896), [#904](https://github.com/caolan/async/issues/904))
* Various doc fixes :scroll: ([#890](https://github.com/caolan/async/issues/890), [#894](https://github.com/caolan/async/issues/894), [#903](https://github.com/caolan/async/issues/903), [#905](https://github.com/caolan/async/issues/905), [#912](https://github.com/caolan/async/issues/912))

# v1.4.2

* Ensure coverage files don't get published on npm ([#879](https://github.com/caolan/async/issues/879))

# v1.4.1

* Add in overlooked detectLimit method ([#866](https://github.com/caolan/async/issues/866))
* Removed unnecessary files from npm releases ([#861](https://github.com/caolan/async/issues/861))
* Removed usage of a reserved word to prevent :boom: in older environments ([#870](https://github.com/caolan/async/issues/870))

# v1.4.0

* asyncify now supports promises ([#840](https://github.com/caolan/async/issues/840))
* Added Limit versions of filter and reject ([#836](https://github.com/caolan/async/issues/836))
* Add Limit versions of detect, some and every ([#828](https://github.com/caolan/async/issues/828), [#829](https://github.com/caolan/async/issues/829))
* some, every and detect now short circuit early ([#828](https://github.com/caolan/async/issues/828), [#829](https://github.com/caolan/async/issues/829))
* Improve detection of the global object ([#804](https://github.com/caolan/async/issues/804)), enabling use in WebWorkers
* whilst now called with arguments from iterator ([#823](https://github.com/caolan/async/issues/823))
* during now gets called with arguments from iterator ([#824](https://github.com/caolan/async/issues/824))
* Code simplifications and optimizations aplenty ([diff](https://github.com/caolan/async/compare/v1.3.0...v1.4.0))

# v1.3.0

New Features:

* Added constant
* Added asyncify/wrapSync for making sync functions work with callbacks. ([#671](https://github.com/caolan/async/issues/671), [#806](https://github.com/caolan/async/issues/806))
* Added during and doDuring, which are like whilst with an async truth test. ([#800](https://github.com/caolan/async/issues/800))
* retry now accepts an interval parameter to specify a delay between retries. ([#793](https://github.com/caolan/async/issues/793))
* async should work better in Web Workers due to better root detection ([#804](https://github.com/caolan/async/issues/804))
* Callbacks are now optional in whilst, doWhilst, until, and doUntil ([#642](https://github.com/caolan/async/issues/642))
* Various internal updates ([#786](https://github.com/caolan/async/issues/786), [#801](https://github.com/caolan/async/issues/801), [#802](https://github.com/caolan/async/issues/802), [#803](https://github.com/caolan/async/issues/803))
* Various doc fixes ([#790](https://github.com/caolan/async/issues/790), [#794](https://github.com/caolan/async/issues/794))

Bug Fixes:

* cargo now exposes the payload size, and cargo.payload can be changed on the fly after the cargo is created. ([#740](https://github.com/caolan/async/issues/740), [#744](https://github.com/caolan/async/issues/744), [#783](https://github.com/caolan/async/issues/783))

# v1.2.1

Bug Fix:

* Small regression with synchronous iterator behavior in eachSeries with a 1-element array. Before 1.1.0, eachSeries's callback was called on the same tick, which this patch restores. In 2.0.0, it will be called on the next tick. ([#782](https://github.com/caolan/async/issues/782))

# v1.2.0

New Features:

* Added timesLimit ([#743](https://github.com/caolan/async/issues/743))
* concurrency can be changed after initialization in queue by setting q.concurrency. The new concurrency will be reflected the next time a task is processed. ([#747](https://github.com/caolan/async/issues/747), [#772](https://github.com/caolan/async/issues/772))

Bug Fixes:

* Fixed a regression in each and family with empty arrays that have additional properties. ([#775](https://github.com/caolan/async/issues/775), [#777](https://github.com/caolan/async/issues/777))

# v1.1.1

Bug Fix:

* Small regression with synchronous iterator behavior in eachSeries with a 1-element array. Before 1.1.0, eachSeries's callback was called on the same tick, which this patch restores. In 2.0.0, it will be called on the next tick. ([#782](https://github.com/caolan/async/issues/782))

# v1.1.0

New Features:

* cargo now supports all of the same methods and event callbacks as queue.
* Added ensureAsync - A wrapper that ensures an async function calls its callback on a later tick. ([#769](https://github.com/caolan/async/issues/769))
* Optimized map, eachOf, and waterfall families of functions
* Passing a null or undefined array to map, each, parallel and families will be treated as an empty array ([#667](https://github.com/caolan/async/issues/667)).
* The callback is now optional for the composed results of compose and seq. ([#618](https://github.com/caolan/async/issues/618))
* Reduced file size by 4kb, (minified version by 1kb)
* Added code coverage through nyc and coveralls ([#768](https://github.com/caolan/async/issues/768))

Bug Fixes:

* forever will no longer stack overflow with a synchronous iterator ([#622](https://github.com/caolan/async/issues/622))
* eachLimit and other limit functions will stop iterating once an error occurs ([#754](https://github.com/caolan/async/issues/754))
* Always pass null in callbacks when there is no error ([#439](https://github.com/caolan/async/issues/439))
* Ensure proper conditions when calling drain() after pushing an empty data set to a queue ([#668](https://github.com/caolan/async/issues/668))
* each and family will properly handle an empty array ([#578](https://github.com/caolan/async/issues/578))
* eachSeries and family will finish if the underlying array is modified during execution ([#557](https://github.com/caolan/async/issues/557))
* queue will throw if a non-function is passed to q.push() ([#593](https://github.com/caolan/async/issues/593))
* Doc fixes ([#629](https://github.com/caolan/async/issues/629), [#766](https://github.com/caolan/async/issues/766))

# v1.0.0

No known breaking changes, we are simply complying with semver from here on out.

Changes:

* Start using a changelog!
* Add forEachOf for iterating over Objects (or to iterate Arrays with indexes available) ([#168](https://github.com/caolan/async/issues/168) [#704](https://github.com/caolan/async/issues/704) [#321](https://github.com/caolan/async/issues/321))
* Detect deadlocks in auto ([#663](https://github.com/caolan/async/issues/663))
* Better support for require.js ([#527](https://github.com/caolan/async/issues/527))
* Throw if queue created with concurrency 0 ([#714](https://github.com/caolan/async/issues/714))
* Fix unneeded iteration in queue.resume() ([#758](https://github.com/caolan/async/issues/758))
* Guard against timer mocking overriding setImmediate ([#609](https://github.com/caolan/async/issues/609) [#611](https://github.com/caolan/async/issues/611))
* Miscellaneous doc fixes ([#542](https://github.com/caolan/async/issues/542) [#596](https://github.com/caolan/async/issues/596) [#615](https://github.com/caolan/async/issues/615) [#628](https://github.com/caolan/async/issues/628) [#631](https://github.com/caolan/async/issues/631) [#690](https://github.com/caolan/async/issues/690) [#729](https://github.com/caolan/async/issues/729))
* Use single noop function internally ([#546](https://github.com/caolan/async/issues/546))
* Optimize internal \_each, \_map and \_keys functions.