# readdirp [Weekly downloads](https://github.com/paulmillr/readdirp)

Recursive version of [fs.readdir](https://nodejs.org/api/fs.html#fs_fs_readdir_path_options_callback). Exposes a **stream API** and a **promise API**.

npm install readdirp

const readdirp = require('readdirp');

// Use streams to achieve small RAM & CPU footprint.

// 1) Streams example with for-await.

for await (const entry of readdirp('.')) {

const {path} = entry;

console.log(`${JSON.stringify({path})}`);

}

// 2) Streams example, non for-await.

// Print out all JS files along with their size within the current folder & subfolders.

readdirp('.', {fileFilter: '\*.js', alwaysStat: true})

.on('data', (entry) => {

const {path, stats: {size}} = entry;

console.log(`${JSON.stringify({path, size})}`);

})

// Optionally call stream.destroy() in `warn()` in order to abort and cause 'close' to be emitted

.on('warn', error => console.error('non-fatal error', error))

.on('error', error => console.error('fatal error', error))

.on('end', () => console.log('done'));

// 3) Promise example. More RAM and CPU than streams / for-await.

const files = await readdirp.promise('.');

console.log(files.map(file => file.path));

// Other options.

readdirp('test', {

fileFilter: '\*.js',

directoryFilter: ['!.git', '!\*modules']

// directoryFilter: (di) => di.basename.length === 9

type: 'files\_directories',

depth: 1

});

For more examples, check out examples directory.

## API

const stream = readdirp(root[, options]) — **Stream API**

* Reads given root recursively and returns a stream of [entry infos](#3dy6vkm)
* Optionally can be used like for await (const entry of stream) with node.js 10+ (asyncIterator).
* on('data', (entry) => {}) [entry info](#3dy6vkm) for every file / dir.
* on('warn', (error) => {}) non-fatal Error that prevents a file / dir from being processed. Example: inaccessible to the user.
* on('error', (error) => {}) fatal Error which also ends the stream. Example: illegal options where passed.
* on('end') — we are done. Called when all entries were found and no more will be emitted.
* on('close') — stream is destroyed via stream.destroy(). Could be useful if you want to manually abort even on a non fatal error. At that point the stream is no longer readable and no more entries, warning or errors are emitted
* To learn more about streams, consult the very detailed [nodejs streams documentation](https://nodejs.org/api/stream.html) or the [stream-handbook](https://github.com/substack/stream-handbook)

const entries = await readdirp.promise(root[, options]) — **Promise API**. Returns a list of [entry infos](#3dy6vkm).

First argument is awalys root, path in which to start reading and recursing into subdirectories.

### options

* fileFilter: ["\*.js"]: filter to include or exclude files. A Function, Glob string or Array of glob strings.
  + **Function**: a function that takes an entry info as a parameter and returns true to include or false to exclude the entry
  + **Glob string**: a string (e.g., \*.js) which is matched using [picomatch](https://github.com/micromatch/picomatch), so go there for more information. Globstars (\*\*) are not supported since specifying a recursive pattern for an already recursive function doesn't make sense. Negated globs (as explained in the minimatch documentation) are allowed, e.g., !\*.txt matches everything but text files.
  + **Array of glob strings**: either need to be all inclusive or all exclusive (negated) patterns otherwise an error is thrown. ['\*.json', '\*.js'] includes all JavaScript and Json files. ['!.git', '!node\_modules'] includes all directories except the '.git' and 'node\_modules'.
  + Directories that do not pass a filter will not be recursed into.
* directoryFilter: ['!.git']: filter to include/exclude directories found and to recurse into. Directories that do not pass a filter will not be recursed into.
* depth: 5: depth at which to stop recursing even if more subdirectories are found
* type: 'files': determines if data events on the stream should be emitted for 'files' (default), 'directories', 'files\_directories', or 'all'. Setting to 'all' will also include entries for other types of file descriptors like character devices, unix sockets and named pipes.
* alwaysStat: false: always return stats property for every file. Default is false, readdirp will return Dirent entries. Setting it to true can double readdir execution time - use it only when you need file size, mtime etc. Cannot be enabled on node <10.10.0.
* lstat: false: include symlink entries in the stream along with files. When true, fs.lstat would be used instead of fs.stat

### EntryInfo

Has the following properties:

* path: 'assets/javascripts/react.js': path to the file/directory (relative to given root)
* fullPath: '/Users/dev/projects/app/assets/javascripts/react.js': full path to the file/directory found
* basename: 'react.js': name of the file/directory
* dirent: fs.Dirent: built-in [dir entry object](https://nodejs.org/api/fs.html#fs_class_fs_dirent) - only with alwaysStat: false
* stats: fs.Stats: built in [stat object](https://nodejs.org/api/fs.html#fs_class_fs_stats) - only with alwaysStat: true

## Changelog

* 3.5 (Oct 13, 2020) disallows recursive directory-based symlinks. Before, it could have entered infinite loop.
* 3.4 (Mar 19, 2020) adds support for directory-based symlinks.
* 3.3 (Dec 6, 2019) stabilizes RAM consumption and enables perf management with highWaterMark option. Fixes race conditions related to for-await looping.
* 3.2 (Oct 14, 2019) improves performance by 250% and makes streams implementation more idiomatic.
* 3.1 (Jul 7, 2019) brings bigint support to stat output on Windows. This is backwards-incompatible for some cases. Be careful. It you use it incorrectly, you'll see "TypeError: Cannot mix BigInt and other types, use explicit conversions".
* 3.0 brings huge performance improvements and stream backpressure support.
* Upgrading 2.x to 3.x:
  + Signature changed from readdirp(options) to readdirp(root, options)
  + Replaced callback API with promise API.
  + Renamed entryType option to type
  + Renamed entryType: 'both' to 'files\_directories'
  + EntryInfo
    - Renamed stat to stats
      * Emitted only when alwaysStat: true
      * dirent is emitted instead of stats by default with alwaysStat: false
    - Renamed name to basename
    - Removed parentDir and fullParentDir properties
* Supported node.js versions:
  + 3.x: node 8+
  + 2.x: node 0.6+

## License

Copyright (c) 2012-2019 Thorsten Lorenz, Paul Miller (<https://paulmillr.com>)

MIT License, see [LICENSE](http://license) file.