# @nodelib/fs.stat

Get the status of a file with some features.

## :bulb: Highlights

Wrapper around standard method fs.lstat and fs.stat with some features.

* :beginner: Normally follows symbolic link.
* :gear: Can safely work with broken symbolic link.

## Install

npm install @nodelib/fs.stat

## Usage

import \* as fsStat from '@nodelib/fs.stat';

fsStat.stat('path', (error, stats) => { /\* … \*/ });

## API

### .stat(path, [optionsOrSettings], callback)

Returns an instance of fs.Stats class for provided path with standard callback-style.

fsStat.stat('path', (error, stats) => { /\* … \*/ });

fsStat.stat('path', {}, (error, stats) => { /\* … \*/ });

fsStat.stat('path', new fsStat.Settings(), (error, stats) => { /\* … \*/ });

### .statSync(path, [optionsOrSettings])

Returns an instance of fs.Stats class for provided path.

const stats = fsStat.stat('path');

const stats = fsStat.stat('path', {});

const stats = fsStat.stat('path', new fsStat.Settings());

#### path

* Required: true
* Type: string | Buffer | URL

A path to a file. If a URL is provided, it must use the file: protocol.

#### optionsOrSettings

* Required: false
* Type: Options | Settings
* Default: An instance of Settings class

An [Options](#options) object or an instance of [Settings](#settings) class.

:book: When you pass a plain object, an instance of the Settings class will be created automatically. If you plan to call the method frequently, use a pre-created instance of the Settings class.

### Settings([options])

A class of full settings of the package.

const settings = new fsStat.Settings({ followSymbolicLink: false });

const stats = fsStat.stat('path', settings);

## Options

### followSymbolicLink

* Type: boolean
* Default: true

Follow symbolic link or not. Call fs.stat on symbolic link if true.

### markSymbolicLink

* Type: boolean
* Default: false

Mark symbolic link by setting the return value of isSymbolicLink function to always true (even after fs.stat).

:book: Can be used if you want to know what is hidden behind a symbolic link, but still continue to know that it is a symbolic link.

### throwErrorOnBrokenSymbolicLink

* Type: boolean
* Default: true

Throw an error when symbolic link is broken if true or safely return lstat call if false.

### fs

* Type: [FileSystemAdapter](http://./src/adapters/fs.ts)
* Default: A default FS methods

By default, the built-in Node.js module (fs) is used to work with the file system. You can replace any method with your own.

interface FileSystemAdapter {

lstat?: typeof fs.lstat;

stat?: typeof fs.stat;

lstatSync?: typeof fs.lstatSync;

statSync?: typeof fs.statSync;

}

const settings = new fsStat.Settings({

fs: { lstat: fakeLstat }

});

## Changelog

See the [Releases section of our GitHub project](https://github.com/nodelib/nodelib/releases) for changelog for each release version.

## License

This software is released under the terms of the MIT license.