# require-directory

Recursively iterates over specified directory, require()'ing each file, and returning a nested hash structure containing those modules.

[**Follow me (@troygoode) on Twitter!**](https://twitter.com/intent/user?screen_name=troygoode)

[NPM](https://nodei.co/npm/require-directory/)

[build status](http://travis-ci.org/troygoode/node-require-directory)

## How To Use

### Installation (via [npm](https://npmjs.org/package/require-directory))

$ npm install require-directory

### Usage

A common pattern in node.js is to include an index file which creates a hash of the files in its current directory. Given a directory structure like so:

* app.js
* routes/
  + index.js
  + home.js
  + auth/
    - login.js
    - logout.js
    - register.js

routes/index.js uses require-directory to build the hash (rather than doing so manually) like so:

var requireDirectory = require('require-directory');

module.exports = requireDirectory(module);

app.js references routes/index.js like any other module, but it now has a hash/tree of the exports from the ./routes/ directory:

var routes = require('./routes');

// snip

app.get('/', routes.home);

app.get('/register', routes.auth.register);

app.get('/login', routes.auth.login);

app.get('/logout', routes.auth.logout);

The routes variable above is the equivalent of this:

var routes = {

home: require('routes/home.js'),

auth: {

login: require('routes/auth/login.js'),

logout: require('routes/auth/logout.js'),

register: require('routes/auth/register.js')

}

};

*Note that routes.index will be undefined as you would hope.*

### Specifying Another Directory

You can specify which directory you want to build a tree of (if it isn't the current directory for whatever reason) by passing it as the second parameter. Not specifying the path (requireDirectory(module)) is the equivelant of requireDirectory(module, \_\_dirname):

var requireDirectory = require('require-directory');

module.exports = requireDirectory(module, './some/subdirectory');

For example, in the [example in the Usage section](#usage) we could have avoided creating routes/index.js and instead changed the first lines of app.js to:

var requireDirectory = require('require-directory');

var routes = requireDirectory(module, './routes');

## Options

You can pass an options hash to require-directory as the 2nd parameter (or 3rd if you're passing the path to another directory as the 2nd parameter already). Here are the available options:

### Whitelisting

Whitelisting (either via RegExp or function) allows you to specify that only certain files be loaded.

var requireDirectory = require('require-directory'),

whitelist = /onlyinclude.js$/,

hash = requireDirectory(module, {include: whitelist});

var requireDirectory = require('require-directory'),

check = function(path){

if(/onlyinclude.js$/.test(path)){

return true; // don't include

}else{

return false; // go ahead and include

}

},

hash = requireDirectory(module, {include: check});

### Blacklisting

Blacklisting (either via RegExp or function) allows you to specify that all but certain files should be loaded.

var requireDirectory = require('require-directory'),

blacklist = /dontinclude\.js$/,

hash = requireDirectory(module, {exclude: blacklist});

var requireDirectory = require('require-directory'),

check = function(path){

if(/dontinclude\.js$/.test(path)){

return false; // don't include

}else{

return true; // go ahead and include

}

},

hash = requireDirectory(module, {exclude: check});

### Visiting Objects As They're Loaded

require-directory takes a function as the visit option that will be called for each module that is added to module.exports.

var requireDirectory = require('require-directory'),

visitor = function(obj) {

console.log(obj); // will be called for every module that is loaded

},

hash = requireDirectory(module, {visit: visitor});

The visitor can also transform the objects by returning a value:

var requireDirectory = require('require-directory'),

visitor = function(obj) {

return obj(new Date());

},

hash = requireDirectory(module, {visit: visitor});

### Renaming Keys

var requireDirectory = require('require-directory'),

renamer = function(name) {

return name.toUpperCase();

},

hash = requireDirectory(module, {rename: renamer});

### No Recursion

var requireDirectory = require('require-directory'),

hash = requireDirectory(module, {recurse: false});

## Run Unit Tests

$ npm run lint

$ npm test

## License

[MIT License](http://www.opensource.org/licenses/mit-license.php)

## Author

[Troy Goode](https://github.com/TroyGoode) ([troygoode@gmail.com](mailto:troygoode@gmail.com))