```
const DependencyTree = require("@11ty/
        dependency-tree");

DependencyTree("./my-file.js");
// returns ["./my-local-dependency.js"]

DependencyTree("./my-file.js", {
        nodeModuleNames: "exclude" });
// returns ["./my-local-dependency.js"]

DependencyTree("./my-file.js", {
        nodeModuleNames: "include" });
// returns ["./my-local-dependency.js",
        "@11ty/eleventy"]

DependencyTree("./my-file.js", {
        nodeModuleNames: "only" });
// returns ["@11ty/eleventy"]
```

(Deprecated) `nodeModuleNamesOnly`

(Added in v2.0.0) Changed to use `nodeModuleNames` option instead. Backwards compatibility is maintained automatically.

- `nodeModuleNamesOnly: false` is mapped to `nodeModuleNames: "exclude"`
- `nodeModuleNamesOnly: true` is mapped to `nodeModuleNames: "only"`

If both `nodeModuleNamesOnly` and `nodeModuleNames` are included in options, `nodeModuleNames` takes precedence.

dependency-tree

Returns an unordered array of local paths to dependencies of a CommonJS node JavaScript file (everything it or any of its dependencies `requires`).

- See also: `dependency-tree-esm` for ES Modules.

Reduced feature (faster) alternative to the `dependency-tree` package. This is used by Eleventy to find dependencies of a JavaScript file to watch for changes to re-run Eleventy's build.

Big Huge Caveat

⚠ A big caveat to this plugin is that it will require the file in order to build a dependency tree. So if your module has side effects and you don't want it to execute—do not use this!

Installation

```
npm install --save-dev @11ty/dependency-
tree
```

Features

- Ignores `node_modules`
- Or, use `nodeModuleNames` to control whether or not `node_modules` package names are included (added in v2.0.1)
- Ignores Node's built-ins (e.g. `path`)
- Handles circular dependencies (Node does this too)

Usage

```
// my-file.js

// if my-local-dependency.js has
        dependencies, it will include
        those too
const test = require("./my-local-
        dependency.js");

// ignored, is a built-in
const path = require("path");

const DependencyTree = require("@11ty/
        dependency-tree");

DependencyTree("./my-file.js");
// returns ["./my-local-dependency.js"]
```


`allowNotFound`

```
const DependencyTree = require("@11ty/
        dependency-tree");

DependencyTree("./this-does-not-
        exist.js"); // throws an error

DependencyTree("./this-does-not-
        exist.js", { allowNotFound:
        true });
// returns []
```

`nodeModuleNames`

(Added in v2.0.1) Controls whether or not node package names are included in the list of dependencies.

- `nodeModuleNames: "include"`: included alongside the local JS files.
- `nodeModuleNames: "exclude"` (default): node module package names are excluded.
- `nodeModuleNames: "only"`: only node module package names are returned.

```
// my-file.js:

require("./my-local-dependency.js");
require("@11ty/eleventy");
```