libnpmfund

`libnpmfund` is a Node.js library for retrieving **funding** information for packages installed using `arborist`.

Table of Contents

Example

```
const { read } = require('libnpmfund')

const fundingInfo = await read()
console.log(
  JSON.stringify(fundingInfo, null, 2)
)
// => {
  length: 2,
  name: 'foo',
  version: '1.0.0',
  funding: { url: 'https://example.com' },
  dependencies: {
    bar: {
      version: '1.0.0',
      funding: { url: 'http://collective.example.com' }
    }
  }
}
```

Install

```
$ npm install libnpmfund
```

Contributing

The npm team enthusiastically welcomes contributions and project participation! There's a bunch of things you can do if you want to contribute! The Contributor Guide outlines the process for community interaction and contribution. Please don't hesitate to jump in if you'd like to, or even ask us questions if something isn't clear.

All participants and maintainers in this project are expected to follow the npm Code of Conduct, and just generally be excellent to each other.

Please refer to the Changelog for project history details, too.

Happy hacking!

API

`> fund.read([opts]) -> Promise<Object>`

Reads **funding** info from a npm install and returns a promise for a tree object that only contains packages in which funding info is defined.

Options:

- `countOnly` : Uses the tree-traversal logic from **npm fund** but skips over any obj definition and just returns an obj containing `{ length }` - useful for things such as printing a `6 packages are looking for funding` msg.
- `workspaces` : `Array<String>` List of workspaces names to filter for, the result will only include a subset of the resulting tree that includes only the nodes that are children of the listed workspaces names.
- `path` , `registry` and more Arborist options.

`> fund.readTree(tree, [opts]) -> Promise<Object>`

Reads **funding** info from a given install tree and returns a tree object that only contains packages in which funding info is defined.

- `tree` : An `arborist` tree to be used, e.g:

```
const Arborist = require('@npmcli/arborist')
const { readTree } = require('libnpmfund')

const arb = new Arborist({ path: process.cwd() })
const tree = await arb.loadActual()

return readTree(tree, { countOnly: false })
```

Options:

- `countOnly` : Uses the tree-traversal logic from **npm fund** but skips over any obj definition and just returns an obj containing `{ length }` - useful for things such as printing a `6 packages are looking for funding` msg.

`> fund.normalizeFunding(funding) -> Object`

From a `funding` `<object|string|array>` , retrieves normalized funding objects containing a `url` property.

e.g:

```
normalizeFunding('http://example.com')
// => {
  url: 'http://example.com'
}
```

`> fund.isValidFunding(funding) -> Boolean`

Returns `<true>` if `funding` is a valid funding object, e.g:

```
isValidFunding({ foo: 'not a valid funding obj' })
// => false

isValidFunding('http://example.com')
// => true
```

LICENSE

ISC