libnpmorg

npm v4.0.4 license ISC Node CI failing coverage 100%

`libnpmorg` is a Node.js library for programmatically accessing the npm Org membership API.

Table of Contents

Example

```
const org = require('libnpmorg')

console.log(await org.ls('myorg', {token: 'deadbeef'}))
=>
Roster {
  zkat: 'developer',
  iarna: 'admin',
  isaacs: 'owner'
}
```

Install

`$ npm install libnpmorg`

API

`opts` for `libnpmorg` commands

`libnpmorg` uses `npm-registry-fetch`. All options are passed through directly to that library, so please refer to its own `opts` documentation for options that can be passed in.

A couple of options of note for those in a hurry:

- `opts.token` - can be passed in and will be used as the authentication token for the registry. For other ways to pass in auth details, see the n-r-f docs.
- `opts.otp` - certain operations will require an OTP token to be passed in. If a `libnpmorg` command fails with `err.code === EOTP`, please retry the request with `{otp: <2fa token>}`

`> org.set(org, user, [role], [opts]) -> Promise`

The returned Promise resolves to a Membership Detail object.

The `role` is optional and should be one of `admin`, `owner`, or `developer`. `developer` is the default if no `role` is provided.

`org` and `user` must be scope names for the org name and user name respectively. They can optionally be prefixed with `@`.

See also: `PUT /-/org/:scope/user`

Example

```
await org.set('@myorg', '@myuser', 'admin', {token: 'deadbeef'})
=>
MembershipDetail {
  org: {
    name: 'myorg',
    size: 15
  },
  user: 'myuser',
  role: 'admin'
}
```

`> org.rm(org, user, [opts]) -> Promise`

The Promise resolves to `null` on success.

`org` and `user` must be scope names for the org name and user name respectively. They can optionally be prefixed with `@`.

See also: `DELETE /-/org/:scope/user`

Example

```
await org.rm('myorg', 'myuser', {token: 'deadbeef'})
```

`> org.ls(org, [opts]) -> Promise`

The Promise resolves to a Roster object.

`org` must be a scope name for an org, and can be optionally prefixed with `@`.

See also: `GET /-/org/:scope/user`

Example

```
await org.ls('myorg', {token: 'deadbeef'})
=>
Roster {
  zkat: 'developer',
  iarna: 'admin',
  isaacs: 'owner'
}
```

`> org.ls.stream(org, [opts]) -> Stream`

Returns a stream of entries for a Roster, with each emitted entry in `[key, value]` format.

`org` must be a scope name for an org, and can be optionally prefixed with `@`.

The returned stream is a valid `Symbol.asyncIterator`.

See also: `GET /-/org/:scope/user`

Example

```
for await (let [user, role] of org.ls.stream('myorg', {token: 'deadbeef'})) {
  console.log(`user: ${user} (${role})`)
}
=>
user: zkat (developer)
user: iarna (admin)
user: isaacs (owner)
```