Lerna is dead, now what?

I've used Lerna in the past and it worked really well, but recently it has been retired, as documented in Lerna is largely unmaintained

It appears that someone has taken over as documented in Announcement: Passing the torch.

But even if I were to trust that Nrwl will remain a good steward for Lerna, it remains to be seen if the project remains viable in the long term.

So what are the alternatives?

Alternatives to Lerna

So what's available that would Replace Lerna when working with a monorepo?

Bazel

Bazel is the open source version of Google's Blaze system, their answer to working with a monolith.

Blaze provides a set of rules for building Node-based projects. There are other rule set for Node; still evaluating which one is better.

I documented a Node/Javascript build example in Bazel build system: Typescript and Node.js.

The best thing about Bazel in my opinion is that it can work with multiple languages in the same repository. The Google monolith repo (the one where Blaze, the closed-source version of Blaze was first developed) has code ranging from Javascript and Python to Scala, Go and C++.

My main reservation is still the complexity of the tool itself. Learning Skylark, the Blaze DSL for writing build files, is not trivial and requires a lot of trial and error prone.

But if you're familiar with Bazel it may be a good way to create reproducible builds from your Javascript files along with any other languages.

NPM Workspaces

Workspaces is a generic term that refers to the set of features in the npm cli that provides support to managing multiple packages from your local file system from within a singular, top-level, root package.

This set of features makes up for a much more streamlined workflow handling linked packages from the local file system. Automating the linking process as part of npm install and avoiding manually having to use npm link in order to add references to packages that should be symlinked into the current `node_modules` folder.

You may automate the required steps to define a new workspace using npm init. For example in a project with the following structure.

```
.
+-- package.json
`-- packages
   +-- a
   |   `-- package.json
   `-- b
       `-- package.json
```

If you want to add a dependency named `abbrev` from the registry as a dependency of your workspace a, you may use the workspace config to tell the npm installer that package should be added as a dependency of the provided workspace:

```
npm init -w ./packages/a
```

This command will create the missing folders and a new `package.json` file (if needed) while also making sure to properly configure the “workspaces” property of your root project `package.json`.

This would make it possible to manage multiple packages in the monorepo using our standard tools.

NX

NX

Bits

Bits

Links and Resources

- https://medium.com/autodesk-tlv/lerna-has-gone-which-monorepo-is-right-for-a-node-js-backend-fc075cad51b0
- https://nx.dev/migration/adding-to-monorepo