Integrating npm into the Guix ecosystem

Jelle Licht, jlicht@fsfe.org

21-03-2016

1 Overview

This project will allow Guix hackers to more easily package software that is distributed through the Node Package Manager (npm), as well as allowing Node developers on Guix to make use of the reproducible builds guarantee of Guix.

After completing this project, it should be possible to easily make use of the less-problematic packages in the npm registry on the Guix Software Distribution.

2 Project structure

Depending on findings in the early stages of the project, I foresee the following distinct parts:

1. Extend Guix so it can 'simulate' the dependency graph generation of both the old and new npm [1].
2. Extend guix with an algorithm that matches npm's package.json flexible version specification to a specific version.
3. Add a `guix import` backend for the npm registry
4. Package npm modules in guix
5. Interface

Right now, dependency resolution in npm is as stateful as can be, with even the installation order [1] making a difference for where npm expects to

[1] `https://docs.npmjs.com/how-npm-works/npm3`

find a certain dependency. As two different dependency resolution mechanism are in use, of which especially the newer one is problematic because of its habit of propagating dependencies upwards in the folder structure, both have to be supported by a guix module.

npm uses SemanticVersioning range patterns [2] to declare dependencies between packages. A problem with this approach is that the same package declaration can lead to an entirely different dependency graph, which defeats the purpose of having a system with a focus on reproducible builds. If npm packages are to be used, these version numbers need to be locked down to a specific version, corresponding to the version that npm would install if left to its devices. A potential problem is that the entire dependency graph has to be known ahead of time in order to pinpoint a 'correct' version of the dependency. A consistent ordering for 'installing' dependencies also has to be decided upon.

After the dependency resolution has been worked out, it should be possible to create a guix import backend to leverage the code that has been produced up till now to allow the packaging of npm modules. The last part of the project essentially serves as a starting point for packaging up useful npm packages.

A stretch goal for the summer would be to create a guix build system for a subset of npm packages, such as gulp.

3 Planning

As I am currently a novice with regards to the internals of guix and the guile programming language, up to the start of the actual project I will mostly be reading up and hacking on guix. As such, the planning becomes:

April 22 - May 22:

- Getting to know Guix(SD)
- Package programs using the guix import module
- An informal specification of the npm dependency resolution mechanism
- Getting the know the guix community and what everyone is working on
- Getting familiar with the contributing work flows

[2] `http://developer.telerik.com/featured/mystical-magical-semver-ranges-used-npm-bower/`

May 23 - Jun 5:

- Formal specification to allow guix to simulate npm dependency resolution (1)

Jun 6 - Jun 19:

- npm Version pinning should be working (2)
- start working on the guix import backend

Jun 20 - Jul 3:

- Guix import backend should be finished by now (3)
- Start testing npm packages

Jul 4 - Jul 10:

- Holidays!

Jul 11 - Aug 7:

- Solve any problems and corner cases with building and installing npm packages (4)

Aug 8 - Aug 23:

- If life goes a planned, Get all contributions ready to be merged back in the main Guix codebase.

4 About me

My name is Jelle Licht, and I am currently studying Data Science at the University of Technology Delft. I finished my BSc in Computer Science in 2015, at the University of Technology Delft as well. Somewhere in the second semester of my studies, my roommate was using Vi in front of me, and since then I've been falling into the rabbit hole that is GNU/Linux, free software and the 'open source' community. Somewhere along the line I became a huge fan of Clojure, and for a short while was a member of the very much unofficial Lisp Community Delft.

I have been a small-time contributor to some free software projects on github, as well as having a day job to make ends meet using mostly Node. My online handles include 'wordempire', as I am quite fond of reading, and 'jlicht'. For the past month, I have been lurking on-and-off again in the #guix irc channel, as well as reading up on some of the motivations behind reproducible research and reproducible builds.

Besides one week for visiting family and general holidays, this project would be my full time focus during the summer.