A System for JavaScript Package Health and Usability Overview

IMT 542 Project Presentation

Xinshuo Lei

Image source: https://www.freepik.com/

Information Story: The Risks of External Packages

- In web development, engineers often rely on external packages to avoid building everything from scratch

- **The problem: not all packages are reliable**

- If developers aren't careful when choosing packages, they may end up building their project on top of:
  - **Abandoned packages** that no one maintains anymore
  - **Outdated packages** with unresolved security issues
  - **Unstable packages** that break when updates are released

Real Impact of Unreliable Packages

r/programming · 9 yr. ago
_ar7

An 11 line npm package called left-pad with only 10 stars on github was unpublished...it broke some of the most important packages on all of npm.

left-pad/left-pad

#4 npmjs.org tells me that left-pad is not available (404 page)

193 comments

silkentrance opened on March 22, 2016

github.com

Open

What happens to abandoned packages? #82

Closed | Answered by MylesBorins | joshbressers asked this question in Registry

joshbressers on Nov 3, 2020

Hi all,

The package trim has a recently reported security vulnerability
https://nvd.nist.gov/vuln/detail/CVE-2020-7753

It's fairly widely used (3.5M weekly downloads) and appears to be an abandoned package as it's not seen any commits in over 8 years.

There is a PR to fix this problem from a community member that I verified works as expected. There is an open PR that's been open since 2016, the last PR was closed in 2014.

In this particular instance the security vulnerability isn't critical, I don't think any rushed decisions need to be made.

Does npm have an unresponsive maintainer policy or process?

An example of prior art here is Fedora's Non-responsive maintainer policy
https://docs.fedoraproject.org/en-US/fesco/Policy_for_nonresponsive_package_maintainers/

↑ 3

Projects can suddenly break

— even when developers haven't changed a single line of their own code

Broken Packages Lead to Broken Trust

Here's some advice for every junior out there, when you go implement your own solution and some nerd goes and tell you "duh, you so dumb why reinvent the wheel", don't listen to them; that code will most likely be simpler and have less to no dependencies and do exactly what you need it to do; don't make the same mistake I did, every single library I installed to do something that I could've done myself has backfired, every single one of them, no exceptions. The closer to the basics you are the better, otherwise node_modules will suck you in, your life, hopes and dreams.

– u/boisheep on r/webdev

https://www.reddit.com/r/webdev/comments/1ake0hd/rant_javascript_dependency_hell_is_worse_than_i/

Developers are forced to act like detectives

— piecing together scattered clues just to understand a package's **health and usability**

LemonAncient1950 • 1y ago

Yeah it can get gross. The ESM/Commonjs conflicts really grind my gears. I've become much more hesitant to grab external dependencies. I start by checking open issues, how recently the project was updated, and how many contributors it has.

121 Reply Award Share ...

Solving Node.js Security

npm packages are amazing, and let us build software at an unprecedented pace. You should definitely keep using npm packages – but there's no reason to do so blindly. We covered 7 questions you can easily answer to understand better, and reduce your security exposure:

1. Which packages am I using? And for each one...
2. Am I still using this package?
3. Are other developers using this package?
4. Am I using the latest version of this package?
5. When was this package last updated?
6. How many maintainers do these packages have?
7. Does this package have known security vulnerabilities?

Controlling the Node.js Security Risk of NPM Dependencies

By Ferenc Hámori (2024)

Existing Information Structure: Scattered Health and Usability Signals

Source	Available Package Information	Issue
npm (npmjs.com)	Basic information like description, downloads, version, maintainers, etc.	Lacks information on whether the package is actively maintained (e.g., repository activity)
GitHub	Code, issues, stars, commit history	Users need to navigate through a lot of information to extract health and usability signals
libaries.io	Dependency stats, repository metadata, and version history	Only lists raw stats without helping users interpret package usability or overall developer experience

Requirements for a New Information Structure: Centralized Health and Usability Signals

- **Aggregates signals across sources**
- **Readily accessible**
- **Provides high-level assessments of package health and usability, backed by raw metrics**
  - **Community Adoption**: whether the package is widely trusted and actively used in projects
  - **Maintenance Frequency**: how often the package is updated and how recently it was maintained
  - **Release Management**: how actively and responsibly the package is being released
  - **Implementation Footprint**: how much size and complexity the package adds to a project
  - **Documentation Completeness**: whether the package includes clear instructions and documentations

Improved Structure


```
{
    "package_name": "echarts",
    "retrieved_at": "2025-06-05T18:04:13.794214Z",
    "npm_data": {
        "name": "echarts",
        "latest_version": "5.6.0",
        "description": "Apache ECharts is a powerful, interactive charting and data visualization library for browser",
        "license": "Apache-2.0",
        "homepage": "https://echarts.apache.org",
        "bundle_size": "53.2 MB",
        "dependencies_count": 2,
        "repository": "https://github.com/apache/echarts",
        "npm_url": "https://www.npmjs.com/package/echarts",
        "last_release": "2024-12-28T07:21:42.839Z",
        "days_since_last_release": 159,
        "releases_last_year": 4,
        "maintainers_count": 9
    },
    "downloads_data": {
        "monthly_downloads": 4376169,
        "weekly_trend": [
            {
                "start": "2025-01-17",
                "end": "2025-01-23",
                "downloads": 983094
            },
            {
                "start": "2025-01-24",
                "end": "2025-01-30",
                "downloads": 953135
            },
            {
                "start": "2025-01-31",
                "end": "2025-02-06",
                "downloads": 995077
            },
```

```
    "github_data": {
        "repo": {
            "stars": 63685,
            "forks": 19736,
            "last_code_push": "2025-06-05T12:32:10Z",
            "is_archived": false,
            "is_maintained": true
        },
        "health": {
            "health_percentage": 87,
            "has_readme": true,
            "has_license": true,
            "has_contributing": true,
            "has_code_of_conduct": true
        },
        "activity": {
            "open_issues_count": 1939,
            "closed_issues_count": 17019,
            "total_issues_count": 18958,
            "last_pr_merged_at": "2025-06-03T12:10:00Z",
            "last_pr_info": "5 hours before merge",
            "last_pr_url": "https://github.com/apache/echarts/pull/21011"
        }
    },
    "health_ratings": {
        "community_adoption": "Strong",
        "maintenance_frequency": "Regular",
        "release_management": "Infrequent",
        "implementation_footprint": "Lightweight",
        "documentation_completeness": "Thorough"
    },
    "success": true,
    "errors": []
}
```

Improved Structure: FAIR Assessment

- **Findable**
  - Each package is uniquely identified by its name and version
- **Accessible**
  - Human-readable scores and metrics
  - Metadata can be retrieved via a documented REST API in machine-readable JSON format
- **Interoperable**
  - Shared vocabularies from source APIs are used
  - Data formats align with standard developer conventions
- **Reusable**
  - Records include provenance (source, fetch time)
  - Endpoints are documented for downstream use

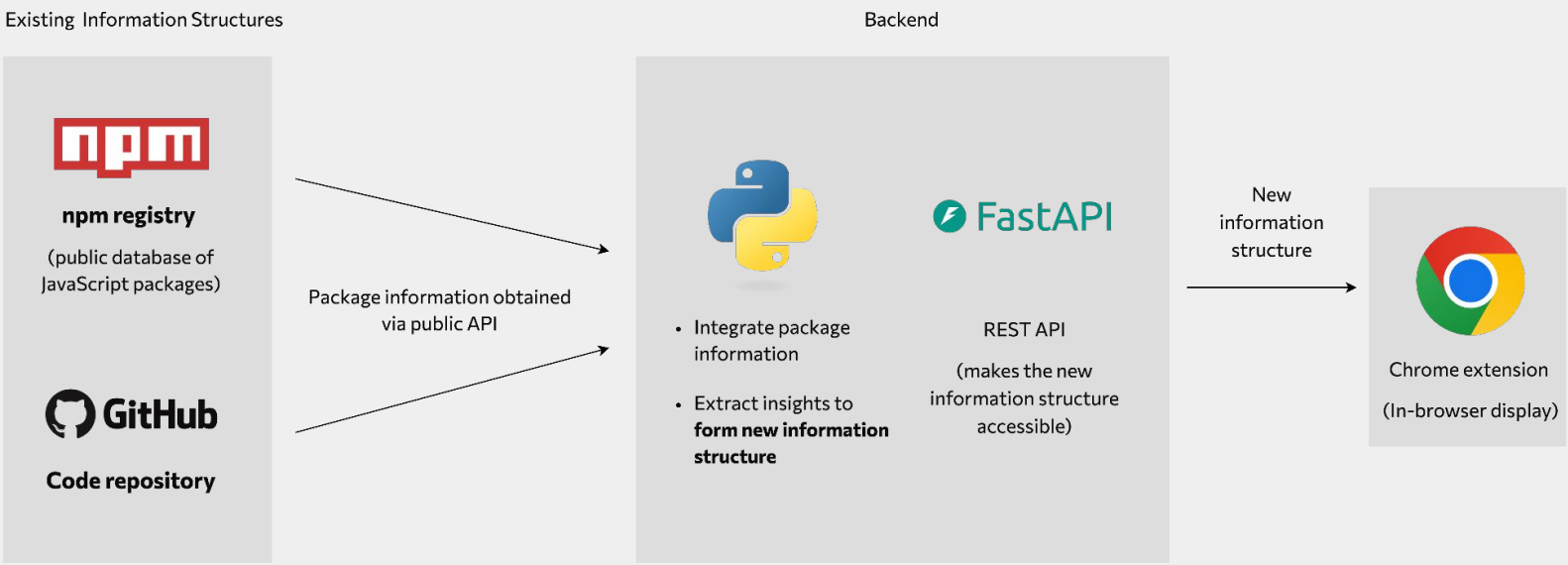
Demo 1: Quick Package Search with Keyboard Shortcut

- https://drive.google.com/file/d/1jIGkkWMrje6W4vvLZ57mp58R_TWgGu6t/view?usp=drive_link

Demo 2: Manual Search

- https://drive.google.com/file/d/1TlMfa-atrZcUlbYrbe4JJjTO04aCE3K9/view?usp=sharing

Making the Improved Structure Readily Accessible



Quality and Performance

- **Functional Tests**
  - **Data Accuracy**: scores should reflect the most current and correct package data ✅
  - **Data Completeness**: each queried package should include all essential metadata ✅
  - **Error Handling**: the system continue functioning even if some data sources fail or return incomplete information ✅
  - **Data Consistency**: results for a given package should remain consistent across requests (assuming no changes in the source data) ✅
- **Performance Tests**
  - **Efficiency**: responses should be returned quickly enough to support real-time use in a browser extension ✅
  - **Scalability**: support multiple package queries in parallel without significant delay ⏳