An analysis of Java release practices on GitHub

Vivian Roest

Supervisor: Sebastian Proksch

EEMCS, Delft University of Technology, The Netherlands

Introduction & Background

- Github is the most popular online code hosting platform
- Most prior research on Java practices utilizes Maven Central [1, 2, 3, 4, 5, 6].
- Existing literature scraping Github for Java repositories is sparse [7, 8]
- Furthermore, how does GitHub packages (and the like), compare to Maven Central?

Research Questions

"What are the Maven release practices on GitHub?"

1. Can we make a dataset of Java repositories on GitHub?
2. Are these projects released, and where are they released?
3. What is their use of external repositories?
4. How is authentication for releasing packages to distribution repositories realized?

Method

We have created a four-stage pipeline for gathering Java repositories from GitHub.

1. Use GitHub API to request the list of all repositories, and filter on Java repositories;
2. Download all `POM.xml` files contained within those repositories;
3. Use *Maven* to create an 'effective pom' of all the pom files (when possible) to get the full data;
4. Analyse the resulting `POM.xml` for the use of external and distribution repositories.

Results

These are the results of a random sample of 500 000 from the 15.5 million Java repositories on GitHub.

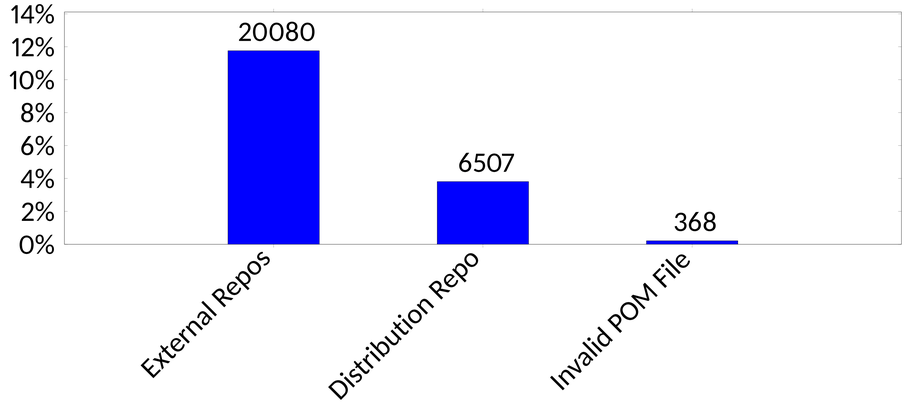

Figure: The percentage of repositories that have external repos, distribution repos or invalid POM files out of a total of 170 798. The y-axis shows percentage, the numbers on the bars are the absolute number of repositories in each category.

The source code and dataset can be found on the 4TU Research Data repository, DOI: 10.4121/67a790fe-b65a-4c30-aae0-c5b2dc7e5d4d

Most used External Repositories

#n	distinct	url
2362	15	`oss.sonatype.org`
393	2	`repository.apache.org`
245	238	`maven.pkg.github.com`
205	5	`s01.oss.sonatype.org`
106	105	`api.bintray.com`
105	3	`repo.spring.io`
105	3	`repo.jenkins-ci.org`
103	2	`repository.jboss.org`
35	2	`maven.wso2.org`

Most used External Repositories

#n	distinct	url
9488	54	`repo.spring.io`
3193	71	`oss.sonatype.org`
1250	4	`maven.aliyun.com`
1064	28	`repo1.maven.org`
1033	11	`hub.spigotmc.org`
1009	135	`dl.bintray.com`
908	29	`repository.apache.org`
866	5	`repository.jboss.org`
763	3	`jitpack.io`

Discussion

Looking closer at the numbers we obtained we see that Maven Central is not as central as it might have used to be. With quite a few uses of different package repositories, both for publishing and consuming. Specifically GitHub packages is an interesting case, being so popular for publishing but not to depend on. Only 89 Java repositories use GitHub packages as an external repository. The main reason is speculated to be the issue of authentication.

Recommendations

- Developers should think twice about releasing on GitHub packages, but not discount it completely.
- Both GitHub and Maven are frustrating the current situation
- Ideal solution would be adapting Maven to not be overly reliant on IDs and allow authentication per hostname.
- Intermediary solution can include:
  - A proxy that adds authentication on the fly
  - Embedding authentication tokens inside the repository URLs (as was inspired by some real world examples).

References

[1] Steven Raemaekers, Arie van Deursen, and Joost Visser.
Semantic versioning versus breaking changes: A study of the maven repository.
In *2014 IEEE 14th International Working Conference on Source Code Analysis and Manipulation*, pages 215–224, 2014.

[2] Lina Ochoa, Thomas Degueule, Jean-Rémy Falleri, and Jurgen Vinju.
Breaking bad? semantic versioning and impact of breaking changes in maven central: An external and differentiated replication study.
Empirical Software Engineering, 27(3):61, 2022.

[3] Raula Gaikovina Kula, Daniel M German, Takashi Ishio, and Katsuro Inoue.
Trusting a library: A study of the latency to adopt the latest maven release.
In *2015 IEEE 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER)*, pages 520–524. IEEE, 2015.

[4] Amine Benelallam, Nicolas Harrand, César Soto-Valero, Benoit Baudry, and Olivier Barais.
The maven dependency graph: a temporal graph-based representation of maven central.
In *2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)*, pages 344–348. IEEE, 2019.

[5] César Soto-Valero, Nicolas Harrand, Martin Monperrus, and Benoit Baudry.
A comprehensive study of bloated dependencies in the maven ecosystem.
Empirical Software Engineering, 26(3):45, 2021.

[6] Dimitris Mitropoulos, Vassilios Karakoidas, Panos Louridas, Georgios Gousios, and Diomidis Spinellis.
The bug catalog of the maven ecosystem.
In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 372–375, 2014.

[7] Thomas Durieux, César Soto-Valero, and Benoit Baudry.
Duets: A dataset of reproducible pairs of java library-clients.
In *2021 IEEE/ACM 18th International Conference on Mining Software Repositories (MSR)*, pages 545–549, 2021.

[8] Phuong T Nguyen, Juri Di Rocco, Davide Di Ruscio, Lina Ochoa, Thomas Degueule, and Massimiliano Di Penta.
Focus: A recommender system for mining api function calls and usage patterns.
In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*, pages 1050–1060. IEEE, 2019.