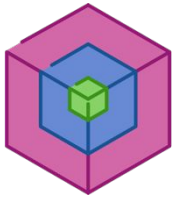


Apache Fineract ®

Community Meeting

October 8, 2025

James Dailey, PMC member
(and current chair)

A top level project (TLP) of the Apache Software Foundation (ASF)

Agenda

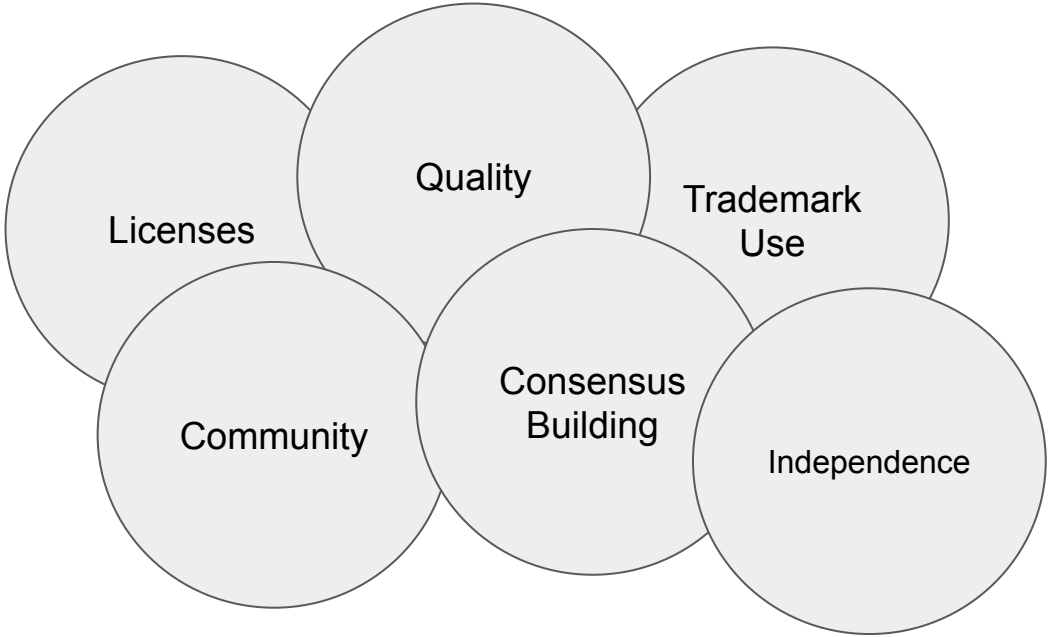
- ASF Maturity Model as frame
- Highlight a few Survey Results
- Getting a release out and thank you Release Manager
- Community Discussion and Channels
- Quality, Security and Roadmap
- Community Over Code (Sept 2025 in Minneapolis, June in Europe TDB)

Meeting notes will be collated here ⇒

https://cwiki.apache.org/confluence/display/FINERACT/Meeting+notes

ASF Project Maturity Model

https://community.apache.org/apache-way/apache-project-maturity-model.htm



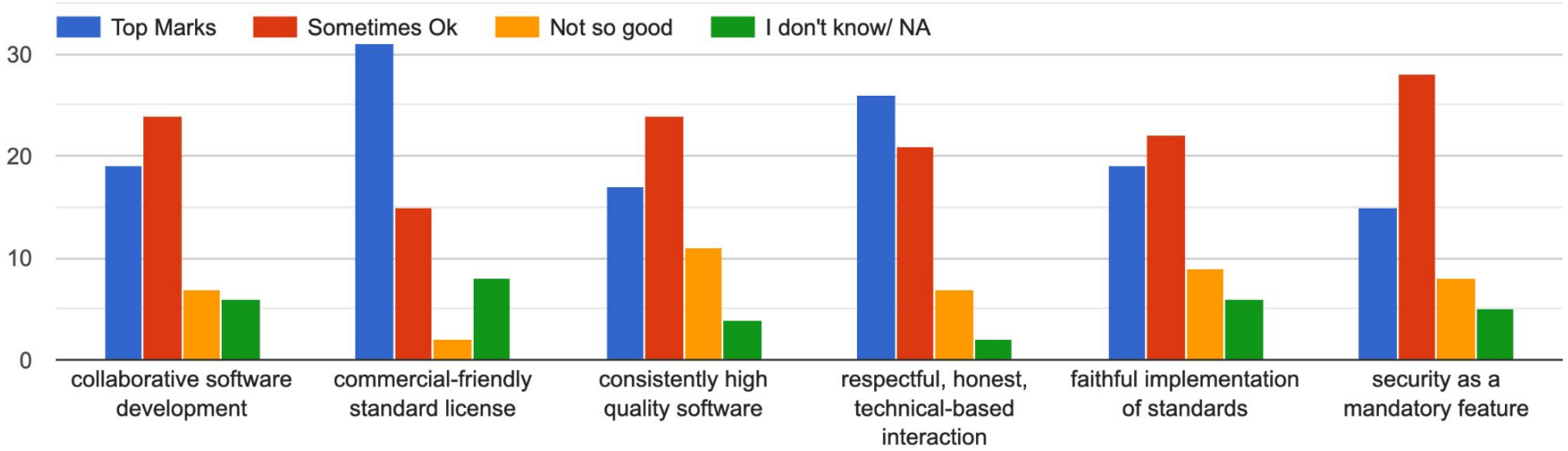
Remember: This model is a guide; it is not a requirements document. The model shows what generally good behaviors in an Apache project look like.

Highlighting Survey Results

Surveying is part of a good community feedback loop.

Maturity Model

How are we (as a community) doing with these Apache Principles ?

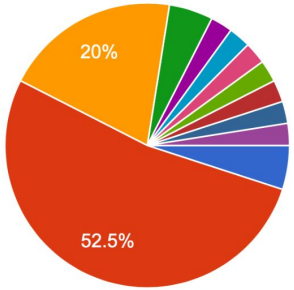


Same question over time…

How did you FIRST learn about the fineract apache project? (pick one)

40 responses

2019

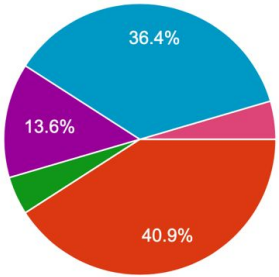


- From an email, article, other posting
- From Mifos
- From a friend
- From Apache, apacheCon, or other sp…
- Fineract was proposed as the core ba…
- the company which hired me for this p…
- googled my problem
- google

How did you FIRST learn about the fineract apache project? (pick one)

22 responses

2022



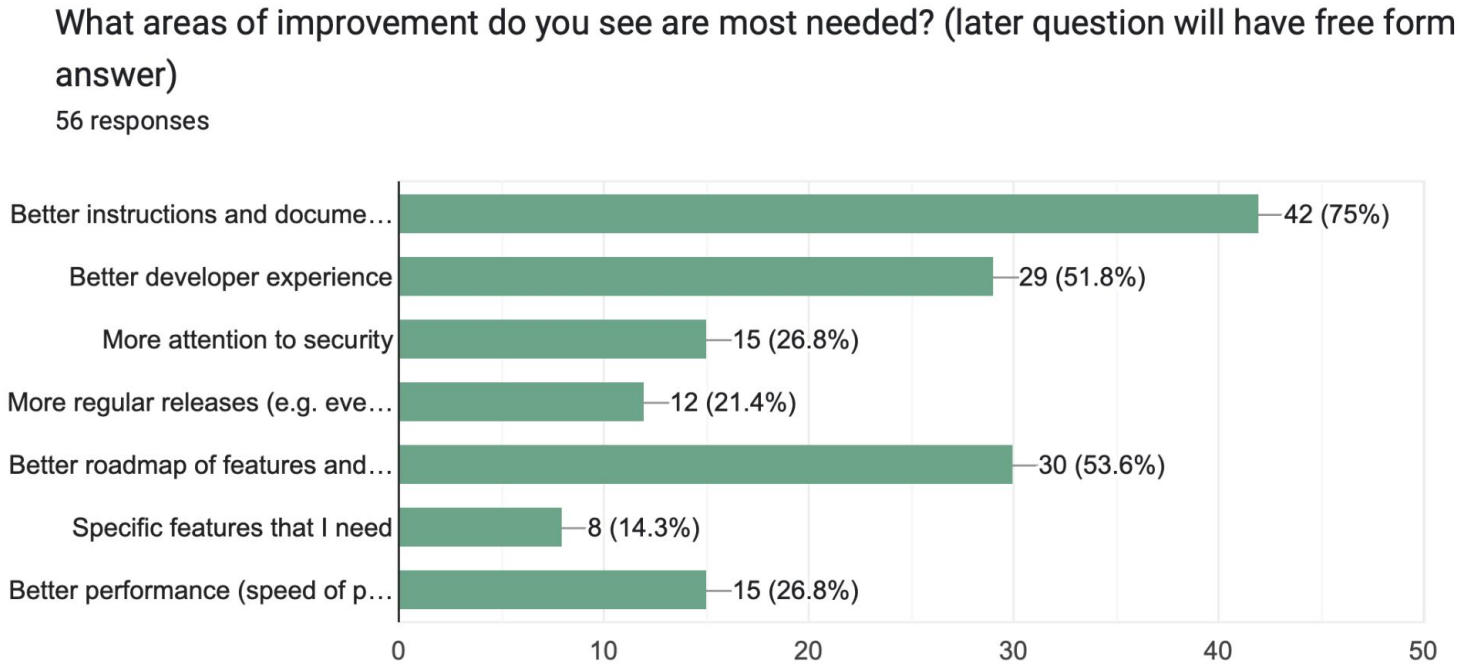
- From an email, article, other posting
- From Mifos
- From a friend, associate, industry connection
- From Apache, apacheCon, or other sponsored Apache event
- From Workplace / required for job
- Googled my problem, found fineract
- As early as 2008

2025 ---same question

How did you FIRST learn about the Apache Fineract(R) project? (pick one)

56 responses

Survey questions



https://cwiki.apache.org/confluence/display/FINERACT/Survey+Results+2025+August

Releases

https://community.apache.org/apache-way/apache-project-maturity-model.html

Releasing Software at ASF: Maturity model

Releases

RE10
Releases consist of source code, distributed using standard and open archive formats that are expected to stay readable in the long term. 6

RE20
The project's PMC (Project Management Committee, see CS10) approves each software release in order to make the release an act of the Foundation.

RE30
Releases are signed and/or distributed along with digests that anyone can reliably use to validate the downloaded archives.

RE40
The project can distribute convenience binaries alongside source code, but they are not Apache Releases, they are provided with no guarantee.

RE50
The project documents a repeatable release process so that *someone new to the project* can independently generate the complete set of artifacts required for a release.

?

Also… it is expected that a project shall be able to do a new release on a regular basis, and required that it resolve open security issues with patches or releases.

https://www.apache.org/legal/release-policy.html

Getting a release out today

1. Release manager announces upcoming release on list {REQUIRED CEREMONY}
   a. Requires a willing Release Manager
   b. Open a Jira ticket for the release - to ensure tracking of open issues for the release
2. Clean up Jira tickets (validate only those closed between releases)
3. Create release branch
   a. Challenge of choosing an appropriate stable commit
   b. Cherry picking commits
4. Freeze Jira for that release // Jira admin process
5. Create release tag
6. Create a Distribution {REQUIRED}
7. Sign Distribution {REQUIRED} // local machine only?
8. Upload Distribution Artifacts to SVN Staging {REQUIRED}
9. Verify Distribution at Staging {REQUIRED}
10. Call for Vote on Mailing List {REQUIRED CEREMONY}
    a. Expectation is that votes include verification of code and signatures
    b. Three PMC members must approve at a minimum
    c. 72 hour minimum
11. Finish Vote (record result) {REQUIRED CEREMONY}
12. Promote to Production within SVN {REQUIRED}

17 days

https://fineract.apache.org/docs/current/#_release_process

Release process

Goals

- Create improvements to process
- Automate
- Quarterly (at least)
- Monthly ?
- Longer term: Roadmap and release

Grails release process (mostly) automated
-->.
https://github.com/apache/grails-core/blob/7.0.x/RELEASE.md

Patch releases for security or bugs

Security Fixes are sometimes not backwards “portable”

CVEs fixed

Requires a new release

Support two releases is the *standard*

But…

So… exceptions to this are required.

Community is informed.

WHY?

Ruleset: No public release with unfixed CVEs are allowed.
 -- Or --

Only fixed versions allowed.

Community Discussion and Channels

ASF Infra - the Archive: Where key discussions belong

Listserv : Discussions elsewhere brought back here, proposals, some user queries, VOTES (the official record)

Jira Tickets: For issues to be described in detail

Github comments: Technical back and forth

Slack at Apache

New solutions at Apache

“If it didn’t happen on list, it didn’t happen.” - commonly cited apache motto

Slack at outside entity (not archived by ASF)

How to bring discussions back…

Examples:

“Some of us were at a conference and discussed ideas about building a better release process, so I want to share our observations and pointer to other projects doing the same…”

“At my work, a few of us working on Fineract observed a bunch of slow tests, and looked into a recent change that was committed… I think this needs to be corrected or reverted… so I have created a ticket for that.”

“On slack, Peter and I have discussed creating a new process for managing the cherry picking process, so I am bringing that back here.”

“Hey Devs - I have a new idea and I think it will solve xyz problem….” (based on a slack discussion which is not shared explicitly, which is ok too)

Quality, Security, and the Roadmap Discussion

Setting our north star

Currently as listed with ASF Board:

The mission of Apache Fineract is the creation and maintenance of software related to a **core banking platform** that provides a reliable, robust, and affordable solution for entrepreneurs, financial institutions, and service providers to offer financial services to the world's underbanked and unbanked.

> Apache Fineract® (\'fīn-,ə-,rakt\) is open source software for financial services, designed to create a **cloud-ready core banking system** that enables **digital financial services for everyone**, including the unbanked and underbanked.

Revise or keep?

What is this project trying to accomplish?

A modular system

Performant and scalable

That serves as a backend system

To fintechs, community banks, microfinance operations, and any financial entity

With core functionality for lending, transactions, and customer records

Such that complex term loans can be configured, stored and managed

And such that additional complexity and integration can occur outside of the core.

A “Bank in a Box” for the smallest orgs and a Digital Core Banking Platform for those that need a lot more.

What else?

https://community.apache.org/apache-way/apache-project-maturity-model.html

Quality

Quality

QU10
The project is open and honest about the quality of its code. Various levels of quality and maturity for various modules are natural and acceptable as long as they are clearly communicated.

QU20
The project puts a very high priority on producing secure software. 7

QU30
The project provides a well-documented, secure and private channel to report security issues, along with a documented way of responding to them. 8

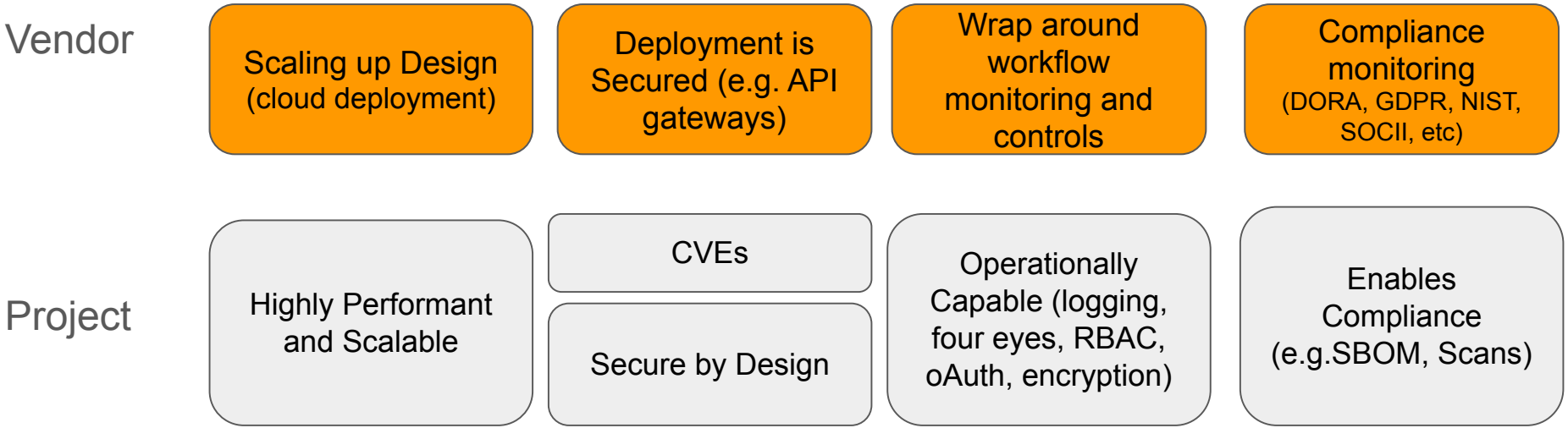
QU40
The project puts a high priority on backwards compatibility and aims to document any incompatible changes and provide tools and documentation to help users transition to new features.

QU50
The project strives to respond to documented bug reports in a timely manner.

Banking Software Standards

? 

Foundational project needs *and* vendor role



Goal

A financial backend system must be *performant*, *scalable*, *secure*, *operationally* capable and *compliant*.

QU20: We place a high value on Security

CVEs - fixing and releasing (emails and wiki)

Dependencies (dependabot)

Auto scanning tools

Advice on how to deploy (very high level)

How are we doing?

Securing Fineract

This section covers best practices in securing the use of Fineract.

Security hardening is a continuum and Fineract is adaptable to your security needs. There's no one way to correctly deploy it and the open source project offers no warranty. It's up to you to deploy and maintain it carefully, according to your organization's needs and compliance requirements.

See https://fineract.apache.org/docs/current/#_securing_fineract

SBOM (Software Bill of Materials)

https://issues.apache.org/jira/browse/FINERACT-2164

The CycloneDX Gradle plugin creates an aggregate of all direct and transitive dependencies of a project and creates a valid CycloneDX SBOM. CycloneDX is a lightweight software bill of materials (SBOM) specification designed for use in application security contexts and supply chain component analysis.

i.e .

Install the plug in

```
# Generate per-project SBOMs

./gradlew cyclonedxDirectBom
```

Software Bill of Materials (SBOM) A "software bill of materials" (SBOM) has emerged as a key building block in software security and software supply chain risk management. An SBOM is a nested inventory, a list of ingredients that make up software components.
https://www.cisa.gov/sbom

SBOM Guidance, Sept 2025
https://www.cisa.gov/sites/default/files/2025-09/joint-guidance-a-shared-vision-of-software-bill-of-materials-for-cybersecurity_508c.pdf

Roadmap Mechanism

Fineract Significant Improvement Proposals

https://cwiki.apache.org/confluence/display/FINERACT/Fineract+Significant+Improvement+Proposals

FSIP 1	Modular Security	Implemented
FSIP 2	Scarf Data Tracking	Implemented
FSIP 3	Interest bearing loans in progressing lending module	Implemented / in process to add more
FSIP 4	Cucumber Testing	Implemented / more tests needed
FSIP 5	(new) Command processing	In process / status update?

Gaps (things we need)

1. Better participation
2. Security fixes done timely
3. Documentation of functionality
4. Easy to set up for devs
5. Test coverage (broadly)
6. Easy to work with code and tests
7. Modularity of code
8. Included front end UI of project
9. Regularity of releases
10. Trimming of functionality
11. Completed refactoring
12. Closing of old jira tickets
13. Removing unused dependencies
14. Automated SBOM

We might assign different values to these in terms of priority.

This is my force ranking.

- James

Progress: Things that are helping…

Readme file update

Docker Scout and other scanning tools → fix dependencies and test

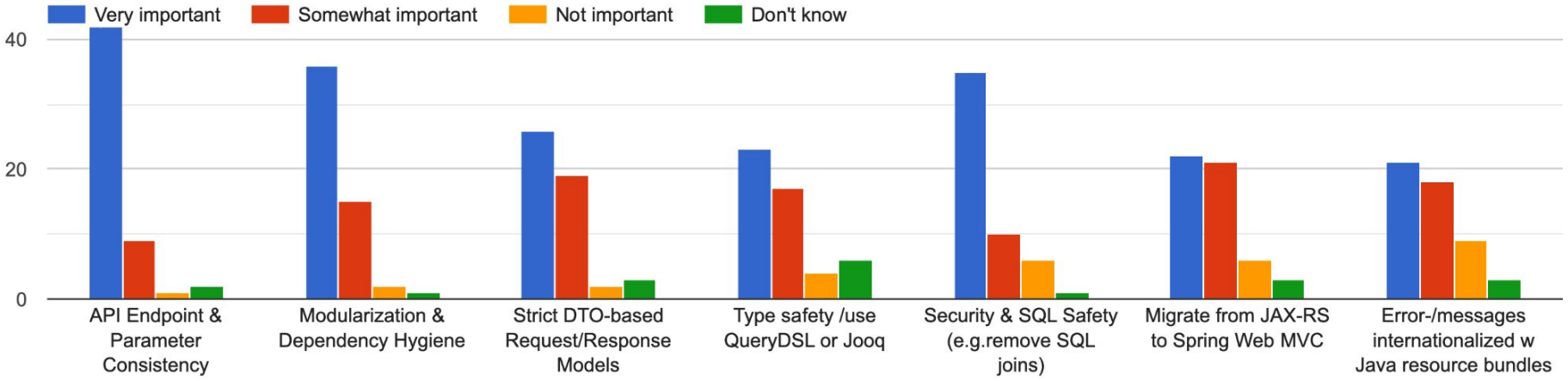
Sharding of tests (to speed them up)

Automation of the build process

Auto generating swagger files ?

Some priorities from the survey

Which of the following areas should the project work on? (see listserv archive for more info...)



Fineract NextGen - more ideas

Modularity & Refactoring / break backward compatibility ?

UUID , no more DB increment

Remove MariaDB and mySQL

WSO2

Co-lending functionality

Revolving line of credit

Scalable current account

Treasury management

Track: Securing Fineract and Fintechs

Evolving Fineract

Securing Fineract

Zero Trust Trinity for workflows

Confidential Computing for Fineract

What Fineract gets right on security

Open collaboration for security fineract

https://communityovercode.org/schedule/

Next Community Over Code conference will be in Europe next summer.