Ship It or Skip It?

When & How to Upgrade Your Open Source Dependencies

Bloomberg

Engineering

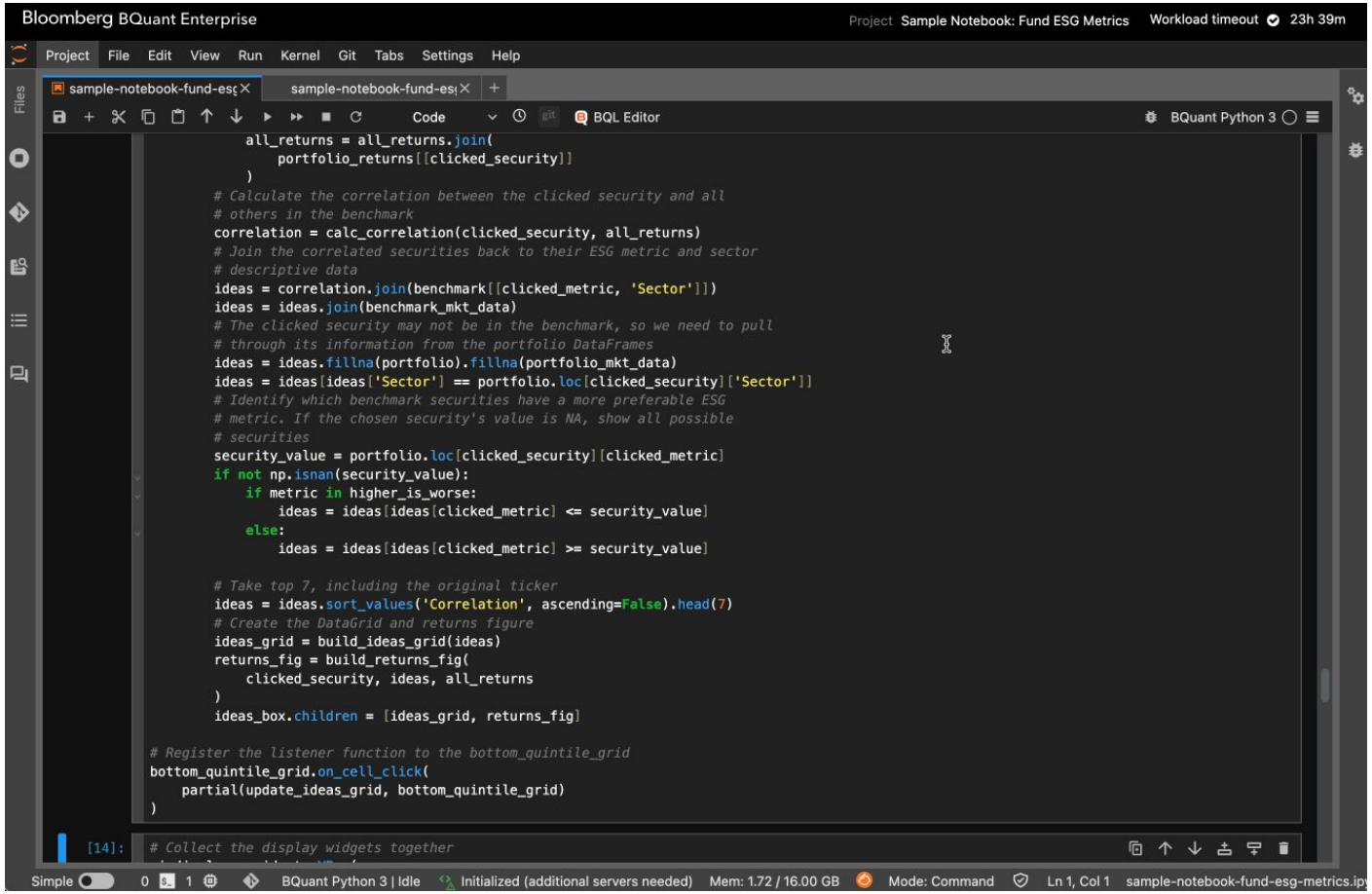
JupyterCon 2025
November 5, 2025

Balaji Sundaram, Shruti Sapre, and Rebecca Ely
Software Engineers, BQuant IDE Tools

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Our upgrade experience



JupyterLab 3.0 ⇒ 3.2 ⇒ 3.6

Voila 0.3 ⇒ 0.4

Jupyter Widgets 7 ⇒ 8

Jupyter Server 1 ⇒ 2

JupyterLab 3 ⇒ 4

Voila 0.4 ⇒ 0.5

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Bloomberg

Engineering

Why do we wait to upgrade?

- 🐌 Few changes, no benefit
- 🤷🏽 Scope uncertainty
- 🚩 Known issues needing fixes
- ☣️ High risk, consequences for releasing breaking changes
- 🪸 Ecosystem catch up
- ⌛ No time
- 👎🏾 No organizational buy-in

© 2025 Bloomberg Finance L.P. All rights reserved.

Why should we upgrade sooner?

- 🤩 Shiny new features
- 🕸️ Bug fixes
- 🔒 Security
- 🚧 Unblock other upgrades
- 😎 No end-of-life scramble
- ❤️ Reduce burden on maintainers
- ➗ Frequent upgrades = less risk per upgrade

© 2025 Bloomberg Finance L.P. All rights reserved.

Defining scope

🔗 Which of your repos depend on the library you're upgrading? ↳⛓️ Which of your repos depend on these repos?
🗑️ Does your repo still need this dependency? ↳♻️ Does your product still need this code?
⚒️ What backwards-incompatible API & UI changes could affect your codebase?
😈 What code sins will be broken by changes to upstream internals?
🚧 What other upgrades are needed to unblock this one?

© 2025 Bloomberg Finance L.P. All rights reserved.

Identifying stakeholders

👩🏿‍🍳 Who owns the affected repos?
📣 Who will communicate changes to users?
📋 Who will test the upgrade before release?
👑 Who has ultimate decision-making authority?
🔥 How will you prevent upgrade burnout?

© 2025 Bloomberg Finance L.P. All rights reserved.

Communicate early and often

- 📅 Notify stakeholders early
- 👥 Host kickoff meetings
- ✅ Assign owners & tasks
- 📋 Shared checklists/docs to track progress
- 📣 Regular status updates

© 2025 Bloomberg Finance L.P. All rights reserved.



Shrink your scope

- ♻️ Don’t upgrade what you can delete
  - ⚰️ Dead code
  - 👻 Unused dependencies, features, & libraries
  - 🎁 Features contributed upstream
- 🩹 Clean up hacky workarounds
- 🎯 Resist scope creep

© 2025 Bloomberg Finance L.P. All rights reserved.

Utilize upstream resources

🗺️ Migration guides

🏷️ Release notes

🕵🏻 PR/Issue searches

❓ Ask questions in GitHub issues, Discourse, or Zulip

🐘 As you discover them, compile a list of:

- ⚠️ Known changes that will break your users' workflows
- ♟️ Strategies to get around them

© 2025 Bloomberg Finance L.P. All rights reserved.

Beyond unit tests

✅ Tests pass, but workflows still break

👤 Real-world behavior needs real-world testing

🖥️ Test across kernels and environments

👁️ Watch out for UI regressions

🔘 New features may be disabled by default

© 2025 Bloomberg Finance L.P. All rights reserved.

Expect curveballs

⭕ Avoid releasing unrelated changes at the same time as a major upgrade

⏪ Have a rollback strategy

🕰️ Factor rollback likelihood into release timeline promises

🔔 Not every breaking change should be fixed

🙅 Users may object

🐁 Prepare for a long tail of follow-up work

© 2025 Bloomberg Finance L.P. All rights reserved.

Happy upgrading!

TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.

Bloomberg

Engineering

Thank you!

Bloomberg

Engineering

Open studio registration: https://bburl/sfos25

We are hiring: bloomberg.com/engineering



TechAtBloomberg.com

© 2025 Bloomberg Finance L.P. All rights reserved.