# Airflow Debugging Improvement Survey

**About the survey**

- The goal of this year's survey is to uncover the pain points of users of all levels when it comes to the Airflow debugging experience. We plan to use these results to create and prioritize action items for Airflow 3.x specifically related to debugging & DAG development.
- The information collected will be anonymized and shared publicly with the community. The respondents will be notified once the results are published. Any personal data collected will not be used for marketing purposes.

**Instructions**

- While you may skip questions or categories, your responses are incredibly valuable. Please take the time to answer as many questions as possible to help us make meaningful improvements.
- In questions with checkbox (□) options, please select all that apply.
- Some questions are sentiment-based; in case you have no opinion on the matter or are unfamiliar with the topic, please skip the question instead of selecting "Neutral", "Sometimes", etc.
- If you wish to elaborate on an answer, and there's no dedicated field for it - please use the feedback box at the last section, and mention the question number.
- You may amend your answers after submission.

**Estimated Time to complete**: 10 mins

1. Email *

## **1.** Logging & Tracebacks

2. **1.1.** What issues have you encountered with Airflow logs?

*Tick all that apply.*

- [ ] Logs are fragmented across different systems/components
- [ ] Logs are not verbose enough
- [ ] Logs are too verbose
- [ ] Logs are non-existent or missing in some cases
- [ ] Logs are difficult to search and filter
- [ ] Logs are not aligned to technical layers/responsibilities (infrastructure, application, business logic)

3. **1.1.1.** Other issues, examples or clarifications:

4. **1.2.** What challenges have you been facing with Airflow's stack traces?

   *Tick all that apply.*

   - [ ] Stack traces are too complex and hard to follow
   - [ ] Stack traces are incomplete or missing critical information
   - [ ] Stack traces do not provide enough context about the DAG execution state

5. **1.2.1.** Other challenges:

6. **1.3.** How would you rate the ease of understanding stack traces in Airflow?

   *Mark only one oval.*

   | | 1 | 2 | 3 | 4 | 5 | |
   |---|---|---|---|---|---|---|
   | Very | ◯ | ◯ | ◯ | ◯ | ◯ | Very Easy |

7. **1.4.** What improvements to Airflow’s traceback information would make debugging easier?
   Please provide your suggestion(s) in the text box below, or choose from the options provided in 1.4.1

8. **1.4.1.** Potential improvements:

*Tick all that apply.*

- [ ] Simplified and more readable stack traces
- [ ] More detailed context about the state of the DAG and task at the time of failure
- [ ] Better correlation between different logs and stack traces
- [ ] Inclusion of specific suggestions or hints for resolving the issue
- [ ] Enhanced visibility of nested errors and their causes
- [ ] Integration with visualization tools to trace execution flow

## **2.** Error Handling

9. **2.1.** What issues have you encountered with Airflow error messages?

*Tick all that apply.*

- [ ] Error messages are vague or non-specific
- [ ] Error messages do not provide clear guidance on resolving issues
- [ ] Error messages lack context or details about the failure
- [ ] Error messages are inconsistent across different components

10. **2.1.1.** Other issues:

_____

_____

_____

_____

_____

11. **2.2.** How would you rate the clarity & actionability of Airflow error messages?

*Mark only one oval.*

| | 1 | 2 | 3 | 4 | |
|---|---|---|---|---|---|
| Not | ◯ | ◯ | ◯ | ◯ | Highly actionable |

(Details for question 2.3)

**Below are multiple suggestions to improve Airflow's error handling. Please choose up to five suggestions which you consider the most useful, and rank them in order of importance (1st = most important)**

A. Make certain error messages more focused and specific
B. Clarify vague or confusing error messages
C. Convert certain errors to warnings instead of halting execution
D. Raise issues earlier in the execution process, possibly in the DAG-parsing stage
E. Provide more detailed context around error messages
F. Group related errors to avoid overwhelming the user with multiple alerts
G. Include actionable steps or suggestions within error messages
H. Standardize error messages across different components for consistency
I. Introduce a user-configurable error threshold before alerts are raised
J. Offer the ability to suppress or customize error notifications

12. **2.3.** Which of the following suggestions would improve Airflow's error handling?

*Mark only one oval per row.*

| | A | B | C | D | E | F | G | H | I | J |
|---|---|---|---|---|---|---|---|---|---|---|
| **1st** | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| **2nd** | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| **3rd** | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| **4th** | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |
| **5th** | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ |

## **3.** Tooling & Integrations

13. **3.1.** Which tools do you use to develop Airflow DAGs?

*Tick all that apply.*

- [ ] Text editor (e.g., Vim, Notepad++)
- [ ] Source code editor (e.g., VSCode, Sublime Text)
- [ ] Integrated Development Environment (IDE) (e.g., PyCharm, Eclipse)
- [ ] Jupyter Notebooks
- [ ] Cloud-based development environments (e.g., AWS Cloud9, GitHub Codespaces)
- [ ] Other: _____

14. **3.2.** How satisfied are you with Airflow's integration with modern debugging tools and the related documentation?

*Mark only one oval.*

| | 1 | 2 | 3 | 4 | |
|---|---|---|---|---|---|
| Very | ◯ | ◯ | ◯ | ◯ | Very satisfied |

15. **3.3.** How often do you use external tools (i.e. besides Airflow's API, UI, and CLI) to supplement Airflow's debugging capabilities?

*Mark only one oval.*

| | 1 | 2 | 3 | 4 | 5 | |
|---|---|---|---|---|---|---|
| Nev | ◯ | ◯ | ◯ | ◯ | ◯ | Always |

16. **3.4.** What sort of external tools do you use in conjunction with Airflow for debugging?

*Tick all that apply.*

- [ ] Log management systems (e.g., ELK Stack, Splunk)
- [ ] Monitoring tools (e.g., Prometheus, Grafana)
- [ ] Tracing tools (e.g., Jaeger, Zipkin, OpenTelemetry)
- [ ] Profiling tools (e.g. cProfile, Austin)
- [ ] Debuggers (e.g., pdb, debugpy, PyCharm)
- [ ] Other: _____

17. **3.5.** What integrations or tooling improvements would you like to see in Airflow to enhance your debugging experience?

_____

_____

_____

_____

_____

18. **3.6.** Which of the following code assistance and inspection tools do you use while developing Airflow DAGs?

*Tick all that apply.*

- [ ] Linters (e.g. Flake8, Ruff-lint, Mypy, Pylint)
- [ ] Formatters (e.g. Black, Ruff-format, autopep8)
- [ ] Auto-completion and AI tools (e.g. Copilot, IntelliSense, Cursor)
- [ ] Other: _____

## **4.** DAG development

Below are several activities a DAG developer typically goes through, which we'll be referring to in this section. In bold are their short names.

- Writing the **business logic**
- **Authoring** the DAG (e.g. decide on the Airflow features to use, define operators and their relations)
- **Iteration** during development (i.e. run the DAG to see if it behaves correctly, amend it if not, repeat as needed)
- **Integration** into production (address any challenges related to interacting with resources that cannot be simulated/mocked during development)
- Post-deployment **maintenance**

19. **4.1.** What is the most time-consuming activity related to developing new DAGs?

*Mark only one oval per row.*

| | Most | 2nd | 3rd | 4th | Least |
|---|---|---|---|---|---|
| **Business logic** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **DAG authoring** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **Iteration** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **Integration** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **Maintenance** | ◯ | ◯ | ◯ | ◯ | ◯ |

20. **4.2** What can be improved about the workflow activities mentioned above?

Feel free to add any other time-consuming activity not mentioned above too.

21. **4.3.** How often do you leave the Airflow UI or CLI (and rely on external tools) to achieve each of the above?

*Mark only one oval per row.*

| | Never | Rarely | Sometimes | Often | Always |
|---|---|---|---|---|---|
| **Business logic** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **DAG authoring** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **Iteration** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **Integration** | ◯ | ◯ | ◯ | ◯ | ◯ |
| **Maintenance** | ◯ | ◯ | ◯ | ◯ | ◯ |

22. **4.4.** Which of the following additions to the Airflow UI could be useful to your debugging efforts?

*Tick all that apply.*

- [ ] An integrated terminal in the UI
- [ ] An integrated debugger (for stepping, pause-on-error, workspace inspection and modification, etc.)
- [ ] A file explorer
- [ ] Ability to make temporary/permanent changes to DAG code, potentially exporting as a new file.
- [ ] Ability to package the code and necessary debugging context to allow reproducing the issue on another airflow instance, to facilitate bug reports and help requests.
- [ ] Integration with SCM platforms (e.g. GitHub, Bitbucket, GitLab) to streamline code modifications.
- [ ] Integration with task management platforms (e.g. Trello, Jira) to streamline the opening of feature/bug tickets.

23. **4.4.1.** Other suggestions:

24. **4.5.** What could make DAG.test() more useful?

*Tick all that apply.*

- [ ] Mock file system queries/reads/writes
- [ ] Mock network requests/responses
- [ ] Mock dataset events
- [ ] Store/load the state of a DAG
- [ ] Ability to jump into a specific task/operator with a user-provided state

25. **4.5.1.** Other suggestions:

26. **4.6.** What kind of remote Airflow environment(s) do you use?

*Tick all that apply.*

- [ ] Self-Managed Airflow on Kubernetes (using official Helm Chart)
- [ ] Managed Airflow by a cloud provider (Astronomer, AWS, Azure, GCP etc.)
- [ ] Airflow on Heroku
- [ ] Bare metal or VM-based install
- [ ] Other:

27. **4.7.** How would you rate the ease of debugging DAGs in a remote Airflow environment (Kubernetes, Docker, etc.)?

*Mark only one oval.*

| | 1 | 2 | 3 | 4 | |
|---|---|---|---|---|---|
| Very | ◯ | ◯ | ◯ | ◯ | Very Straightforward |

28. **4.7.1.** What can make debugging a remote Airflow deployment easier or more efficient?

_____

_____

_____

_____

_____

## **5.** About yourself

29. **5.1.** How would you describe your proficiency level with Airflow?

*Tick all that apply.*

- [ ] Beginner (made a handful of DAGs recently)
- [ ] Advanced (made and debugged several DAGs over the last few years)
- [ ] Power user (made and debugged DAGs for several years using pdb, IDE or remote debuggers, etc.)
- [ ] Administrator (supported Airflow installations at a platform level)
- [ ] Airflow Committer/PMC Member

30. **5.2.** What are your responsibilities in the context of using Airflow?

*Tick all that apply.*

- [ ] DAG development (e.g. Data Engineer, Software Developer)
- [ ] Airflow administration (e.g. Administrator)
- [ ] Airflow monitoring & availability (e.g. SRE, DevOps)
- [ ] Other: _____

31. **5.3.** How would you describe your role in the context where you use Airflow most often?

*Tick all that apply.*

- [ ] Junior team member
- [ ] Senior team member
- [ ] Technical supervisor (e.g software architect)
- [ ] Non-technical supervisor (e.g. project or product manager)
- [ ] Lone developer (e.g. independent consultant)
- [ ] Other: _____

**5.4.** How can you be contacted for a follow-up?
Please leave the below empty if you do not wish to be contacted.

32. **5.4.1.** Airflow Slack username

_____

33. **5.4.2.** GitHub handle

_____

34. **5.4.3.** Email Address

_____

**6.** Final thoughts

We thank you for supporting Airflow by taking the time to submit this survey! If you'd like to get in touch with the community or have any further questions or concerns, feel free to subscribe and ask on the Airflow Users mailing list (users-subscribe@airflow.apache.org).
If there's any other feedback that you'd like to share with the Airflow community - now's the time!

35. **6.1.** Free-form feedback

Do you have any final thoughts or suggestions on how we can improve Airflow or the debugging experience specifically?

36. **6.2.** Feedback about the survey itself (length, thoroughness, relevance, etc.)