Postman Flows: A Visual Programming Tool for Building API-Powered Apps and Workflows

Rebecca Krosnick
Postman, Inc.
New York, NY, USA
rebecca.krosnick@postman.com

***Abstract*—Flows is a new visual dataflow programming tool from Postman for creating API-powered applications and workflows. Users chain together nodes that make API calls, process and manipulate data, and present data visually. Flows enables API producers to demonstrate API use cases, builders to prototype API-powered apps quickly, and individuals with limited programming experience to actually create software. Thousands of developers, solution architects, QA engineers, and hobbyists use Flows every day to build and share new applications and workflows. To make building with Flows even easier, we are also exploring AI-powered approaches to create and edit flows with natural language.**

I. Postman

Postman[1] is the world's leading API collaboration platform. API producers can design, build, test, and document APIs, and do it collaboratively. API consumers can discover, experiment with, and integrate APIs into their own software. Postman was founded in 2014 and over 30 million developers use Postman. Currently Postman users are largely developers and technology professionals, but one of our goals going forward is to make building and using APIs more accessible for everyone.

II. Postman Flows

At Postman, one approach we are leveraging to make APIs more accessible for everyone is visual programming. Our recent product *Flows*[2] is a visual dataflow programming tool for building API-powered applications and workflows. Users drag and drop to add nodes to a canvas and chain them together, enabling them to create programs that make API calls, process and manipulate data, and visualize data. This can be useful for a variety of needs, for example, creating dashboards for monitoring live data, integrations with messaging apps or Customer Relationship Management systems, synchronization across data sources, or any sort of custom data workflow. Figure 1 shows an example flow that retrieves price history for a given stock (Fig 1A), visualizes this data over time (Fig 1D,E), and then specifically calls out whether this week's stock price is higher than in the previous week (Fig 1F).

Flows includes the following primary kinds of nodes:

- **API Request:** Users can make a REST API call and inspect the response using the "Send Request" node (Fig 1A).
- **Data Processing:** Using the "Evaluate" scripting node (Fig 1B,C), users can write custom code snippets in TypeScript or using the Flows Query Language (FQL) to query and manipulate data within their flow, or construct new data to pass along to the next node(s).
- **Control Flow:** Users can create looping and conditional logic using nodes such as the "If" and "For" nodes.
- **Data Types and Variables:** Users can create hard-coded values using data type nodes such as "String", "Bool", "Number", and "List" (Fig 1G), and can also create these as variables to reference elsewhere in their flow.

More recently we have also been exploring LLM-assisted approaches to make creating flows easier, especially for users with limited programming experience. At a low-level, through our "Create with AI" node, users can process data without writing code, simply by specifying their goal using natural language. At a higher-level, users can generate full chunks of a flow through natural language – users attach our AI agent, *Postbot*, to a node in their flow that they want to build on and then type their goal in natural language; Postbot then asks for clarification if needed and generates sequences of nodes. Users can also follow up with refinement requests. An example scenario is shown in Figure 2.

III. Implementation

The Flows node-based editor UI is implemented as a JavaScript web app. Each flow is modeled as a dataflow program, where upon receiving new input from its parent nodes, each node evaluates it new value and passes it along to its child nodes. Our AI-powered code and node generation features leverage 3rd party LLMs and context around a target node the user has specified.

IV. Presenting at VL/HCC

We plan to present live software demonstrations of Flows on our laptop, including a few completed flows to illustrate use cases and complexity, and live demos of authoring and our AI features. We hope that our demos spark fruitful discussion on a variety of topics, for our own learning and to enrich participants' experience: e.g., balancing expressivity versus ease of use, debugging techniques for dataflow programming, human-AI collaboration. We also hope to gather feedback on Flows from conference participants.

[1] https://www.postman.com/
[2] https://www.postman.com/product/flows/

1943-6106/24/$31.00 ©2024 IEEE
DOI 10.1109/VL/HCC60511.2024.00047

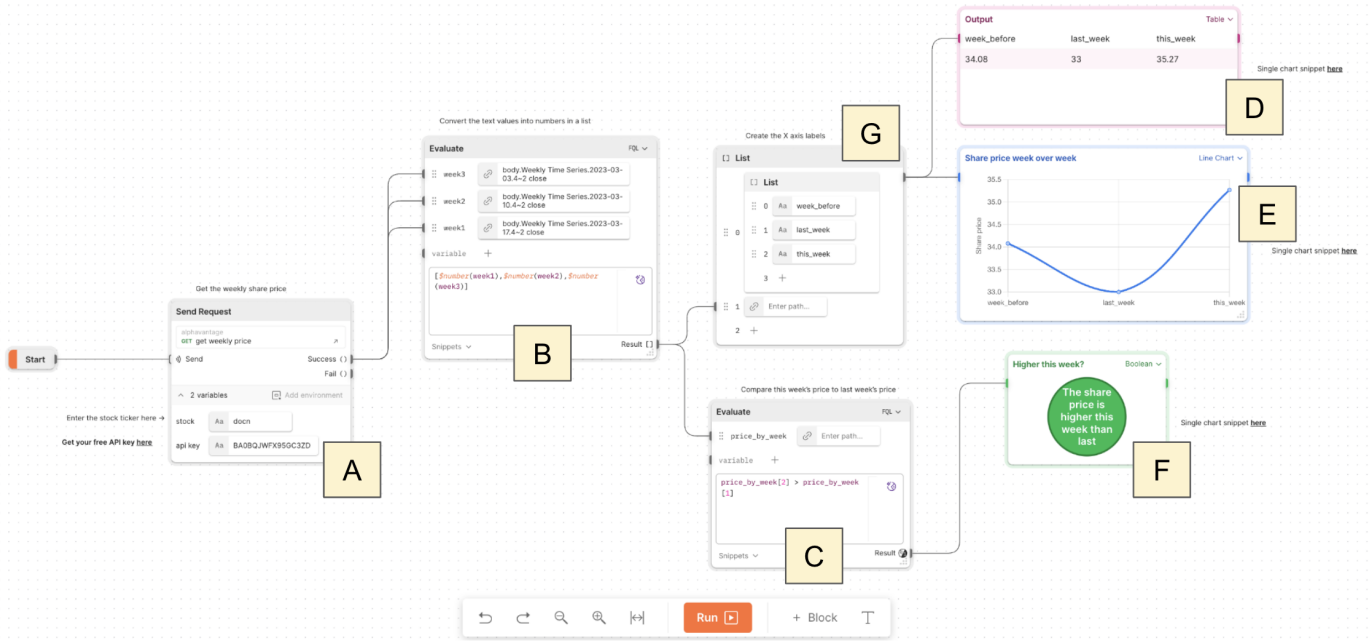

Fig. 1. *Flows* is a visual programming tool for building API-powered apps. This flow first (A) queries stock prices from the Alpha Vantage stocks REST API (https://www.alphavantage.co/) for DigitalOcean (DOCN), and (B) from that API response, extracts particular weeks of interest and places them in a list. Next the flow (G) constructs a new list object to add labels for each week and then visualizes this data (D) in a table and (E) a line chart. The flow also (C) performs a secondary order operation determining if the current week's price is higher than last week's and (F) illustrates this clearly through a boolean yes/no visual.

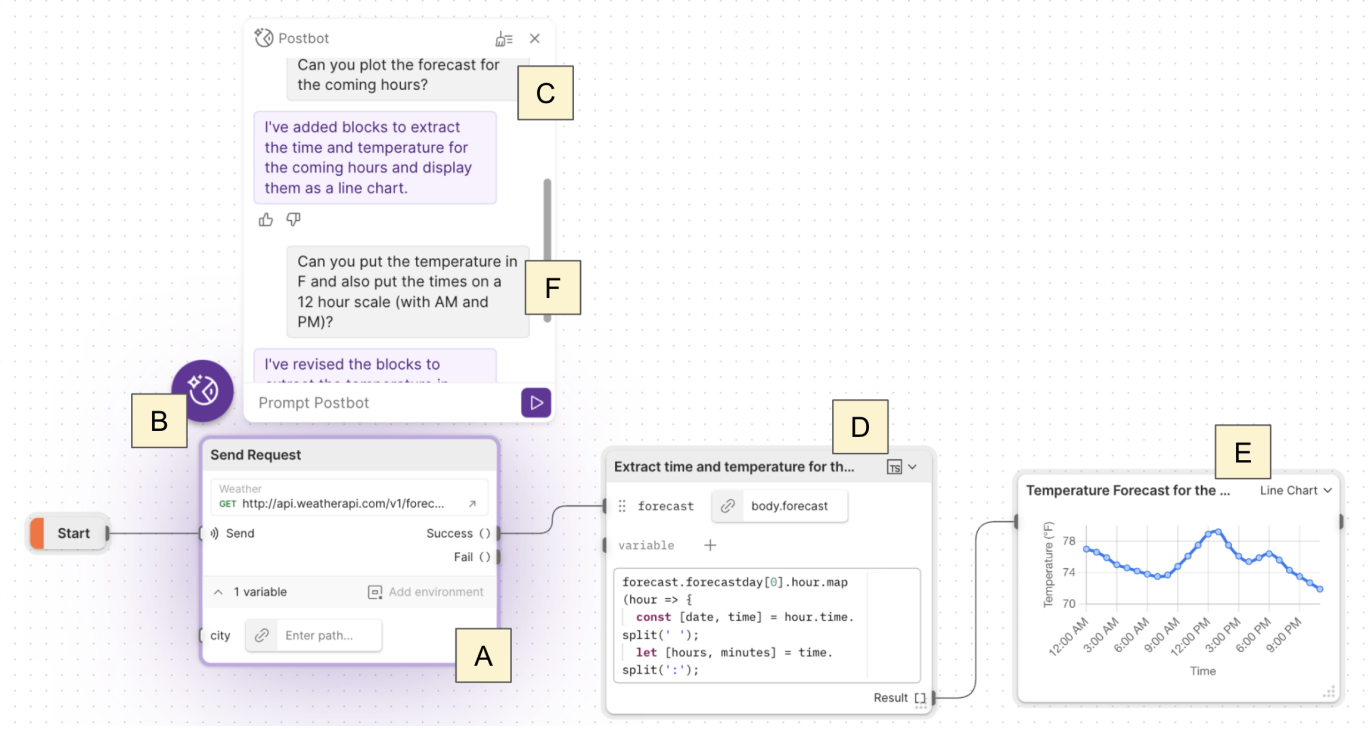

Fig. 2. *Flows*' AI agent, *Postbot*, helps users generate full sequences of nodes. Here the user first starts building their flow with (A) a "Send Request" node to query the Weather API (https://api.weatherapi.com/) for today's weather in New York City. The user then wants to plot today's weather forecast and decides to use (B) Postbot, to save time populating nodes and writing tedious data processing code. The user (C) writes a request to Postbot, which after a few seconds adds two additional nodes to the flow – (D) an "Evaluate" node to extract the correct data from the API response object and appropriately parse and combine strings, and (E) an "Output" node to visualize the data as a line chart. The user is mostly happy with Postbot's initial attempt, except for the temperature units (initially shown in Celsius) and x-axis time labels (initially shown on a 24 hour clock). The user (F) asks Postbot to correct these items, and Postbot updates the flow (final version shown here).

V. Acknowledgments

I am only one member of a large team at Postman that created Flows. Special thanks to the rest of the team: Shrey Bhagat, Shamasis Bhattacharya, Michael Browning, Sterling Chin, Michael Claus, Saswat Das, Joe Fusco, Michael Hudelson, Anand Keshavan, Daniel Kimmelmann, Anudeep Medicharla, Prashasti Nagpal, Romulo Nascimento, Chris Patty, Rodric Rabbah, Alice Robertson, Navneet Singh.