PagerDuty Software Coding Interview Guide

At PagerDuty, we believe in testing out your everyday programming skills and are not expecting you to memorize theoretical algorithms or complex data-structures. We are looking for developers who code well, communicate clearly and seek to understand what is the right user experience for customers. Expect to show-case these skills during the interview.

The structure of your interview

Timeline: 60-min

- 5-min: Introduction to the interviewers/interviewee
- 5-min: Coding Problem definition and clarifying question
- 45-min: Core coding exercise, in your preferred development environment
- 5-min: Time to interview us and learn more about how it is working at PagerDuty

What to expect during the interview?

- Outcomes:
    - We prefer working code during this session and recommend using your time towards stabilizing an initial solution and then optimize/extend it further as time permits. Ask your interviewer for feedback along the way.
    - Focus on well-organized and readable code as feasible, given the time constraints. We understand that time is short, so don't try to be perfect..
    - If your code is not functional at the 40 min mark, it is ok to spend time talking about the rest of the modules and your approach to implementing them. We understand that 60-min is not enough time to showcase years of programming experience.
    - If your code runs into issues in the first-half of the interview, take time to debug the code. We want to see how you approach debugging and fixing the code.
    - We highly encourage thinking out loud and discussing your approach while coding. Our interviewers will be happy to share any guidance.

Lessons learned from our interviews:

- We are not looking for pseudo code or simply explaining the programming language concepts, so we recommend spending under 5-min on initial clarifying questions in order to ensure you have enough time for implementation.
- Please ask questions and clarifications along the way and expect your interviewer to partner with you but ensure you are maintaining focus on getting a working solution.

PagerDuty Software Coding Interview Guide

How should I prepare for this interview?

- You may choose any language and any development framework to write your code but here is our recommendation based on candidates that do really well:
    - You will have an hour to showcase your skills. Languages that require a lot of package installs or dependencies may make it hard to complete your code in the time allotted.
    - Due to the time constraints, we do not recommend compiled languages such as Java, C++, C# that rely on heavy dependency setup/libraries. If you still choose to use Java, please consider using our provided Java skeleton (below) as a starting point.
    - We **recommend** prototyping-friendly languages like Python, Javascript, Perl, Ruby, etc which are ideal for coding interviews and showcasing your knowledge of programming v/s dealing with build and/or framework hiccups.
- Please have your laptop prepared with boilerplate client code to make HTTP calls and parse JSON data.
- You will be making various HTTP calls and parsing JSON responses using your code. Test a few PagerDuty API calls with your client code to ensure you are familiar with our APIs.

- Familiarize yourself with our API documentation here:
    - https://developer.pagerduty.com/api-reference/reference/REST/openapiv3.json
- Test API Token: y_NbAkKc66ryYTWUXYEu

- Ensure your code is well-built (if using a compiled language) and runs reliably **before** the interview and you are not dealing with dependency errors/environment issues during the interview.
- Extra points:
    - While not necessary, we would like to see how you think about making your code easy to use and operate.
        - If you are backend developer with minimal UI experience, think about how you might package the final code into a CLI tool
        - If you are a front-end/full stack developer with extensive UI experience, feel free to prototype using a simple UI page. Don't get too carried away with bells and whistles, the goal is to ensure you can implement the requirements.
    - Unit Tests / Test Driven Code
        - If time permits, try to write some unit tests or start your code with tests that you will use to ensure different methods are reliable.

Leveraging our basic skeleton/sample code

- If you don't want to build your own starter client, you may choose one of the starter clients we have to get you going
    - Java : https://github.com/PagerDuty/interview-skeleton/tree/main/java
    - Elixir : https://github.com/PagerDuty/interview-skeleton/tree/main/elixir
    - Ruby : https://github.com/PagerDuty/interview-skeleton/tree/main/ruby
    - Python: https://github.com/PagerDuty/interview-skeleton/tree/main/python
    - Javascript : https://github.com/PagerDuty/interview-skeleton/tree/main/javascript

Instructions on how to use the clients are available in the README for each code example.