Open Source Blog Template for Micro-internships

This document describes a template for writing a technical blog post for your open source contribution. Five key sections are listed. Please replace the questions with your responses as separate paragraphs.

After completing the blog post using this template, please feel free to reorder, revise or add additional sections or subsections as you see necessary to make your blog better and easy to read.

Example blog post written by one of the CTI/CodeDay open source mentors, Natalia Gill.

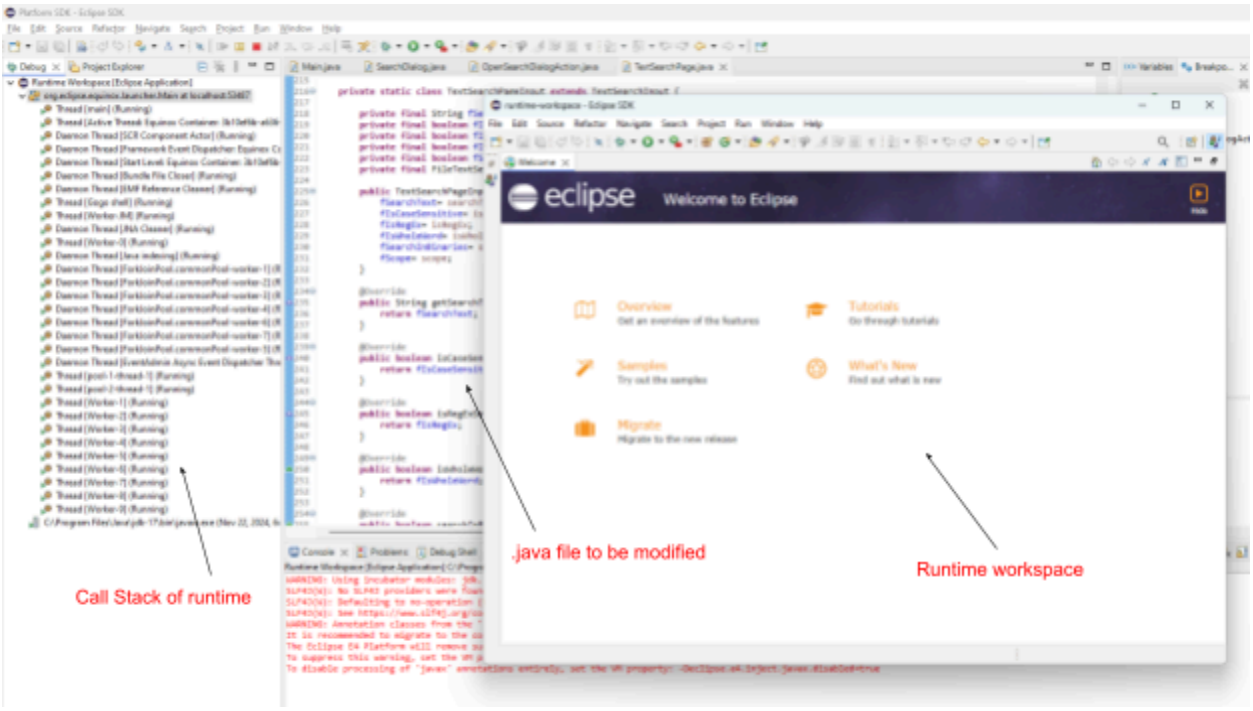
Section 1: About the Project

A. **Required:** What is the open source project about? (Example)
   - Eclipse Platform.UI Issue #1943 is about file pattern search. Currently, when searching for a specific file in the Eclipse project explorer, the user does not get recommended paths for possible locations of where it might be. Because of this, they need to reside to other third-party softwares to fulfill this need.

B. **Required:** Why is this open source project important? (Example)
   - This issue proposes a feature that makes the search process more efficient and user-friendly. Developers often deal with large projects containing thousands of files, and the ability to filter by path helps narrow down results quickly, saving time.

C. **Required:** Who is the typical user of the project and what do they use it for? Give an example scenario where the app could do something useful for someone. (Example)
   - Primary users for the project are software developers (mainly those who program in Java, but other languages are also welcomed), open-source contributors who work on solving Eclipse issues, and educators/students who are learning a new programming language, mainly Java.

D. **Required:** Show a screenshot of the app running on your local machine (or Github Codespaces) and annotate it with arrows to explain what is happening in the screenshot. (Example)



E. **Optional:** How many users does the project have? (If the project is not live yet and therefore does not have users, you can skip this question). It may take some digging/research to answer this question or at least get an estimate. (Example)
    - Eclipse has over 20 million monthly downloads

Section 2: The Issue

A. **Required:** What is the issue that you addressed for the project? Link to the github issue. (Example)
    - The issue addressed for this topic is the Eclipse Platform.UI Issue #1943
    - Link to GitHub: Issue #1943
B. **Required:** Why is this issue important to the open-source project? (Example)
    - This issue allows for the enhancement of the core functionality of a programming IDE. Introducing a file pattern search feature would allow for smoother navigation through the project explorer for the developers as they work with large codebases consisting of several thousands of files.

    - Additionally, instead of manually sorting through irrelevant files, developers can quickly filter search results based on path patterns, significantly reducing the time spent locating specific files. This not only saves valuable time but also improves workflow efficiency

C. **Required:** Identify the key location(s) of the codebase that are relevant to solving the issue. Describe the purpose of the file, function, or folder you identified. (Example)

- The target window in Eclipse is called “Search Dialog Window”, which would deem it reasonable to look for a “SearchDialog” related file.
- Confirming it through breakpoints, one of the target files is “SearchDialog.java”
- Additionally, reading through the CONTRIBUTIONS.md on the issue, and using the available commands to us, another key file is “TextSearchPage.java”
-
- One relevant class would be SearchPatternData class, which contains multiple other functions like SearchPatternData create(IDialogSettings)--which seems to be responsible for parsing and checking for various condition.

D. **Optional:** Show any supporting materials/artifacts related to the issue. For example, was there an error message? It might be helpful to show a screenshot of the error message in your blog post. (Example)

==========

Choose one out of the remaining 3 sections. You are strongly encouraged to do more (ideally all 3) but for the purpose of the Micro-internship you are only expected to choose one.

==========

Section 3: Codebase overview

A. **Required:** Describe the tech stack. For each component of the tech stack, describe its purpose in relation to the overall app. We recommend you create a table with two columns: The left column can name the technology, the right side describes the purpose. (Example)

B. **Required:** Show a system diagram for the codebase. At least one part of your diagram (either a component or an arrow) should correspond to the section of the codebase you identified in your answer to 2C (Example)
C. **Required:** Describe the full workflow of the code from start to finish by walking through one specific use case. Start with the user actions / inputs and explain what happens at each step of the diagram in your own words all the way to the final end result. Optional: Consider numbering the relevant components in your System Diagram so that you can refer to them by number as you describe the workflow. (Example)

Section 4: Challenges

A. **Required:** Clearly identify at least one specific technical challenge you faced in the project. (Example)
    - One technical challenge I faced was
B. **Required:** List out at least three different self-directed problem-solving attempts you made to address the challenge(s). (Example)
C. **Required:** If you couldn't solve the problem by yourself, share who you reached out to get help (Example)
D. **Required:** What is the status of the technical challenge? (Example)
E. **Optional:** Show a code snippet (Install the Code Blocks extension for Google Docs to get easy formatting and line numbers). Explain in your own words what the code is doing at one-two specific points in your code snippet. Reference the code by line number if possible. (Example)

Section 5: Solution

A. **Required:** Describe your final solution. Explain in your own words what you did and how it solved the original problem. (Example)
B. **Required:** When describing your solution, reference at least 1 key technology / framework from your tech stack summary (answer to 3A) **and** reference at least 1 component of your System Diagram (answer to 3B). Be sure to get feedback on this part from your mentor to make sure your usage of the technical terms is accurate. (Example)
C. **Required:** How did you test / verify that it works? Can you show some kind of proof that it works as intended (perhaps show a screenshot)? (Example)

D. **Optional:** Update the System diagram. Highlight a relevant component, add a new arrow somewhere to represent the impact of your solution, or zoom in to a more granular view of a specific component to illustrate where exactly your solution fits in. (Example)
E. **Optional:** Show a code snippet of your solution. Explain how it works in your own words or with pseudocode / comments. (Example)
F. **Optional:** Link to the pull request if your team submitted one (Example)