Google Summer of Code 2020 Proposal

Project-

Replace FreeType's tracing and debugging facilities with an external logging library

Priyesh Kumar

Mentors:

Werner Lemberg, Alexei Podtelezhnikov, Toshiya Suzuki (FreeType), Armin Hasitzka.

Personal Details:

Name	Priyesh Kumar
Country	India
University	Indian Institute of Information Technology, Una
Degree	Pursuing B.Tech in Computer Science Engineering
Email	priyeshkkumar@gmail.com
Contact Details	+918318691545
Time Zone	Una, Himachal Pradesh (GMT+5:30)
Github	https://github.com/Priyeshkkumar

About Me:

I am Priyesh Kumar, a third-year student of the Indian Institute of Information Technology, Una India pursuing a Bachelor of Technology in Computer Science and Engineering. I am very much interested in open source software development. I am proficient at C programming language and Unix build tools which makes me a suitable candidate for this project.

Project Proposal:

Title: Replace FreeType's tracing and debugging facilities with an external logging library

Proposal Abstract: FreeType is an open-source library used to render fonts. Please find details @ here . FreeType is packed with a lot of font-related features. Each feature in itself has complex mathematics and memory management involved. This makes logging a very important aspect of this library. Freetype does have a home-brewed tracing and debugging facility which is well thought and designed to fulfill the debugging and development needs. But then there is a limitation of this facility to write logs only on standard error. This logging feature helps and supports debugging on platforms that have inbuilt support for standard error but fails on platforms lacking standard error. Since FreeType is used on a variety of platforms, there is a strong need to enhance the current logging facility beyond a standard error in order to ease the development and debugging.

The idea under this project is to enhance the existing design of writing logs to standard error to writing logs to the file system considering almost all platforms support input/output to files.

Writing the code for file I/O for multiple platforms might be time taking and might require a lot of testing and later might take a lot of time to stabilize the FreeType library as a whole.

There are multiple logging libraries freely available in the market. They are stable and have the capability to write logs in files. Hence the proposal is to explore such stable and capable logging libraries already available in the market and pick one of them which best fits the FreeType debugging requirements and integrate it in FreeType.

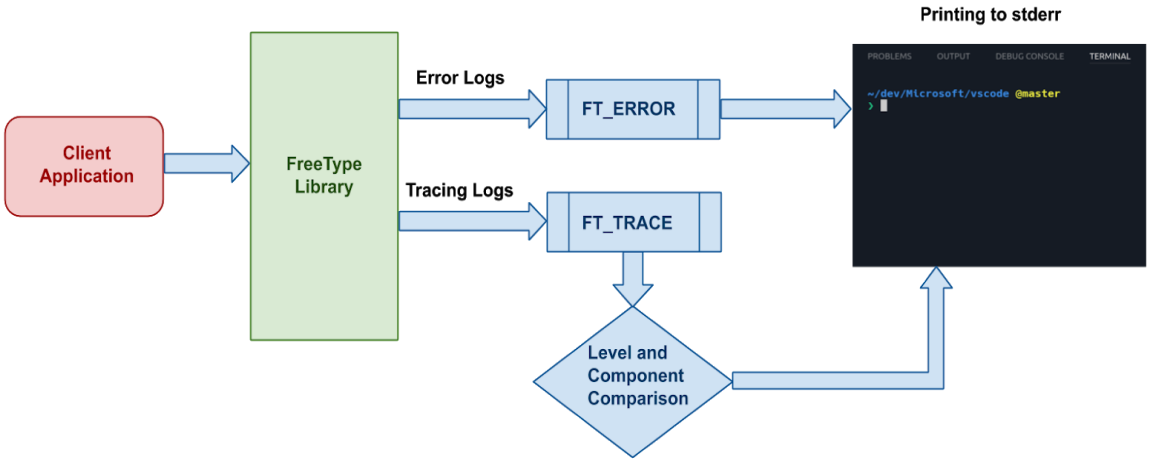
Hence, this project involves exploring the existing logging libraries, selecting one of them, integrating the same in FreeType and finally touching upon the testing part.

Detailed description of the project proposal:

- ### Current Logging Facility:

A detailed description of how current tracing and debugging feature works could be found @ here.

Program Flow of current logging feature:



★ **FT_DEBUG_LEVEL_ERROR** is a macro present in ftoption.h file in the FreeType project. This macro enables logging of error messages in FreeType. Once this is enabled, FT_ERROR macro gets enabled. FT_ERROR, in turn, invokes FT_Message which uses vfprintf() standard function to write the logs on standard error.

Hence after enabling FT_DEBUG_LEVEL_ERROR macro, all the log messages in the code base logged via FT_ERROR macro are written to standard error

★ Similarly, there is another macro **FT_DEBUG_LEVEL_TRACE** also present in ftoption.h file in the FreeType project. This macro enables logging of error and tracing messages in FreeType. Enablement of this macro firstly calls **ft_debug_init()** function whenever **FT_NEW_Library()** function is called. This ft_debug_init() initializes FreeTypes tracing sub-system. Additionally, it retrieves the environment variable **FT2_DEBUG.** Via this environment variable, the user could control the enablement of trace logs by defining the debug levels for the components. Enablement of FT_DEBUG_LEVEL_TRACE macro also enables FT_TRACE macro. When **FT_TRACE** is invoked from the

codebase**,** the debug level needs to be passed as an argument. This argument is compared with the debug level of the current component of FreeType (received via FT2_DEBUG environment variable). If former stands less than latter, FT_TRACE, in turn, invokes FT_Message which uses vfprintf() standard function to write the logs.

★ There is another macro **FT_DEBUG_MEMORY** present in ftoption.h file in the FreeType project. This macro enables memory debugging in FreeType. Once it gets enabled, messages are printed using the standard printf() function.

- Proposed enhancements to Logging feature:

  My proposal will include following stages:

  Stage 1: Requirement gathering

  This stage will include the study of current logging facility in detail and list down the pros and cons ending up to gathering the requirements.

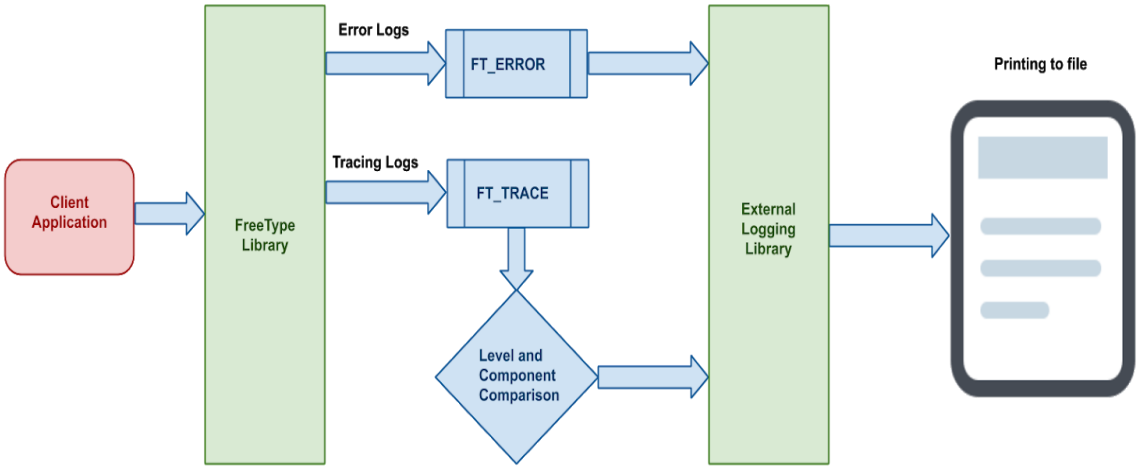
  Stage 2: Selection of an external library

  This stage will include the exploring of available logging libraries, their pros and cons and then discussing with mentors and finalizing the best fit for the requirement.

  Stage 3: Integration of the selected logging library in FreeType

  Once an external library is selected, it will be integrated in FreeType for multiple platforms. For this, a new interface will be written which will act as a bridge between FreeType and the library. At this stage, it will be ensured that backward compatibility is maintained. In case someone wants to stay with the existing logging feature, she should be able to do that. In order to maintain the backward compatibility, a new macro will be introduced. Users could un/define the value of the macro according to the need.

  Please refer the below architecture diagram:



Stage 4: Testing and bug fixes

Once integration is ready, basic testing would be done. This will cover the verification of logs being printed via new library and at the same should verify that backward compatibility is maintained.

Note: Stage 3 and Stage 4 need to be iterated for each platform. The list of platforms that will be targeted under the scope of this project are as follows:

1. UNIX like platforms
    a. GNU/Linux
    b. ChromeOS
    c. Android
2. BSD platforms
    a. macOS
    b. iOS
3. Windows and ReactOS

Project Plan:

I will complete the project in 4 phases.

- Phase 1: Requirement Gathering
- Phase 2: Exploring external libraries
- Phase 3: Integrating external library
- Phase 4: Testing and Bug Fixing

A detailed project plan according to the GSoC 2020 timeline:

Days	Work to be done
April 28 - May 18	**Community Bonding Period**
April 28 - May 3 (1 Week)	• Getting familiar with FreeType workflows • Project-related environment set-up • Closure on project related meetings frequency, code review process and status tracking process
May 4 - May 24 (3 weeks)	**Phase 1: Requirement Analysis**
May 4 - May 18 (2 Week)	• Discussion on FreeType/current project related queries • Basic requirement gathering
May 18	**First Coding Period begins**

May 18 - May 24
(1 Week)
	• Listing requirement Gathering
	  ○ Asking questions on mailing list
	  ○ Requirements from new logger

Phase 1 Completed !!	
May 24 - June 14 (3 weeks)	**Phase 2: Exploring External libraries**
May 24 - June 14 (3 Week)	• Start exploring external logging libraries. • Testing them according to FreeType's requirements. • Documenting a formal report of all the external libraries explored, mentioning their pros and cons. • Discussing the report with mentors and

	selecting an appropriate external library to integrate.
Phase 2 Completed !!	
June 15 - June 19	Midterm Evaluations
June 19 - July 31	**Second coding period begins**
June 19 - July 13 (6 Weeks)	**Phase 3: Integrating external library**

June 19 - July 1 (2 Weeks)	• Understanding the working of current debugging and tracing facility on Windows (and ReactOS, it uses the same APIs as Windows).
	• Integrating the external library with FreeType on these platforms.
	  ○ Writing new logic for printing error messages (**FT_ERROR**).
	  ○ Writing new logic for printing tracing messages (**FT_TRACE**).
	• Basic testing.
	• Documentation.

July 2 - July 13 (2 Weeks)	• Understanding the working of current debugging and tracing facility on UNIX-like platforms (GNU/Linux, ChromeOS, Android).
	• Integrating the external library with FreeType on these platforms.
	  ○ Writing new logic for printing error messages (**FT_ERROR**).
	  ○ Writing new logic for printing tracing messages (**FT_TRACE**).
	• Basic testing.
	• Documentation.

July 13 - July 17	Midterm Evaluations

July 18 - August 10 (4 Weeks)	**Third coding period begins**

July 18 - July 31
(2 Weeks)
	• Understanding the working of current debugging and tracing facility on BDS environments (macOS and iOS).
	• Integrating the external library with FreeType on these platforms.
	  ○ Writing new logic for printing error messages (**FT_ERROR**).
	  ○ Writing new logic for printing tracing messages (**FT_TRACE**).
	• Basic testing.
	• Documentation.

Phase 3 Completed !!	
August1 - August 10 (2 Weeks)	**Phase4: Testing and bug fixing**
August 1 - August 10 (2 Weeks)	• Testing new logger on different platforms. • Changes according to reviews. • Immediate bug fixes.
Phase 4 Completed !!	
August 10 - August 17	Students submit their final work product and their final mentor evaluation
August 18 - August 24	Mentors submit final student evaluations
August 25	Final results of Google Summer of Code 2020 announced
Onwards	Keep contributing to FreeType and will keep resolving the issues faced by users while using the new logger.

Note:

- ★ Open to change the timeline according to the mentors.
- ★ Holidays and other occasions are also considered while designing this timeline.
- ★ GSoC timeline events are marked in red, the start of phase are marked in light green and completion of a phase is marked in dark green.

Why this project?

I was always fascinated by production-level software and their working. As during our undergraduate degree we do not have any chance to work for such production-level software, which is surely a great motivation for me to work with FreeType and to write this proposal.

Further learning about new technology also pushes me towards the completion of this project.

Regarding GSoC Timeline and Working hours :

I do not have any commitments this summer and I will be able to manage 40 hours per week for this project. My working time will be a little less (approximately 2 - 3 hours) from 04 May to 20 May due to my end semester practicals and exams, but the loss will be compensated during the community bonding period.

Freetype Community :

I have been active in the Freetype community for the past two months and feel comfortable with the codebase of FreeType.

I am very thankful to all the members and mentors of the FreeType community for supporting me in my early stages in open source contribution.

GSoC participation :

This is my first time applying for Google Summer of Code and I would like to thank all members of FreeType, to provide me the opportunity to write this proposal and to work with this team.

I am looking forward to feedbacks, regarding this proposal and will be glad to discuss any changes.