Submit your case study at https://goo.gle/gsod-case-study

README: The purpose of this case study is for others to learn from your experience. You should provide an honest assessment of how things went. We are interested in hearing about mistakes and challenges as well as successes. Do your best to answer ALL of the following questions to the best of your ability.

FontTools: 2024 Google Season of Docs Case Study

Organization or Project Name: FontTools

Season of Docs link:
https://github.com/fonttools/fonttools/wiki/Season-of-Docs-2024-project-proposal

Organization Description:
FontTools (current version 4.49.0; first release 1991) is an MIT-licensed Python library for manipulating fonts. It provides modules and command-line tools for building, decompiling, troubleshooting, analyzing, drawing, and experimenting with font data in a variety of contemporary and historical font formats, including TrueType, OpenType, AFM, UFO, and several platform-specific file formats.

Authors:
Behdad Esfahbod, Cosimo Lupo, Nathan Willis

Summary

Fill out the following table and provide brief answers to the questions. You can provide more details in later sections. ***We recommend you write out the detailed sections first then come back to write the summary.***

# Tech Writers	TW Project Hours	Budget	% Project Completed
1	250	$15,000	90%

The main problem that we set out to solve with our documentation was to make it easier for a new user of fontTools to start from the documentation landing page and find what components they need in order to solve the problem that they are facing (whether that component is a command-line utility, a converter, a module for drawing glyphs, or a package for font engineering).

We took that problem to involve both the contents of the documentation for each constituent piece of the fontTools library and also the set-up and arrangement of the documentation overall. As a large library, finding your way through the layers can be a challenge in its own right.

In the end, we had significant changes to the documentation structure and organization of our docs site, we had multiple missing pieces of documentation added, and we had new user-orientation added. We did not get every module and script fully documented, but we did achieve some clarity on how to better guide newcomers to the most relevant components, and we made significant progress chipping away at the overall body of modules, classes, and functions that would benefit from more thorough documentation.

Generally speaking, the only difficulties we encountered had to do with the hassles of coordinating schedules. Our technical writer and our maintainers and lead developers were in different time zones and had their own work and real-life obligations that made it non-trivial to set up meetings. But we did get the hang of that process eventually, and we refined our goals as we went along over the course of the project.

Perhaps the key piece of advice that we came away with that could be useful for other projects is to be flexible in what you set out to accomplish: what looks like the top-five items on the to-do list on day one may not be what you think is the most important at the end. It's also important to focus on the communication aspect between the project and the technical writers; we had some interruptions, but once we had our communications running smoothly, revisiting the to-do list became easier and doing and assessing the documentation writing became easier, too.

Project Description

Project Proposal

https://github.com/fonttools/fonttools/wiki/Season-of-Docs-2024-project-proposal

Proposal Creation Process

We had considered applying to Season of Docs in several previous years. It is sort of a known issue that fontTools has a lot of constituent packages and modules, and that they are diverse in their purposes. They have also been written by different developers and some of them have been around for many years. So the state of the documentation varies quite a bit, depending on whether the module in question is actively developed or not, when it was written and by whom, and what other packages in the fontTools library it touches. That means there are a lot of bits and pieces that can be uneven where documentation is concerned, and that the overall library can seem a bit intimidating to new users as they look at it for the first time.

Some of our big contributors have worked on other documentation projects in the past, including Mozilla's MOSS program, so we had discussed Season of Docs as an option off and on. This year, a technical writer who we had had that conversation with in previous years indicated some interest in taking it on, so we decided to submit a proposal.

As far as the contents of the proposal are concerned, we looked at what documentation-related issues regularly come up on the issue tracker and what people tend to ask about on the Discussions forum on GitHub or on various font-related forums, but we also decided it would be a good idea to take stock of the status quo and prioritize based on what packages and modules were in most need of help, even if they were not generating user questions or requests for help.

We wrote the proposal collaboratively via email and posted it to the project's GitHub wiki, then asked for feedback from the community. We did not receive much in the way of feedback from that part of the process, however. A few comments from social media, but little else.

In the end, the proposal-writing stage was fairly brief (about two weeks), but that does not take into account that we had had preliminary discussions on several occasions over the

previous few years. Because of the earlier discussions, we sort of went in with a rough estimation of what needed doing and what was doable.

Budget

Answer the questions in the table below.

How much money did you ask for?	15,000
How did you come up with this estimate?	We estimated the number of hours available for the technical writer and the standard rate used in contracts by similar projects
How many hours of work did you budget for the project?	150
How many hours of work were actually needed for the project?	250+; there is still some work ongoing
What other expenses did you include in your budget?	none
Did you run into any budget surprises during the project (e.g. misestimates)? If so, please explain.	No

Tech Writer Recruitment

Before submitting the proposal, we discussed the project with a tech writer who was already a member of the community and with whom several of us had worked in the past. Some project members had considered proposing a GSoD project in prior years, but it has always been an obstacle that FontTools as a software library is highly domain-specific. Consequently, recruiting writers from the general tech-writing never felt like a good option, since that would require the writers — and the developers / maintainers — to spend considerable amounts of time just covering background information.

When the writer that we knew from some other font-engineering-related documentation projects was available this year, we discussed the project and agreed on general terms before submitting the proposal.

The writer did stay for the entire duration of the project, and has indicated that they are interested in continuing to participate in ongoing documentation work.

Timeline

The initial timeline anticipated that the documentation audit and gap-analysis phase would take place in April and early May. That would be followed by establishing a prioritized list of documentation bugs, missing pieces to be added, and other improvements, which the contracted writer would work on over the months of June, July, and August.

In practice, the schedules of the participants ultimately required pushing back the start date, and there were some slips in the mid-summer months (also due to scheduling conflicts, not with difficulties encountered in the project work itself).

The audit phase was conducted in late May. We developed the list of documentation needs and work started in June. There was a gap in July and at the beginning of August; at the end of that, we re-evaluated the list, and work on the documentation-needs list resumed. The technical writer has continued to work into fall.

The June and August work consisted largely of improving and cleaning up the documentation build chain and of squashing documentation bugs. From September on, the writing work has consisted more of adding new material and rewriting existing documentation. Some of the new and improved documentation is still in pull-request state.

Deliverables

Fill in the tables below to describe what was created, updated, or otherwise changed during the project. Include both planned and unplanned deliverables. Be sure to include things that were planned but not completed.

Planned Deliverables

Relevant links might include published docs, pull requests, or other artifacts.

Deliverable	% Complete	Relevant Links	Notes
Audit	*100*		*The documentation audit and gap analysis involved cataloguing the state of the existing documentation, both that that lives in the project Doc/* tree and that that lives inside the source annotations, and comparing the gaps found with the topics that come up most often in forum questions and GitHub issues.*
Fixes to docs build system and bugs	*100*	*https://github.com/fonttools/fonttools/pull/3625 https://github.com/fonttools/fonttools/pull/3712*	*These were mostly implemented in a series of pull requests*
Fixes to docs site structure and navigation	*100*	*https://github.com/fonttools/fonttools/pull/3611 https://github.com/fonttools/fonttools/pull/3627 https://github.com/fonttools/fonttools/pull/3730*	*This is a process that likely will see further refinement, but we feel that a good milestone was achieved in the time available.*
Newly written material and rewrites	*80*	*https://github.com/fonttools/fonttools/pull/3637 https://github.com/fonttools/fonttools/pull/3716 https://github.com/fonttools/fonttools/pull/3720 https://github.com/fonttools/fonttools/pull/3721 https://github.com/fonttools/fonttools/pull/3724 https://github.com/fonttools/fonttools/pull/3727*	*These were mostly implemented in a series of pull requests. Not all of the packages originally put on the priority list were addressed, but that is because the priorities were rearranged.*

Metrics

In our proposal, we suggested two possible metrics. First, by using the number of “how to” questions and requests for clarification posted on our issue tracker and GitHub Discussions forum. Second, by calculating the number of Python modules lacking documentation.

For the coverage metric, at the start of the project there were 310 Python module files or scripts, out of which 63 were not included in the documentation (although some of those had docstrings or comments in the source, or command-line `--help` switches available). At present, there are just 26 module files or scripts not part of the documentation, although eight of these we identified to be deprecated, fully private, or dead code.

Our proposal targeted a reduction of 80% in the number of undocumented components. The actual reduction was just over 70%. However, we also determined that the percentage-of-coverage was likely not a granular enough metric to reflect the usefulness of the documentation set to users. For example, most of the member packages have only a few designated entry points intended for public use, and/or a small set of command-line tools. We prioritized adding documentation for these user-facing pieces and explicitly highlighting them in the docs structure and navigation. We suspect that they offer a more meaningful improvement to users of the library overall than filling in missing pieces in general.

Measuring the number of questions asked on the issue tracker and forum proved to be a less clear metric, because the additions to the documentation and changes to documentation structure have rolled out slowly. It might also be difficult to gauge a correlation between the documentation improvements and the number of questions asked because there are other factors that can affect the question cycle: such as the release of a new major version of the library or the seasonal fluctuations of users.

Between June and December (when the documentation project was active), we saw six questions posted in the forum. In the previous six months, we saw 11 forum questions. In the same time periods, between June and December we saw one issue posted to the tracker that asked an explicit question; in the previous six months we saw two issues posted to the tracker that asked an explicit question. Those raw numbers amount to approximately 50% reduction in questions, but that might be more influenced by the time of year (including northern hemisphere summer) and release cycle than by any specifics of the documentation.

Moving forward, it is not clear if these metrics are easy enough to correlate to the state of the documentation to be useful in the long run. Users who don’t find what they want to know can more easily leave without asking a question at all. There are also side-issues like ease of navigation and new-user-friendliness that it is unclear how best to measure.

Analysis

Overall, we consider the project to have been a success, even though we also acknowledge that there is a lot of documentation work that still needs to be done. The nature of the work required was somewhat of a surprise. At the beginning, we envisioned that having a dedicated documentation effort would involve our technical writer writing large chunks of text, either as narratives to explain the purpose of packages or modules or as Python docstrings and examples.

In practice, there were a lot of structural reorganizations and clarifications needed. FontTools is a large library that includes more than 25 individual packages and its main purpose is to serve as an engineering toolkit: users enter fontTools with a specific problem that they need to solve, and the first big job that the documentation needs to do is help these users understand what tools are provided and find the tool that fits their need the best.

So there was a lot of time involved in the project to disentangle pieces, link pieces together, provide context to new readers of the documentation, and shed light on core ideas that might not be obvious to first-time users.

FontTools is also a library that has had many contributors from different organizations over many years, often with their own documentation writing styles. So another aspect of improving our documentation from users was to improve consistency between the various packages and modules members.

Practically speaking, although several packages ended up requiring large chunks of new written material, we decided to elevate the priority of improving the structure and consistency of the documentation. That meant that not all of the individual packages needing new material received it (although some did). But we think that the overall documentation site will be easier to maintain and we can continue to improve it for all of the member packages.

We knew at the beginning of the project that the overall size and age of the codebase meant that not every page (or every member package / module) could be improved to perfection in the course of one project. So we decided to start with an audit of the existing state of the documentation and follow that up with the creation of a prioritized list. That made for a good start, but along the way, the technical writer

We also found the idea of assessing the success of documentation via metrics to be a thorny topic. We are looking forward to seeing if the latest updates to the documentation have a lasting effect and seeing whether the number or the type of questions we receive changes as a result.

There were some obstacles encountered to getting the project completed. The first was simply the matter of scheduling; the maintainers and major contributors as well as the technical writer were all located in different cities (and, as “industry veterans”, frequently had real-life interruptions to work around). Another hurdle encountered was the toolchain used for documentation itself; we found that Sphinx (which is the base package used for our manpage and HTML documentation generation) can require workarounds to run smoothly on a codebase that includes multiple, perhaps independent, Python packages.

We did find several processes along the way that may prove to be useful practices to adopt in the future. For example, a commonly encountered problem was determining what the entry points are for a particular module. Various fontTools modules differ in whether or not they provide the __all__ variable and in how they distinguish public from private objects. When working on documentation consistency, we found that defining __all__ at the module level provides a simple clarifier that the Sphinx documentation system can consult and use to explicitly highlight public entry points, without otherwise forcing module authors to alter their coding style or internal documentation style. So it might become a standard practice across the library.

Lessons Learned

What went well?

For lessons learned, add your own or indicate a "plus one" for any of the existing GSoD Best Practices.

Topic	What we did	Lesson Learned
Budget		
Communication	Use shared, editable documents to ask questions of developers	Issue trackers rarely elicit replies; they are primarily associated with bugs needing to be triaged and fixed. So when technical writers need an answer or clarification, it works better to use a document-based tool like Google Docs that exists outside the regular development process.
Mentorship		
Metrics		
Onboarding	Set aside considerable time to get the technical writer oriented	Library projects can be more difficult to understand and internalize the structure of than an end-user-facing application is, precisely because there are many possible use cases
Participants		
Project Deliverables	Regular video chats to determine high-priority elements	A live discussion is far better for reaching a decision about whether a particular documentation component should go on the front burner or the back one than any email thread or PR is.
Project Deliverables	Emphasize orienting new users	Word counts and function coverage are not necessarily indicators that new users investigating your project for the first time will find what they need to get started
Project Deliverables	+1 to the "sustainable processes" Best Practice	
…		
Other		

What could be improved?

For lessons learned, add your own or indicate a "plus one" for any of the existing GSoD Best Practices.

Topic	What we did	What we would do differently	Lesson Learned
Budget			
Communication			
Mentorship			
Metrics	Defined metrics in terms of users asking questions or seeking help	This is not yet entirely clear. Probably some more direct engagement with the user community itself, and what it sees as its documentation needs, ought to come first.	You can't know if people (including your users or potential users) are leaving without asking you a question. If they leave in silence without finding what they need, it is hard to detect them.
Onboarding			
Participants			
Project Deliverables	Established a to-do list early, which ended up evolving several times, sometimes in big ways	Establish goals, decide on a top priority goal, then repeat that process iteratively after each top goal is achieved (or, potentially, if the top goal is blocked)	
Project Management			
Recruiting & Hiring			
…			
Other			

Appendix

If you have other materials you'd like to link to (for example, if you created a contract for working with your technical writer that you'd like to share, or templates for your documentation project, or other open documentation resources, you can list and link them here). The Appendix is also a good place to list links to any documentation tools or resources you used, or a place to add thanks or acknowledgments that might not fit into the sections above.