Sarah Murto   
WRIT 4662W  
12/04/15

DITA Memo

I chose to write my DITA document about Jekyll. Although the topic of my historicizing project was Flash, I had written my website using Jekyll. That project was my first experience using Jekyll. When we were offered the opportunity to switch topics, I decided that Jekyll suited itself better to writing software documentation than Flash.

The hardest challenge I encountered was deciding what topics to include in the project. There are many aspects of Jekyll I could have written about. I knew even trying to summarize all of the potential main topics would fall far outside the expected scope of the project. Subsequently, the final project feels more like a few pages taken from a larger document.

I started my project by simply creating empty files for every task, concept and potential reference I thought would be necessary for a full Jekyll documentation. I ended up with about 20 files. As I looked through the topics, I kept seeing other topics that would probably be needed that I forgot. This ever-expanding list of potential topics helped keep me from feeling overly ambitious. Having a list of what I could include in my project helped me pick the topics I would include in my project.

I decided to focus my documentation on tasks around creating posts and sharing data with variables. Originally, I was going to focus more on making layouts. While a layout is certainly needed when making a Jekyll website, it also is not easily discussed without explaining Jekyll’s variable system. I felt that the creating and sharing of variables in Jekyll was a necessary concept to understand first, so that was what I included in my documentation.

The following topics are included in my DITA document:

**What Is Jekyll?** – a concept topic that summarizes basic uses and functionality in Jekyll.  
 **Create a Configuration File** – A task explaining how to configure a Jekyll site.  
 **Configuration File Settings**  - A reference table of available configuration items  
 **What are Variables?** - A concept topic explaining the what and why of variables.  
 **Create Default Variables**  - A task explaining how to make variables in the configuration file.  
 **Write a Post** – A task explaining how to write and name a Jekyll post.  
 **What is YAML Front Matter?** - A concept topic explaining YAML in posts.  
 **Override Variables** – A task explaining how to override default variables from a post.

Finding the proper organization of these topics is also important. Although at first glance, there seems to be a clear order in which to do these tasks, it’s quite misleading as many tasks are done simultaneously. For instance, it seems like a user would be writing a post before they are overriding variables. However, they are likely to need to override variables as they are writing the post, not as a separate step and not at a specific time in the writing process. This made it difficult to create an order for the document.

For my final organization, I have three main topics with several subtopics. Throughout the project, this organization was reshuffled, but I feel this final organization makes the most sense. ‘What is Jekyll?’ stand by itself; the concept is an overarching intro to any topic that would have been included in my documentation. The other two main topics have subtopics.

‘Create a Configuration File’ is the second main grouping. Putting the reference topic for the configuration file was a rather obvious decision. It was harder to decide whether to separate out discussion of default variables to its own grouping, or to nest it. In the end, since default variables are set in the configuration file, it was decided the topic should be nested. I felt the concept of variables needs to be introduced before explaining to implement them, so the ‘What is a Variable’ concept was also nested under the configuration file grouping.

The third grouping is under Write a Post. Writing a post is a completely separate step from writing a configuration file, so it seemed necessary to separate this topic. Following the logic that default variables are discussed under the configuration file grouping, overriding variables was nested under the post grouping. The YAML Front Matter concept was also nested here, since it is not only required in posts, but also because it is required knowledge if one wants to override a variable.

Understanding that these topics may to be referenced within their grouping, it was important to pick an appropriate collection type. When I tested the document with the family collection type, I knew that was perfect. It didn’t force and order, and properly identified when a topic was a concept or reference that may interest the user.

Challenges were also encountered outside of the document structure. One challenge was how to properly format code. This was first encounter when I was describing that posts belonged in the ‘\_posts’ folder. I knew that in most software documentation \_posts would get its own special font: it’s a required folder name and needs to be typed that way. But I didn’t know how to do this in DITA. With the aid of Google, I discovered the codeph element. This was more appropriate than the codeblock element because I only needed one word highlighted and wanted to keep it in-line with the rest of the text.

I also experienced challenges with creating links to other topics in steps. There isn’t an obvious order to my documentation, so often steps to complete a task needed to reference other topics. I eventually used a xref and typed relevant text around this link, but I do wonder if there was a way to do that in a more reusable manner.

Overall, I think my project is well structured, although I could have been talked into another organization. I also think I picked appropriate topics for the scope of this project, although I could have easily chosen to focus on other topics. I don’t think there is one best organization or structure, but I hope this memo adequately explains why I made the decisions I made.