To: Chris Lindgren

From: John Buckeye

Subject: DITA topic model design rationale

Date:12/23/15

One of the dangers when writing the content for the concept topics giving background on the rendering methods was that they give any reference to the doing of the task that is those methods. I had to be very careful to talk exactly about what those methods were and not make mention of how to do them until the tasks themselves. Similarly, with the tasks I needed them to be very barebones in what they command, provided the user has met the given prerequisites. And so the tasks needed avoid saying anything which could have been explained in the tasks.

I chose to create a simple table for the reference topic in the topic model in some ways to get my feet wet creating visual aids within DITA, but also because I thought it was a pretty decent fit for the information I was trying to convey. As any simple table might use, there were only two potential categories of items. I wanted to make sure to keep the definitions relatively short, so they would not look too clunky or out-of-place in the structure. But as such, the format of the element allows for me to reference it in other topics while still giving it a stand-alone wholeness.

The history and kerning topics were all necessary to include because they give the user a context for how they are carrying out their desired task. For example, that they might have an idea about what is going on with the rendering of the fonts, I included “kerning”. So that they might have an idea of where kerning itself even comes from, there was a reference to it in history. And the module gives them a background on who has made the final decisions regarding the whole of web fonts.

My topic model’s goals are represented in some ways simply by to whom it becomes apparent the tasks are oriented. Within the main topic, Choosing a Font, the only prereq is that the user know what HTML and CSS are. When it comes to the other two tasks the prereqs get a little heavier. The sIFR task requires that the user have a knowledge of flash and Javascript. The @font-face task requires that the user at least have their CSS style sheet set up, and therefore have a working knowledge of how to write in CSS.

Each of the short descriptions are present, and I think accurately represent the content of the topic without going too in-depth or making any number of the faux-pas referred to in Bellamy, et al. They try to explain the benefits of a specific task, or the main idea of a specific concept without giving detailed information or referring to upcoming steps or sections of content.

I am not sure, after formatting my main task into a PDF file, what the point of including a reference to the other task or tasks is. It appears to just be an awkward remnant of a format to which this model is more suited, because you cannot click the link, even when reading it in an online PDF reader. This made me think I ought to write a short explanation attached to the choice between the two, at least so that it becomes more clear in different formats what the user is being asked to do. I decided instead to make the final step of the main task optional and offer the two choices, because each of them is indeed an optional choice.

There were a couple rather large decisions that I had to make regarding the way I presented the three main tasks to the user. I realized that I would need one sub-map in the course of my writing because before any user took up choosing which font they wanted to use and see how that would interact with their website, they would have to know which method they were using to render it. Keeping in mind that there are more simple ways to render font, simply using CSS, I’ve used the two which through my research I felt were the most historically significant. SIFR was the rendering technique which it would appear forced the hand of the W3C and others to make access to a wide array of fonts easier for those seeking open-source. @font-face, of course, completed that evolution to the place we are now, and for good reason is the most widely used.

There is a history behind both of these technologies, though, which I felt needed to be shared. If that were not the case, then it would not have been necessary to include a concept right next to the actual task topic for carrying out the method. I have tried to be as clear as possible, in juxtaposing those pairs of topics, which one in each pair is that which tells the user *what* the method is and which one tells *how* to do it. I was originally going to put the sIFR topics and @font-face topics under separate sub-maps, or even parent topics of some kind, but I thought this would bog down the flow of the project. It also became necessary to distinguish them from the final task, that of picking a font. But I also needed to allow for those who needed to know more about choosing their font to be able to easily go back to those tasks, once they had concluded that pre-required (for them) task.

Preparing the user for looking at each of these tasks is a limited history of how web fonts came about and how they have evolved and been organized. The module topic is often not going to be necessary to look at, but I felt that that the history and the tutorial on kerning could be helpful for those who are trying to get more involved in the rendering and even creation of their own fonts. The history intro, kerning concept and module concept were also going to originally be a part of a shared sub-map, but it became apparent as I wrote the other topics and one sub-map with which I ended up, that they frankly aren’t as important as the task topics themselves. Another reason for not coupling them together in a sub-map is that they are as related to each other as they are the other topics in that level of the topic model.

My topic model overall actually got a rather large transformation between when I came to the last class and when I left it. The original thought was that I would make several ditamaps with subtopics which supported three large topics: the background and the two rendering methods. The ladies in my group helped me understand that those would not be as necessary as I at first thought. I didn’t need a ditamap for each individual rendering method (sIFR and @font-face) when I could more neatly offer them as a choice right next to each other and juxtapose them with the other task-topic, that of choosing your font. This was a well-rounded discussion, as there were certain ditamaps, such as the history sub-map, which I was reluctant to part with. This was just going to make the structuring itself easier, and still allowed for at least one sub-map to further compartmentalize those topics which I felt needed to be.

I wanted to be sure I put the font-choosing task near the end for a couple reasons. It was the main task with which I started, the basis for this topic model as well as the basis for the content which stems from my historicizing project. But the truth is, it also may not be necessary for a certain level of user of this topic model. If they are the type with a large amount of experience designing pages, or even just seen a lot of sites which they admire the design of, the user may be coming in with a good idea of which strategies they are going to use, and simply need help carrying them out.

If that is the case, those people can skip right over that particular topic and go to the final choice. The font-choosing topic does not even mandate that the user choose between those two choices. In fact, it is perfectly possible to render a certain amount of fonts on just about any browser without using either sIFR or font-face. And the two “outside” methods for rendering font are still available for them to use at the end of the topic.

Not many of the steps between @font-face and sIFR are that similar, considering they stem from two very different font-rendering technologies. Between the two tasks, therefore, it was not reasonable to try to re-use any content. Nonetheless, I was having trouble linking to different task topics from my main task topic. That was because it was not working to for me to use the <topicref> within the task element. I realized that what I needed to employ was the inline link, <xref>. This enabled me to create links to the two rendering methods, and from the @font-face task to the Generic Families reference page. I simply had to make sure they would be read correctly, so I nested the Generic Families page within the <postreq> element on the @font-face page, and created an optional step for the main task. This was important, given that the user will most likely need to remember which Generic Families are which.

I chose to not directly create it as a super-task because there were not enough options which you needed to choose at least one of. I see one of the possible results of this project that those learning about fonts choose to *not* use one of the rendering methods listed, but instead choose to simply pull them using CSS.

Finally, I added a folder to help better organize my topics. I did not have that many topics to begin with, but enough of them were separate from the others that I wanted to give them a safe “home”. It was pretty easy to adjust the links to those topics from the main ditamap, because I only had to add one level to the address. And I added a level to the links in the main task as well. I put the Generic Families reference in the folder so that it would be easier to link to from the @font-face task.