THE EVERGREEN PROJECT

LIBRARY SOFTWARE CONFERENCE ASCIIDOC AND ANTORA

1:3O PM – E:30 PM

MAY 24, 2021

CAPTIONING PROVIDED BY:

CAPTIONACCESS

contact@captionaccess.com

[www.captionaccess.com](http://www.captionaccess.com)

>> It's 2:30 Eastern, 1:30 Central. We are going to get started here with Lynn Floyd and Blake Graham-Henderson for AsciiDoc and Antora. They are going to take us through a live demo and it will be exciting to watch them work. I do want to say thank you really quickly to our captioning sponsors duck sponsor, Mobius, as is the Evergreen community development initiative, which is our platform sponsor this week. And the consortium of Ohio libraries has underwritten the pre-conferences specifically. We appreciate all other support.

We'll put the link for our captioning there in the chat. And I do see that it is being live captioning. We are going to put all the questions that we have into the chat and they will pause periodically or as they go through answer the questions, we will make sure that they are read out so they are on the YouTube recording. So without any further ado, I will stop sharing my screen. And you guys go ahead and take it away.

>> I'm going to start sharing my screen. First of all, we are going to go through and do a brief overview of the documentation as it exists now and it through a history of the documentation. To start off, this is the 3.6 to take in here. This is anything 3.6 and above. This is the way we are going to move the platform from the straight AsciiDoc to Antora. Most of you don't know, the documentation didn't exist before really the first Evergreen conference back in 2009. In 2009, at that point it was a really big growth year for Evergreen. That was also the year of the first Evergreen conference. Anti-duck imitation was developed at that time.

To allow us in our first set of documentation we will go over, the first published set of documentation was 1.6. And had 2.0, 2.1. At 1.6, around about then, we started writing the -- in -- there was too much about. I think they have moved it off of -- yet. If anybody still has that 1.6 documentation, I don't get exists anymore. On any of the current servers.

Evergreen in action was published in 2012. It was a combination of things that they put out. It was 2.3. It was created by -- this is the basic set up for Evergreen. From 2.3. A lot of that has change, some of it hasn't. Since then, every year we have published new documentation.

In 3.1 and 3.2 we started doing the individual modules. 3.3, individual modules, and 3.5, was the last that was truly AsciiDoc and we moved on to 3.6. So as of now, your current documentation it says 3.6, if you click HTML, it takes you to the Antora documentation site. The only problem that we have with Antora is that it does not have a PDF version. Blake?

>> I will take it from here. One thing we skipped over is I am Blake Graham-Henderson and I work at Mobius. I have been a proud Evergreen community member since 2012 or so. Some of the stuff that Lynn we never saw about the documentation in the history, I don't member or wasn't a part of in 2009 and the trials and tribulations of all of the documentation.

I think we have come a long way as a community in terms of the documentation. We have arrived here at this interface that Lynn is showing, and I just kind of wanted to go over some of the highlights of what we are getting out of this documentation site.

First of all, it might be pretty prominent on your screen, the left-hand side, we have the navigation. The navigation is on every sickle page of every documentation page that you might come to. And it is divided by modules and sort of contiguous with what we have with 3.5 as Lynn was talking about having the things broken out into modules. So they are sort of broken out into modules here on the documentation's eight-page itself. So you don't need to go and choose a module and then browse the module. You have access to all of them all the time here in the navigation.

So what Lynn is showing is also sort of a part of the navigation. You will notice there are little triangles that indicate that there are sub navigations. So if you click on it or the series as a whole, it will give you each individual piece of cereals.

If you click a page, preferably the page having more going on, just go through here until we have one that has a pretty decent table of contents. That only has two. There is one. On any given page you have the title header that mirrors or should mirror close to what the navigation says it is. And that is the heading 0, and there is only one heading 0 per Page. It has the large text at the top. Followed by the table of contents, which is automatically generated for us based on all the subheadings below the page. And sometimes, this is a good example, there is a sub table of contents. So settings overshoot as a sub, subheading data types. And I can go all the way down to 4 or 5 levels, heading levels.

So you can see as many as five indents.

Helpful on the right-hand side we have this thing called content, which is the same thing a table of contents except it floats. It's always floating there and telling you, see how highlights, it might be kind of small on your screen, but it does highlight the section of the documentation that you are viewing currently. As you scroll through the page. Some of you may be familiar with this, but just giving you the overview of the documentation as it is today.

In addition to that, across the top of the page as you can see here, we are in the local administration library settings editor. That's called a breadcrumb trail. So the bread come trail is the root, the document root/module name/page name. So it's another way of knowing where you are all the time. On the left-hand side you will see the navigations bulleted on the page you are on.

And then we get a really neat feature, which is version changing. We didn't specifically pointed out on the previous page, but where all the documentation is listed, starting at 1.6 and going up to 3.6, we have a version number. We have a version number 3.4, click here, 3.5 click here, it's sort of degree to hear. Starting at 3.6, you can switch between 3.6, 3.7, Lynn, are there plates clicking around?

>> Doing dishes.

>> 3.6, we were looking at the 3.6 version of that document and you can switch to 3.7 and see the same exact document, the 3.7 version. By and large between any given version, 3.73.6, there probably not any changes on the particular page, but there might be a difference in the way Evergreen works between 3.7 and 3.6.

Moving forward using this layout, users can change the version of the documentation to match the version they are interested in on any given page they are on.

So 3.6 is the oldest one that we have in this view, followed by 3.7, and then latest -- 3.7 is the latest, but technically there is nearest for 3.7 after 3.7 gets cut. And latest is associated with the word master. You probably hear the word master being thrown around a lot.

In this case we are looking at the absolute last fragment the documentation that was contributed to the code. That's the latest.

There are two places to find the version change. One place is in the bottom left, and there is another place on the top right. You can click that drop-down and then click versions from there as well.

One last feature that we should touch on is the search feature, which is probably going to be used the most. When you're looking at the navigation, you just see a ton of text, tons of stuff, and how do you find anything? You search for it. So if you're looking for hold, you can type and hold in the search. And you will get instant results of any page that has the word old.

One caveat here is that the search phrases are reduced to just the headings. By and large the search results only hit on the headings. Which is good and bad. You don't want to get a whole lot of results for the order hold might have been in the wrong context. These are all the headings that contain the word, hold. You can see a ghost text on the left-hand column, the page on which it appears and then you see the heading inside that page. And of course your word is highlighted period holding's or hold.

So if you are to click on any of these, it would jump you straight to the page. That one is working with hold templates. So that's kind of a bird's eye tour of the we have organized the documentation presentation to the users. That is how that search feature works. It gives you the page name and a ghost text at any of the headings it matters. I would say this documentation we have here is still pretty new and rough around the edges.

Most recently that search feature has been changed after we introduced version 3.7, we suddenly found ourselves with duplicate search results. It was giving a search result for the 3.6 version and a search response for the 3.7 as well. That sort of the shortcoming of this documentation site. There wasn't a way to signify any search results which version you are getting hits on.

So we made the search box only search the latest. It's always searching the latest branch, not the previous. So if you were to use the search feature and you click on any one of these, it should take you to the latest. You'll notice it says latest in the top right corner. So if you actually were interested that weren't interested in the latest and you wanted to see a -- an earlier version, click on that and it goes back to the other version.

This page is a lucky hit. This has a ton going on. This is a good example to show off the table of contents. At the top of every page, you have your table of contents, and this one has a lot of headings and subheadings. Are there any examples of triple, a subheading of a subheading? Thereon. It gets a little unwieldy when you add triples and quadruples for the subheadings. Makes the table of contents though it's organized, left the call point. So if it's a sub idea of a bigger idea which is us a subheading of a bigger idea, you will have multiple indents like that.

So that is my dog and pony show of what we are currently working with in terms of documentation. Lynn is going to talk about the next item on our list. Lynn, you are muted.

>> Okay. The next thing were going to talk about is the Evergreen documentation needs. I will put the link in chat. These are things that we know needs to be documented. And there are lots of them. There is an easy task for new contributors, we can do one of those in a second. And there's everything from easy tasks all the way down to tips and quick starts on document features and changes taken from release notes because there are a lot of those.

And also you can go and see the release notes that they are taken from. And it's supposed to be actually looking at these and you can see that these are actually edited, added to pick if you keep scrolling down after all the release notes, different shared documents are out there, and the other requested documentation. And these are pieces of documentation that have never been documented in Evergreen. A lot of server administration has never been documented. We need to actually go through and document them. We are going to go over patrons with negative balances.

So that is what needs to be documented. Just click on one. If you do not have -- access, you can edit this page and go down if you don't know how to edit a wiki, you can choose it, make sure you make your notes about it it's hard to go through that and know that you're going to document this one.

If you have made a documentation on what more information, we are all here ready to talk now we are going to move on and this is the one we are going to look at it, IDL can amuse, but never defined. So we will go to the details and that information is on there. IDL sense for interface definition language. This is not defined anywhere within Evergreen. So we're going to add this one. But before we would do that, Blake are you going to show git?

>> You can just use your screen, the we have a couple spots here. One thing I think you said is that it wasn't in Evergreen. The IDL isn't document it in the documentation. So everywhere, where the word IDL is, if anyone is confuse, there wouldn't be any documentation to ask point what IDL meant.

>> Because the documentation does have a glossary.

>> The shortcoming here is we don't have a line in the glossary that would spell out the meaning of IDL. So that is a need we need to put into our docs, the definition of IDL in the glossary.

I have here on my spreadsheet, to talk about the folder structure of the documentation actually on your hard drive. I think we are going to go over how to get this on your computer, but for now I think we will just take a look at the folder structure and loop back to how to get this on your hard drive.

Can we blow that up little bit? And let's get the icons bigger.

You may not have seen this before, but these are all the files, the main file structure for our documentation. We are going to go over how you get this on your computer, but this is what you would see once you get on your computer. And you can probably barely make out the word, but it says modules. That is where most of the stuff is. You double-click on that and in modules you'll see a group of folders that basically mimic what you see in the navigation. You have the acquisition, cataloging, circulation.

Each of those contain the documentation for those modules, obviously.

Now, within each of the folders, there are two subfolders pick there is a folder called pages, and that is actually the pages. That is where you actually get to the nitty-gritty. That is where each file is delineated for and usually married by what you see in the navigation.

I don't know if you want to look at circulation on the page real quick, but we should find if you go to the website, under circulation, you should find booking. Says here booking module. That page there is generated from this file over here called booking. A file called booking right there. And she's opening it on chrome, but is just a text file that contains the words of the page, booking module, on the documentation.

What I'm trying to accomplish here in my little segment here, and it is a brief overview of the folder structure. So back to the folder structure we have gone over modules and then your actual module, so circulation is an example, and within that you have pages, a pages folder that contains the pages, and there is another folder called assets. Assets can contain lots of different things, but mostly, 90%, probably 95% of things in the assets are the images. Which is probably one of the hot topics. We will go over that today. Anytime documentation contains a picture of what you are trying to document, you're going to have to put that picture file somewhere.

So the picture file goes separate from the documentation words. And then you call out the picture file in your documentation. And you do that with some basic syntax. Blow that up so we can see that. When you get to the part where you're going to put a picture in, you would type out image: In the path to the file. We are going to go over that more when we actually documents nothing for the first time on camera, life.

We feel so confident that we are going to be able to do this life for you here today. Anyway, I know that's not something you would type in English, but that's all you call out a picture file. This page is rendered on our site, and will say 1, in the staff client, select booking. Can probably find that part in there. So that's what it looks like rendered. It says 1 parentheses, and there's this arrow. That little arrow is actually drawn by having a. And it renders that as an actual arrow.

And then there's the picture right below. That is the image that we get. And show the code for the image again. And the path. Back on the file structure we can show the file we are calling out.

There are a lot of pictures in here in the circulation, the screenshots, I think I'm counting 191 there. So there is the one idea she has highlighted. That is the picture that is being made into that page.

It's not magic. It's just typing out the word for the file and then it puts it on the page for us. That is what I have now Lynn is going to take over. And go through the starring procedure. If you open up the documentation page, I'd like to show off something that I did recently to the documentation. I added, under the introduction, there is a new thing called how to contribute documentation. This is new, so if you haven't seen it before, don't feel bad. It's just been added in the last few weeks to the documentation.

This is taking a stab at getting everybody jumpstarted on contributing documentation. Lowering the bar as low as possible to make it as easy as possible to contribute documentation. So we are going to go through this crib sheet together. And get started on creating documentation.

>> The first thing were going to do is go to GitHub. If you don't have a GitHub account, please sign up for one. I really have one. And this is my GitHub account. I'm moving things around. Give me half a second. So I signed into my account. I have been using this for a while, as you can tell. So the first thing you're going to set up in your GitHub account, then you have to export the Evergreen repository. So it starts with here. This is a copy of the actual -- you want to fork this.

>> It's in a top right corner there.

>> I have to think of that sometimes. So I'm going to fork it and I already have, so --

>> Forking a repository is copying the whole repository into your own account. So you can change it.

>> Once you fork it, I just fixed the current version of it. Because I just forked it again. Once you fork it, you have a copy of it in your GitHub account. Now, you want to install the GitHub desktop software. If you go back to the GitHub --

>> There is a link to it.

>> And I would use the Windows 64 download. It usually only takes a couple seconds to download. Showing folder. There is my download. And at this point you want to sign into getup took GitHub. And you'll see it pulled in my account. It's a simple as that. Now, you have a GitHub account. So here is my GitHub. No the fork I just created I'm going to bring it to my desktop. So I'm going to get the repository from the Internet, click on my Evergreen.

>> You have already cloned it.

>> I have. I'm going to change something. There it goes. So I will create my repository. I will use Evergreen, and Evergreen will be cloned.

>> OneDrive is fighting you, I think. Use the one you have already you must live right next to a highway.

>> I do. So since we already have a clone, and going to use I'll try --

>> Just click on add existing peer there's a button for it on the page there. And then just pick out that one.

>> For those of you out there who haven't done this before, you want have any of those screens.

>> You can see it's refreshing the repository. It's getting all the latest information from the GitHub. And updating all the files. With all the changes that happen since the last time I cloned it.

>> Shutting down OneDrive might help. Sucking up the bandwidth enter microphone is crackling. The OneDrive can be turned off temporarily, I think. Maybe not. I think OneDrive has messed you up. Maybe make a fresh clone. You have to choose the URL or choose it from GitHub. Go to file and clone, choose that and change the directory. That's weird. We could use my screen.

>> I will stop sharing and he can start.

>> Let's see here. I can pull up GitHub. This is what it looks like after it has been cloned. Can everybody see that?

>> It's pretty tiny, at least on my screen.

>> Too small?

>> Very small.

>> In a way, but we were trying to get to we were on the step up down here we need to GitHub, and help we go to repository, and we are installing the GitHub desktop software, and now we are on the set for the Evergreen repository. There were some steps in here that I think she's all reluctant and it shows how to sign into your account. And now we are calling the repository. Clinically repository should be easy. Let's see if I have one I can clone.

>> I'm having a little bit of trouble with the audio, so I wanted to ask the audience if anyone else is having disruption in their audio. Okay. So Blake, you may want to move your microphone closer to you. Because it was okay earlier, or we could try muting whoever isn't talking.

>> Is it choppy?

>> Yes.

>> I figured out what's going on, but I couldn't clone it. I forgot I had signed into the VPN this morning. So I was logged into the VPN network and they are blocking it right now.

>> Okay. We can take a look at that here.

>> I got disconnected from the VPN.

>> Can you hear me?

>> Much better now.

>> Cool. You had an idea of what you are going to document, so you already have the Evergreen repository on your hard drive. You are just showing it. So I would say go from there.

>> All right. So we will go back here. So we go to the modules -- specter you want to share your screen, Lynn?

>> Yeah. Sorry. There we go.

So we go to the modules in the glossary is actually this shared module. We going to pages, and it's there. No, it's not.

>> One of the tricks to finding it is to go to the site and the look at the URL for the site.

>> Find out where I had it. I've lost it. I want to look for glossary.

>> I see the URL there.

>> So go back here under pages, and there is the glossary. So I'm actually going to open this up with -- and this is the glossary. A lot of text on this page. So we go down here to the Is this is just one of the things you can use. You can use these two write your ASCII doc file, and it gives you Visual Studio codes, you actually have a preview of what it looks like. So I will go back and find out what I was going to document.

>> There's the IDL again.

>> There is my IDL so I will copy that. Now, I need to figure out where it needs to go here. It's after ICL, before ILS. So come in here, and I'm actually going to go into -- and putting in the square brackets, the IDL for a link. I can link -- I'm going to put IDL and in parentheses interface definition language, and because this is a definition, two colons because it's a definition and I'm going to go back, and that is the definition. And I put my definition in. It also told us in the bug that the documentation also refers to the IDL as the -- IDL. So I'm going to include that in the definition because if I actually go over here, I don't have IDL over here. To me, that's something that actually needs to be included in the documentation. So IDL is sometimes referred to as the Fieldmapper IDL. IDL this point, would also add -- put a reference here and it actually is a link. What this does is if you look on the left hand side, you have Fieldmapper IDL.

>> Does it work in --

>>The alphabet works.

>> I'm not sure if it's absolutely clear --

>> Something else is screwed up. It is working because you see, this is the reason why it's always good to have a visual representation of it. It actually is popping into the right location, but something is wrong --

>> Does it need another return carriage?

>> That's what it needed. Now, see how they are two different definitions. So the term has to be on one line, the definition on another. So it pops into the right location. So we can save this.

>> There was part of this routine where you would make a branch, but on your computer the desktop doesn't work --

>> I'm actually -- (talking to self).

>> I see what it's doing.

>> It's trying to update it from --

>> Yeah. This is all on your OneDrive cloud and screen the files back down. This is not normal. Whenever you clone the Evergreen repository to your computer, be sure you are not using OneDrive. It will do all the stuff trying to sink and eyes it twice. It just goes onto your hard drive. Lynn just had her computer formatted.

>> You should be able to. If this wasn't refreshing the repository 52 times, you would create a new branch before you make any changes. But it's not giving me the new branch.

>> I think I can probably show it. Have got it over here. My problem earlier was the VPN, too. So it should be better now. So GitHub. I will show my GitHub account, too. Oh, no device verification. Let's switch it to Mobius consortium. All right. There we go. I'm going to do the same thing. Going to go to the -- refer to the documentation on the matter. And it is called -- there is. And going to fork that. I've already forked it.

Note GitHub we can show off again. A new version. Let's see here. Don't you guys love logins itself? This is a good lesson. This is definitely my login here. Maybe it's because it's out of date.

>> Is it dot org?

>> Anyway this is the repository where you would make your branch. This what we are trying to show, right, Lynn?

>> Yeah.

>> In the GitHub software which is being cantankerous today, you want to make a branch for your new change. The reason being is you want to compartmentalize your change pick we just made the change on Lynn's computer. The example was changing the glossary document. And she just changed it right there on her computer, on the file on her computer in the GitHub folder. If it ever gets done centralizing, you would make a branch in GitHub and you do that by selecting the branch icon and to give the branch a name, name it editing glossary IDL or something you actually doing.

>> I can help with my computer if you want me to hijack your presentation.

>> We are desperately wanting to show this step.

>> Daniel, good question. We probably should have mentioned that some of you have seen the Evergreen repository living in that URL. And in fact, that is the authority place that is the central place for all the Evergreen code lives. But the Evergreen community has created a copy of that and a mirror of that mostly for exactly this reason picked to make it easier for folks to see and edit it. Do you follow me? Thank you complicated. This is exactly what we are talking about. So you cloned it, and thus pretend you were going to make, this is actually better because this is an example of what you guys will be doing on your computer.

By default, quick, this is in the documentation, but I'm going to point out that in the top left corner of the screen, it indicates which repository you are currently look at. It says they are, Evergreen.

Next to that is another drop-down menu for the branch. If you click that, you're going to get a huge scary list of all of the branches that have ever existed in Evergreen, and some of those are, like, 13 years old. Tons and tons of branches. And you are unfortunately going to be making another branch whenever you're making your changes to the repository. This is been a little bit disconnected because of these technical difficulties, but if you remember, we made a fork of the Evergreen code repository on GitHub. And as I said before, and I want to get back to what we are talking about, a fork is a copy of the Evergreen repository, but in your own account.

So you have control of your own copy of Evergreen. So you can do things like make branches and stuff. There is a whole security layer here for you cannot do that sort of thing directly to the Evergreen repository, but you can do whatever you want to on your own copy. So that's why you make a fork. You make a fork of it which is your own personal copy and that you're going to make your own personal branch on your own personal copy.

And that is what I'm hoping we are doing here. So we are going to click on new branch, it's what the routine would be for making changes, and you would do this before you make your changes. So we skipped over that and we went straight to editing documentation because of the technical difficulties. But before you add your documentation, you do this tomorrow and you're going to fix all the problems that we have in documentation, so tomorrow on your computer you will go through those steps there, fork the thing, and by the way, this is all written out in the documentation with screenshots and everything to help you get to this place.

You would create a branch for what it is you're doing. In this case, we might as will call it complicated, glossary IDL. Something like that.

A note on branches, you're not allowed to have spaces in them. They usually are going to have underscores. You can do what you do, just jam all the words together and use casing to delineate the words. But underscore is allowed. And so is a hyphen.

Noted that you have your branch named, you're going to create the branch.

>> This is actually helpful because I haven't done this since last fall.

>> We mentioned this GitHub tool, it's because it's less confusing than using the GitHub website which you could do, and I hope that the difficulties haven't scared you away from doing any of this. Should work better on your computer.

What you have done here is you have taken a snapshot of the code as it stands today. And you have branched off. You have frozen in time, so to speak. And so this is the Evergreen code repository as it stands today. And you have to kind of imagine that whatever you do to this folder right now, let's actually do something real quick. Click on white space and just do a new text document and you can just press enter. We just created a new file in here and if you switch back to GitHub, look there. GitHub desktop has found that he made a change. All we did was we created a new anti-file called new text document .txt. Switch back and show it one more time. New text document was a file that you created directly in the Evergreen directory. And this is the reason why we think this is easier to use.

The GitHub desktop software is monitoring that whole folder. So anything you do in this entire folder, change a file, added file, edit picture file, any of those things, will immediately be reflected in this changes think. So you can delete that file now. And if you switch that, it goes away. So it's no longer -- this directory is pristine and unchanged since you created your branch.

If you want to, Lynn actually give you the stuff, you can do what Lynn just did on your screen complicated. If you remember it was modules, appendix, glossary. Pages, glossary. You can edit that with it looks like you're using.

>> I have Zoom on here, and it has been markup colors. But people like a variety of text editors.

>> I recommend, and in the documentation I mention this, there are three that we call out. But for editing AsciiDoc in particular, AsciiDoc FX which I'm sure Lynn is smiling, she got hold of a funny version of it and install the virus or something. I don't know. AsciiDoc FX -- go ahead and Google AsciiDoc FX. Let's see what we get. Actually there is a link to it on the documentation page. This editor shows you the preview on the right-hand side and shows you the documentation on the left-hand side and it gives you a file browser so you can change files and so on. This editor is not too bad.

>> It doesn't handle the live preview.

>> The AsciiDoc FX does. So I would say, and Lynn said earlier, having that live preview is pretty handy.

>> That popped up first.

>> I got this, too. It's just because it's unassigned program. It's a Java program or something.

>> I am not a state employee period may 501(c)(3) employee.

>> So this little program gives you the preview and it gives you the code, generated table of contents. One thing it does that none of the others seem to do is it also put the pictures in the preview.

>> Nice.

>> It gives you a rudimentary preview of what the documentation page would look like. Now, you just have to open the file -- you chose not to put it on the start menu.

>> Associated with the file type.

>> Choose another app period.

>> There we go.

>> There you go. One thing I will say, AsciiDoc FX is a bit slow to open. So if you take my advice and use AsciiDoc FX, it will open slowly. There you go.

Since you opened the glossary AsciiDoc file, it rises to the pages directory right inside the appendix color module. You can press the up arrow button to navigate upwards, see the up arrow button just above that? Yeah. Now, you're in modules and you can see all the different modules and you can browse more if you wanted to.

Pointing out here, the glossary page doesn't have any images on it. But what we are seeing on Katie's screen is the AsciiDoc FX rendering, interpretation of our glossary. If you open up the actual glossary on the final page, in Firefox for example or chrome whatever, thank you. I just wanted to show the audience the pages rendered in our documentation. Glossary at the bottom. There.

>> These are the links in the --

>> What I wanted to show is see the different the difference in font? Evergreen glossary is in black. If you if you go back to AsciiDoc FX look at the rendering over there. It's not exactly the same, it's just trying to do its best, is the point in trying to make. It gives you a rudimentary idea of what it is going to look like on the final page. Does that make sense?

>> It does to me.

>> A little side note on AsciiDoc FX, if you scroll on the side and you click on any area, it should jump to that area on the left. Did you see that happen? It jumped, but that didn't --

>> Highlights some text.

>> I have that same problem. Should be scrolling to the spot on both sides. If you click on brick on the left, it should take you to that spot. Now, go to your new addition. So there she has added the blurb about IDL. It says interface definition language. And beneath it, indented are the words in regular font were says it's a module that shows how data is structured.

And there was another snippet of documentation that Lynn put in chat that says Fieldmapper IDL. And that would go down in the Fs. I always have to sing the alphabet song to remember the letter. I tried to sing at the other day and I kept forgetting letters. It's right at the beginning of the Fs.

>> It takes over as the first F entry. And look at there. You can click it and it should take you -- it was just delayed. So there you go. We just did it. We totally did it. And save, control S you should be able to save that, and you can minimize and go back to GitHub and it should've show that you made a change.

This is in the documentation. The GitHub desktop will spell out all the files on the left-hand side of all the files you have edited. If you click on one of the files, in this case, there is only one, but if you click on it, on the right-hand side it will jump down to the areas of the file that you have made changes. You passed by an area that I want to show.

>> Sorry.

>> This is a very good thing to review before saving your changes pick you might have made an unintentional change and didn't realize it.

>> I did something weird appear expect GitHub desktop is telling you have made a changer that is probably unintended. This is a really great example. I didn't get a good look at what it was talking about.

Here it says the line, it's even highlighting, there used to be a space there or something?

>> The word insertion got inserted into resources.

>> So it did.

>> So now -- Jessica thinks that the F needs to be moved.

>> Across the top, where we inserted the Fieldmapper addition --

>> That listing of links across the top, is looking literally to the F between a bracket.

>> It was on FIFO.

>> Good catch, Jessica. That's absolutely right.

>> So we want to save this, and it's going to --

>> It should have detected that that change is no longer a change.

>> I was looking for this.

>> Break it down a little bit. The line there that is in kind of pink with a , that is the line that was stripped that's the way it used to be. And what is happening here is we are deleting that line and we are replacing it with the following two lines. We are replacing it with the Fieldmapper entry and following that is the FIFO entry. Make sense? And that is GitHub desktop's method of displaying the differences. So pink is out, green is in. Scrolling down a little more, you've added another line, there. That's it. Those are the changes you are making to the file.

So if we are satisfied with this change, there are no other changes, this should address this documentation need.

And what you want to do here is for the people that put the goat -- called back into the Evergreen repository, give them a favor and give them more detail you can give them a discussion. If you don't type anything there, to GitHub desktop will just say update lottery. This is what we were working on.

>> Thank you.

While you are typing that, I want to tell the crowd here that you are typing this in the description. There is also a title. So with every change to a get repository, you're going to supply a title, which is what people would see as the subject line of an email, and the body of the message is down here in the description. If this makes it all the way into Evergreen proper, that stuff will be retained forever. It will be in the list of things that changed in the repository. So all the stuff that you put down there will also be saved into the repository history.

>> , things with documentation. Always make sure you start with doc or docs. Then we know it's documentation, it's a documentation entry.

>> I can assign it to myself.

>> Now, we are getting really advance here.

>> Always make sure that your document start with docs and you have the number in there so they know what the commit is for.

>> And they were they in this context is the corps commanders. The people who review your changes and put it actually into the repository.

That looks really good to me. You will notice that the button that says commit to and it has the name of your branch. So you're going to commit that to your branch. We skipped over the part of this, is this a fork?

>> Yes. This is my fork.

>> Perfect. So you can publish that. Go ahead and do that step. That center change that to GitHub. But here again, where only synchronizing the change to your copy of the repository, not the actual Evergreen official repository per this is your copy. And that's why GitHub desktop gives you one more step, which is the create folder step, if you press that, it will open up GitHub and ask you to log in as you.

>> Okay. You'll see her all that stuff just appeared here and you are all set. It does say on the left-hand side, the base repository is master and your repository is glossary IDL branch spiritual you're all set. You can treat the folder and you just did it. You just contributed to the Evergreen documentation. Give a round of applause.

>> I am sure that they are going to cover this in the session on launch pad, but I want to go ahead and drop in here.

>> Perfect.

>> Make sure if you can include a link to the documentation, do so.

>> Oh, yes.

>> There should be a link you can grab there is.

>> Don't use that link for anything else.

>> That's a really good piece of advice, Lynn. You can move on and get excited and start contributing, and let's do some more. Soon as you start changing that directory and you forget to change your branch, you will suddenly be appending to your glossary IDL branch on the GitHub desktop which will mirror on GitHub and all that. So what you want to do, this is where it's easy to make a mistake. You have to think ahead of time, you need to make your branch first before you change anything. You make your new branch. Knowing ahead of time what your branch is going to contain. So your new branch in this case I think we are going to show off, Patron negative balances. And you can create that branch.

Step 1 was the glossary for you. Editing the glossary was fairly straightforward because it was editing a single document that existed and there were no pictures involved. So it was just editing text. And you already had the glossary to work from, so you can see all the other glossary terms and things and all you needed to do was copy and paste exactly with the syntax there.

Here we are going to create a brand-new piece of documentation that does not exist at all. A new file and at least one image. Thank you so very much complicated.

>> This is the best way to learn, is to do it. So everybody here, just schedule a time to do this with Lynn and Blake.

>> I was going to say this is actually even better. This is a real-world user using a real-world computer and it's that easy. Once this is on your hard drive, you can make changes all day using the GitHub desktop to facilitate synchronizing to the cloud.

I think on my dock I was going to take this opportunity, we have what, 30 minutes left, do you have some questions?

>> I wanted to do know if anybody else had questions.

>> Any questions?

>> I don't see any. There was a question about extensions. I actually have a couple of extensions in my ASCII code that I actually do the preview with AsciiDoc.

>> There is a disconnect between what is being shown on the screen and what Lynn was saying. Clint is talking about an extension for VSCode. VSCode is an editor like AsciiDoc FX. You can edit the stuff in anything -- anything you're comfortable with. VSCode is a very popular option. Although the AsciiDoc preview on the right-hand side which I think is so crucial when you're editing documentation, that preview does not work in the is code on this you install that a software called extensions. And the question was for you, Lynn, have you used VSCode without extension. I can only imagine you go into the extensions interface search for AsciiDoc, you would find it.

>> There are three or four of them. Our VSCode has the AsciiDoc extension, HTML, there is a GitHub extension. You just going to the extensions and you can add whatever extension you want, software extension. There are several that will do preview.

>> We don't want to confuse you for this presentation.

The next question was if you're comfortable with working with git command line get -- that's okay to do for docs, right?

I like that you can quickly click through them. Four-speed reasons.

So now we have a decent idea of git and branching. So for those folks out there, the concept of what is happening in git. Let me do they real quick thing. I want take too much time here. I hope I don't go because a period we will see what happens. I'm going to draw a line. This is Evergreen. That's the Evergreen repository. And it's going this direction. And let's say today we made the glossary IDL branch. And at this point we went downward like this and now we have ourselves our own branch called glossary IDL. And we continue to edit the code this direction. And meanwhile, while you are working, in theory months to go by. Lots of time could go by, and that a lot of time you don't expect everybody to wait for you. Nobody is going to wait for you to get done with whatever you are doing. They are just going to keep on hacking on the Evergreen code up here. So Evergreen keeps going, and more people are contributing to the code and to the documentation. And just like what you saw there in that GitHub desktop differences, these are being done all the time. You're down here on a copy of it from this point in time. Let me draw a circle, from right here, you have a copy of the code from that exact moment in time. So everything down here on glossary IDL for the number of months and weeks that you're editing it, you have the state of the code at that moment. And you don't get any of the changes that occur appear on this upper line.

Later on, when you are ready to go and you publish your results, you publisher changes, you tell everybody in the community that your stuff is ready to go, somebody will come in and look at this line down here, look at what you have done and if everything looks hunky-dory, they will push it back up into the mainline. Do you follow me? This is usually absolutely fine. The only trouble is, when something that you change was also changed by someone else in this time of here. So that's called a merge conflict. And that all gets resolved by the core committees.

So we only made a tiny little change to one little file out in the corner of the documentation that is being edited. So your change will probably get merge back into the master, and then your change will be in the code for every master moving forward. And this process would go on and on again.

It's also possible to branch off of branches, and it basically the same concept. But there is always the core branch, the main branch called master. That's all I wanted to say.

>> My computer finished doing anything you're supposed to do.

>> You want to try the Patron negative balances?

>> Let me share it. So here's my branch. Here are my folders. So I'm going to show you what is document up here log in. Is this thing. Patrons with negative balances. And you go to administration. Patrons with negative balances. So you have a Patron with a negative balance in here. So this is what is that committed. It's a simple page that you can go in and find all your patrons with negative balances. I used to do this regularly before we got some other things like in 1.9 into .0 software you had lots of patrons with negative balances. Things that pointed outward, the system got weird, and you would have hundreds upon hundreds of negative -- patients with negative balances. And $0.20 find that went negative. So I would come in here every so often and fix them up.

Because we don't want to refund that money. So I just going here and fix them. But it would give me a list of patrons with negative balances. So this is what we are going to document.

Because we know where this is, I 90 to go into modules, local administration. That is where I would put it. Pages, and this is on your introduction page right now. If you go to a doctor, we will edit this in a minute. It's an important page. That goes in and shows you what the navigation on the left-hand side is. So if I go into the documentation, I go to local administration, that document is what this shows. So if you want to add documentation you have to add it into the navigation especially if you create new documentation. So I know I need to add back to here. I need to add my page first. So going to go back to pages and I'm going to open a new file program. I'm going to name it negative balances. Yours have to include the adoc.

>> There are standards for the casing. They are all lowercase.

>> So I wrote it out earlier I'm going to copy this and put it in here. I just keep your version -- I don't have preview on.

>> Is probably worth mentioning some of the syntax.

>> I will get to that in a minute. So here's the text that I added in here is the preview. And there are -- this is some of the syntax you can do. This is a big old file I created were basically -- I will fix that later.

>> I think you need a return carriage after the colon.

>> I think I took it out accidentally earlier. Some of the basic stuff is you have the estrus that makes it bold, you can do bold and italics, colored text, this is a nice blue. You can do block text if you want a simple break, edit + at the end of the text. If you have a bunch of short sentences, that send them medically to the next line.

You can also do source code, basic HTML. And this is what HTML will look like. All of this will format as the format within AsciiDoc. This is a piece that is took the action sequences within documentation, it's word, arrow, word, arrow, word. And it shows up just like this. That little arrow translate behind the pencil is behind the AsciiDoc.

Always include alt text in there if you can. I see more and more images, I want to go back to include alt text images.

>> Alt text are for the visually impaired so they can move a mouse over an image that is broken, it will tell you what that was.

That is the image that actually broke.

>> You can do bulleted text, ordered lists, and this is where you will find the glossary terms. The term, the two colons, it has to be on a second line,

This is an anchor, and you go to the top and here's a link to more information, if you click on that link it goes down to the anchor point in the document the same way as we were having that anchor point in the glossary available.

These are two really good websites. This is the official AsciiDoc website here. There is lots of good information here and there is actually a teaching. There are all sorts of ordered lists. It has to have it out in front of it. And you don't actually have to include that -- it will make a nice ordered list. They bulleted list. You have the indentions and everything like that. There are several ways to format these other than what I have here. But these are some of the basics. The action sequencing always has to be in bold. This is the reason I like AsciiDoc. Working with Visual Studio code, because it added those extensions the AsciiDoc preview, and the recycle support, it actually allows me to see this code in the different ways and it goes in the way I have a coded and it allows me to do the preview.

Going back to the code we are committing, there is also a couple of ways that I have patients with negative balances, and that let the -- 0, you can see how I'm accessing patrons of negative balances, level 1. You usually have the bold and that is really the only think -- you can see as we go through here, I had bold and Patron barcode as italics because that's what I wanted to click on. This was short and sweet. And I added a note this is only branches can be selected. Systems and consortiums cannot be. When you view the patrons of negative balances. Because you start there anything else just get everybody in the consortium or the system, you might have to do you have to go branch by branch.

And I have a screenshot in here. So I'm actually going to use my patrons with negative balances. One thing I do with my screenshots is I just do the actual screen to the small she wanted to be. Because I would do this complete screen here. I'm just going to open up and I'm using the screen set that is built-in with Windows 10. And I'm going to copy that section. I can open it up and it's going to open up in the sniffing software. I can profit if I want to. I can save it as whatever I want to say that as. And because this is on my computer drive, I actually go right into here, go to my docs, my module, create a new folder and call it assets.

>> This is going to be unusual. Assets should be pre-created. Expect and images, go back here, including new folder here because we didn't actually have anything that had -- I'm going to open up images and this thing is always screenshots. Now, I got that saved. Now, make sure this actually says -- because I saved this image directly in, and you see I have it right here, I have my assets, images, negative balance, it helps if I can spell, my negative balances, because I stored this directly into the images folder, and nothing media folder, if you save an image directly into the image folder or if you create a folder in the image file if you ever do multiple images, you can create your folder in the images file as the name of the page that you're editing and include all your images in that particular folder. But that's the only image I have for the screen.

So at this point, I save all my changes they should save automatically, but one thing I at like about AsciiDoc is I can click save also it saves them all. And you see how the screenshot is my text here. You can edit that is a screenshot -- and have as long a description as you want in the alt edit. So I say that again, and I open my GitHub, and you see how I have my two files. Because this is a new file, it has no negatives, only positive spirit entity save the negative balances, there is my screenshot. I normally would edit the site of a more. Basically I want to add to my dock, so this is a brand-new doc. And at this point is the same as what we were doing earlier. I need to go to my map. I have committed just to my branch. So nothing has gone further than that. And now I'm going to include this -- the actual pages taken with native -- and this is in local. So I have made that change so it would show up in the navigation. Say that and if I go back to my GitHub you see how it adds that particular one.

>> This is a new concept where you are committing another commit on top of the previous commit. But it's a good example because you can do that. You can do that any number of times because you are committing to your branch and you are stacking changes on top of each other.

But it's all happening in one branch. And you can make as many changes as you want to your branch before you finally publish it and tell anybody about it.

In your entire branch, all of the commits are what is going to be added to the Evergreen master branch.

>> Right. So I'm going to commit back to my branch. You have to make sure you add that navigation piece because otherwise, the doc is not going to fit.

>> It searchable. The search feature will search that document, it just won't navigate there.

>> So now that I have actually committed my changes to my branch, I save my branch.

>> That's okay. We are ready showed that. The idea would be to publish it.

>> I'll get it published in other bit. Because this is something new, I would have to add a -- watch for bugs for the system. The court commit or scan commit this documentation to the master documentation.

>> We are getting closer time. Go complicated.

>> I was going to say, does edit but have any questions? Feel free to put them in the chat. I do want to thank our sponsors one more time, the consortium of Ohio libraries were bringing is the three pre-conferences today, which I really been fun. I have learned a ton. As well as the Evergreen community development initiative for sponsoring this platform and Mobius for bringing us the sponsorship for our captioning today which is been really useful as well.

>> Thank you, Debbie, for making it to the end.

>> There is one other documentation.

>> That is tomorrow at 1:00. It's called building our collage. And she will be covering I think more like the soup to nuts approach of the docs. So people are wanting a little bit more background, that could be really good. And if you want to go and follow along and practice what you learned today, you can do that too.

>> This video will be available to review and the syntax stuff that we were getting too, the debtors and =, there is a reference at the bottom of that page.

>> If you go back to here, I'm going to go to the latest and how to contribute documentation, AsciiDoc resources, with the quick syntax reference guide and the DIG style guide.

>> I refer to that syntax guide recently when I am editing the docs.

>> And when I refer back to is power man's to cheat.

>> And the doctors on Friday. It will be free, definitely sign up and take the opportunity. It's a great opportunity to try things out because people will be around to answer any questions if you get stuck anywhere, you can hop on video and chat with people about it. And everything that I was showing with my setup and stuff I had done in the fall of 2020. So that is a great way to get started and practice and hopefully you will be able to continue on your own.

>> I will be around that morning from 9:00 a.m. to 2:00 p.m.

How do you manage your documents locally? It depends. There are organizations, Evergreen Indiana uses -- we are looking at setting up our own. That's going to be some work, but that's one of the things we are looking at.

I know of other organizations that use a form of online --

>> The question is a little more general here. How do you manage your local documents? Do you clone the Evergreen docs? Are you talking about local documentation about Evergreen question specifically how to use Evergreen? Hopefully the Evergreen documentation can standard on its own as a manual for using Evergreen. But if you are talking about specific documentation that doesn't really -- is not appropriate to have in the Evergreen documents -- yeah. There's a certain amount of documentation, as you get bigger and bigger in consortium, you have to become a little more organized, you will generate your own documents. A lot of people use wiki player, and we are Mobius use AsciiDoc with Antora. We relock that presentation everybody is going to do their own thing.

>> We are seeing a lot of wiki player, Google docs, PDFs. We use Google docs because we have multiple people in our consortium editing. So that's why we use Google Docs. Whatever you are comfortable with is what you should use.

>> One thing I'm hoping that can happen is since the Evergreen documentation is in the repository it is theoretically possible that each -- unit could override documentation and have your own local version of the community documentation inside of the Evergreen staff client. It's something that we are maybe going to walk toward Sunday.

>> That's the overall goal is to get the documentation for the actual product expect so now that it's all been published on the web and can be easily accessible on the web and different versions can be easily accessible on the web, it's just one step further to put the help button in the corner of every interface. That help and could pop out, go straight to the documentation on that interface you are on, and then you can take it another step further and allow individual systems and branches to override the documentation tree.

So if you didn't like are needed to change the documentation for your consortium, and furthermore, just like everything else in Evergreen, you can override the org unit three.

>> That's what we hope will happen.

>> So come on Friday.

>> Come on Friday, and we will do it. We will do it all.

>> Hopefully my computer will cooperate.

>> All right. We are definitely over time now.

>> Thank you, everybody for being here and we will see you tomorrow for our official opening of the keynote of the conference and our keynote speaker. Have a great evening.

>> You, too.

>> Thank you, everybody.