Zendesk Develop User Group

ZCLI & Custom App Demo

Fri, Apr 28, 2023 5:51PM • 48:02

Summary

- Welcome to the second developer user group meeting. 0:01
    - Welcome to the second developer user group meeting.
    - Setting up WSL on a Windows machine.
- How to get started with WSL on a Mac. 2:24
    - Pivoting to housecleaning instead of demonstrating.
    - How to get the xecli environment set up on macOS.
    - How to install WSL first.
    - How to access WSL after installation.
- How to install the latest version of Node.js onto the latest Version of Ubuntu. 7:26
    - Reviewing the documentation through the CLI.
    - How to install the latest version of Node.js on Ubuntu.
    - Checking which versions of Node.js are available.
    - Installing the CLI for Linux.
- How to get started with Zendesk app development using the XE CLI. 12:23
    - XE CLI is now installed.
    - How to access Zendesk apps.
    - Copy and paste the URL to the second desk instance.
    - Choosing an IDE.
- Demonstrating the difference between Zendesk's different environments. 17:25
    - Hello world for the basic Zendesk application.
    - Defining the first screen or iframe.
    - Getting familiar with the HTML for the application.
    - The hello world client.
- What is JSON? How does it help you package data? 22:14
    - X-ray is a beautiful way to package data.
    - The X-ray app.
    - Manifest JSON and assets folder.
    - Basic HTML structure of the app.
- Why is this available in HTML and why isn't it in JavaScript? 27:44

    - The basic connection between JavaScript and HTML.
    - The basic idea of a function.
    - How to make an API call.
    - Request types: GET, POST, and DELETE.
- Why isn't this working? 33:44
    - Making a request to the client.
    - Another function, show info, receiving the information.
    - Creating a template using Handlebars and JavaScript.
    - Creating the HTML template using HTML.
- What's going on in the app. 39:17
    - The main function of the app.
    - The other functions in the JavaScript code.
    - Opening up the floor for questions.
    - A video of the demo is available on...
- Is there anything else that's available? 43:20
    - Keeping the secret safe with OAM.
    - Using OAM for authentication and making API calls.
    - Tony gives time back for questions.
    - Check out the developer documentation on developerzendesk.

SUMMARY KEYWORDS

html, app, json, application, javascript, manifest json, ubuntu, file, function, instance, zendesk, define, information, request, developer, id, installed, user, install, cli

SPEAKERS

Zack Olinger, Rafael Santos

Zack Olinger 00:01
All right. Thanks, everybody, for joining for the second developer user group meeting. Just so you know, I'm on my Windows PC. My cameras right here, and my monitors right here; that's why it looks a little funny. I'll put a space in camera for right now.

Zack Olinger 00:19
Today, we're gonna go over setting up WSL, Windows machine to get that environment set up for ZCLI. Development, we'll go through the simple Hello World app. And time permitting, we'll go through the there's an x ray app, and sorry, an x ray app, and desk pads is a is a demo that shows some information about the agent that's logged in.

Zack Olinger 00:45
Before we begin, there's some housecleaning stuff. Some things just kind of want to go through for anybody that's new to the group for anybody that's watching the video afterwards, just general kind of

maintenance. This since it's a user community, this isn't it send us sponsored, there's no since we're developer community, there's no official Zendesk support for anything that we made develop. We're kind of out here on our own, it's a little bit of the Wild West; I say that in the sense of like, anything that we do here or develop or kind of explored is should be treated as highly experimental; and as is that there's no official support for those things; it's, it's an exploratory thing. And I just want to put that out there as a general, no. Another thing is, is that some of the stuff that I may have that I would like to demo, to that success for you to share that with them, so they can, you know, just ensure stability and security, that's pretty much their main thing. They don't want, like something to be deployed that may jeopardize the stability or security of a pod, or anything like that; reasonable, reasonable things. This stuff today is totally okay, because we're not doing anything, anywhere that that could remotely do anything like that. And then another thing as well, from the previous from my first meeting, I did speak to about some, like an application that I have developed about creating tags for followers of tickets. And that's definitely had some interest. And what I am finding is that certain things I need to navigate, before I can share certain things. And so I'm doing that, and that's part of the reason I've pivoted today, and going through what we're going to go through instead of necessarily demonstrating that particular aspect. Being a creative, I'm super keen, I would love to be able to just share. I personally slightly, you know, I do have my own little frustrations around that; that's my own thing. But I just wanted to put that out there that housecleaning stuff.

Zack Olinger 02:57
We'll go ahead and get started. And for those of you that have Mac's because I know that I asked last time which environment you may have I know not everybody's got a Windows machine, I do not have a video for how to get the ZCLI environment set up on a Mac, although I believe it's more straightforward, I do have a MacBook and it's already set up. I don't know of a way to kind of reset it, put it back so I can record how to set it up with WSL, it's pretty easy, I can uninstall and reinstall it. I'm just saying that because I'm mindful of different developer environments, and I am going to research ways to make that available to demonstrate how to get that environment set up on Mac OS. Just want to put that out there. Without further delay, we'll go ahead and just start getting into the weeds of it. For ZCLI, the, it's a node type of js application, which just means there needs to be a way to serve that, that application of when you're doing it. When you're doing this development locally, on a Windows machine, the Windows Linux subsystem, the WSL is the easiest way to access a way to that type of environment that allows them serving up this information to your Zendesk instance. I do also want to say that, at the sense, that seems a lot of us here are very new to development. I'm gonna go slow, I'm going to try to go slow anyway, and kind of start with the basics. And so if there's anything that feels a little frustrating if you're a developer or if there's things that I'm going too far ahead because I'm taking things for granted, let me know so keep trying to keep it at a pace that's pleasurable for you. The first thing I'm going to do is on my Windows machine, I'll start sharing my screen. I'm just going to go through the process of installing WsL first And all of this information, the links and all of this will be provided to the email and also the community posts. I just want to let people know that as well. Okay.

Zack Olinger 05:12
I'm just gonna go ahead and share the entire screen. Okay, so everybody able to see this, okay? If anybody wants to do a verbal confirmation, that'd be great. Yes. Okay. Awesome. Thank you.

Zack Olinger 05:30
Okay, so basically, this page here tells you how to install WSL. And I'm going to go super quick, we don't need to read all of this, the main command that we care about is this. When it come down here,

and my Start button, I'm going to type in CMD. Right click on this and tell it to run as administrator, you may not see this window that popped up. But it's asking me if I've given permission, I say, Yes, this window comes up. Now, I'll go ahead and type it in. That's all we need to do. Enter. This will take a moment, because it has to download Ubuntu. It's fairly quick, fairly small, depending on your internet connection, but I think probably most of us.

Zack Olinger 06:34
Then after it's installed, it will automatically launch us right into the Ubuntu instance. But I'm going to close that out because I'd like to show you how to access it later on after you've installed it. Here we are, at the prompt my command line of my Ubuntu instance. And like it's saying I'm gonna shut it down, or just close it. Once that's installed, in the future, if you want to access that, you just come down to your Start button, and you can just start typing in Ubuntu should have something like this, just click that. And it will bring up another window, it may take a moment for that prompt to show up because it'll have to boot up the Linux instance. But since I closed it down, it's still fresh or cached, it's pretty quick to boot up, it doesn't take long to come up anyway. But in case you experienced a delay, that's typical. Once this is now that we're inside the Windows Linux subsystem, we're ready to start reviewing the documentation through the CLI. If we come over to Zendesk site, we'll see basically the requirements that we have to have. And so down here says to install ZCLI, we need the following. We need node.js, this version or later. But we also need some of these other items.

Zack Olinger 08:02
The first prerequisite that we're going to have to satisfy as is no three as well. I prefer this, this how to from DigitalOcean. On how to install the latest version of Node js onto the latest version of Ubuntu. Now, they have three different options on this site on this on this particular article. And I personally prefer option number three, which is using the node version manager. And so you can go through and read this, but we're gonna go through step by step, this first one here is just basically I'm gonna go quick on this, this is just showing you how to review the shell script that is calling so that you can verify for yourself that there's nothing malicious happening in it, we don't need to do that I trust this. It's my own instance. I'm going to go for it. This one down here, the second one will actually download and execute the shell script to install the node version manager. I'm gonna go ahead and copy that. And then I'm going to come over to my WsL instance, I'm just going to right click and paste it. And then I'm just going to hit Enter. And we're going to see that NPM is already installed. That's because I've already done this before. Once you do have this installed, you could just do source, SRC. Hit enter, I could go into what each one of these things is doing. If anybody's interested, we could do that. But there's we start to get into a much larger conversation. Once we've done that, we can come back to the page. And then we can see which versions of Node js are available. And since this is the node version manager, like a massive amount of the versions of Node js are available. Just to show you, we copy the command. We'll come back over and we'll execute it There's just a ridiculous amount of versions of Node js. Pretty much what we're looking for. If we come back over to this index documentation, we just need to have a version of Node js that's greater than or equal to this version. We can obviously see we've got versions that are greater than that. And these are okay to use. We come back over to the Digital Ocean, we just basically when we want to install a version of Node js, we just pretty much just tell it node version manager NPM, install the version of which is listed right here. What I did before, on this site just did version 19. That's what I did before. If I wanted to start version 20, this, it's going to go through, it was saying, download the latest version, and install it. Then once I do if I do a list remote, again, it's going to list on that. And then you'll see here that the arrow is on version 20.

Zack Olinger 11:13
There's another way to show the status of that when I thought I'd just show the command. That's pretty much how you install the latest version of Node JS. We've satisfied this requirement, we have our we have our Linux instance set up with no GS. And now what we need to do is install this since we're using an Ubuntu distribution, if you're using a different Linux distribution, we would do these different commands. Just copy this. And again, it's telling me that it's already installed, I have the latest version, because I've done that before. We've satisfied that that's now installed. And the next thing is to install the CLI itself. And so we just copy this command, come over here, paste it in. And it's going to go through and do its thing, download the dependencies and get these installed. It's telling me that there's a new version of the NPM package manager, I'm not going to worry about updating that right now. The xe CLI is now installed. And to verify that we can come over and just type in the CLI. And there's no errors, we have this information now that tells us some of the commands that ZCLI receives. And more or less at this point, we have a very basic setup, where we can start to do Zendesk app development and using the CLI that's truly the fundamental pieces of it. Now at this point, we can go through and do like a simple like hello world in demonstrate how the app shows up inside of Zendesk that what you have to do to access that locally.

Zack Olinger 13:05
I have a couple of demos in a folder already set up. To access my C Drive Linux subsystem, this is where it's going to be. It's not have to necessarily know anything about Linux, a whole lot other than kind of navigating the directory structure, which is the thing a little bit. I do understand that some of this may, for some people may seem a little overwhelming. And I can appreciate that. There's there's pieces involved, I totally understand that you can see from point A to point B, kind of how to get there can be a lot of other steps. Inside of this folder inside of this temp folder, I have two two folders, two applications that I was looking to demo once the Hello World. And the other one is that x ray apps that I was speaking of. To serve this up to start to access these apps that are on my local machine. How do I get that into my Zendesk instance. Using the CLI, what we do is we just type into CLI, we say, apps, colon server. And so basically, this is telling ZCLI to act as an application server. And then we just tell it the folder or the directory where the application lives. And then Linux, the dot forward slash is like the current folder, and then like the next one, so I can do hello world. And that's, that's this current folder, the hello world in this folder. I just run this and then I get this output and the ZCLI will actually tell you what to do. He says add this to the end of your Zendesk URL to load these apps on your Zendesk account. Okay, so I just take this Oops. All right, let's copy that. And then I'm going to come over here to my second desk instance, I'm on a ticket already. And so I'm going to come up here to my Zendesk, my URL, and I'm just going to paste that in the question mark ZCLI, underscore apps equals true. And once I have that in there, just hit enter.

Zack Olinger 15:29
And now what happens is, you can see over here, the apps that I did have loaded with a time tracking app, the REPL, the our approval, all of that stuff is nice, SweetHawk apps, all of those are gone. And I just have that local app that is running on my local machine. And so see kind of the, what's involved in that, how did we get the HelloWorld to display. And so now we can get into the actual structure of the app and the files that make this up and how to do that.

Zack Olinger 16:08
Before we get into the part of the actual code and the files, there, there's a part of development, there's if there's a tool, integrated developer environments, or IDE s, this is a personal thing, for sure. It's a

very, when you're coding, there's definitely a flow, the visualization, the syntax highlighting for the language that you're using, there's considerations, code completion, GitHub integration, there's a lot of things to consider when choosing an IDE. Right now, if you're a new beginner, it really doesn't get that complicated. It's more of what feels right. That being said, the two that I use, the one that I started off with was notepad plus plus on a Windows machine, on a Mac. TextMate is the IDE that I that I use that I started off with. And then on both platforms on Mac OS, and Windows, I've moved to Microsoft VS code, not Visual Studio, not their full, not their bigger package, but VS code. And I enjoy that because it has some other integrations and some other apps that are embedded into that environment that are helpful that that I enjoy. I'll demo this, so showcase the different like the difference between the two.

Zack Olinger 17:33
Come over here. For the Hello World. For the basic Zendesk application like this, there's going to be a real general structure, you're going to have a readme markup file, the manifest json file, to the translations folder, and an assets folder. The manifest json, if we were to open this up, looks like this. And then I'll show you the difference between the two environments as well. This is the same manifest json, this is notepad plus plus over here. And under this is VS code over here.

Zack Olinger 18:18
This is, again, whatever feels right. And again, on the Mac OS side vs VS code is going to look exactly the same, it's going to feel exactly the same. TextMate is obviously a different product. It's one that I like as a text editor. The manifest json is the real basic structure of the application is going to define the first like screen or the iframe, the HTML that makes up the first interface of viewer of the app, it's going to define the location of the app, it's going to define whether or not the app runs in the background. It's basically defined the description, any variables that you may have, that you need to have captured for using inside of the application. So unlike the admin panel, where there's fields that you might fill out, you could define these manifest json. So this one is real simple. It doesn't have any, anything too crazy has the basic stuff. It has what product ID of Zendesk where the application is for where inside of support, it's meant to be ticket sidebar, and here is the URL or the path that assets folder right here. So assets iframes. Inside the assets folder, there is an iframe file. This is that HTML there. Now, there's these other files here too, which are used by the HTML file. So that's why it's all that's bundled inside the assets folder. The main one for this app is maintenance as far as functionality goes is going to be the iframe and so on. We'd open this up, I'm going to go ahead and actually just drag it here. And this is what the HTML looks like.

Zack Olinger 20:11
With the, I will say this, as you're probably becoming aware, as we go through this, that to do some of this development, there's some navigation of Linux will be involved, some knowledge of HTML will be needed. Depending on how far you get JavaScript is going to be needed for these applications. And so there are the things to get familiar with, as we go through. Start to explore these things.

Zack Olinger 20:39
And as you can see, right now, just in this JavaScript, we do have the script heading here. And this right here, I don't think this client is actually used, because we're pretty much just printing this Hello World. Although we do have the script here, this is how we start to initiate that connection to the Zendesk environment where we can start to pull information about users, tickets, and things like that. This is the very, very beginning of that type of connection, even though we're not doing anything with it.

Zack Olinger 21:14
So let's see what else I wanted to. This is the reason it says Hello, World Two, if we come back, it says hello world right here. This is from the HTML, the Hello World Two is actually coming from, from the manifest json. So this is showing us the name of the application. So this is the name of the app. And this is what the HTML is generating. So just to be aware of what what that's about.

Zack Olinger 21:48
And for those of you that may have come from Michael's group, when we were talking about ZIS, we talked about JSON. This is an example of what that looks like. We won't necessarily go through all of that here. But it's basically a way of having key value pairs like, here's the name, which is key, here's the value, here's author. And then we have things about the author's sometimes we have to have multiple elements, right? Here's the name of emails, here's a key, here's a value, another key other value. And it's a way to structure this information. That way you can take it from one system to another. JSON is a beautiful way to package data so that we can move in from one application such as in desk into another maybe Asset Management System, or mail platform, or distribute, I don't know, whatever it is, it's a good way of packaging, packaging, and transferring information from one system to the next. I think that at this point, I'm gonna go ahead and move into the X ray app to show a little bit more of the JavaScript side and the connection that we that I was just kind of talking about connecting to the Zendesk environment, since that's primarily what we're looking to do, right.

Zack Olinger 23:01
Then I'm going to come over to my WSL instance. And I'm going to ask you to stop serving up the Hello World, which is basically hitting Ctrl C on the keyboard. And then I'll stop. Basically, what I'll just do is tell it to run X or the X ray. Ideal, hit enter, and now it's serving up that app, all I need to do is come back over to my Zendesk instance, and literally just hit the refresh button. And this hello, world will go away, and the X ray app will be loaded up.

Zack Olinger 23:45
Okay, and so basically, the requests are X ray is, this is giving us some information about the requester has the name, the date, the requester was added and the last time that they signed in. This is all information that we need to know that's available on the user profile. And that we can access. Sorry. This is an example of how this application is making connection towards Zendesk instance and pulling information. How is it doing this, and this is where we get into the JavaScript side of things. This is definitely going to be beyond just HTML, we're going to start to need to make calls and format the data and present that information. If we want to come over to the files, come over to the X ray app. Again, we have the manifest json for this one. If we take a look at it, it's very much the same. If not, it's the only thing that's different our version numbers, but this is basically the exact same JSON that I have for. For mine. It's Is that the names have changed, but otherwise is the exact same. This, we noticed the same, you know, the apps not doing anything too different than what we were using before, at least as far as structure wise, where it's running inside as a desk and that type of thing. But if we look at the assets folder, we do have, we have a couple more items listed, we have our iframe HTML, which is this box over here. Then we still have our, you know, PNGs, which is for the logos. And but we also have the main css, a cascading stylesheet, and JavaScript file. These are new, my other app didn't have these. Let's take a look at the HTML. And this is obviously more involved in a way. What we're looking at is, first we have the head pulling in the the CSS that's inside here, inside the assets. And then in the body, we have the basic table, or the div structure for the table to create this. And we have some, to me, this looks like liquid markup. If anybody has corrected me on this, let me know because I haven't actually

employed this type of approach in my HTML. We can see that, in my mind, there's some liquid markup here, this is an if statement, if there are tags to display them on the user profile. And if there's not, do not display this, we don't see any tags over here. But there's other information and we just see that this, this is the variable that it gets populated. And down here at the bottom, we have some more JavaScript libraries that we're pulling in. This is the basic HTML structure. And then what is going to happen here with me, let's take a look at the JavaScript. See what's going on.

Zack Olinger 27:20
Okay, so here, we have our first connection to our Zendesk client can remember from that other HTML, the Hello World.

Zack Olinger 27:39
Hello, World One, we have this, even though we weren't doing anything with it, I was missing. This is the basic connection. Over here, in this JavaScript, and the reason might, if it's confusing the HTML that is living inside of the script, block inside the HTML, you can move this into just a straight up JavaScript file, like it is here, call it or have it available just by including that JavaScript file into your HTML.

Zack Olinger 28:09
A case that, for those of you that are starting out with coding is JavaScript is how you can that may have been confusing to me is starting out today. I just want to call that out that why is that available in HTML? And why isn't showing up in JavaScript? And how does that work? The reason it can live here that HTML is because we're defining it as a script, that over here, we're explicitly defining it inside of a JavaScript file. And then we're calling it that JavaScript file. One way is putting it into a file. That's explicitly JavaScript, calling that into the HTML, the other ones that embedding it directly into the HTML. Having things inside of their own files is easier for code maintenance. It does make more sense, in my opinion, to keep that separate JavaScript. If it's something super simple, then this may make sense. But anyway, just best practice kind of thing or an idea. Back to the main js, we can see that now that we've attached to, we've initiated a client connection to artisan desk instance, which allows us to start accessing different functions or information from our Zendesk instance, we can call functions to get information.

Zack Olinger 29:20
Before I go further, if you don't know what a function is, it's a block of code inside of a inside of your application or program that is for a specific function for, you know, doing a specific task. We'll say like, if I needed to make an API call to get the user's name, when I make an API call, they can all to get a user. I'm gonna get a chunk of JSON back, you know, like one of these. We're gonna get some information that looks similar to this, right. Really, I just, I just want this I just want to Do I, when I call that function, I just wanted to give me just that. The function would basically take something and do something with it and return the final output that you need. If I were to write that function for getting the name, I would pass it this information, this JSON. And then I would define how to extract that information out and return the name. Now that that may sound kind of difficult. The nice thing about JSON is that that's actually been easy. But that's the basic idea of what a function is, is it's meant to take an input and return something specific as an output.

Zack Olinger 30:40

Here, we have like this, right? This right here, we're calling a function of getting the ticket requestor ID. And then we are assigning this ID to a variable to find here. And then there is a function where we pass the client connection and the user ID, you can see this right here, this request user function, here's a function that's defined. Somebody wrote this function to request user information, it takes in as input, the connection to the client, and the ID of the requester, then it's going to make an API call. And so it's creating a variable called settings, which is basically the parameters of the of the request of the HTTP request. You can see the URL, it's making a, this is a string, right here, this is what's called the string. And it's basically a string as text in programming languages. And what it's doing right here is it's doing string concatenation. What that just means it's putting it's inserting in the ID, that was passed to the function into the string. That way, this string is very dynamic, whatever ID gets passed to this function gets inserted into the string. And the reason it's doing that is because that way, you can pass it in the user ID, and it will make the API call and it's going to get that JSON for that specific user.

Zack Olinger 32:16
Down here, we're typing, we're telling it the type of request that we're that we're doing, when you're making an API call, there's a number of different request types, the most common ones are going to be get, put, post, and probably delete, the ones that are going to be the three that are going to be get posted, put are probably going to be the biggest ones get is read only, you're just receiving information, posting. And putting information, you're either creating information that's new, or you're updating information that exists. So it's going to be an overwrite or right operation involved there. Delete is pretty straightforward, you're getting rid of something. So get operations or get requests are going to be totally safe, you won't have to worry about harming anything. So I just want to put that out there. So we're reading in this scenario, we're calling and we're making a request to read it.

Zack Olinger 33:13
To get the user information by the users ID, or the requesters ID sorry. And then the datatype is, is JSON, which means this is how we're packaging the information to transfer. And so that way the server knows how to interpret what it's receiving, or in a get request, that's less of a thing. But more when you're doing a put in a post, the server needs to understand what it is you're sending it. So defining the datatype is very important on foot post operations. That has caused me some trouble, like why isn't this working? Once we have created this variable? For the settings, we're going to make a request. So we do we take the client connection, we tell it that we're making a request, and we're passing it these settings. So we're making this type of request to the client, we're going to say is it as a client, we need to go to this URL, this type of request for this user ID. So it makes makes that call. And then there is inside of here inside of this request in JavaScript is I don't want to take anybody's opinion. It's a it's a little different beast than a lot of other programming languages.

Zack Olinger 34:30
So if this seems a little confusing, it can be confusing to me too. It's kind of hard to share the information. So you can see down here that there's another function that was defined. So we see show info down here is the function for show info. What what show info is doing is it's receiving the information that JSON because we're making this client request, right? We're taking this user ID we're making this call synced. to client, we need to ask for this information. So then we receive that information, we're going to call this function, which is this down here, we're going to send that function, what we got back by making this call, which is going to be a JSON, that looks something like this, it will look like this, because it's going to be structured like this. And so then this function is going to take this information, take this type of JSON, right? And it's going to say, Hey, I'm going to create my own little

package of data structure, I'm going to have a key, that's called name. And the value of that is going to be data, user name.

Zack Olinger 35:44
And the way that the way that you can access elements like so to speak, like, like, if we legitimately use this JSON, we wouldn't take out the user part and just do name, data dot name, right. This can be a little confusing. And I may not be able to articulate this very well, because I feel like this. Understanding why this works, it requires an understanding of the relationship between key value pairs, how they exist inside of a JSON file. This, understanding why this is that way, or named this way, is going to require an understanding of how JSON is structured. Just to let you know, we're taking, we're taking that information, this is the value for this key. And we're doing this for all of the other things, right, the name the tags created at the login, last logged in. Now that we've pulled this information from that JSON that we got from this request. Now we're down here, we've created our own little data structure.

Zack Olinger 36:54
We're going to come down here now. And we're going to look at the HTML. And so we're now in the JavaScript where you now that we've made a connection to the client, we've grabbed all the data, we've created a data package, we want to display that data. Now we're going into the HTML and we're finding this requester template ID, if we come over to the HTML, we can find that we have this right here is so what this is going to do, this is where this table lives is inside of this script ID. This JavaScript is going to find that HTML ID. And then it's basically going to create the template of that using handlebars. This is a built in JavaScript HTML template, where you can take it and then insert elements into the HTML posted back. It's creating a template of HTML. And what it's doing then is passing along this data structure that we have. And then what it's going to do is it's going to get the content. There's a, there's an element or an ID inside of the HTML called content. Should be Yeah, like right here. And so basically, what that is doing is it's, it's reading in this table that's defined right here, and it's creating a template of that. And then basically, these tags right here are what get populated right here.

Zack Olinger 38:42
And that's what this is doing. We're pulling in this for the template. That's why it's pulling in a template, the template is basically this. Once it has the template right here, we're going to insert our requested data into that template. And then we're going to write that updated template with this with this data back into this div here, which is how we're able to get this information populate. It seems it's just a real high level, like, what's going on.

Zack Olinger 39:21
And I may have kind of gone over some of that in a way that's not entirely accurate. But that's basically what's happened. And that's the main function of, of this app. The rest of what we see down here is if for some reason, we have an error, and we have a show error function. There's that. If we did have an error instead of seeing these items. We would see something about status and status text. And it would it's just doing the same thing. It's creating a data structure. It's grabbing the div in the HTML over the air, a template is just right here. And it's basically populating these two variables in the HTML with these two, just like it was doing up here. But inside of this to this ID and then the other. The other, the only other function inside of the JavaScript is this function that formats the date that's used up here. And this is basically to make this more human readable. And so she's taking that, that date from the user profile and converting it into what we what we see right here. And that is the gist of that. of that application.

Zack Olinger 40:43
It's basically just connecting to the Zendesk client, it's making an API call for the requester. By the requesters ID, it's receiving back the JSON response like this, it's then processing that JSON response, and then packaging that and updating the HTML, so that it can create the table that gets populated and displayed in our user in our agent user interface. Then the other, the other rest of the code is just handling any errors, and formatting a date. And again, down here, you can see that this function actually returns something up here, these two don't, they just automatically, they just perform a task. But they don't return anything, the format date, which is up here, we needed to return something we're calling this function, and we need something back. So the format date actually returns the date. And so you can see the date variable is, you know, all of this, that's basically this function is receiving some data, it's receiving input, and then it's doing something to that input so that I can return the desired output. This is a great example of what I was explaining earlier.

Zack Olinger 42:00
There is that, and that actually went a lot faster than I anticipated. I'm going to stop sharing. I am totally open to questions. Or if there's anything that I need to go back over anything on the fence. Let me please look into it. I'm opening up the floor.

Rafael Santos 42:32
Thanks, like I went on a beat off a spam streak on the chat. If anyone has any questions, please ask away either voice or chat.

Zack Olinger 42:40
Awesome. Yes. Thank you. This is great. Thank you for that.

Zack Olinger 42:57
Is there? Is there any interest? I did do? Michael had asked me in the User Group slack if I if there was anything to do to be prepared before this. And I had sent through a video I had gotten through that very beginning part of getting WSL setups. There is on the YouTube channel a video of that, because I didn't send it out because I needed to pivot for this demo today. I just included in what, what we do here. Is there I'm just curious if there's anything else like, like I say, I'll put some together for the MacOS side. Just so that's, that's available. Is there anything else interesting?

Zack Olinger 43:40
Okay, yeah, so Jonathan's question about keeping the secret safe and things like that. This is pretty interesting. Raphael and Ahmed both really helped to be here recently around OAuth connectivity and authentication and making API calls. That took me a minute to get my head wrapped around. To answer this question, though, what I personally do is have a YAML file. For for for apps like this, and we back up a little bit for apps like what we're doing here, you're gonna see my screen. For internal apps, like we were just demonstrating, that's actually using the authentication of the agent that's logged in. We don't have to do any type of credentials for those types of calls, which makes that really awesome. If we're doing outside of the system type of calls, like I've got my Python scripts that I run, and that makes API calls. If there's something like that that's not running inside of this desk instance. In that scenario, I use a YAML file, and I store my API key or bearer token inside the YAML file. And then I just access that as a variable I read in the YAML file and the Python, just access it that way. And then I have my YAML file set up by instance. That way, I can just define the instance inside of my, my Python,

and then it pulls in all of that relevant config information. And once the scripts that way. That's how I've handled that.

Zack Olinger 45:06
See, um, see if the user group slack that I think I don't know if I have any uses for this group leaders. I'm not sure if there was just for us, you're just like a big group. That would be pretty cool. But as far as I'm aware, and psyche. Okay, cool. All right, awesome. Well, I mean, I'm cool to hang out. But I'm also cool to give you some time back. Like I said, this, this went way faster than I thought. But if there's no questions, I don't have to necessarily keep you guys either. But I'm totally good with anybody has. Thanks for being here. Thanks. Thank you guys for your support your time. Thank you. Thanks. Thank you. Thanks, Tony.

46:00
I wanted to mention something, feel free to go through the developer documentation on developer.zendesk.com. And if you have any questions, reach out either on the community forums topic for developers, or even on the user groups topic. For developer were the things where you can ask us any questions and first can guarantee a quick response, but like we and the rest of the community will try to get back to you and get you a solution. For sure.

Zack Olinger 46:36
Gotcha, cool.

Zack Olinger 46:43
Well, it's good. Yeah, I'm excited about that. We Raphael and I were talking about some other stuff to kind of go through for the next meeting, maybe some other apps that are out there. I haven't necessarily quite decided on what that's gonna look like. If you guys have any input, feel free to respond or even post on the community forum, that I create this for this meeting. will kind of kind of go from there. Yeah, but we'll do definitely support and showcasing things but yeah, but I want to have accessible so just feel free to communicate if anything, is striking. I believe I believe there's a way to email me through the use of groups. That's why I say that. Okay. Yeah, absolutely. Cool. Yeah. Well, then I'm gonna give you guys 10 minutes back, definitely enjoy your weekend. Thanks for Thanks for being here. And thanks for hanging out. And thanks for all the help. Alright, I'll see you guys. It'll probably be at the end of May. Like a monthly cadence but also that a lot. Alright, thanks again guys.