Neon AI SDK Installation

In today's video, we're going to look at installing the Neon SDK on an Ubuntu machine and then giving it a quick test to make sure our installation went smoothly. So let's get started.

Checking Hardware Configuration

Okay, we're going to continue our Neon setup here. So we've just gotten our account credentials all generated, and we're going to move on to hardware. We're going to go to our computer, make sure it's running Ubuntu 18.04, and we're going to make sure we have audio devices. So, if we go to this top right corner, go down to settings, sound, and our output device – our speakers, we're going to test those and make sure they're the correct ones. And we get audio so that's right, go over to input and make sure we have the right microphone, and we do see some activity here, so we know that's the right microphone. Close that out. A webcam is optional. Skills like the USB cam skill require a webcam, and then most webcams will also give you a microphone, so you could use that for input. The last item on our list is an active internet connection, which if we go up here, wired and connected. So, we're all ready to install Neon.

Developer vs User Install

So there are two ways to install Neon, you can either install it in a development environment, or a user environment. The development environment is designed to be connected to an IDE like PyCharm, so you can modify your core and skills, or create new skills. It also defaults to some more logging. A user environment is designed to be installed on something like a voice assistant device, it's a device that you would use for Neon, not so much for making changes, but for actually using the skills. It's also useful to have a user environment setup so you can test any changes you make, test the update process as you make changes in your developer environment. So, as I said, we've already setup our service accounts.

Checking for Credential Files

We're going to go open up our home directory and make sure we have our neonSetup file, as well as our access keys from Amazon, our google JSON file, and our wolfram keys.

Running neonSetup.sh

Go ahead and open the terminal here. Type in bash neonSetup.sh, press enter and yes to developer mode. I'm gonna say no for quick settings, just so we can go through them here. If you've made any changes to neonSetup.sh that you want to make for the git repository you're pulling from and your default settings, you could say yes here. So moving along, we're going to install from gits, we're not going to use the default repository, just so we get this pop up to make sure, and this is the correct core repository and branch, and this is the correct skills repository and branch. It's going to output those to the terminal here, if we made a mistake here, or if you want to change to a different repository or branch, you can go back at this point. It will bring you back to the last

question, where you could say no again to make changes. In this case I'm gonna say yes, because I know they're correct. Now back to auto run we're gonna say no for the development environment, and updates we're also going to say no to automatic updates. On a user device, these two settings would probably be yes, so you can turn on the device, automatically have neon ready to go, and get updates as they become available. Default install locations for developers are the subdirectory NeonAI, relative to neonSetup.sh. So, neonSetup.sh here is in our home directory, the default would be home/neonAI. I'm going to say yes to that. OpenHAB is a piece of open-source software used to control smart home devices, like lights, outlets, thermostats. In the development environment, it's useful to have to test any changes you make to the skill, make sure it still communicates properly. I'm going to say yes to installing that here. If you want to learn more about OpenHAB, you'll see in our instructions a link to their website. It's gonna ask if we want to install mimic for offline text-to-speech; this can be useful, so if you lose an internet connection and you're typing in commands, or if you're running some kind of local speech to text, you can still get responses. It takes a while to build, so I'm going to say no for this tutorial. Now it's going to ask us to confirm our installation settings. So we are installing from git, in developer mode, we're not running neon at login, we're not getting updates automatically, we are installing OpenHAB, we're installing to this home/neon/NeonAI directory. Got our git repository and branch, skills repository and branch. All of that looks good, so I'm going to say yes and enter my sudo password.

Installing Packages and downloading Neon

So because I've set this up on this machine before, it did not ask to connect to the Neon Gecko server; the first time you install it will ask you to set up SSH keys. These are required for getting brands and coupons updates, as well as sending diagnostic emails from your device and transcripts. So now it's asking to sign into github, and this is to download our core. Sign in there. Now that we have the core downloaded, will see our status window pop up, and we're going to follow that through the rest of setup. I'm going to sign in one more time to get skills, and now we'll see here we're just going to follow our status window. Completing a new setup, we got things from git, we've downloaded our skills sub modules from git, able to source our functions, and right now it's adding our neon core to our Python virtual environment path. If this is the first time you're installing, you probably won't have all of these packages cached as I do, so this might take a little bit longer; but it's just installing all of the Python dependencies for Neon.

Validating Credentials

And we'll see here, our virtual environment is ready, our Wolfram, Google, and Amazon credentials were all validated. If we ran into an issue with any of these, we would have seen a FAIL, and we also would have gotten a window that popped up prompting us to correct the credentials or find them if they weren't there. Note that the Google credentials are the only ones that are explicitly required here, so that we get our continuous speech-to-text, our transcriptions,

and our non-english speech recognition. The Amazon credentials are required for translations, non-english speech input, and output. You could get away with using mimic, or a different speech-to-text engine, if you didn't care about translations. And then lastly our WolframAlpha credentials, not explicitly required, but very useful so that you can get general inquiry responses like weather, stock prices, math questions, other general knowledge questions.

Neon AI First Run

And as we heard a moment ago, Neon is starting up. We'll see every time it starts up, the date and time logged, along with some information about the current state. So, version, auto start, developer, update sources, number of skills we have, it will log if had any issues updating our brands and coupons from the Neon Gecko server. And it will count down as we've heard. We'll see all of our skills loaded successfully and that Bell indicates that startup is completed and it is ready to go.

Quick Test of Neon AI

As you can see, it's already started transcribing what I'm saying. Just to give it a quick test:

what time is it? (It's 1:58)

what is the derivative of x squared? (the derivative with respect to x of x squared is 2 times x. Provided by Wolfram Alpha).

So as you can see, we validated all of our credentials again here. If Google wasn't working, we wouldn't get our transcriptions here, if Amazon wasn't working we wouldn't get the responses read back, and if Wolfram wasn't working we wouldn't have gotten an answer to that question we asked here. So with that, we have our setup completed.

Check out neongecko.com

Thanks for watching this Neon AI tutorial. Be sure to head over to neongecko.com for more information, including a written transcript of this tutorial and any code snippets we may have used. See you next time.