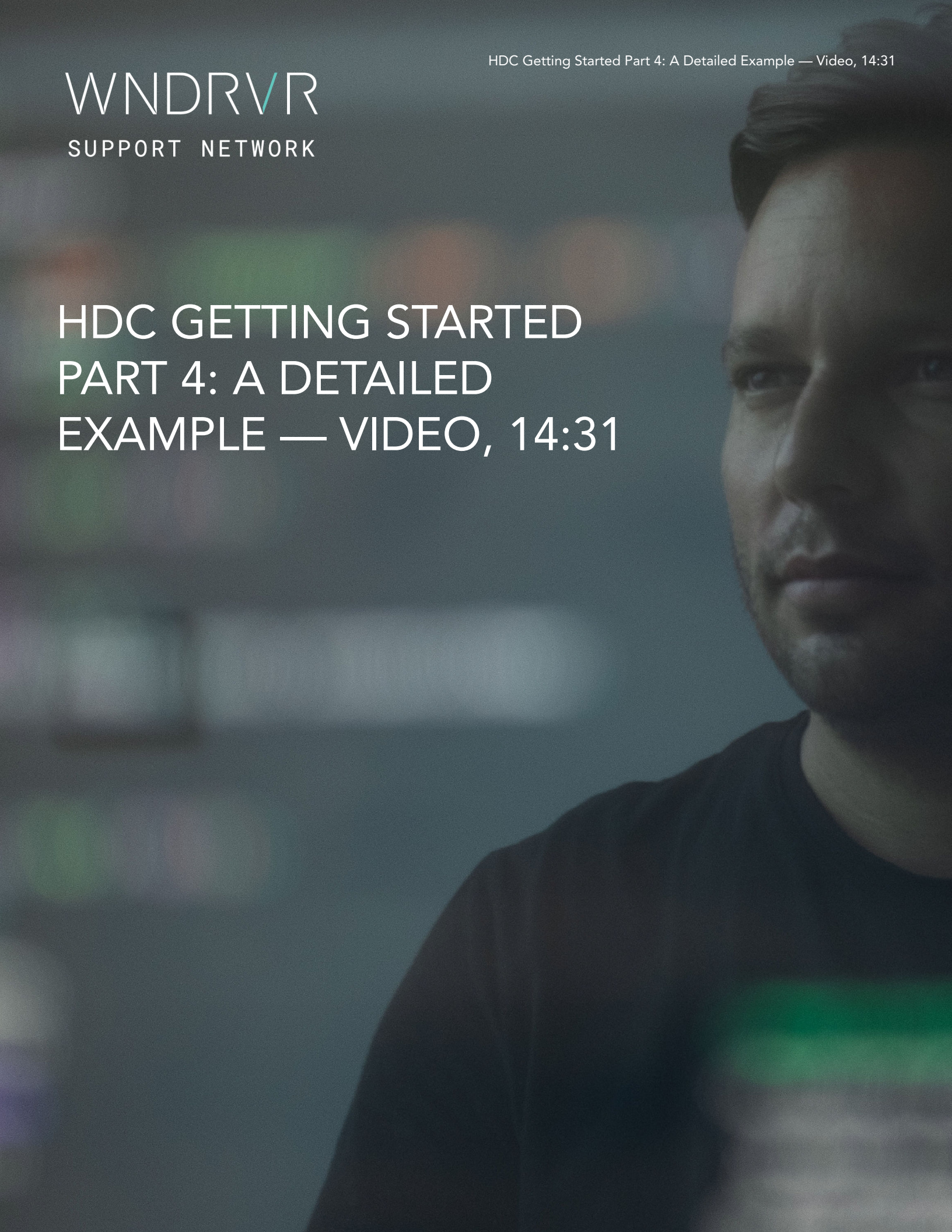
WNDRVR
SUPPORT NETWORK

HDC GETTING STARTED PART 4: A DETAILED EXAMPLE — VIDEO, 14:31



Corporate Headquarters

Wind River
500 Wind River Way
Alameda, CA 94501-1153
U.S.A.
Toll free (U.S.A.): +1-800-545-WIND
Telephone: +1-510-748-4100
Facsimile: +1-510-749-2010

For additional contact information, see the Wind River website:

www.windriver.com

For information on how to contact Customer Support, see:

www.windriver.com/support

HDC Getting Started Part 4: A Detailed Example — Video, 14:31

29 January 2019

1. HDC GETTING STARTED PART 4: A DETAILED EXAMPLE — VIDEO, 14:31

Published on 6 October 2015

Transcript

Time (mm:ss)	Narration
From: 00:12	Ok, in this section, we're going to actually work through an example of preparing a package, uploading that package into the Helix Device Cloud, then defining some deployment instructions for that package. We're then going to deploy that package to our target, and then we're going to test and verify that it actually got deployed correctly. And so, let's walk through this example. If you go into the VM environment, then you'll notice that I'm in this blds/code/ lua_setup directory, and I want to take a first, a quick look at some of the code that's in this, in this directory. And so if we go down into this lua-tools directory, you're going to see that there's these are the scripts and, of course, these have already been loaded onto the target, but we're going to walk
From: 01:01	through the example of seeing what I did to prepare this as a package to be downloaded. And so I want to show you, how this actually works, and you'll notice that there's an install script, and in the install script, you can see it's really just one line of code to copy some files, but it needs to run those with root permission, and so we're going to see how we can actually invoke this script through the platform and have the files get installed correctly into the proper directories into the lua path, so that we can then later run these lua scripts. So, the first thing that we have to do, and of course I've already created the lua tools versioned tarball, but simply creating that tarball is just done like this, and then you specify the path for that.
From: 02:02	Ok, and it actually just puts all those files into a tarball, and you can see that there's not really a whole lot there. I did include the lua files, the shared libraries that need to be installed into the lua path. Then there is the install script, and then there's a few lua scripts that we will use later. Ok, so now if we go up into the platform, and now, if we take a look here at our asset, and you can see that our asset is still running. I rebooted it and we're going to try installing this package now. And so, what we'll do is we'll come over here into the Content tab, and in the Content tab, we're going to specify that we want to "Create a new software package", and I'm just going to call this the "lua-tools" package, and it's only intended for a certain model type,
From: 03:02	and so this is the model type of what's running in our virtual machine here, and then I'm going to give it a version number and since I name the tarball 1:0:0:0, I'm simply going to make the version number match here as well. So then, I'll come down and just simply go through. We're going to accept some of the defaults here, I don't have any dependencies, and then I'm going to come to the next section where I need to define some instructions for this package, what do I want to do with this package, as part of the deployment process. Well, the first step I want to do is I want to download the file from the server into the remote target, and so here I need to specify the file that I want to deploy, and so I'm going to upload it onto the server, so I'm going to select it from my local file system. And then I'm going to specify to upload. Ok, once I've uploaded it, I'm going to come down here and just determine some
From: 04:04	other conditions that I might want to set. So, in this case this is a file - It's a tar.gz archive - and I want to unpack it in the directory, so I'm going to select that one, and then be, if it were a version two, three, four, whatever, you might want to also specify to overwrite in the existing files, and so you could turn that one on as well, and the other options that may or may not apply, but it's not an executable, we don't want to run it at this time. So this is the first instruction, just, we've uploaded the package to the cloud platform, and then I want to define

	this instruction to extract that package when it's downloaded into the target. Then my next instruction, simply unpacking a tarball doesn't really invoke any action, so I want to now tell it to execute an application. And in this, in this case, I want to run the install script, and I want to run it
From: 05:04	with root permission, so I, I have a little utility that I've created called sudoit, and this was actually installed as part of the platform build, and so it's already in the /usr/bin directory. But, the install script, when it's unpacked from the tarball, does not have execute permissions so I'm simply going to execute it like this. Ok, and I've specified the full path that I want to execute it, and this is an application that is already registered with the agents so I can use this because it's a registered application with the agent. I'm going to choose not to select any of these other options for now, and I just simply want to say my first instruction is to, after extracting the tarball, I just want to run the install script. And then I'm going to actually define a second instruction, that I actually want to run one of the lua scripts,
From: 06:02	and so I'm going to come down and I'm going to specify, in this case, just do it, because I do not need root permission here. I want this one to run as wrauser, and so, and my arguments here are simply going to be run the lua script, and I'm going to run the "registerapp" lua script, and we'll take a look at that script in a moment, but what it does is it actually registers some additional applications or functions with the agent running in the target, and this can actually run asynchronously, and I'm going to go ahead and select that, because I want to leave it running continuously so I'm going to define that as my action for this one as well, and I'm going to add that instruction to my package as well. Alright, so now I've defined several instructions. You can see down here is
From: 07:00	the list, and if I needed to rearrange the order, or remove them, I can modify the list later. But I'm just going to go ahead and say finish here. I want to complete the instructions, and I have nothing else to do so I just want to go ahead and say "Finish". Alright, so now I have defined a package. Now, I can choose to define rules for when to deploy it, or I can simply go do a manual deploy right now. And what I'm going to do is I'm going to simply come over here and choose to manually deploy my package into the target, and I'm going to simply step through a few mouse clicks here. I have no special routing criteria. And I'm going to simply say that yes, this is the device I want to deploy to. And obviously there would be a long list here if you had many devices. I only have one, but I want to say that that's the device I want to deploy to.
From: 08:00	And then, I'm going to just say "Finish" because I'm going to accept all the other defaults, and now I'm simply going to say "Deploy", and you can see that it will say over here that it's "in progress", and if I come back over here to the Home screen, you can see that it will say that the package deployed and it's delivered to the agent, but it doesn't mean that the package is actually been deployed yet, and so if we come over into our target, and we can look at our target, and you can see if we come over to, there, you'll see that the scripts are there, of course, they were already installed from before, but the install script would have placed them into the proper place. You can see if we go over to the lua path directory, you can see that the .so files are already here. Again, these were already here from before, but the installer script would have placed them here for us, and I just wanted to show you where it would have put things if,
From: 09:03	when it runs the installer script. The other thing though I wanted to show you was that you can look at the log file, and that log file is in the /var/log, and it's wra.log, and you could see that it specifies that it's downloaded the file, but you'll also notice that it's saying that it has registered lua functions, and that was part of that register app description that we defined. So let's come over here, for a moment, and you can see now it says the package completed deployment, and so then let's come back over and take a look now at that script itself, and so if we come down into that lua scripts directory, and this is the one that we registered, and you can see the what we've done here is we've simply defined a function in lua, and it's a very simple function. It just simply, you know, does a couple of printfs essentially, and then it defines a data value,

From: 10:03	and posts that data value up to the cloud, and so, when you execute this function, all it's really going to do is output some message on the console, and then post a data item. And so you'll have one, a function one, function two, and you can simply see that all we have to do is subscribe these actions, by specifying the name of the function, and specifying the actual vector to the function in the script. And then we're simply in a loop. The loop is simply waiting on those functions to be called, And so this is why we started this process up asynchronously. It will run forever in this loop, and waiting for these functions to be called. So let's go over and invoke one of the functions. If we come over here now, we can come over to the side and see that one of our actions has been defined for lua function one. So let's go ahead and execute the lua function, and once we do that,
From: 11:03	we can come over here, and you'll actually be able to see that it will, it takes a moment to complete the action, and we can see that the function actually executed. And then, if we come back into our platform, and we go and look at our data, we'll see that it is actually defined the function value with "hello". That was in, defined in the script, the lua script that we created. And now let's go back, and execute the other function, the lua function two, and let's execute that one as well. And once we execute that one, and going back to our target again, you'll see it actually executed the "good bye". And then coming back to the target again, you can see, that it will show that we actually executed our functions here,
From: 12:04	and then you'll also be able to see, in our data, that we have defined "hello" and "good bye". Those were the data values that the script sent up to the platform. Let's go back and review that script for a moment. You can see here, that that's what we were defining here, is that when the function runs, it would actually post a data item up into the platform. So what have we covered? We've actually seen in this one example, we've seen defining a platform package, we've defined an install script for that package, we've uploaded the package to the platform, we've defined some deployment rules, or instructions for the package. We then deployed the package - it ran the install script, it registered some application functions with the agent - and then we were able to start using those
From: 13:04	application functions in the agent. So, in a single example, we've covered a number of different scenarios that were important for our common use cases that we looked at earlier. You know uploading packages, deploying packages, and defining values. Now, of course, creating a package like this for executable applications is probably not the way that you would normally do this, it would probably be a better choice to create an RPM file that could be signed, and then have the package manager install it instead. So there are multiple different ways that you could go about defining and executing some of these rules and actions but, I just wanted to show you this particular example, in showing that you can do it this way, as well as defining it in the package manager. Of course, if you wanted to install
From: 14:03	this at the time of your build, then you would put this into a layer, and just add it into your platform build in the first place. But this is an example of how you might deploy this these applications to a target that was already out running in the field. All right, so that completes this particular example, and then we'll move on into the next section now.

Contact: nlyons

Content ID: 045831

template('WindRiver/function/JS/wrConditionalContent');