OK, welcome back everybody.

So, we’re doing video number four and in this video, just to kind of recap, video number one was an overview of the API Management console and the API Management portal and the development area.

Video number two was basically on the WebSphere side, was to define and create your RESTful API, deploy it in WebSphere, and test it to make sure it’s working.

Video number three was basically exposing that API within WebSphere and in API management, and testing it out to make sure that it does work.

And lastly to complete the end to end story, I am going to do video number four which is, I am taking that API that I exposed to API Management, and I am going to use it within a sample Worklight application and I will deploy it within Worklight, use an emulator to show that you’re able to test it that way and use it.

I will be using the adapters in that case.

The actual mobile app that I am using is just some simple one that I will walk you through.

The purpose is not to define the mobile app itself, the purpose is to create that adapter, show you how it’s done, how you can use it to define the adapter, and what parameters to use to configure it to be able to run against to call that API management API which, underneath, will call the WebSphere Application Server layer and call the API that is defined in WebSphere, the RESTful API, and that completes this series of videos that I talked about that I started.

I wanted to give you a piece of info in here, is that as I was doing the videos, I have updated my installation of API management so you might see that some of the features that exist in the API management console may be a bit different, it’s actually becoming prettier and you have more options, so that you’re not confused to why that’s the case that video number one may not show you some item that video number four will. Video number 1, 2 and 3 are actually done at the same level of API Management.

Video number four, however, that includes the mobile case, does include the latest release of API Management, which is as of January 26, 2015.

Just to recap, I created an organization that’s called Soloman World Bank, swb is the path to it, and that’s the one that I will be defining everything from an API Management and I’m going to expose it that way.

If you look inside of the application you will see that swb is actually the URL to get access to it.

I do have an application that I have defined.
Of course I have the API that is called getInterestRatesByType and the type is going to be the interest rate type, it’s going to be long, short or jumbo, these are the three types that we have if you remember from video number two.

And I have defined the plans as you can see in here is the subscription plan that I have, subscribe plan is silver, which basically gives me limited number of access to the APIs, five APIs per minute, five calls per minute.

If I go into the API Management console under the plans you will see the sandbox that I have, I have deployed with the silver.

Of course, I could define it if I want to with the gold which gives me unlimited access but I’m not going to do this in here, the purpose is just to show you how to make the call end to end.

I could easily switch between the silver and gold, I'm not going to do this right now.

So let's get started.

I will start on the Worklight area.

As you can see here is my Worklight area, on the left side you will see the adapters.

I have an adapter that’s created.

I will upload the code for the end to end solutions so you can actually download and play with it if you want to.

I will be in some cases defining some samples, for example for the adapter, I will show you how to create the adapter, but I will come back and actually walk you through what I have configured for my adapter.

To create an adapter, right click on the adapter folder here, and do new, Worklight Adapter.

Of course, this is my Worklight Soloman World Bank, this is my application, again the purpose is not to show you how to define a mobile application or use it, the purpose of it is to show you, if you do have a mobile application, how do you go about defining an adapter for that particular API or APIs and how do you go about using that particular adapter to make the call to that particular API that’s exposed in API Management.

Under adapter type you would select http adapter and you can give it whatever name, and in this case I'm just going to give it a junk name here, TestSimpleAdapter and Finish.

Now I am not going to modify it, like I said I want to show you what gets generated because you do have a template, and then I will show you how my adapter looks like, that's going to connect and how I can go about finding the actual values to what I want to connect to.

So as you can see, you will have the JavaScript and you will have the template and this is very critical, this template.

This particular template comes in with a connection policy, protocol, and the domain, and in this case it was just a sample CNN feed that you're trying to get to on the port number.

Now let me show you how my adapter looks like.
If I go back and I'm going to just simply close this one for this XML file, if you look at my adapter, of course to connect to the API Management exposed APIs, API Management is going to run on all the https secure protocols so that's what you're going to get.

I have this IP address, 192.168.153.5 and this port.

Now this is the secure port for API Management, I'll show you how to find that but for https this is actually the port number you would use.

In terms of what the domain is that I'm going to be using, if you go to your API itself, where you defined your API, and as you can see in here when I click on APIs, if I click on this particular API, I click plus, you will see that this is the url, and the url that I have is 192.168.153.5 and this is what I'm actually going to be using for my domain.

So if I go back to the domain in this case, you will see that this is what I used.

In terms of the 443, like I said you could just simply remember it, this is exactly what the API is or what the port is, but if you actually want to find out what it is, if you go back to the cloud management console, click on the Clusters tab, and under the gateway clusters, click on Settings, you will see that you have the port 443 as that's what's being used actually for us.

Let me cancel out of this and continue the discussion on the API management, on Worklight.

Beyond that, that's all that you need to do.

Now at this point I am configured from an XML file and connectivity perspective to be able to connect to my API management.

Of course that is just one part of this adapter.

The second part of the adapter which is added, I want to define the function that I created.

Under the test simple adapter it does give you a sample program, or just a sample to use things, as you can see there's the function getStories that has interest or getStoriesFiltered, and it gives you a sample of how it looks like.

Let me show you how you can configure that on the actual adapter that we created.

I’m going to go to the one that I used for the RestSolomanWBRate, which is the World Bank rate, and if I click on the RestSoloman, you will see that I have only one API that I'm exposing.

And that API is getInterestRate that takes the type of interest rate that I am interested in, the customer name in this case the person so I can greet you, and then the client ID and client secret.

If you remember when we did configure the APIs I configured it that I would like it to be with a client ID and secret.

This is an option that you can do and that's what most people would do anyway, particularly if you're trying to expose that API externally that you want to be able to, for example, charge money for it, so you want to be able to not give the key to everybody.

I covered this already in part three of this video, but just to kind of walk you through it very quickly, as you can see in here if I go to Application, by the way the view that you see in here is a little different than what you saw before and that's part of the enhancement that was done on the latest fixpack that I have applied from API Management it makes it look really nice.
As you can see in here, the application that I created, which has this particular API, getInterestRate, it tells me that it is silver plan, it tells me what the average of calls that I have, and of course it tells you have the client id and the client secret.

If you do not know what they are, you could just simply reset them and that would allow you to regenerate the client ID and client secret, and you can use it if you want to take these and use it for your next call.

Just be careful if you do reset it, the people who already have the old client ID and client secret will no longer be able to work and they need to update their settings.

So if I go into my Worklight again, and you'll see the method that I used is get, obviously that is straightforward and simple, I am using, if you look at my API, it is a get API that I am using.

If I go back again to the Worklight and you will see that for the path I am going to take it all the way up to the records after that IP address that I created.

So if I go back in here, you saw in the connectivity in the XML file I added the IP address, so at this point I am going to take the swb all the way to the records, which is up to the point where I'm going to be passing the parameters, but I'm not going to include those parameters when I set it up inside of the Worklight environment because those are going to go where the parameters are, and in this case, the parameters that I am interested in passing are going to be the following, because you can see again based on the API, I have the type, I have the fname, I have the client ID and I got the client secret.

That's what this tells me that it's needing so therefore I want to make sure I add those parameters.

So going back in here you will see that the parameters I pass, I do type, which is going to be ptype, I have the fname which is the pname that I have in here, client ID and I put the parameter name to be f and pclientid and of course the pclientsecret for the client secret.

And beyond that this will actually compose my input and I will invoke the http as you can see in here, very straightforward, very simple.

Once I'm done with this, of course I want to be able to validate that it does work, so before I start using this API I should be able to, I need to compile the adapter that I created, and I want to test it.

Worklight does give you the ability to do that.

If I go to the API that I just created, of course again I'm going with the one that I actually added some real valuable data, so if I go to the RestSolomanWBRate, if I right click on it and I go Run As and Deploy Worklight Adapter, that would allow it to get deployed, it says no functional changes were detected, and it will just rebuild it for me.

You may not be able to see it through here but it says adapter build and deploy build and right now I seem to be ready to go and I want to test it, I right click on it and I do Run As and I go to Invoke Worklight Procedure.

Now when I do this, remember when I put function inside of the JavaScript, it's smart enough to go list all the function in here, in this case I only have one, and it tells me what the signature is for that particular function, if you have multiple of them you can select which ever you want.
And that’s where you pass the parameters, and in this case I’m going to say it’s a jumbo loan and I’m going to say this is Soloman789, that’s the value that I want to pass in, and I put the client ID and the client secret, and I hit run.

Now as you can see at this point I passed that information in, it is being run, it’s going to be connected to WebSphere, and it’s coming back saying this is the first name, everything worked ok, the status is 200, and it tells me by the way the rate is 3.075, that’s perfect.

To also validate that, I can go back to the WebSphere area, you’ll see on the WebSphere side, remember I have multiple layers here, I have the WebSphere layer, I have a separate environment, think of it as separate machine, and it is a different Eclipse environment, and I called the REST APIs and indeed that’s what happened, it’s the old 75 is what that returned and this is what the customer name was.

I printed these in WebSphere just to make sure that I see those values as they come in, and the type of loan that was requested was a jumbo loan.

So perfect, that works just great.

If there are any issues, you would debug these issues until you can get things working.

Now one of the things that I did not mention in here which is kind of necessary to make things work, as you can see, when we do make the call when you want to validate, I am using https so if you look at the adapter again, that particular adapter is actually using, from a connectivity and protocol, I am using https.

That means it’s going to have to be a secure connection.

To do that, I will need to make sure I do the security settings.

So if I go back in here, these are kind of steps that I have done to be able to get the certificate, I need to get the certificate from the API Management system, or the DataPower in this case, and I want to take there’s a default certificate that comes in with it, and I’m going to take that and I’m going to import that certificate into my environment, and in this case it’s Worklight.

You can look at the Worklight documentation and it tells you that or I’m going to walk you through it very slowly, it’s very straightforward.

In my case, I downloaded OpenSSL and once I downloaded that I literally just simply executed the OpenSSL.exe and I went into the command and I issued s_client minus connect and this is, again this is the actual 192.168.153.5 calling 443, and if you remember when we looked at this, this is exactly the port number and IP or domain that I used when I defined my environment, which is where my API is, and going back there again that would allow me to actually get the certificate and it will print it out for me on the screen.

You would need to copy from include in the begin certificate and end certificate lines.

You will need to copy this into a file of your choice, in my case I called it SolomanWorldBankswb_wl for Worklight, and then self-signed certificate dot cer, and once I do this you need to make sure that you upload it into the certificate store, and to do this, typically you need to do it in kind of one or the other because the system will look for the certificate, remember Worklight has a keystore that you configure it with, and I will show you that, but also you need to configure it on the JDK itself if you want, just to make sure that you capture it if Worklight doesn’t catch it for example.

So in my case, I know that Worklight is configured.
If you look at how I’m doing it in here, I go to the JDK that I’m running with and I’m running on Windows and in this case I’m using Oracle 1.6 JDK and I basically use the key tool and I did minus import and as you can see minus trustcacerts minus alias and I created, this is the name of the alias that I will be using to actually create that particular certificate when I import it into my keystore, and I put the keystore to be the place, and as you can see the keystore is the location of the certificate store within the JRE and this is the location of it which is the JRE home lib security certs, and as I said minus file and the file would be the actual file that I just generated, or that I copied and pasted the self-signed certificate in.

And once you do this, it asks you a question and it asks you if you want to enter the password, and of course that’s straightforward you can enter the password.

The default password for the JDK is changeit and the default password for Worklight keystore is worklight.

Now if I go back to the Worklight, I want to show you how to import the certificate into Worklight which is the same exact thing.

I’m using keystore.exe minus import, I do the alias, I do the keystore, and in this case my keystore is going to be this is the workspace that I have for my Worklight environment, and as you can see it takes me all the way to the key dot JKS and that is actually the keystore for Worklight.

And of course the minus file points to the same exact file that I did earlier.

So how do I find out where that keystore is in your environment when you set it up?

Let’s go back to the Worklight item.

As you know, Worklight is going to run your environment inside of the WebSphere Application Server Liberty profile as an option, of course you have other options, so if you open this server configuration file, which gives you this one, you will see in the server configuration file there is an entry if you look for keystore, you will see there is the default keystore and it’s being set with a password called worklight and this is the location where that keystore is located.

So I did specify that and if you look at the source code you could see it’s the same way, its keystore, ID default keystore and the location of it and the password.

So that’s how you can find out what your password is when you install it, this is the default, and you can change it obviously if you want to, or even change the keystore to something else.

But that allows you now to know how you can import that certificate and be able to issue the call that’s going to be used in https protocol to run, because API Management only runs with https protocol.

Now that we’ve got this piece, what is the next item for me?

I’ve tested it, like I said and I actually was using https to do the testing, so I know things are working just fine, and as you can see from WebSphere everything is working just great.

Of course, what I’m after right now is to actually test it in a way that allows me to start a Worklight application and I will be using an emulator to actually test it out.

The next level up for me is that after I create that adapter, and again// my adapter is called RESTSolomanWBRate.
I would go ahead and I created an application called SolomanWorldBank and I generated the code for it for Android, which is what I’m trying to simulate here.

If I go to the index XML file, again the purpose of this session is not to show you how to become a mobile developer, I literally just created a quick and dirty mobile application that I can just deploy in my emulator to just show you that I’m able to make the call and return the values.

So in this particular case, I created just a simple form and in that form I wanted a long term input so it’s just two inputs and a customer name, and I did specify by the way for the long term I would like the options you have as short, long or jumbo, and as you can see they are type text and straightforward and I click process the form and I send out form and if I go into the process form I will extract the value of the term first name.

I would hardcode the values of the client ID and the client secret, remember those are the two values that when I sign up with a company and they give me the client ID and secret, I can take these things and put them in my application.

I’m using the simplest form of doing things.

You might be able to call another function to extract that value from another place, that’s basically up to you how you write your program or how you architect your solution.

And then at the end I would go ahead and basically call SolomanWorldBank_getRatesPerType and I pass those values in.

As I do this and this is the function that I created, again within that index.xml file, that’s when I call the adapter and invoke it.

As you can see I am invoking the adapter by saying, here I am doing an invocation, the adapter is RESTSolomanWBRate and this is my adapter RESSolomanWBRate, and the procedure name is getInterestRate, and if you go back like I showed you earlier into the procedure name with a function it is getInterestRate, and beyond that, the parameters I’m passing in which is basically the same parameters that came in into the actual function, and then I just call an invoke of that procedure.

I have an onSuccess and an onFailure, I am doing something simple in here, I just put a quick alert in the case of a failure.

In the case of a success, I do the same way, I put a quick alert the basically shows me here is the interest rate that I got.

As you can see, I do success rate equals and I put the rate.

As I mentioned earlier, I will upload all these samples so you can at least be able to look at them yourself once the videos are complete and uploaded and you’ll be able to download them and take a look at them yourself and do something similar.

The first thing you do after that is you want to make sure that you go ahead and compile your code, so if I go to the SolomanWorldBank, this is my application, and I do Run As, and I do Run on Worklight Server, and that’s what it’s doing right now, just let it go.

Of course there were no changes detected because I have not made any changes.
It would just simply build it quickly because it’s not going to really compile it, and once that is done I will go ahead and take it to the next step which is I have already generated this for Android, so because that is the emulator I’m going to be running, so I will go ahead and right click on it and do Run As, and I’m going to do Android Application, and doing so is going to run that application into an emulator, and I will be able to basically execute what you can see in here.

I’m going to have the form and I will be able to execute that and be able to call API Management, call to get the interest rate API within the API Management, which in turn make the call to WebSphere, which is completely hidden from me as I mentioned earlier, and I will be able to abide by the rules that the developers or the people who own that API dictate how many calls per minute they would allow, and being able to charge and being able to monitor the calls that are taking place.

If you look at this, this is what I’m going to begin, this is from the emulator, today is January 26, the date is actually correct.

If I go ahead and login, it tells me Welcome to Soloman World Bank, this is again the simple application that I did, that’s within the index.xml file, nothing really too complicated here.

Welcome to Soloman World Bank and it gives me the form that I’m going to ask.

Now let’s say I’m going to go ahead and ask for, the default was long, I’m going to change it to short, and I’m going to say that for the name I’m going to say SolomanB.

Now if I do get interest rate and I click on that, it will actually go ahead and make the call, it comes back to me with success saying interest rate is 3.375.

Now let’s walk through this.

I passed in SolomanB as a customer name and long term being short.

This particular Worklight API uses https called the API Management API that was exposed there.

API Management itself made sure that you were the right person based on the client ID and the client secret and made sure that you abide by the rules as is you have not reached your quote for the number of calls you are allowed per minute, or whatever plan that you are under.

And of course after that it maybe passed that call onto the application server layer and within the application server layer, which is WebSphere in this case, it passed in and it called the REST API that is running inside of WebSphere.

And let’s just confirm that that’s the case because remember I said I do print those values in just to make sure that things are coming in.

So if I go to WebSphere and here you go you, will see that it says the rates were type short, that’s exactly what I had, and the rate was 3.375 and it says the name was SolomanB.

Success.

And as you can see in here I am able to run that, I am able to extract that information and make it work just fine.

So if I go in and I change that one into long just to make sure here and keep it with Soloman, it told me 4.375, and if I go back there again you will see that indeed it is still Soloman B and it’s 4.375.
I want to just do a couple of calls with different names for SolomanA, and then SolomanC, and then D.

I’m just making a few calls in here because I want to get some statistics so I can show you that things are actually being generated and being tracked by the API Management solution.

Now if I go back there, I guess maybe I could just do another one I’m going to call it jumbo, jumbo for this one and SolomanJ, and of course I got 3.075 just to confirm the jumbo one one more time.

As you can see I got Soloman B, A, C, D, and then the last one was SolomanJ with a jumbo and 3.075.

Let’s go back to the API Management APIs, the statistics, let’s take a look at the impact manager, and let’s take a look at the analytics and you will find out in here when this one comes back that it just kept track of the, this is different from the last time I showed you this, it gives you the environment, it shows you the plan, it gives you the APIs, you can really select what plan, whether the silver or plans, this is new in the latest one and is actually becoming very very cool, and you can actually also do it for all APIs or be specific to any particular API, in this case I have the API developers and again you can do it for application or the actual API user, so if you have Soloman and if you have Jonathan and if you have Mike, you can get it for everybody or you can actually say, I see a lot of people calling APIs, I want to see who’s making these calls, and you can drill down exactly to that person and maybe raise the rates on them or whatever you want to do from a business perspective.

And you can specify whether it’s the last 24 hours, seven days or thirty days.

And as you can see in here, the total number of calls in the last seven days, even in the last 24 hours, you will see that this is the number of calls and as you can see at 10:15pm, which is really past 10:15 pm, which is when I made the calls and you can see an increase, I have about seven, that’s the number of calls that I have gotten.

And with that, this is the conclusion of this series of end-to-end solution where I defined an API in WebSphere Application Server Liberty profile, RESTful API, exposed it in the IBM API Management solution, and used the IBM Worklight offering, in this case just simply the simple Worklight Studio to go ahead and develop a simple application and develop an adapter to call and test that particular API and then show you how that can be exposed very very easily.

And now I’m ready to go and I can deploy that application on a normal Android system and I could have a system that takes advantage of the APIs.

Again, my name is Soloman, thank you so much for tuning in to listen and watch for more future videos on a lot of these, not necessarily just API or WebSphere Application Server videos perspective, I will try to include more videos throughout the solutions that IBM offers in that space that might be very useful for you which may include multiple areas of the IBM products.