

This video will demonstrate how to create a web service using the host access transformation services tool kit. The example web service will take his input and account number and will return detailed account information which can be utilized by a larger business process that requires it.

To begin creation of the web service click on the new hats project link.

In the hats project creation wizard you need to give your project a name. You will also need to specify that you will be creating a web application and in this case targeting the application to be deployed to a websphere application server 6.1 environment.

In the next panel configure the name of the host which contains the transaction which you will like to capture as a web service. Note you will also indicate the type of connection you would like to use. In this case it will be to a 5250 backend system.

The project theme is not important for the creation of the web service.

Also the template is not important so just click finish to complete creation.

The basis of a hats web service is a macro. A macro automates interactions with a host. You can sync commands with a host, enter data into entry fields, extract data, and be used to navigate host screens on behalf of the user. To create a macro we'll need to start a host terminal. Click the host terminal icon on the toolbar.

Before recording the macro I would like to take you through the screens that highlight the transaction that we will capture as a part of our web service. The initial screen is a sign on screen so I will type the user ID.

Next I will enter the password and press the enter screen to move to the next screen.

On the main menu screen I will type the command to go to the example application that I need to access.

Once I'm in the application I will select option one to perform a customer inquiry.

On this screen I will enter an example customer number.

This screen shows the specific account details associated with that customer number. This is the screen on which I will extract data which will be returned as part of my web service.

On this screen I just press pf12 to exit out of the example application.

Once I return to the main menu I will enter the sign out command.

We are now ready to begin recording our macro.

Go to the toolbar and click the macro record button.

You will be prompted to give your macro a name. In this case we'll call it get customer details.

You can then click the finish button to proceed.

When recording a macro you need to uniquely identify each screen in the navigational flow. In this case we need to identify the entry screen to our macro. It is the sign on screen and I have given it the name signon so that it will be easily recognizable when I show it in the visual macro editor later on in the demo.

Once the screen has been uniquely identified you can begin interacting with it. Also note that the macro navigator to the left hand side not only records which screens you have visited but the key strokes you have entered.

In this case we'll type in the user name of atdemo.

We then type in the user password and press enter to proceed to the next screen.

The next screen that we encounter is the display program messages screen. I will uniquely identify it by the title on the top of the screen and give it a more readable screen name.

Once the screen has been defined I press the enter screen to continue to the next page.

The next screen we encounter is the main menu screen. We uniquely define it by roping off the word main in the upper left corner and give it a meaningful name.

We then type in the command to go to our sample application that we will use to retrieve the user specific information that we require.

We have now entered our sample celDial communications application. Note that I will go to the tool bar and click on the define screen recognition button which I have done on previous screens so that we can uniquely identify this screen.

Once we have roped off the unique identifying characteristics for the screen I also give the screen a unique name.

For this particular web service we will always want to do the customer inquiry path so I will type option 1 to continue.

Our next screen is our customer inquiry screen. I will define the unique screen definition criteria for it.

This screen contains the input field in which we will place the account number received in the application that is invoking this service. In order to get the information required to satisfy this field we will add a prompt action to our macro by clicking on the prompt button on the toolbar.

We'll be asked to provide a prompt name. In turn this prompt name will be used to name the input parameter used for our web service.

Click the ok button once the prompt name has been defined.

You'll next be asked to enter a value for the prompt so you can continue recording the macro.

The value that you enter is placed in the input field. You may now press the enter key to continue to the next screen.

The next screen that you encounter contains details about the customer account. We will be extracting the web service output data from this screen. Note that I have clicked the define screen recognition button on the toolbar so that we can uniquely identify this screen.

Once the screen recognition panel is displayed, I select the area at the top of the screen for recognition and give the screen name a unique identifier.

I will now begin to extract my output data from the screen. First I rope off the area to be extracted, in this case the customer name, and the go to the toolbar and select the add/extract action.

I will be prompted to give the extract a name, in this case custname, and that will be used as the output parameter of the web service. I will the click the finish button to continue.

Next I rope off the customer phone number and select the add/extract button from the toolbar.

I am then prompted to give the action a name. Again this will be used as a name of the output parameter for the web service. I click finish to continue.

I next enter pf12 to begin exiting the application and return to the main menu.

Again I enter pf12 to continue exiting the application.

Once on the main menu I enter the sign off command to return to the sign on screen.

Once here I go to the toolbar and click the icon to stop recording the macro.

I'll then be asked to define the screen recognition criteria for this screen because it represents the exit screen for my macro.

Once my macro has completed recording I will close the host terminal window.

I will be prompted to save the macro changes that I have made and I'll click yes to complete the macro.

Now that my initial macro has been created I would like to go back and edit it to make a couple of changes required to make it more robust and to add another extraction to the detail screen. To do this I will go to the navigator tree on the left hand side of the hats perspective and click the plus sign next to the macros label.

I will right click on the macro named getcustomerdetails and select to open it with the visual macro editor. This is a new technology preview that is available with hats 7.1 and allows you to visually edit macros.

When you open the visual macro editor notice how each box represents a screen. Along with each screen you'll also see the actions that will be performed when that screen is encountered. Lastly you'll see the error lines that show the navigational flow between screens.

In our particular example I know that sometimes the display messages screen is not displayed so I need to introduce an alternate flow. To do this I click on the next screen connection in the palette on the right hand side and select to connect the sign on screen directly to the main menu screen. My macro will be more robust.

I will now right click on the background of the editor view and select to reset the layout.

Next I'll go to my customer detail screen and right click on the customer detail box so that I may select properties and add a new action to extract some more information that I would like to return.

Once the properties panel is displayed I will select the add button to add my new extract action.

As part of adding the new extract action I'll be shown the screen so I can identify the area to be extracted and then be allowed to give that area a name.

Notice that the new extract action has been added to the bottom of my action list. I'll need to move it up so that it occurs before the pf12 command.

To move it I highlight it and click the up button.

I am now extracting 3 pieces of data instead of 2. Those 3 pieces of data will be returned as part of the web service.

Now that my macro is complete I am now ready to generate the web service. The first step is to right click on the macro and select to create an integration object. This is a java bean that encapsulates the macro. It is the basis for our web service.

Once the integration object is created, you will go to the source item in the navigator tree and expand it by clicking on the plus sign.

Right click on the integration object and select the option to create web service support files.

You'll need to give the service a name. In this case I'll call it getcustomerdetails.

Once the web services support files have been created, expand the view for the web service classes and right click on the java file associated with the web service. You will then select the option to generate the web service itself.

The web service wizard will be displayed. Click finish.

Once the web services creation wizard has completed, expand the web content file and look for the wsdl file under the web services definition folder.

At this point you can right click on the wsdl file and select to test it with the web services explorer.

In the web services explorer test environment enter the customer number for your test.

Click the go button to drive the web service in your test environment. Double click on the status line so you can expand that view and see the results returned by the web service.

Notice that the customer input data shown as well as customer details that were extracted.

Our demo is now complete. We have successfully generated a web service using the hats toolkit. The web service will take the customer number and return specific account details. The wad file that was created as part of this web service can be consumed by any calling program to evoke the web service and use those details as part of a larger business process. Thank you!