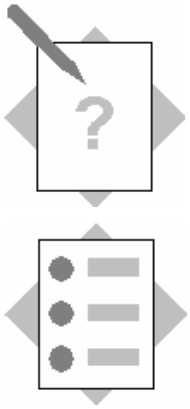
Models Exercise



Chapter: **Models, Web Services**

Theme: **Using Web Dynpro to access a Web Service**

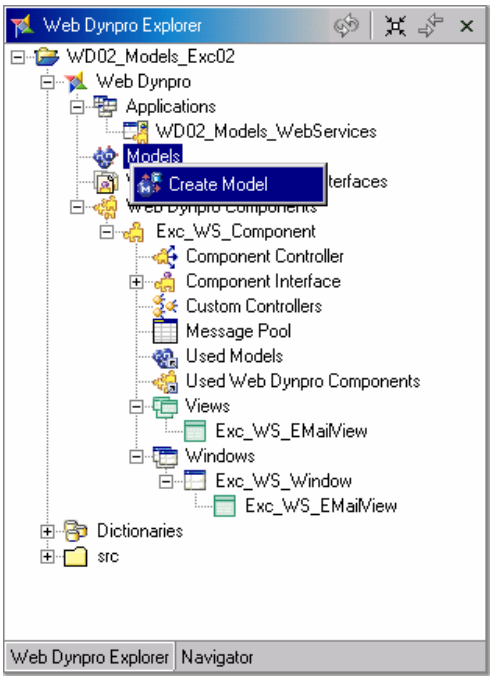
At the end of this Exercise, you are able to:

- Access a Web service from Web Dynpro

4 Developing, Step-by-Step

4-1 Create a Web service model.

4-1-1 Generate a Model from the WSDL Description



In the project structure, expand the node Web Dynpro Models.

From the context menu, choose Create Model.

The appropriate wizard appears.
Choose the Import Web Service Model option and press *Next*.

Enter:

Name	EmailModel
Package	com.sap.training.wd.exc21.model_ws

Under Select WSDL Source, choose the radio button UDDI or URL, followed by *Next*.

Enter:

Wsdl field	http://webservices.matlus.com/scripts/emailwebservice.dll/wsdl/IemailService

You do not need to make any entries in the next popup (Proxy Definition / URI Package Mappings).

Close the input dialog by choosing *Finish*.
The corresponding Java proxies are then generated as client stubs, and the model classes are generated for the subsequent binding of context elements.

4-1-2 Make HTTP Proxy Settings

Open the Package Explorer.

Open the following node:
src / packages / com / sap / training / wd / exc21 / model_ws / proxies

Choose the file *lport1_1.lp*.

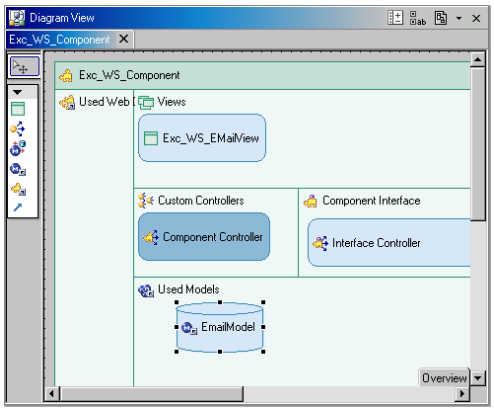
After having selected the checkbox *Use HTTP Proxy*, make the appropriate entries in the fields *Proxy Host* and *Proxy Port*:

The field *Proxy Host* represents the host name or the IP address of the proxy server, and *Proxy Port* is the port to which the proxy server listens.

Save your settings by choosing *Save Editor Contents* in the toolbar underneath the menu bar.

4-2 Create the binding: Component Controller Context and Model.

4-2-1 Add the Model defined in Step 4-1 to the Web Dynpro Component using the Data Modeler.



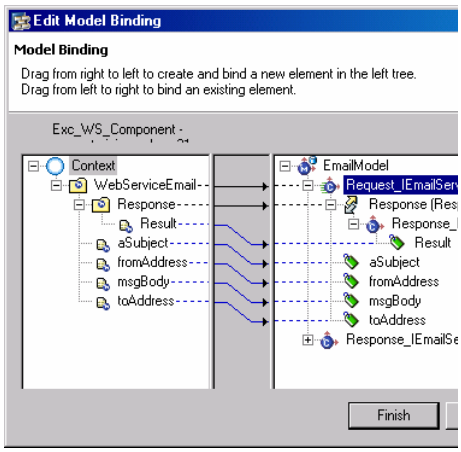
Open the Data Modeler.

In the toolbar on the left, choose the icon *Add a model to the component*. The icon will turn gray.

Place the cursor on the *Used Models* area and left-click.

Select *EmailModel* and choose *Ok*.

4-2-2 Bind the component controller context to the Web service



Open the Data Modeler.

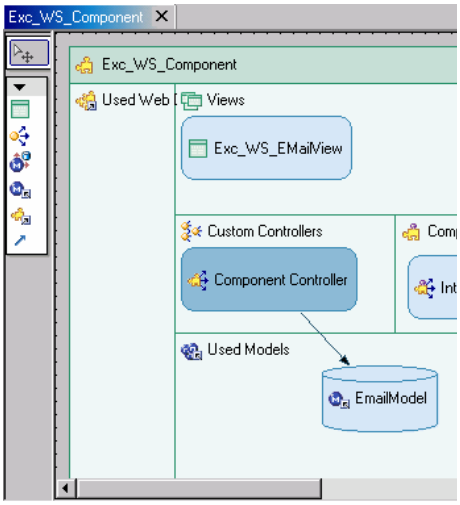
In the left toolbar, choose *Create a data link*.

Starting above the Component Controller rectangle, press the left mouse button, and keep it pressed.

Draw a line to the *EmailModel* rectangle and release the left mouse button. The Model Binding Wizard starts automatically.

Drag the node *EmailModel / Request_IEmailService_sendMail* of the model class to the root node of the component controller context.

In the dialog box that appears, select the model node *Request_IEmailService_sendMail* with all subcomponents.



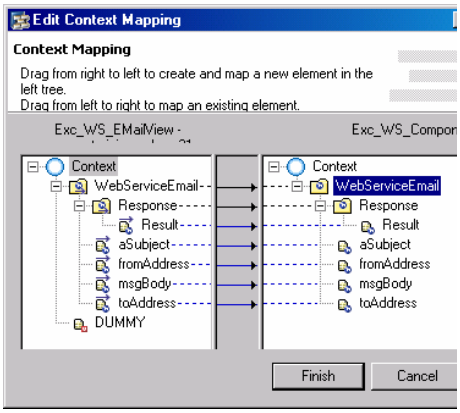
Rename the new Model node from *Request_IEmailService_sendMail* to *WebServiceEmail*, by editing the appropriate entry in the column *Name*, and then choosing *Ok*.

The resulting Model Binding between the model node *WebServiceEmail* and the corresponding model class is then displayed in a dialog box.

Close the Model Binding Wizard by choosing *Finish*.

4-3 Map the View Context Elements to the Component Context Elements

4-3-1 Define a Context Mapping in the Data Modeler



Open the Data Modeler.

In the left toolbar, choose *Create a data link*.

Starting above the Component Controller rectangle, press the left mouse button, and keep it pressed.

Draw a line to the Component Controller rectangle and release the left mouse button.

Drag the model node *WebServiceEmail* of the context of the component controller to the root node of the view controller context, and drop it.

In the dialog box that appears, select the model node *WebServiceEmail* with all subcomponents and choose *Next*.

In the final dialog box, the context mapping declared between the two model nodes WebServiceEmail is displayed graphically: After having chosen *Finish*, the model node – together with its model attributes – is mapped to the component controller context.

4-4 Update the View *Exc_WS_EMailView*

4-4-1 Define the Data Binding between the value of the input field of the View *Exc_WS_EMailView* and the corresponding context attributes.

Open the view *Exc_WS_EmailView* in the View Designer by clicking the *Layout* tab.
In the *Properties View*, define the following bindings between input field values and context model attributes:

Name of input field	Context Model Attribute
fromAddress	*WebServiceEmail.fromAddress*
toAddress	*WebServiceEmail.toAddress*
aSubject	*WebServiceEmail.aSubject*
aMessage	*WebServiceEmail.msgBody*

4-4-2 Create the Action *SendEmail*.

Open the View Designer for the predefined view *Exc_WS_EmailView*.

Choose the tab *Actions*.

Choose the pushbutton *New* to start the dialog box for defining a new action.

Enter the name *SendEmail* for the new action.

Enter *Send Email* in the Text field and then choose *Finish*.

4-4-3 Bind the action *SendEmail* to the *onAction* event of the UI element *SendButton*.

In the View Designer of the view *Exc_WS_EmailView, Properties View:*

Choose the UI element *SendButton*.

Bind the event *onAction* of the Button UI element *SendButton* to the action you have created, *SendEmail*.

4-5 Implement the source code for sending the Email via the Web Service Connection

4-5-1 Implement the Generic Event Handler *wdDoInit()* of the View Controller

In the View Designer, click on the *Implementation* tab for the view *Exc_WS_EmailView*.

After the generation routines have been run once again, the updated source code of the view controller implementation is displayed.

Now add the following Java code into the User Coding Area:

```
public void wdDoInit(){
  //@@begin wdDoInit()
  // create a new instance of the Web Service ModelClass
  Request_IEmailService_sendMail req =
      new Request_IEmailService_sendMail();
  // bind new instance of the Web Service ModelClass to the
  // independent Model Node
  wdContext.nodeWebServiceEmail().bind(req);
  //@@end
}
```

4-5-2 Implement the action event handler *onActionSendEmail()*.

In the *onActionSendEmail()* method, add the following source code:

```
public void onActionSendMail(wdEvent) {
  //@@begin onActionSendMail(ServerEvent)
  IWDMessageManager msgMgr=
    wdThis.wdGetAPI().getComponent().getMessageManager();
  try {
    // call Email Web Service
    wdContext.currentWebServiceEmailElement().
    modelObject().execute();
    wdContext.nodeResponse().invalidate();

    int result =
      wdContext.currentResponseElement().getResult();
    String msg = "Email Web Service returned " +
                 Integer.toString(result);
    if (result == 0) {
      msgMgr.reportSuccess(
        "The email was successfully sent (" + msg + ")!");
    } else {
      msgMgr.reportWarning(
        "The email was not successfully sent (" + msg + ")!");
    }
  } catch(Exception ex) {
    msgMgr.reportException(ex.getLocalizedMessage(),true);
  }
  //@@end
}
```

5 Building, Deploying, and Running, Step-by-Step

	Deploy and run the Web Dynpro application.
	In the Web Dynpro Explorer: Expand the nodes *WD02_Models_Exc_WS* / *Web Dynpro* / *Applications*. Open the context menu for *WD02_Models_WebServices*. To deploy and run the application, choose *Deploy new Archive and Run*