TIPS ON INTERACTIVE FORMS DEVELOPMENT BASED ON ADOBE SOFTWARE

How to Handle Table Input and Output

Marc Chan
SAP NetWeaver RIG
SAP Labs

Copyright

Table of Contents

Overview 4

Prerequisites 4

1. Displaying Table Outputs in Interactive Forms Based on Adobe Software 5

    1.1. Creating Context in Web Dynpro 5

    1.2. Adding a Table to the Interactive Form 6

    1.3. Testing the Table Output Display 10

2. Taking Table Input in Interactive Forms Based on Adobe Software 12

    2.1. Adding Actions to the Existing Form 12

Overview

SAP® Web Application Server 6.40 provides a new service called *Adobe Document Service* for generating and rendering interactive forms based on Adobe software. This how-to paper demonstrates how to handle the input and display of tables.

Note: This instructional material is based on the SAP NetWeaver ’04 release of SAP Web Application Server 6.40.

Prerequisites

It is assumed that the reader is familiar with the basic concepts of Web Dynpro development. Web Dynpro terminology and concepts are outside the scope of this paper. Users can refer to relevant Web Dynpro documentation for details.

It is assumed that the reader has reviewed the basic concepts of how to create an online interactive form based on the first paper in the Tips on Interactive Forms Based on Adobe Software collection, “How to Create Online and Offline Forms in Web Dynpro.”

Users should have SAP Web Application Server 6.40—SAP NetWeaver Developer Workplace installed on their development machine. Details on installing SAP NetWeaver Developers Workplace can be found in the installation guide of SAP Web Application Server 6.40.

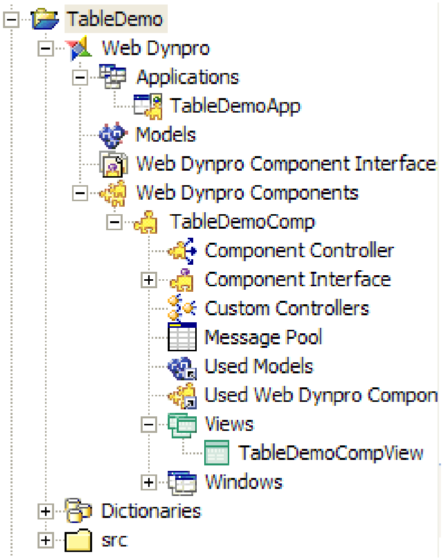
Adobe Document Service should be installed and ready for use. The detailed instructions on the installation and configuration of Adobe Document Service can be found in the installation guide. Adobe Reader (version 6.0.2 or higher) and ActiveX Control Framework should be installed on the developer machine. For more information, please see the installation guide.

1. Displaying Table Outputs in Interactive Forms Based on Adobe Software

This section will demonstrate the steps for displaying a table structure output in an online interactive form inside Web Dynpro. As an example, we will build a sample data context in Web Dynpro with multiple instances/records at runtime.

1.1. Creating Context in Web Dynpro

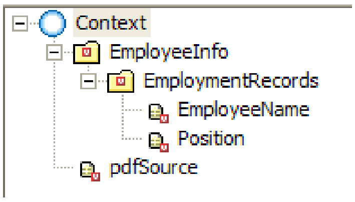
1. In SAP NetWeaver Developer Studio, create a new Web Dynpro project called *TableDemo*, with an application called *TableDemoApp* and a component called *TechDemoComp*. Put them all in a package called *com.sap.demo.tabledemo*. A TableDemoCompView should be generated by default.



2. In TableDemoCompView, create the following context structure:

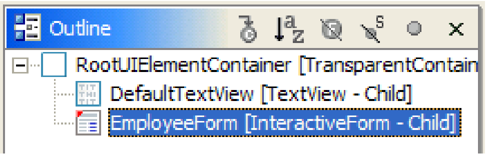
   - **Value Node**: EmployeeInfo
     - Cardinality: 1..n
     - Singleton: true
     - **Value Node**: EmployeeRecords
       - Cardinality: 0..n
       - Singleton: true
       - **Value Attributes**:
         - EmployeeName (type: string)
         - Position (type: string)

3. Create another Value Attribute in the Context root and call it *pdfSource*. Set the type to binary. The resulting Context will look like that below.



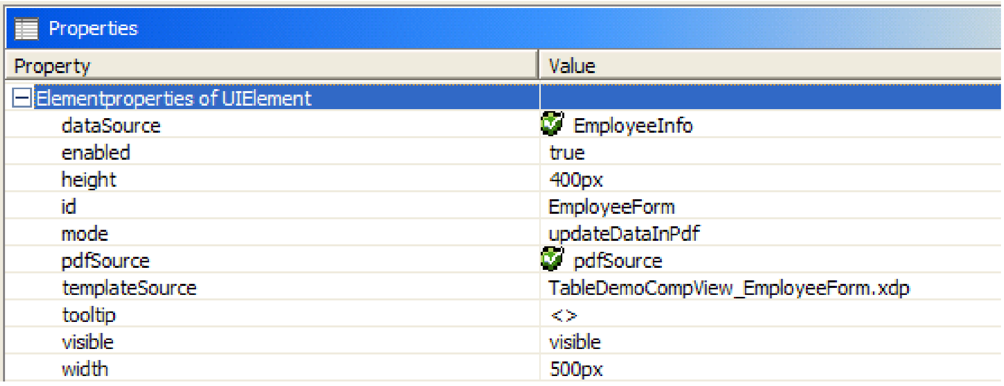
1.2. Adding a Table to the Interactive Form

4. Go to the Layout tab of TableDemoCompView. Create an Interactive Form called *EmployeeForm*.


Outline
RootUIElementContainer [TransparentContain
DefaultTextView [TextView - Child]
EmployeeForm [InteractiveForm - Child]

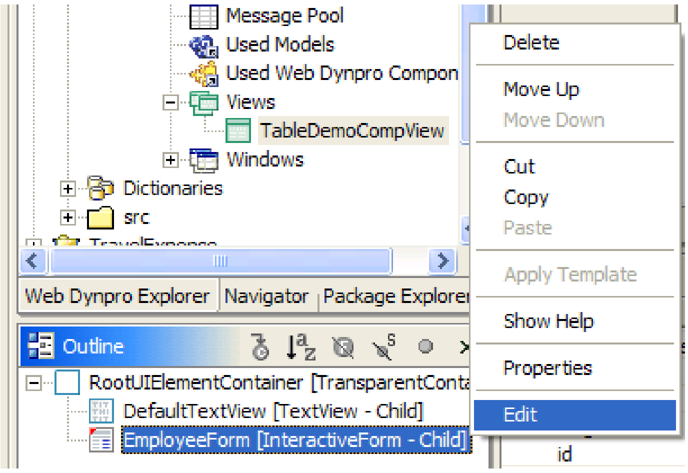
5. In the Properties of EmployeeForm, set the following parameters:

- DataSource: EmployeeInfo
- Height: 400px
- pdfSource: pdfSource
- Width: 500px

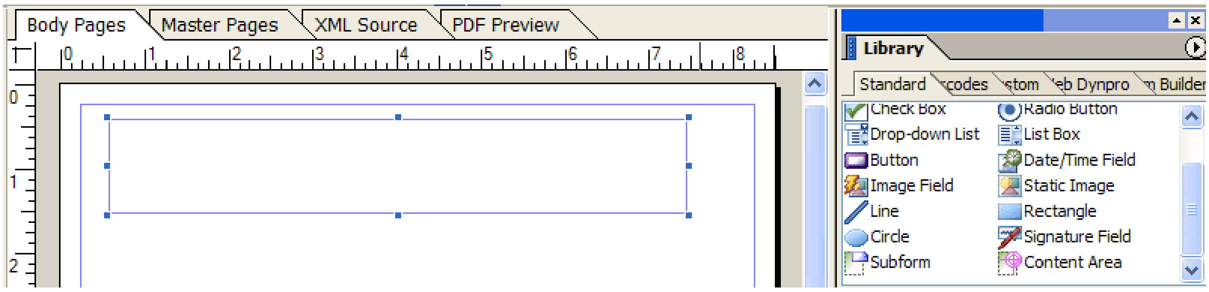


Property	Value
Elementproperties of UIElement	
dataSource	EmployeeInfo
enabled	true
height	400px
id	EmployeeForm
mode	updateDataInPdf
pdfSource	pdfSource
templateSource	TableDemoCompView_EmployeeForm.xdp
tooltip	<>
visible	visible
width	500px

6. Edit the EmployeeForm (right-click and choose Edit). This will show the Adobe Designer for editing.

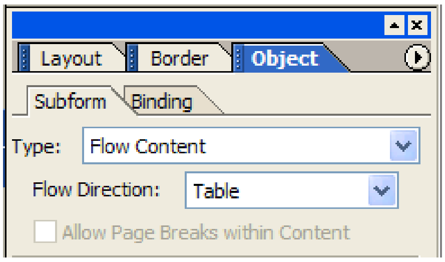


7. Place a subform UI element onto the form. To do this, drag the subform icon under the Standard Library tab onto the form.



8. In the Properties of this subform, set the following parameters:

- Type: Flow Content
- Flow Direction: Table

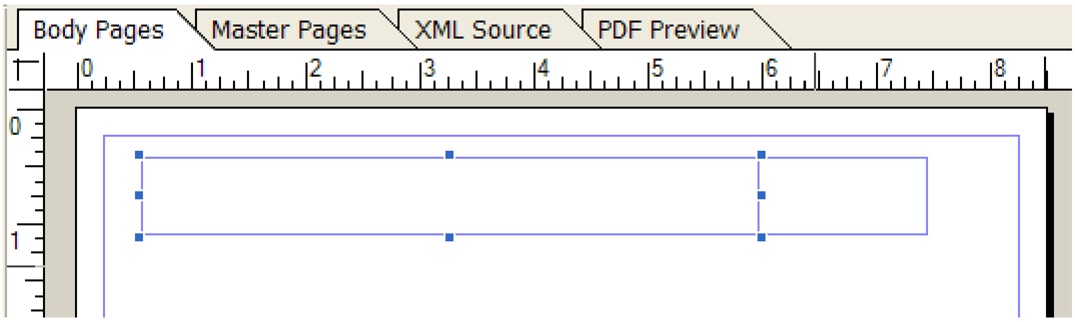


Type: Flow Content

Flow Direction: Table

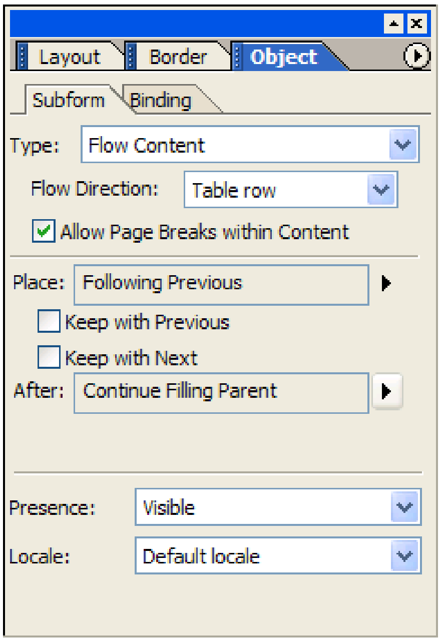
□ Allow Page Breaks within Content

9. Place another subform element inside this subform.



10. In the Properties of the inner subform, set the following parameters:

- Type: Flow Content
- Flow Direction: Table row



Type: Flow Content

Flow Direction: Table row

☑ Allow Page Breaks within Content

Place: Following Previous

□ Keep with Previous

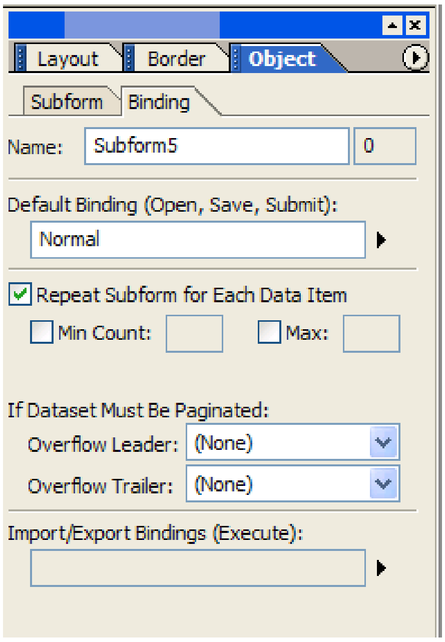
□ Keep with Next

After: Continue Filling Parent

Presence: Visible

Locale: Default locale

11. On the Binding properties of the inner subform, check the Repeat Subform for Each Data Item checkbox. Uncheck the Min Count checkbox.



Name: Subform5 0

Default Binding (Open, Save, Submit): Normal

☑ Repeat Subform for Each Data Item

□ Min Count: □ Max:

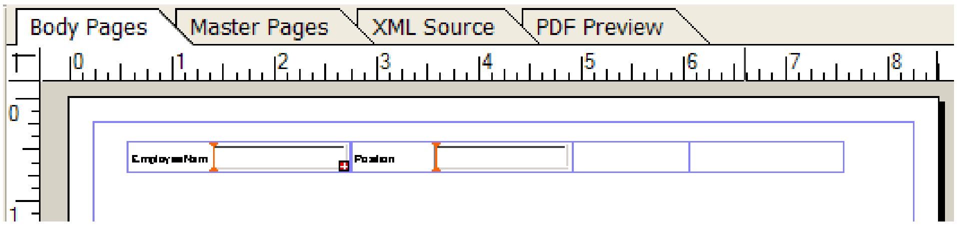
If Dataset Must Be Paginated:

Overflow Leader: (None)

Overflow Trailer: (None)

Import/Export Bindings (Execute):

12. Drag the EmployeeName and Position attributes from the Data View into the inner subform.



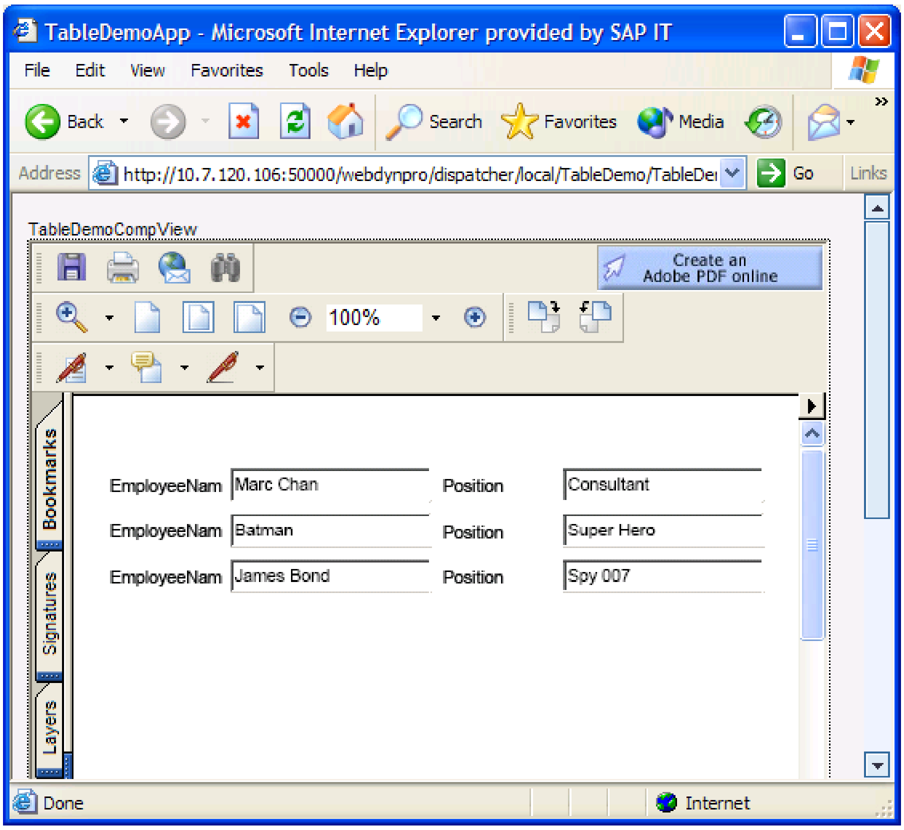
13. Click on Save All Meta-data.

1.3. Testing the Table Output Display

14. To test the application, we will create some testing data for the EmployeeRecords. Go to the Implementation tab of TableDemoCompView and add the following code to the wdDoInit() method.

```
public void wdDoInit() {
        //@@begin wdDoInit()
        Collection myRecords = new ArrayList();
        myRecords.clear();

        IPrivateTableDemoCompView
                .IEmploymentRecordsElement employmentRecordElement1 =
                wdContext.createEmploymentRecordsElement();
        employmentRecordElement1.setEmployeeName("Marc Chan");
        employmentRecordElement1.setPosition("Consultant");
        myRecords.add(employmentRecordElement1);

        IPrivateTableDemoCompView
                .IEmploymentRecordsElement employmentRecordElement2 =
                wdContext.createEmploymentRecordsElement();
        employmentRecordElement2.setEmployeeName("Batman");
        employmentRecordElement2.setPosition("Super Hero");
        myRecords.add(employmentRecordElement2);

        IPrivateTableDemoCompView
                .IEmploymentRecordsElement employmentRecordElement3 =
                wdContext.createEmploymentRecordsElement();
        employmentRecordElement3.setEmployeeName("James Bond");
        employmentRecordElement3.setPosition("Spy 007");
        myRecords.add(employmentRecordElement3);

        wdContext.nodeEmploymentRecords().bind(myRecords);
        //@@end

}
```

15. Deploy and Run the TableDemoApp.



TableDemoApp - Microsoft Internet Explorer provided by SAP IT

Address http://10.7.120.106:50000/webdynpro/dispatcher/local/TableDemo/TableDe

TableDemoCompView

EmployeeNam	Marc Chan	Position	Consultant
EmployeeNam	Batman	Position	Super Hero
EmployeeNam	James Bond	Position	Spy 007

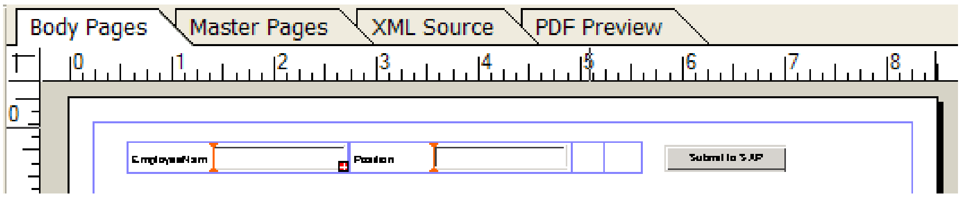
2. Taking Table Input in Interactive Forms Based on Adobe Software

In this section, we will modify the interactive form created in section 1 to take table input for the employee records.

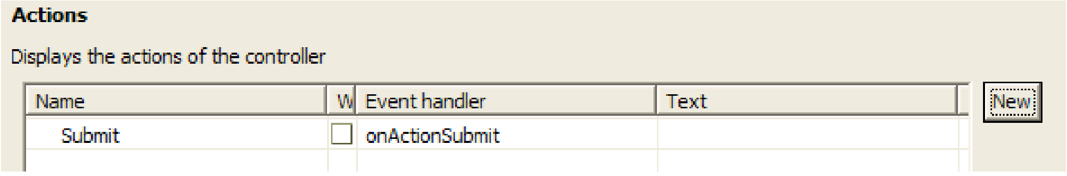
2.1. Adding Actions to the Existing Form

1. We first need to remove the testing data we created in section 1. Go back to the Implementation of TableDemoCompView. Remove all the sample code we created in wdDoInit().

2. Edit the EmployeeForm again. Add a Submit to SAP button to the form.



3. Go back to the Actions tab of TableDemoCompView. Add a new action called *Submit*.

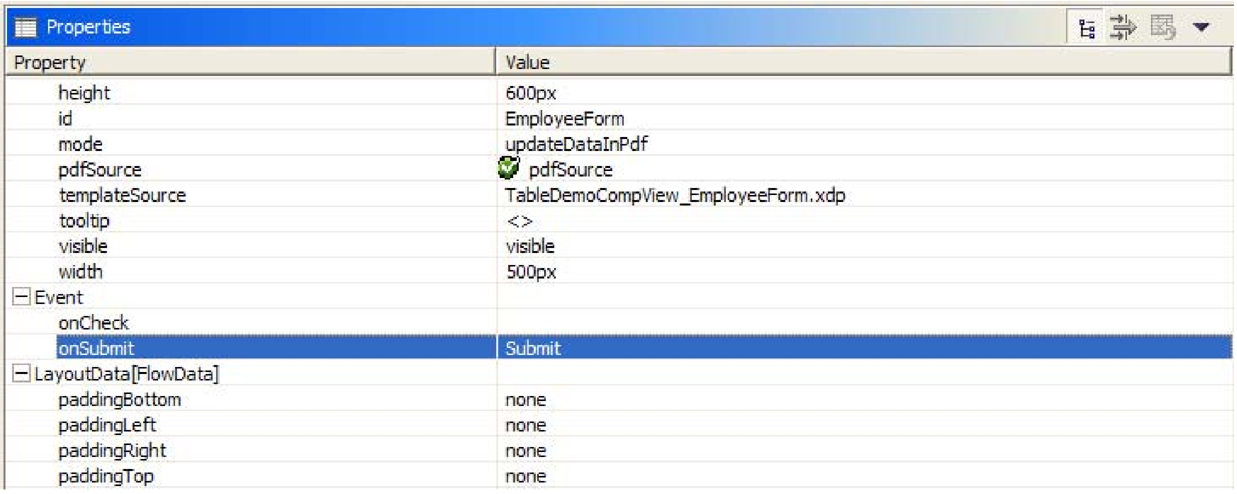


Actions

Displays the actions of the controller

Name	W	Event handler	Text
Submit	□	onActionSubmit	

4. Click on the Layout tab. Go to the Properties of EmployeeForm. Set the Event – onSubmit to "Submit."



Properties

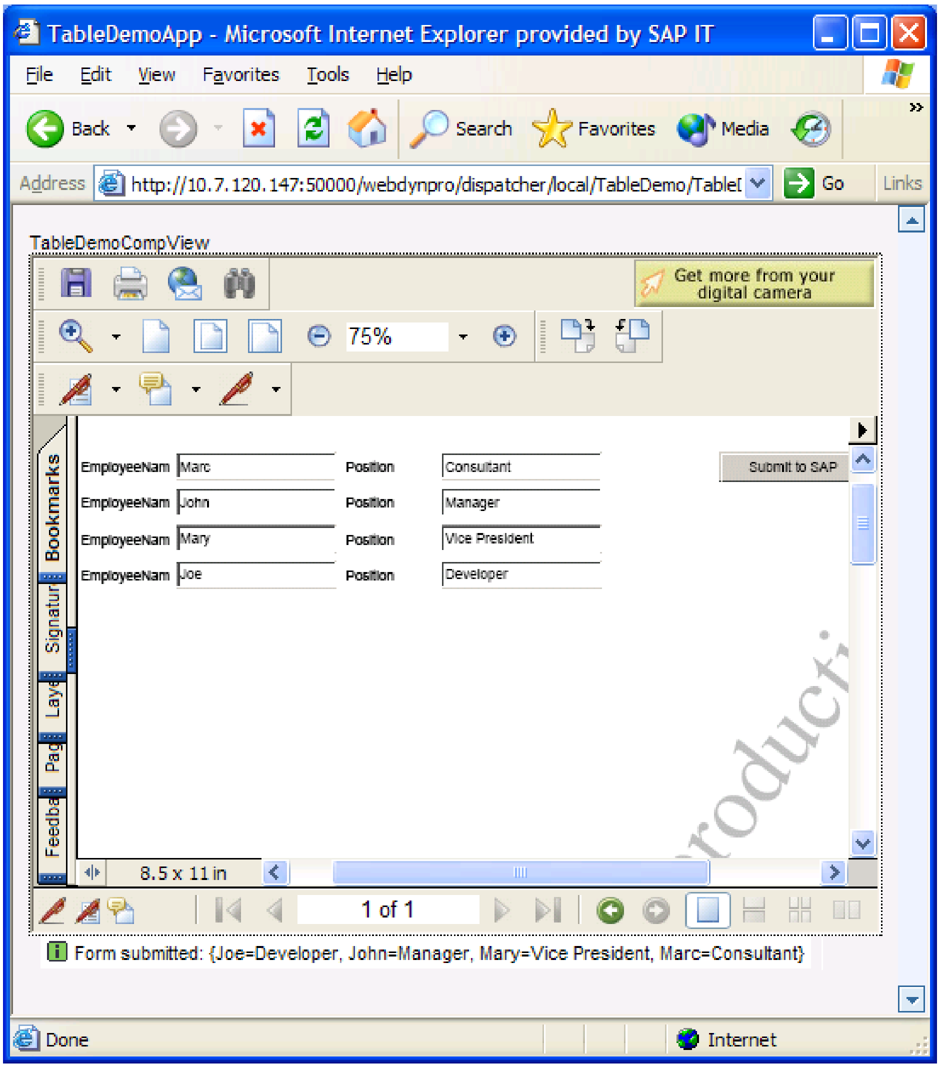
Property	Value
height	600px
id	EmployeeForm
mode	updateDataInPdf
pdfSource	pdfSource
templateSource	TableDemoCompView_EmployeeForm.xdp
tooltip	<>
visible	visible
width	500px
⊟ Event	
onCheck	
onSubmit	Submit
⊟ LayoutData[FlowData]	
paddingBottom	none
paddingLeft	none
paddingRight	none
paddingTop	none

5. Click on the Implementation tab. Add the following code to the onActionSubmit() method. We will display the input employee info as a message in Web Dynpro.

```
public void wdDoInit()
{
  //@@begin wdDoInit()
             Collection records = new Vector();
             IPrivateTableDemoCompView.IEmploymentRecordsElement employmentrecord =
                    null;
             int NUM_RECORDS = 4;
             for (int i = 0; i < NUM_RECORDS; i++) {
                    employmentrecord = wdContext.createEmploymentRecordsElement();
                    records.add(employmentrecord);
             }
             wdContext.nodeEmploymentRecords().bind(records);
  //@@end

}
```

6. Click on Save All Meta-data.

7. Deploy and Run TableDemoApp. Enter some testing data for the four records and press Submit to SAP. You should be able to see them as a message from Web Dynpro.



Address: http://10.7.120.147:50000/webdynpro/dispatcher/local/TableDemo/TableI

TableDemoCompView

EmployeeNam	Marc	Position	Consultant
EmployeeNam	John	Position	Manager
EmployeeNam	Mary	Position	Vice President
EmployeeNam	Joe	Position	Developer

Form submitted: {Joe=Developer, John=Manager, Mary=Vice President, Marc=Consultant}