**Technical Information on developing Dynamic UI based on data**

The idea behind the Dynamic UI is based on the data,

* Object Page Header,
* Object page section,
* Form inside Object Page section
* Table inside Object page section are being created.

Data format will be like this

[{

"Stlan": "",

"Apprflag": "",

"Configtype": "",

"Stlal": "",

"Asset": "",

"Columnno": "0000000001",

"Equnr": "",

"Forapp": "",

"Saknr": "",

"Attachmandtry": "",

"Fieldvalue": "",

"Fromstore": "",

"Reqstatus": "",

"Tax": "",

"Mdgtype": "AM",

"Tokunnr": "",

"Tosales": "",

"Tovendor": "",

"Companycode": "",

"Fromsales": "",

"Fromsdivi": "",

"Mdgtext": "Asset Master",

"Roleid": "A1",

"Tocompanycode": "",

"Todist": "",

"Todivi": "",

"Fromsdist": "",

"Purchaseorg": "",

"Rolename": "General",

"Matnr": "",

"Topurchaseorg": "",

"Maktx": "",

"Mbrsh": "",

"Mtart": "",

"Viewid": "H",

"Viewname": "General Data",

"Viewdespc": "General Data",

"Secsequenceno": "001",

"Sectionname": "Header",

"Subsequenceno": "001",

"Subsecname": "Header",

"Fieldname": "ASSET",

"Fieldlabel": "Main Asset Number",

"Fieldtype": "CHAR",

"Fieldlength": "0012",

"Sequenceno": "001",

"Ismandatory": false,

"Hasf4": true,

"Iseditable": true,

"Url": "",

"Iscritical": false,

"Fieldproperty": "INPUT",

"Attachmenttype": "",

"Attachmentid": "",

"Attachname": "",

"Subsectype": "F",

"Message": "",

"Data": "",

"Validate": "",

"Dupcheck": "",

"Action": "",

"Reqid": "",

"Edit": false,

"Reqtxt": "",

"Save": "",

"Lifnr": "",

"Kunnr": "",

"Materialto": "",

"Fromplant": "",

"Toplant": "",

"Tostore": ""

}]

**Step1: Segregation on data into Object sections, forms, and tables**

Refer ***\_prepareDetailViewofMDGType*** method

In the above data, property ***Sectionname*** and ***Subsecname*** are used to create the sections for the ObjectPageLayout. Data with same ***Sectionname*** belongs to same ObjectPageSection and data with same ***Sebsecname*** belongs to same ObjectPageSubSection

In the above data, property Subsectype says whether the data belongs to form or table.

* + - Subsectype = ‘F’ ---> Form
    - Subsectype = ‘T’ ----> Table
    - Columnno -----> defines the row no.

**Step2: Creating Object Sections, forms, and tables using fragments**

Refer ***\_prepareFormData*** method

Fragment:

* ***DynamicForm.fragment.xml*** – for Form creation
* ***DynamicTable.fragment.xml*** – for Table creation
* ***DynamicAttachment.fragment.xml*** – for Attachment dialog creation

\_prepareFormData: function (path, modelName) {

var aGroups = [],

aFlag = [],

j = 0;

var aForms = [],

aFormFlag = [],

aTableFlag = [],

k = 0,

isTableItem = false,

IsContainerEditable = true;

for (var iFieldCount = 0; iFieldCount < path.length; iFieldCount++) {

var oProcessItem = path[iFieldCount];

if (aFlag[oProcessItem.Subsecname || oProcessItem.subsecname]) {

switch (oProcessItem.Subsectype || oProcessItem.subsectype) {

case "F":

aGroups[j - 1].SectionFormControls.push(oProcessItem);

break;

case "T":

if (parseInt(oProcessItem.Columnno || oProcessItem.columnno) === 1) {

aGroups[j - 1].SectionTableColumns.push(oProcessItem);

}

if (aGroups[j - 1].SectionTableItems.length < parseInt(oProcessItem.Columnno || oProcessItem.columnno)) {

aGroups[j - 1].SectionTableItems.push({

SectionTableControls: []

});

}

aGroups[j - 1].SectionTableItems[parseInt(oProcessItem.Columnno || oProcessItem.columnno) - 1].SectionTableControls.push(

oProcessItem);

break;

}

} else {

aFlag[oProcessItem.Subsecname || oProcessItem.subsecname] = true;

switch (oProcessItem.Subsectype || oProcessItem.subsectype) {

case "F":

aGroups.push({

SectionName: oProcessItem.Sectionname || oProcessItem.sectionname,

GroupHeader: oProcessItem.Subsecname || oProcessItem.subsecname,

FormId: oProcessItem.Subsecname || oProcessItem.subsecname,

FormHeader: oProcessItem.Subsecname || oProcessItem.subsecname,

SectionFormControls: [],

IsFormVisible: true,

IsTableVisible: false,

});

aGroups[j].SectionFormControls.push(oProcessItem);

break;

case "T":

isTableItem = false;

aGroups.push({

SectionName: oProcessItem.Sectionname || oProcessItem.sectionname,

GroupHeader: oProcessItem.Subsecname || oProcessItem.subsecname,

FormId: oProcessItem.Subsecname || oProcessItem.subsecname,

FormHeader: oProcessItem.Subsecname || oProcessItem.subsecname,

SectionFormControls: [],

SectionTableColumns: [],

SectionTableItems: [{

SectionTableControls: []

}],

IsFormVisible: false,

IsTableVisible: true,

});

if (parseInt(oProcessItem.Columnno || oProcessItem.columnno) == 1){ aGroups[j].SectionTableColumns.push(oProcessItem);

aGroups[j].IsTableVisible = true;

}

aGroups[j].IsContainerEditable = true;

aGroups[j].IsTableItem = true;

aGroups[j].SectionTableItems[parseInt(oProcessItem.Columnno || oProcessItem.columnno) - 1].SectionTableControls.push(

oProcessItem);

break;

}

j++;

}

}

aGroups.forEach(function (oGroupItem) {

if (oGroupItem.IsFormVisible) {

if (aFormFlag[oGroupItem.FormId]) {

aForms[k - 1].Groups.push(oGroupItem);

} else {

aFormFlag[oGroupItem.FormId] = true;

aForms.push({

SectionTitle: oGroupItem.SectionName,

SubSectionTitle: oGroupItem.FormHeader,

FormLayout: {

ColumnL: "",

ColumnM: "",

EmptySpanL: "",

EmptySpanM: "",

LabelSpanL: "",

labelSpanM: "",

labelSpanXL: ""

},

Groups: [],

IsFormVisible: true,

IsTableVisible: false,

});

aForms[k].Groups.push(oGroupItem);

k++;

}

} else if (oGroupItem.IsTableVisible) {

if (aTableFlag[oGroupItem.SectionId]) {

aForms[k - 1].Tables.push(oGroupItem);

} else {

aTableFlag[oGroupItem.SectionId] = true;

aForms.push({

SectionTitle: oGroupItem.SectionName,

SubSectionTitle: oGroupItem.FormHeader,

Tables: [],

IsFormVisible: false,

IsTableVisible: true,

});

aForms[k].Tables.push(oGroupItem);

k++;

}

}

});

for (var i = 0; i < aForms.length; i++) {

var columnAr = [];

if (aForms[i].IsFormVisible) {

for (var j = 0; j < aForms[i].Groups[0].SectionFormControls.length; j++) {

columnAr.push(aForms[i].Groups[0].SectionFormControls[j].Columnno);

}

columnAr = columnAr.filter(function (x, i, a) {

return a.indexOf(x) == i;

});

columnAr.sort();

var ary = [];

ary.length = columnAr.length;

for (var k = 0; k < columnAr.length; k++) {

ary[k] = {

SectionFormControls: []

};

for (var l = 0; l < aForms[i].Groups[0].SectionFormControls.length; l++) {

if (aForms[i].Groups[0].IsFormVisible) {

if (columnAr[k] == aForms[i].Groups[0].SectionFormControls[l].Columnno) {

ary[k].SectionFormControls.push(aForms[i].Groups[0].SectionFormControls[l]);

}

}

}

}

aForms[i].Groups = ary;

}

}

aForms.forEach(function (oFormItem) {

if (oFormItem.IsFormVisible) {

var iColumnLen = oFormItem.Groups.length;

oFormItem.FormLayout.ColumnL = iColumnLen;

oFormItem.FormLayout.ColumnM = iColumnLen;

oFormItem.FormLayout.EmptySpanL = 0;

oFormItem.FormLayout.EmptySpanM = 0;

oFormItem.FormLayout.LabelSpanL = 12;

oFormItem.FormLayout.LabelSpanM = 12;

oFormItem.FormLayout.LabelSpanXL = 12;

}

});

var secAr = [];

for (var i = 0; i < path.length; i++) {

secAr.push(path[i].Sectionname || path[i].sectionname);

}

secAr = secAr.filter(function (x, i, a) {

return a.indexOf(x) == i;

});

var mainAr = [];

for (var i = 0; i < secAr.length; i++) {

var obj = {

SectionName: secAr[i],

subSec: []

};

mainAr.push(obj);

for (var j = 0; j < aForms.length; j++) {

if (secAr[i] == aForms[j].SectionTitle) {

mainAr[i].subSec.push(aForms[j]);

}

}

}

var oOPLModel = new sap.ui.model.json.JSONModel();

oOPLModel.setData([]);

oOPLModel.setData(mainAr);

this.getView().setModel(oOPLModel, modelName);

},

The above method will segregate the data into Object sections, forms, and table. And with the data we will create model and bind to the view.

Below code will create Object Page sections in controller level and add Form (DynamicForm.fragment.xml) and Table (DynamicTable.fragment.xml) based on data.

\_addDynamicSections: function (aForms) {

var sectionForm = aForms;

var oObjectPage = this.getView().byId("idDetail");

for (var i = 0; i < sectionForm.length; i++) {

var oObjectPageSectionControl = new ObjectPageSection({

title: "",

titleUppercase: false

});

for (var j = 0; j < sectionForm[i].subSec.length; j++) {

oObjectPageSectionControl.setTitle(sectionForm[i].SectionName + " (" + sectionForm[i].subSec.length + ")");

var oObectPageSubSection = new ObjectPageSubSection({

title: sectionForm[i].subSec[j].SubSectionTitle,

titleUppercase: false

});

if (sectionForm[i].subSec[j].IsFormVisible) {

var oFragment = sap.ui.xmlfragment("app.DynamicApp.fragment.DynamicForm", this);

oFragment.bindElement("sectionsModel>/" + i + "/subSec/" + j);

oObectPageSubSection.addBlock(oFragment);

} else if (sectionForm[i].subSec[j].IsTableVisible) {

var oTableBlock = sap.ui.xmlfragment("app.DynamicApp.fragment.DynamicTable", this);

oTableBlock.bindElement("sectionsModel>/" + i + "/subSec/" + j);

oObectPageSubSection.addBlock(oTableBlock);

}

oObjectPageSectionControl.addSubSection(oObectPageSubSection);

}

oObjectPage.addSection(oObjectPageSectionControl);

}

this.getView().getModel("sectionsModel").refresh();

this.getView().getModel("sectionsModel").updateBindings();

},

**Step3: Add new line item for table section**

Refer ***onPressTableItemAdd*** method

onPressTableItemAdd: function (oEvent) {

debugger;

var title = ovent.getSource().getParent().getParent().getParent().getParent().getTitle(),

mainData = this.getView().getModel("sectionsModel").getData(),

columns = oEvent.getSource().getParent().getParent().getBindingContext("sectionsModel").getObject().Tables[0].SectionTableItems[0].SectionTableControls;

columns = columns.map(item => {

var itemCopy = Object.assign({}, item);

itemCopy.Columnno = "";

itemCopy.Fieldvalue = "";

return itemCopy

});

var obj = {

SectionTableControls: columns

};

for (var i = 0; i < mainData.length; i++) {

for (var j = 0; j < mainData[i].subSec.length; j++) {

if (mainData[i].subSec[j].SubSectionTitle == title) {

for (var k = 0; k < obj.SectionTableControls.length; k++) {

obj.SectionTableControls[k].Columnno = (parseInt(mainData[i].subSec[j].Tables[0].SectionTableItems.length) + 1).toString();

this.data.push(obj.SectionTableControls[k]);

}

mainData[i].subSec[j].Tables[0].SectionTableItems.push(obj);

break;

}

}

}

this.getView().getModel("sectionsModel").refresh();

},

**Step4: Object Header creation**

* The demo header data is stored in ***this.ObjectHeaderData***
* Refer ***onInit*** method line no. 17
* Refer ***\_addHeaderFragment*** method

\_addHeaderFragment: function () {

var oObjectPage = this.getView().byId("idDetail");

var oFragment = sap.ui.xmlfragment("app.DynamicApp.fragment.Header", this);

oObjectPage.addHeaderContent(oFragment);

},

**Step5: Adding SearchHelp for required fields.**

Refer ***handleValueHelpRequest*** method, ***Dialog.fragment.xml***

**Step6: Creating the final Payload**

Refer ***onInputChange*** method

This method is called every time when the Input control value changes. Here we update the payload data.

*for (var i = 0; i < this.data.length; i++) {*

*if (oEvent.getSource().getName() == this.data[i].Fieldname) {*

*this.data[i].Fieldvalue = oEvent.getSource().getValue()*

*}*

*}*

Refer ***onCheckBoxClick*** method

This method is called when check box state changes

for (var i = 0; i < this.data.length; i++) {

if (oEvent.getSource().getName() == this.data[i].Fieldname) {

if (oEvent.getSource().getSelected()) {

this.data[i].Fieldvalue = true;

} else {

this.data[i].Fieldvalue = false;

}

}

}

**Step7: Form Validation and Save**

Refer ***onSave*** method

In the data, we have a property ***Ismandatory***, which says that the field should not be blank. Using the same property, we do the validation. After the data is validated, the data is being sent to backend.