1. **Foundational work: Marking staff as is\_system**

Access to the Form Info hyperlink is controlled by the is\_system database field. This field cannot be directly added to a form, but a pre-existing form can be found at: <https://github.com/myEvolv-Development-Community/myEvolvCode/blob/main/Form%20Design/Exports/Staff%20Is%20System%20Flag.json>

Add this field to the a staff setup form, or open a staff profile and reload the profile using the imported form using the script here: <https://github.com/myEvolv-Development-Community/myEvolvCode/blob/main/JavaScript%20Functions/Update%20Form%20Parameters.md>

If the is\_system flag is completely unavailable, use the following code in the JavaScript console to open the form designer:

openRadWindow('',`https://${Form.formObject.serverHandling||window.parent.Form.formObject.serverHandling}/FormDesigner.aspx?parent\_value=${Form.formObject.formFamilyId}&form\_family\_id=${getDataValue("form\_family", "form\_family\_name", 'Form Designer', 'form\_family\_id')}&fdmode=DEV&keyValue=${Form.formObject.formHeaderId||getDataValue("form\_header", "form\_code", Form.formObject.formCode, 'form\_header\_id')}#!`)

1. **Helper subreports for Form Designer:**

**Snippet 1.***Events That Use a Specific Form – include category\_name \*, event\_name \*, and form\_header\_id\*\**

select

event\_category.category\_name ,

event\_category.event\_category\_id,

event\_definition.event\_definition\_id,

event\_definition.event\_name,

form\_header.form\_header\_id,

form\_header.form\_name

from

form\_header, event\_definition

inner join event\_category on event\_definition.event\_category\_id = event\_category.event\_category\_id

where form\_header.form\_header\_id

= event\_definition.form\_header\_id\_new

or form\_header.form\_header\_id

= event\_definition.form\_header\_id

**Snippet 2.**

*Other Forms Using this (Sub)Form – include form\_family\_name \*, form\_name \*, and sub\_form\_header\_id \*\**

select \* from form\_view

**Snippet 3.**

*Navigation Tabs Using this Form – include breadcrumb \* and form\_header\_id \*\**

select all\_form\_set\_modules\_view.\*,

module\_name + ' > ' + sub\_module\_name + ' > ' + set\_name + ' > ' + all\_form\_set\_modules\_view.tab\_caption as breadcrumb,

form\_name,

form\_header\_id

from all\_form\_set\_modules\_view

inner join form\_set\_members on all\_form\_set\_modules\_view.form\_set\_members\_id = form\_set\_members.form\_set\_members\_id

inner join form\_header on form\_header.form\_header\_id = form\_set\_members.default\_form

*Note*:  
\* Visible on Subreport

\*\* Join field

1. **Modify the Form Designer**

Navigate to Form Designer, and open any form.

Open the form header section for the form (before the form lines) and in the console, enter the following:

**Snippet 4.***Launch Form Designer*

openRadWindow('',`https://${Form.formObject.serverHandling||window.parent.Form.formObject.serverHandling}/FormDesigner.aspx?parent\_value=${Form.formObject.formFamilyId}&form\_family\_id=${getDataValue("form\_family", "form\_family\_name", 'Form Designer', 'form\_family\_id')}&fdmode=DEV&keyValue=${Form.formObject.formHeaderId||getDataValue("form\_header", "form\_code", Form.formObject.formCode, 'form\_header\_id')}#!`)

Add a new variable to the form. Name the variable form\_header\_id. Set the default value to keyValue.

Add the subreports we created earlier.

1. **Updates to Event Setup**

**Create a new virtual view subreport:**

**Snippet 5.***Workflow usage for Events*

select

workflows.workflows\_id,

workflows.description as workflow\_grouping,

workflows\_rules.workflows\_rules\_id,

workflows\_rules.description as workflow\_caption,

event\_definition.event\_definition\_id,

event\_definition.event\_name,

'I' + stuff(

iif(

exists(

select \*

from workflows\_event\_triggers

where

workflows\_event\_triggers.workflows\_rules\_id = workflows\_rules.workflows\_rules\_id

and workflows\_event\_triggers.event\_definition\_id = event\_definition.event\_definition\_id

)

, ' and is a trigger for', '') +

iif(

exists(

select \*

from workflows\_event\_triggers

inner join workflows\_events on workflows\_event\_triggers.workflows\_event\_triggers\_id = workflows\_events.workflows\_event\_triggers\_id

where

workflows\_event\_triggers.workflows\_rules\_id = workflows\_rules.workflows\_rules\_id

and workflows\_events.event\_definition\_id = event\_definition.event\_definition\_id

)

, ' and is scheduled by', ''),

1, 6, '') as event\_use

from workflows

inner join workflows\_rules on workflows.workflows\_id = workflows\_rules.workflows\_id

cross join event\_definition

where stuff(

iif(

exists(

select \*

from workflows\_event\_triggers

where

workflows\_event\_triggers.workflows\_rules\_id = workflows\_rules.workflows\_rules\_id

and workflows\_event\_triggers.event\_definition\_id = event\_definition.event\_definition\_id

)

, ' and is a trigger for', '') +

iif(

exists(

select \*

from workflows\_event\_triggers

inner join workflows\_events on workflows\_event\_triggers.workflows\_event\_triggers\_id = workflows\_events.workflows\_event\_triggers\_id

where

workflows\_event\_triggers.workflows\_rules\_id = workflows\_rules.workflows\_rules\_id

and workflows\_events.event\_definition\_id = event\_definition.event\_definition\_id

)

, ' and is scheduled by', ''),

1, 5, '') <> ''

Subreport includes:

Event\_use \*

Workflow\_grouping\*

Workflow\_caption\*

Event\_definition\_id \*\*

*Note*:  
\* Visible on Subreport

\*\* Join field

Navigate to Event Setup and open any event form.

Right click on a worker role in Security for Event subform, and use the script from Snippet 4.

In Form Designer, add a box for the Amend permission and mark it visible on subform.

Ensure the Driving Form Line is set to “Worker Role”

Right-click on the main form and open the form designer

Add a variable, formatted as a regular string or unique identifier. Mark the field as not visible and set the default value to keyValue.

At the end of the form, add a new variable, formatted as a Boolean field. Add the following to the on click script:

**Snippet 6.**

*Copy event contents*

function resolveAfterNSeconds(n){

return new Promise(

(resolve) => {

setTimeout(

() => {resolve()},

n)

}

);

}

async function f1(n) {

var x = await resolveAfterNSeconds(n);

if(!newWin.document.querySelector(".blockUI")) {

function readFormLine(fl) {

if (fl.typeCode == "FK") {

return [fl.lutValue.value, fl.lutValue.description, fl.lutvalue?.prompt]

} else if (fl.typeCode == "SF") {

return fl.

sfValue.

filter(row => row.formMode != "ADD" && row.formMode != "DELETE" && row.formMode != "NEW").

map(row =>

Object.fromEntries(row.

FormLines.

map(fl => [fl.columnName, readFormLine(fl)]).

filter(entry => entry[1] !== null && !(Array.isArray(entry[1]) && entry[1].length < 1))

)

)

} else {return fl.value}

}

newWin.updateCell = (row, keyValuePair) => {

let subformcell = row. // Find the column in the empty subform row to update

FormLines.

find(c => c.columnName == keyValuePair[0]);

if (subformcell) {

let newValue = keyValuePair[1];

Array.isArray(newValue) ? subformcell.SetValue(...newValue) : subformcell.value = newValue;

subformcell.subFormHasData = true

DirtyFormField.\_defaultChangeDetector(subformcell) // Register subform as dirty to prompt myEvolv to save

} else console.warn(`This subform does not contain a ${keyValuePair[0]} field!`)

}

newWin.commitSubformChange = (subform, subformRow) => {

subform.isDirty = true

subform.subFormHasData = true

subformRow.isDirty = true

subformRow.hasData = true

subformRow.subFormHasData = true

}

newWin.insertIntoSubformJSONById = (formLineId, cellContents) => {

let subform = newWin.Form.getFormLineById(formLineId); // Get the subform element

if (subform) {

let subformRow = subform. // Find the empty subform row at the end

sfValue.

find(row => !row.isDirty && row.keyValue?.slice(0, 3) === 'new' && row.keyValue?.length > 3)

Object.entries(cellContents).forEach(entry => newWin.updateCell(subformRow, entry))

newWin.commitSubformChange(subform, subformRow)

subform.RefreshGrid()

} else console.warn(`No subform was found with the ID ${formLineId}`); // Also return undefined if no subform found

}

Form.

formObject.

FormLines.

filter(fl => fl.columnName && fl.typeCode != 'RSF').

map(fl => ({formLinesId: fl.formLinesId, typeCode: fl.typeCode, value: readFormLine(fl)})).

filter(entry => entry[1] !== null && !(Array.isArray(entry[1]) && entry[1].length < 1)).

forEach(entry => {

if (entry.typeCode == 'SF') {

entry.value.forEach(i => newWin.insertIntoSubformJSONById(entry.formLinesId, i, newWin));

} else if (entry.typeCode == "FK") {

newWin.Form.setFormElementById(entry.formLinesId, entry.value[0]);

} else {

newWin.Form.setFormElementById(entry.formLinesId, entry.value);

}

}

)

Form.setFormElement("event\_name", getFormElement("event\_name") + " " + new Date().toISOString().slice(0, 10));

newWin.Form.setFormElement("event\_name", newWin.getFormElement("event\_name") + " - copy")

} else {

f1(n);

}

}

var newWin = window.open(`https://${Form.formObject.serverHandling}/Form.aspx?form\_header\_id=${Form.formObject.formHeaderId}&parent\_value=${parentValue}&key\_value=&is\_add\_allowed=true&is\_edit\_allowed=true&is\_delete\_allowed=true&mode=ADD&isCompleteScheduledEvent=false#!`);

f1(1000);

1. **Adding navigation scheme maintenance to formset maintenance**

Navigate to Formset Maintenance

Open any formset member

Launch form designer

Add a new sub form

Family: Security Formsets

Subform: Security Formsets

Save and reload the formset maintenance form.

Launch the form designer for the security formsets subform.

Set the driving form line to “Security Scheme”

Now admins can pull up a formset member and specify all navigation schemes and their access to the form at once.