

Quick Entry Page Instructions

1. **Build the Quick Entry data record.** Decide which fields to put on the Quick Entry pages. Copy these fields to a separate data record with G_FORM_ID as the key. (Drag and drop from the delivered record to retain lookup information.)
 - a) EMPLID
 - b) EMPL_RCD
 - c) EFFDT
 - d) ACTION
 - e) ACTION_REASON
 - f) POSITION_NBR
 - g) DEPTID
 - h) JOBCODE
 - i) PAYGROUP
 - j) COMPRATE

2. **Build the Quick Entry page.** (You may want to just add the subpages and skip to making the component, so you can test while you add fields.)
 - a) Put subpage G_FORM_SMTASK_SBP at the top,
 - b) Then the data fields (you can wait to add most of these until you have the component registered) ,
 - i. Drag and drop Related Display fields from the delivered pages.
 - c) Then subpages G_FORM_COMMENT_SBP (for form comments) and
 - d) G_FORM_NAV_SBP (for action buttons).
 - e) Set the page's stylesheet.

3. **Build and register the Quick Entry component.** Set it to Add Mode. For now, put G_FORM_INSTALL as the search record. Turn off the Folder Tabs and Display Hyperlinks. Disable the Toolbar. Put the following pages on it:
 - a) The Quick Entry page
 - b) G_FORM_ANY_RSLT (or a G_FORM%RSLT page with an appropriate data subpage)
 - c) G_FORMLIST
 - d) G_FORM_WRK

CHECKPOINT: You should be able to navigate to the page, manipulate the fields, and save it.

4. **Build Form Type.** Add the Form Type.
 - a) Set it to Active.
 - b) Fill out the Form Type Definition section with Virtual Approver turned off.
 - c) On Form Task Navigation page, make an Add Task entry.
 - d) Enter a Step row for your Quick Entry component.
 - e) Save.

5. **Add Quick Entry Homepage Navigation.** If you don't already have a Quick Entry home page, follow these steps to make one. If you have one already, skip to Step 6.
 - a) In App Designer, copy component G_HOMEPAGE_TMPLATE to G_QEADD_HOME.
 - b) Register the new component.
 - c) Add a Form Category Table entry for Quick Entry.

d) On Homepage Table, add an entry for Quick Entry Home Page.

6. Add Navigation for this Quick Entry component.

- a) From Manage GT eForms™ -> Homepage Management -> Homepage Table, pull up the Quick Entry Home Page.
- b) Click Add Action. Fill out the page. For Link Type, use Form Link, and the Form Type values you created above. OK, and Save.

7. Add Component PeopleCode. Add the following to your Quick Entry component:

a) Component PreBuild (Update form type):

```
Global any &GLIB;
```

```
&GLIB.CPC.AddPreBuild("[form type]", "ADD", "DEFAULT");
```

b) Component PreBuild:

```
Global any &GLIB;
```

```
&GLIB.CPC.AddPostBuild();
```

CHECKPOINT: You should be able to navigate using Home Page navigation, see the Task Header information, and appropriate action buttons. Save won't work right now.

8. **Insert a Search Component.** This is a component that hosts the search record you want to use. Name should be in the form G_FORM_xxx_SRCH. Find one with keys that match your Quick Entry record (except for G_FORM_ID). If none exists, you'll need to make one; copy another, or follow the steps in the Developer's Guide.

- a) Back on the Form Type Table, under the Add Task, enter a Step row for the Search Component and move it first. Turn off the Count checkbox for this Step.
- b) Add another Step row (last) for the G_FORM_ANY_RSLT page.
- c) Save.
- d) Change the search record for your Quick Entry component. Name should be in the form G_FRMLST_xxx_VW. Find one with keys that match your Quick Entry record (including G_FORM_ID). If none exists, you'll need to make one; copy another, or follow the steps in the Developer's Guide. Save your changes.

CHECKPOINT: From the Quick Entry Home Page, you should see an appropriate search page. When you choose a row from the search results, the key fields should carry forward onto your Quick Entry Page. When you click Submit, you should be asked to confirm, and then go to the Results page, where it says you have Authorized the form.

For the following steps, the GT eForms™ 2.8 Developers Guide has further instructions.

9. **Create the Component Interface to the target component.** This is how you'll update the system from your Quick Entry form.

- a) File->New->Component Interface. Name it G_[component name]_CI.
- b) Select Component
- c) Click No on when asked "Do you want to default the properties based on the underlying Component definition?"

- d) Drag fields from Component to Component Interface (left to right) that will need to be read/updated.

10. **Setup Component Interface Security.** (Note: For PeopleTools 8.48 and above, Integration Broker uses the security of the final approver to access the associated component interface(s).)

- a) Open Permission List (Use G_FORM_CI unless another is desired)
- b) Go to Component Interface tab
- c) Add the component interface
- d) Click Edit
- e) Click Full Access
- f) Save

CHECKPOINT: Test Component Interface in Application Designer. Open the Component Interface, Select Tools > Test Component Interface. Enter in key values and click Get. Change values. Right click the CI at the top of the Component Interface Tester and select Save. The target component should be updated.

11. **Create the Component Interface Wrapper.** This is a PeopleCode program that directs how data is fed into the Component Interface.

- a) Create new application package (name it the same as the CI) and class (Name it based on the description of the component, like JobDataCI).
- b) Follow example code. For example, Code 6: Component Interface Wrapper Example (G_ALTERNATE_CI:AlternateCI) in the appendix of the Developer's Guide, or use a sample from your GT eForms™ installation. Your class should:
 - i. Extend the GT-delivered class G_CI:Parent.
 - ii. Have a constructor (method named the same as the class itself) with an input parameter for each high-level key in your target component, plus booleans for **&DisplayAll** and **&Correction**. Example:

```
method JobData(&EMPLID As string, &EMPL_RCD As number, &DisplayAll As
boolean, &Correction As boolean);
```

- iii. Have a method for each distinct action you want to perform on the target component, such as **InsertRow**, **DeleteFutureRows**, etc.
- iv. **Component Interfaces expose the rowsets and rows of the underlying component as Collections and Items.** For convenience working in the CI Wrapper, you'll make an APIObject property for each Collection (rowset) and Item (row) that you need to work with in the target component, like this:

```
property ApiObject oJobCollection;
property ApiObject oJob;
```

You can basically just update the listings from your sample to the names of the records in your target component that you are going to work with. Follow the sample code carefully to point your objects at the right data in the CI's collections.

- v. **Model the order that you fill in fields after the way users enter fields manually. Remember that you are kicking off Field-level PeopleCode events as you enter the values.**

OPTIONAL CHECKPOINT: The CI Wrapper that you just wrote is typically triggered from Message Handler PeopleCode, which we are setting up next. There is quite a bit of Integration Broker setup before you can test your CI Wrapper. If you like, you can test your CI Wrapper by calling it from a test PeopleCode event, like FieldChange on a button. Alternatively, you can just finish the IB setup and test it then.

12. **Create the Message Handler.** Completing your Quick Entry page is going to trigger an Integration Broker service operation. This will pass the workload to the Integration Broker to actually enter the transaction into the system through your CI and CI Wrapper. The Handler is the PeopleCode that runs when the service operation is processed.
 - a) **Create a new application package with the same name as your Quick Entry Form Type (G_FORM_XXXX) and a new class in it (Descriptive of your Form Type, with Handler in the name - e.g. JobQuickEntryHandler).**
 - b) **Create the class by following example code. For example, Code 7: Handler Example (G_FORM_ALTN:AltnHandler) in the appendix of the Developer's Guide, or use a sample from your GT eForms™ installation. Your class should:**
 - i. **Extend the GT-delivered class G_FRAME:IntegrationBroker:GTHandler.**
 - ii. **Declare the &DATAREC component record variable (`Component Record &DATAREC;`).**
 - iii. **Have a constructor (method named the same as the class itself) with no parameters, and only this line of code:**

```
%Super = create G_FRAME:IntegrationBroker:GTHandler();
```

- iv. **Have an `OnNotifyHook (&MSG as Message)` method. This is where you put your handler code. You're going to instantiate your CI Wrapper object to open the Component Interface, and then you're going to call the action method(s) you defined, and pass in the data from the Message. (GT eForms™ will have put the data from your Quick Entry component in the Message.) Then you'll save the CI to commit the transaction.**
- v. **Make sure that in your `OnNotifyHook` method you point `&DATAREC` to the Quick Entry Data Record in the message rowset, like this:**

```
&DATAREC =  
&_MSG.GetRowset() (1).GetRowset(Scroll.G_FORM_QEJOB) (1).G_FORM_QEJOB;
```

13. **Create Integration Broker Message.** This is the container that will carry your data from the Quick Entry pages to the Handler, so it can be inserted into the system by your CI Wrapper through the Component Interface.
 - a) **PeopleTools>Integration Broker Setup>Messages>Add a Value**
 - b) **Type=Rowset**
 - c) **Name the Message with the same name for the form record (G_FORM_XXXX)**
 - d) **Insert "v1" for version: click Add**
 - e) **Insert appropriate description and owner**
 - f) **Click Add Record to Root; Select G_FORMLIST as the root record**

- g) Click on G_FORMLIST and add the form data record G_FORM_XXXX as a child record
- h) Save

14. Create Integration Broker Service.

- a) PeopleTools>Integration Broker Setup>Services
- b) Use the same name as the form record (G_FORM_XXXX)
- c) If you get an error saying the broker isn't configured correctly and it won't save, a possible solution is to go to IntegrationBroker>Configuration>Service Configuration and enter ":" for Target Location.
- d) Insert appropriate description and owner
- e) Save

15. Create Integration Broker Service Operation.

- a) On the Service page from the previous step, enter in the name of the service operation. Use the same name as the form record (G_FORM_XXXX).
- b) Select "Asynchronous - One Way" for the Operation Type
- c) Press Add
- d) Add the following information:
- e) Description
 - i. Object Owner ID
 - ii. Check Generate Local-to-Local
 - iii. Version: v1
 - iv. Message.Version: G_FORM_XXXX.v1
 - v. Queue Name: G_FORM_ACTIONS
 - vi. Save
- f) Go to Handler Tab and enter the following:
 - i. Handler name - Same name as your Handler class
 - ii. Type: OnNotify
 - iii. Implementation: Application Class
 - iv. Status: Active
 - v. Press Details and enter:
 - 1. Package Name: G_FORM_XXXXd_UUU
 - 2. Path: ":"
 - 3. ClassID: The class name of the handler
 - 4. Method: OnNotify
 - 5. OK
 - vi. Save

16. Web Service Security (These steps allow the eForms Administrators to see the error messages for the IB messages)

- a) In 8.5, there is a link from the Service Operation>General tab. Just click Service Operation Security, enter GTFORMADMIN, and Save.
- b) For 8.49 or below: Open GTFORMADMIN Permission List
- c) Go to the Web Services tab
- d) Add the G_FORM_XXXX service that was created in step 7.
- e) Click on Full Access (All)
- f) Save

- 17. Link Integration Objects to Form Type.** The final step is to link the form to the integration objects that we have created.
- a) **Manage GT eForms > Form Type Table**
 - b) **Specify the Service Operation created in step 6.**

The integration is now complete and the objects are all linked in the following order:

1. **Form Type Table**
2. **IB Message**
3. **IB Service Operation (>Service>Queue)**
4. **Message Handler**
5. **Component Interface Helper**
6. **Component Interface**

CHECKPOINT: Authorize a form and open Integration Broker > Service Operations Monitor > Asynchronous Services Monitor and look for the transaction. If the Service Operation appears and goes to Done, the service op successfully published. If the Subscription Handler appears and goes to Done, it should mean that the Handler and the CI Wrapper worked. Check the target component for your data!

- 18. Pre-populate the Quick Entry page.** For production use, you'll want the Quick Entry page to pull in existing data and allow changes to it. The GT Framework has a generic load method that can populate the page for you.
- a) **In the Component PostBuild event, add the following line to the end:**

```
%This.FRM.PopulateFormFromRowset([Rowset structure]);
```

The Rowset structure just needs to be a standalone rowset object built from the records you want to pull information in from. It shouldn't be actually filled with data. You can create the rowset right in the method call. In this example, we indicate we want to pull data from the JOB record, and from subordinate JOB_EARNS_DIST records.

```
%This.FRM.PopulateFormFromRowset(CreateRowset(Record.JOB,  
CreateRowset(Record.JOB_EARNS_DIST)));
```

The method will handle all the key assignments, and will match fields by name. The method should work for most Quick Entry forms as long as you use matching field names.

- b) **You can also add a Reload button to your page, which will pull the original data in again. It will also do so based on the value of an Effective Date field if there is one; so if you set the effective date to before the current row, it will pull in the data that was effective as of your date.**
- c) **Add a button to your page that runs the FieldChange PeopleCode for the field G_FORM_WRK.G_LOAD_BTN.**

CHECKPOINT: Open the page for an employee, and it should load the page with data. Change the Effective Date (if you have one) to a date before the current row in the target component and hit **Reload**; the data should change to the appropriate historical values. You can then change the Effective Date as you wish.