

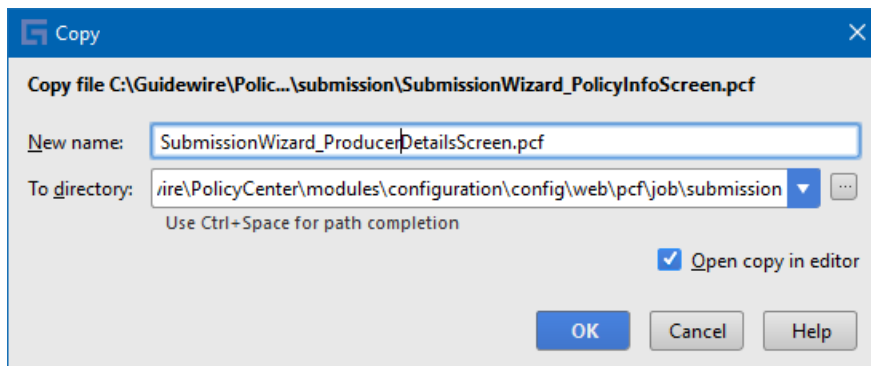
3.4 Solutions



Solution

1. Create a new screen for Producer of Record.

Copy `SubmissionWizard_PolicyInfoScreen.pcf` and name it `SubmissionWizard_ProducerDetailsScreen.pcf`



2. Create content in the new screen.

- In `SubmissionWizard_ProducerDetailsScreen.pcf`, delete every component except the toolbar.
- Copy and paste the `InputSetRef` for Producer of Record from `SubmissionWizard_PolicyInfoDV.pcf` to `SubmissionWizard_ProducerDetailsScreen.pcf`.
- You'll need to rename the parameters in the `InputSetRef` to match variables in the new screen. For example, change `policyPeriod` to `period`.

SubmissionWizard_ProducerDetailsScreen.pcf

Screen : SubmissionWizard_ProducerDetailsScreen submission, period, openForEdit, jobWizardHelper

PreviousNext...

Toolbar Button Set Ref

Shared section mode: Audit

Toolbar Button Set : JobWizardToolbarButtonSet policyPeriod, job

Release Lock

[button](E)

Save Draft

Handle

The rate/quote request is being processed.

Detail View

Input Column

Input Set Ref

Input Set : PolicyInfoProducerOfRecordInputSet policyPeriod, forUseIn, producerC

Producer of Record

Organization *producer

Producer Code *policyPeriod.EffectiveDatedFields.Produ...

Page Configuration

Text

Properties Properties

Input Set Ref

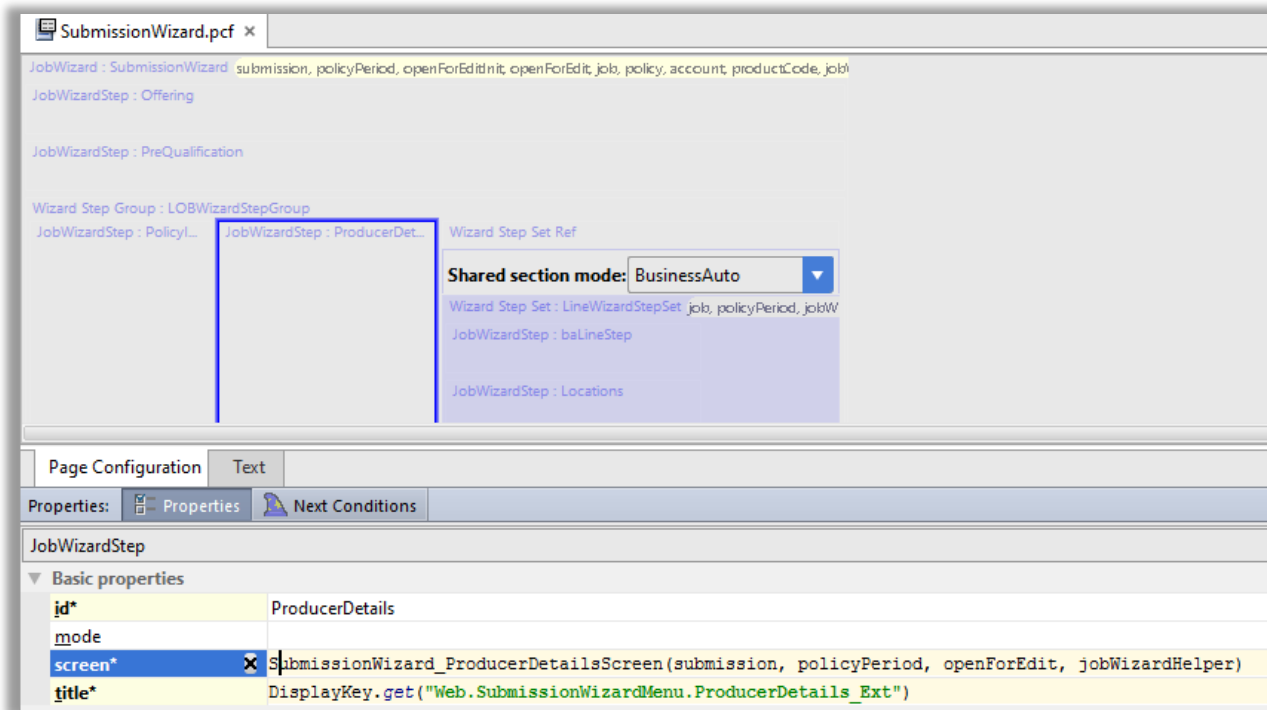
Basic properties

def* PolicyInfoProducerOfRecordInputSet (period, ProducerStatusUse.TC_SUBMISSIONOKAY)

id

3. Add a new JobWizardStep for Producer Details in SubmissionWizard.pcf.

- Duplicate the PolicyInfo step
- Change the screen attribute and point it to the new Producer Details screen.
- Create display keys.



4. In the PolicyCenter application, press Alt + Shift + L to reload changes to pcf.

3.5 References



Reference

Job Wizard

A **job wizard** has job wizard steps that individually points to a screen and therefore the screens in a job wizard may have an order. Only one screen is active at a time in a job wizard.

The screens in a job wizard have an order (though it may be possible for a user to traverse the screens out of order). Job wizards collect information and perform operations while guiding the user through a complex business transaction. The UI functionality of job wizards is captured in a PCF. The actual Job process is usually defined in a Gosu class which coordinates the policy transaction and the job wizards interacts with. So, SubmissionWizard is a PCF, but SubmissionProcess is a Gosu class.

Information in a job wizard includes:

- Actions menu.

- Dependent steps, which are an ordered set of labels, each of which represents a single job wizard step and maps to a single screen. Some steps may not be accessible until a given step is complete.
- Independent steps, which are a set of labels, each of which maps to a single screen. Independent steps do not have a logical order and can be accessed at any time, regardless of what steps have been completed in the job wizard.
- Tab bar, which is the set of tabs that run across the top of the application.
- Job wizard infobar, which lists information about the policy, the Account, or the Line of Business.
- Job wizard toolbar, which contains Back and Next buttons to navigate through the job wizard as well as other buttons relevant to the current step.
- Screens, which contains the content for each step of the job wizard.



Reference

Job Wizard Configuration

There are four levels of configuration for job wizard

- JobWizard
- JobWizardGroup
- JobWizardStepSet and JobWizardStepRef
- JobWizardStep

JobWizard

- The **CountAsWork** property allows a user to resume working in a wizard after leaving the wizard without saving. If set to true and user navigates away before finishing a wizard, wizard step is listed as Unsaved Work within the same active user session.
- If **canEdit** evaluates to true, the job wizard is editable when in edit mode. Some conditions include: edit permission, job state.
- If **canVisit** evaluates to true, the users can view the job wizard. Some conditions include: view permission.
- Entry point definition
 - Syntax: `EntryPointName(ObjectName1: ObjectType1, ObjectName2: ObjectType2 ...)`.
 - Example: `SubmissionWizard(submission : Submission, policyPeriod : PolicyPeriod)`.
- Each object expected in each entry points must be declared on Variables tab
- Navigating to a job wizard
 - Syntax: `EntryPointOfLocation.go(objectsToPass)`.
 - Example: `JobForward.pcf → RenewalWizard.go(job as Submission, policyPeriod, wizardStep)`.

A **WizardStepGroup** is a set of wizard steps which are indented and have a single parent label. For example: SubmissionWizard -> Policy Contract group.

A **WizardStepSet** is a set of wizard steps which have shared logic (visible, available, mode) and can be inserted into a wizard directly or into a wizard group.

- In the base application, there are sets of wizard steps for the submission wizard which change depending upon the line of business. For example, one set of wizard steps is used for personal auto submission and a different set of wizard steps for the workers comp submission.

- The availability and visibility logic for a wizard step set is accessible in the properties window. The step set can use the mode property to select the line of business for the submission.

A **WizardStepSetRef** is a reference to the PCF where the step set is stored.

A **JobWizardStep** can be dependent and independent. The order of the steps is determined by the order in which they appear in the job wizard PCF. Dependent steps are always in top box and Independent steps are always in bottom box. Child steps of a job wizard step group are indented.

- To make a step independent, set its independent attribute to true.
- The independent steps' title and title icon are set in the job wizard widget.
- For example, SubmissionWizard → Tools section contains the independent steps.

Screens are referenced by job wizard steps.

- Syntax: ScreenPropertyOfJobWizardStep = EntryPointOfScreen(objectsToPass).
- A screen PCF contains references to all embedded content.
- Screen related labels.
 - Title determines the text displayed in the title bar of a screen.
 - Label determines the text displayed in the sidebar.
- Buttons on screen
 - Wizard button (previous / next) behavior is handled automatically.
 - Toolbar buttons are displayed based on job type and status.

