

View and edit objects with Popups



February 10, 2014

© Guidewire Software, Inc. 2001-2014. All rights reserved.
Do not distribute without permission.

Guidewire training materials contain Guidewire proprietary information that is subject to confidentiality and non-disclosure agreements. You agree to use the information in this manual solely for the purpose of training to implement Guidewire software solutions. You also agree not to disclose the information in this manual to third parties or copy this manual without prior written consent from Guidewire. Guidewire training may be given only by Guidewire employees or certified Guidewire partners under the appropriate agreement with Guidewire.

Lesson objectives

- By the end of this lesson, you should be able to:
 - Describe the functionality of popups
 - Create popups used to view and edit objects

This lesson uses the notes section for additional explanation and information.
To view the notes in PowerPoint, select View → Normal or View → Notes Page.
When printing notes, select Note Pages and Print hidden slides.

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

2

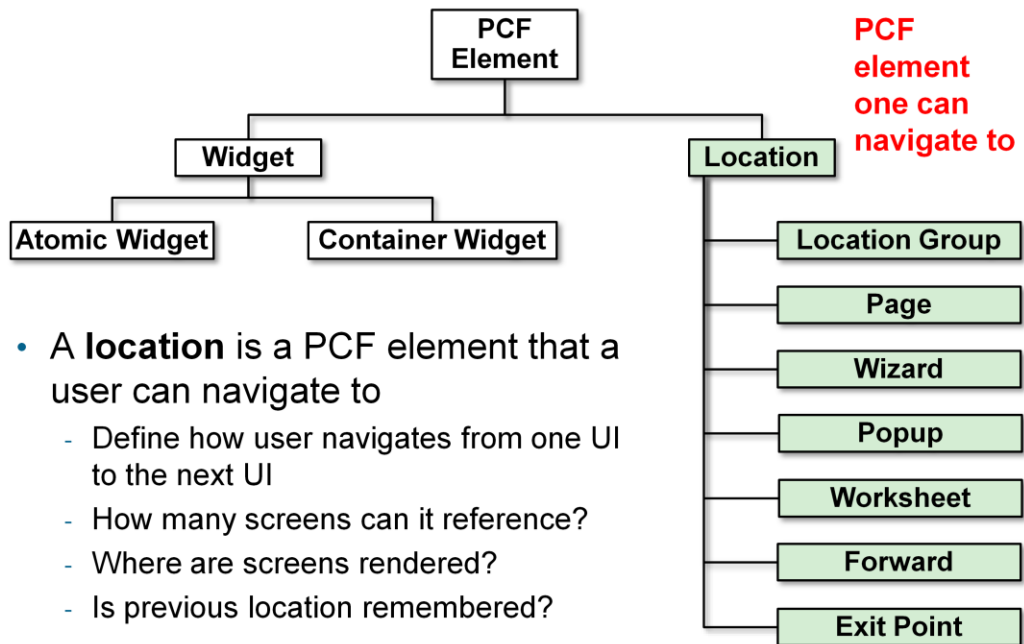
G U I D E W I R E



Lesson outline

- Popup fundamentals
- Create a popup

Locations



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

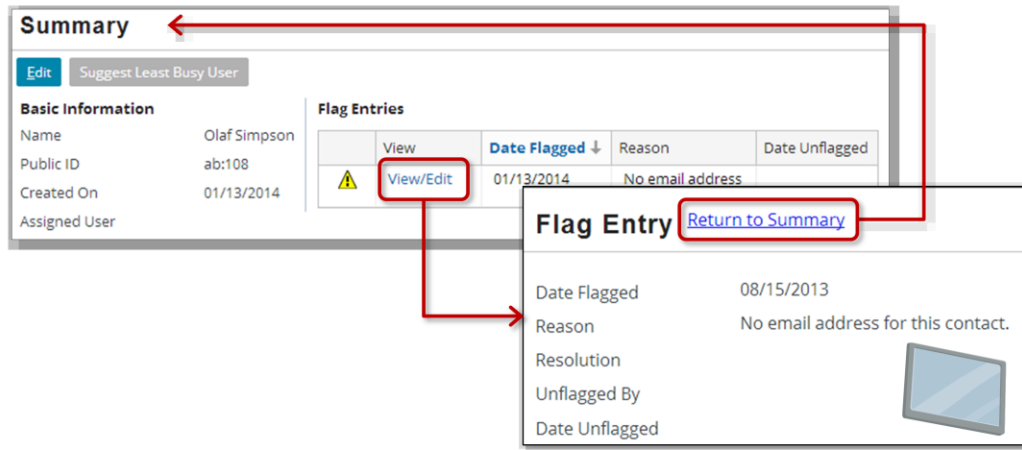
4

G U I D E W I R E

Both Widget and Location are conceptual representations in this diagram. There are no `<Widget />` or `<Location />` elements.

Popups

- A **popup** contains a single screen and returns the user to the previous location once the popup is closed
 - Automatically has "Return to <previous location>" link



Popup screens are designed to mimic the functionality of a true popup (which is an entirely separate window). True popups are difficult to architect because there is no easy way to avoid synchronization errors in which the system tries to update the same object in two separate windows. Popup locations as architected in Guidewire offer virtually the same functionality while avoiding the synchronization and usability issues.

In the slide example, the FlagEntryPopup popup is used to view existing flag entry objects and, if the user has sufficient permissions, edit them. However, it cannot be used to create flag entry objects. Flag entries are created only by the application business logic, not manually by a user.

The slide example shows a popup for a user with permissions to only be able to view the object data.

Why not use browser popup windows?

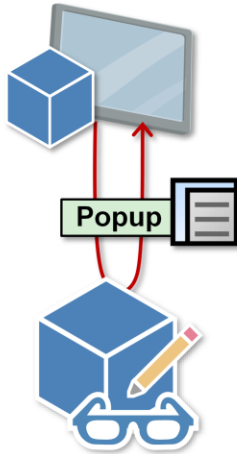
- Many users configure their web browsers to block popups
 - Even if website or URL is from trusted source
- Passing context and objects between browser windows can cause synchronization problems
 - Multiple popups open for same view
 - Popup window is left open while parent (main) window is closed
- Guidewire popups
 - Provide almost all the functionality of true popup windows
 - Avoid blocking issues and synchronization problems

Popup screens are designed to mimic the functionality of a true popup (which is an entirely separate window). True popups are difficult to architect because there is no easy way to avoid synchronization errors in which the system tries to update the same object in two separate windows. Popup locations as architected in Guidewire offer virtually the same functionality while avoiding the synchronization and usability issues.

Popup uses cases

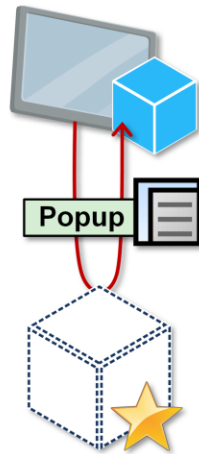
View and Edit

- View and edit existing object



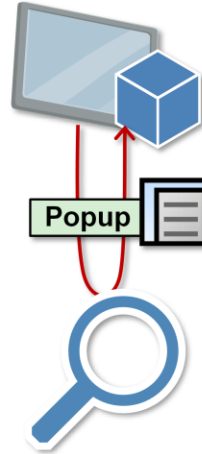
Create New

- Create new object



Search

- Execute "popped" searches



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

7

GUIDEWIRE

You can configure a popup to view and edit an object. You can configure a popup to create new objects. In most cases, the popup can both create new objects and view and edit existing objects. You can configure a popup to search for an existing record.

Example: View and edit existing objects

Summary ←

[Edit](#) [Suggest Least Busy User](#)

Basic Information

Name Olaf Simpson
Public ID ab:108
Created On 01/13/2014
Assigned User

Flag Entries

	View	Date Flagged ↓	Reason	Date Unflagged
⚠	View/Edit	01/13/2014	No email address for this contact.	

Flag Entry [Return to Summary](#)

[Update](#) [Cancel](#)

Date Flagged 02/11/2014
Reason No email address for this
Resolution
Unflagged By
Date Unflagged

See notes for detailed explanation

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

8

GUIDEWIRE

A popup can be used to view and edit an object. When it is either not convenient to edit the object in a location or some of the fields a user might wish to view and/or edit are not displayed in that location, you can use a popup. For example, a list view panel will often only display a limited number of object fields. Some of the cell widgets will be configured as navigation cell widgets. The navigation cells widgets will navigate to a popup so that a user can view and/or edit the object fields.

In the slide example, the FlagEntryPopup popup is used to view existing flag entry objects and, if the user has sufficient permissions, edit them. However, it cannot be used to create flag entry objects. Flag entries are created only by the application business logic, not manually by a user.

When a popup edits an existing object, the popup does not return the object to the parent list view. The parent list view panel already knows about the object, and the object already belongs to the array that is being displayed by the list view. Therefore, the popup simply commits and changes to the object in the database. The parent list view panel simply refreshes its information from the database.

Example: Create new objects

Details

[Update](#) [Cancel](#)

[Company Info](#) [Phone & Addresses](#) [Bank Accounts](#) [Vendor Info](#)

[Add](#) [Remove](#) [Recommend Vendor](#)

<input type="checkbox"/>	Evaluation Date	Evaluator	Total Score
<input type="checkbox"/>	02/01/2014	John Doe	65
<input type="checkbox"/>	01/02/2014	Susan Doe	81

Details

[Update](#) [Cancel](#)

[Company Info](#) [Phone & Addresses](#) [Bank Accounts](#) [Vendor Info](#)

[Add](#) [Remove](#) [Recommend Vendor](#)

<input type="checkbox"/>	Evaluation Date	Evaluator	Total Score
<input type="checkbox"/>	02/01/2014	John Doe	65
<input type="checkbox"/>	01/02/2014	Susan Doe	81
<input type="checkbox"/>	02/09/2014	Marcus Doe	92

Vendor Evaluation [Return to Details](#)

[OK](#) [Cancel](#)

Evaluator ★ Marcus Doe

Evaluation Date 02/09/2014

Status Unverified

Scores (0 to 25 per category)

Score: Timeliness	25
Score: Communication	20
Score: Quality of Work	23
Score: Pricing	24
Total Score	92

Comments
Excellent vendor. Great staff.
Fast shipping. Best pricing.

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

9

GUIDEWIRE

See notes for
detailed explanation

You can configure a popup to create new objects. In most cases, the popup can both create new objects and view and edit existing objects.

When a popup creates a new object, the source container (often a list view panel) does not know about the object. The popup must return the object to the source container and the source must commit the object to the appropriate array that is associated with the parent object. If the popup does not return the newly created object to the source container, the object might get committed to the database (depending on the presence of referential integrity constraints), but the object would not get associated with the parent object array. The result is an orphaned object. From a UI perspective, the object would not appear in the list view panel.

Example: Execute "popped" searches

Official IDs

FEIN: 23-2345678

Organization Type: Common ownership

Industry Code: [Search Icon]

Description of Business: business description

Official IDs

FEIN: 23-2345678

Organization Type: Common ownership

Industry Code: 1522 [Search Icon]

Description of Business: business description

Industry Code Search [Return to Edit Account](#)

Code: [Input Field]

Classification: [Input Field]

[Search](#) [Reset](#)

Search Results

« < | Page 1 of 53 | > »

	Code	Classification
Select	1522	Veterinarians / Veterinari...
Select	0741	Veterinary services for liv...

See notes for detailed explanation

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

10

GUIDEWIRE

You can configure a popup to search for an existing record. When a record is located in some portion of the user interface where it would not be convenient to include search functionality, you will often see an implementation of a popped search.

An application needs a certain level of complexity and functionality to require a popped search. TrainingApp is simple enough that it does not have an example of this type of popup. The example shown above is from PolicyCenter, where a popup is provided to allow the user to search for a valid industry code when creating a new account.

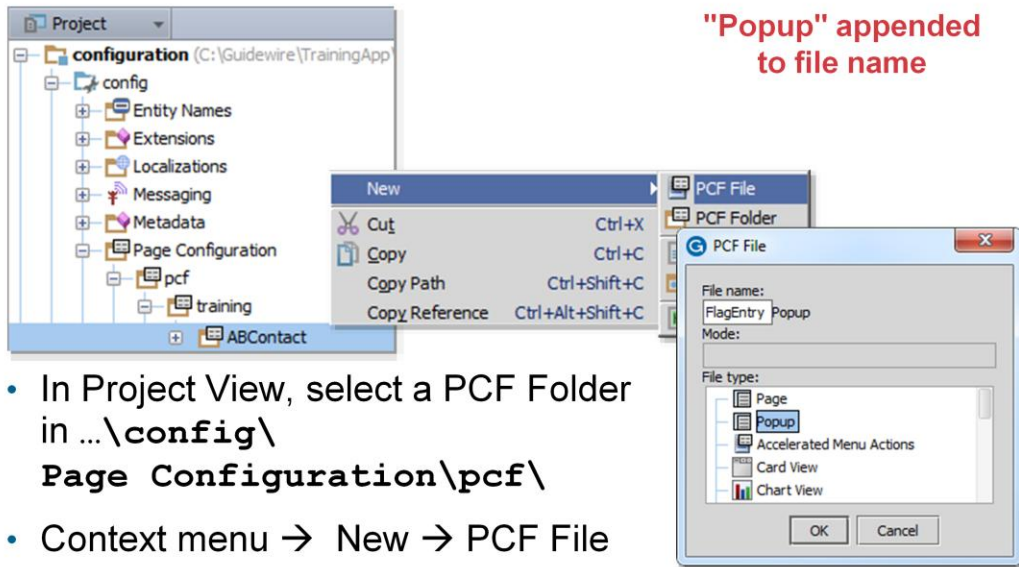
Lesson outline

- Popup fundamentals
- Create a popup

Steps to create a view/edit object popup

1. Create a popup PCF file
2. Specify variable(s)
3. Specify properties
4. Specify entry point(s)
5. Add containers and input widgets
6. Configure the navigation widget
7. Deploy PCFs

Step 1: Create a popup PCF



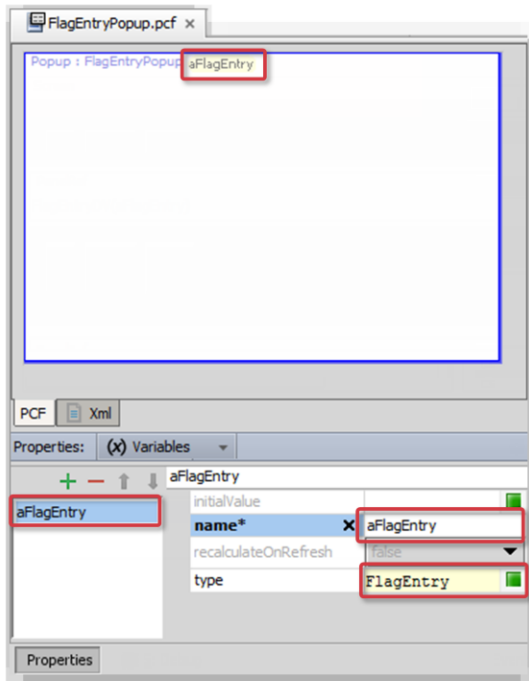
© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

13

G U I D E W I R E

Studio automatically adds "Popup" to the end of the PCF file name.

Step 2: Specify variable(s)



- Variables tab
 - Defines data object variable name and type
 - Example:
aFlagEntry is of type FlagEntry
- Object data can be
 - Data backed (database)
 - Virtual property
- Container data comes from defined variable object(s)
 - Often, at least one variable



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

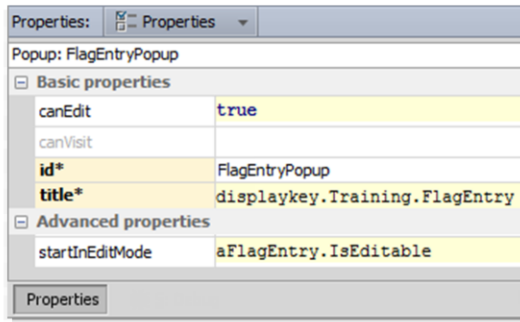
14


G U I D E W I R E

A variable for a popup often serves as the root object for any containers that contains data fields. One way to think of this is that there is at least one root object for a given container. Root objects must be specified on the Popup's Variables tab.

The initial Popup location PCF file always displays an warning in the PCF Editor because the title attribute is required. In the slide example, the screenshot has been modified to not show the error.

Step 3: Specify properties



Properties:  Properties	
Popup: FlagEntryPopup	
Basic properties	
canEdit	true
canVisit	
id*	FlagEntryPopup
title*	displaykey.Training.FlagEntry
Advanced properties	
startInEditMode	aFlagEntry.IsEditable
Properties	

- id – identifier for popup
- title – value for top label
- canEdit – if false, no container or widget in location's screen is editable

- canVisit – if false, no widget navigating to location is visible/clickable
- startInEditMode – determines if popup starts in read-only or edit mode

© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

15

G U I D E W I R E

The title property determines the label in the top blue bar. In the slide example, the title is "Flag Entry".

Every location has a canEdit and canVisit property.

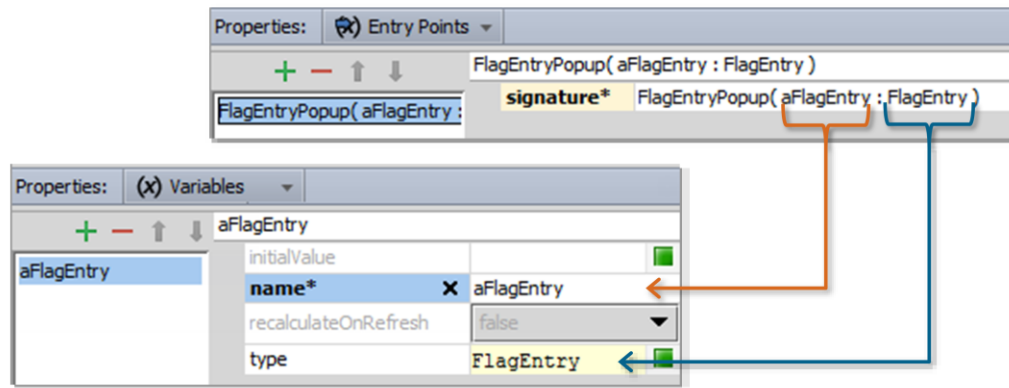
canEdit - This property determines if the screen and widgets contained in the screen are editable or not. If canEdit is false then the screen, every child container widget on the screen, and every atomic widget on the screen will not be editable.

canVisit - This property determines the behavior of widgets that navigate to the location. If it renders to false, then widgets that navigate to it are either not available or not visible. The specific behavior depends on the widget. For example, a cell widget that navigates to a location with its canVisit property set to false is visible but cannot be clicked. A menu item that navigates to a location with its canVisit property set to false is not visible.

The **startInEditMode** property determines if the popup is initially in edit mode or not. It can be set to a static value (true or false) or a Gosu expression, such as "aFlagEntry.IsEditable", which returns true if the flag entry is editable. (A flag entry is editable if it is open and the current user has "resolve flag entry" permissions.)

Step 4: Specify entry point(s)

- An **entry point** is a reference used by widgets to navigate to a given location
- Every location must have at least one entry point
- Syntax: **popupName (objName : objType)**
 - Can specify multiple comma-delimited objects



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

16

GUIDEWIRE

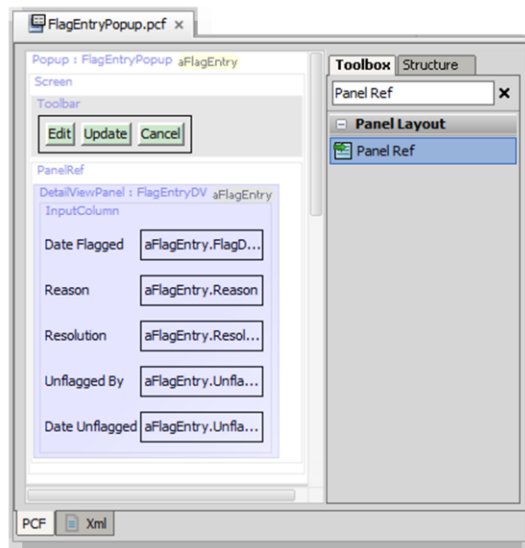
An entry point is a reference used by widgets to navigate to a given location. An entry point specifies the location name and the values required to render the location.

In the slide example, FlagEntryPopup is the entry point name. The name of the entry point must be the same as the name of the file in which the location is declared.

An entry point defines a signature. The syntax for a signature is the same as a Gosu method. A parameter is defined for the entry point, for example, aFlagEntry of the type FlagEntry. The parameter type is the same as the related variable for the Popup. In other words, every object referenced in the entry point signature must be declared as a variable in the Variables tab. In this manner, an FlagEntry object instance can pass as an argument to the Popup. It is possible to declare multiple comma delimited parameters in the signature definition.

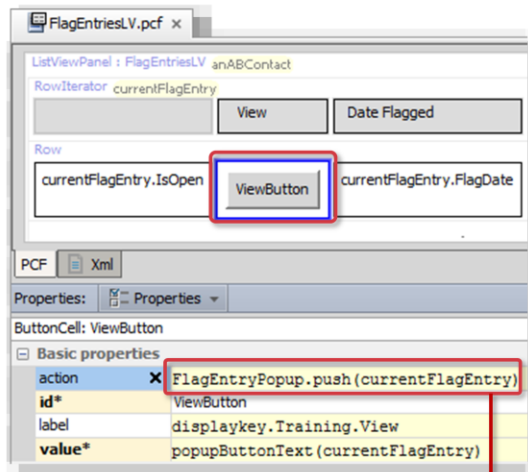
Every location must have at least one entry point. Any given location can have many entry points. An example of this appears later in the lesson.

Step 5: Add containers and input widgets



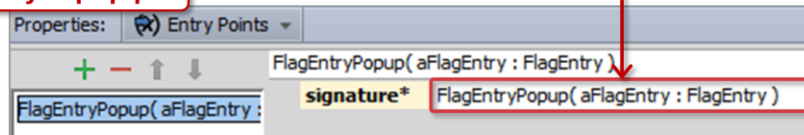
- Add Screen to popup
- If editing...
 - Add Toolbar
 - Add Edit Buttons
- Add inline container and input widgets OR
- Add Panel Ref to reference other containers

Step 6: Configure navigation widget



- For the destination location, determine the entry point
- Specify action property
 - `push()` method for popups
- Syntax:
locationName.push(objectList)

FlagEntryPopup.pcf

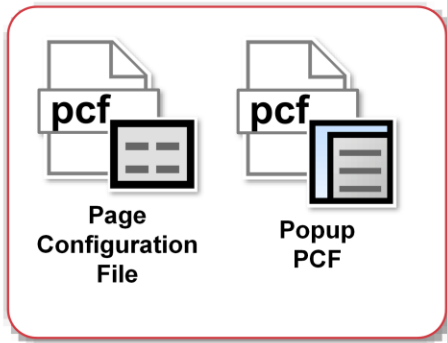


The destination location is the file that you want the navigation widget to navigate to. The source location is the origin of where you navigate from

Step 7: Deploy PCFs

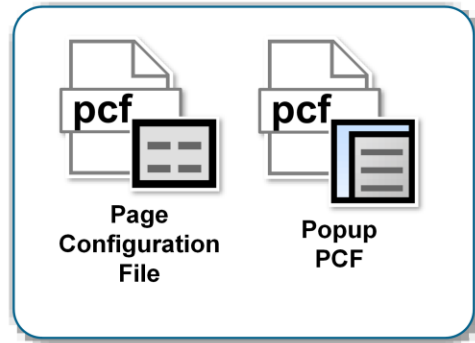
Restart Server

- PCFs read at server startup



Reload PCFs

- ALT+SHIFT+L
 - Internal debug tools enabled
- Internal Tools
 - Reload → Reload PCF Files



© Guidewire Software, Inc. 2001-2014. All rights reserved. Do not distribute without permission.

19

G U I D E W I R E

It is also possible to reload PCF files using the Guidewire API and/or internal server tools. The Reload PCF command can be found on the Reload page in Internal Tools. To access Internal Tools, you must log in as an administrator user, e.g., su/gw. Then, use ALT+SHIFT+T. In the tab bar, select Internal Tools → Reload. On the Reload page, click the Reload PCF Files button. The Reload PCF Files button calls the static method `gw.api.tools.InternalToolsUtil.reloadPCFs()`.

Lesson objectives review

- You should now be able to:
 - Describe the functionality of popups
 - Create popups used to view and edit objects

Review questions

1. What are the three primary use cases for popups?
2. What happens if a location's canEdit property evaluates to false?
3. What happens if a location's canVisit property evaluates to false?

Answers

- 1) Viewing and editing existing objects, creating new objects, and executing popped searches.
- 2) The screen and its contents are non-editable.
- 3) The widget navigating to the location is either not visible or not clickable.

Notices

Copyright © 2001-2014 Guidewire Software, Inc. All rights reserved.

Guidewire, Guidewire Software, Guidewire ClaimCenter, Guidewire PolicyCenter, Guidewire BillingCenter, Guidewire Reinsurance Management, Guidewire ContactManager, Guidewire Vendor Data Management, Guidewire Client Data Management, Guidewire Rating Management, Guidewire InsuranceSuite, Guidewire ContactCenter, Guidewire Studio, Guidewire Product Designer, Guidewire Live, Guidewire DataHub, Guidewire InfoCenter, Guidewire Standard Reporting, Guidewire ExampleCenter, Guidewire Account Manager Portal, Guidewire Claim Portal, Guidewire Policyholder Portal, ClaimCenter, BillingCenter, PolicyCenter, InsuranceSuite, Gosu, Deliver Insurance Your Way, and the Guidewire logo are trademarks, service marks, or registered trademarks of Guidewire Software, Inc. in the United States and/or other countries.

All other trademarks are the property of their respective owners.

This material is confidential and proprietary to Guidewire and subject to the confidentiality terms in the applicable license agreement and/or separate nondisclosure agreement.

This file and the contents herein are the property of Guidewire Software, Inc. Use of this course material is restricted to students officially registered in this specific Guidewire-instructed course, or for other use expressly authorized by Guidewire. Replication or distribution of this course material in electronic, paper, or other format is prohibited without express permission from Guidewire.

Guidewire products are protected by one or more United States patents.