Lab: Gosu for Integration

In this lab, you will practice with the fundamental Gosu integration skills pertaining to classes, blocks, and sequential numbers.

Requirements

This lab requires that you use TrainingApp 8.0, Guidewire Studio 8.0, and a supported web browser. Start Guidewire Studio for Training App. Debug the Server.

To view, edit, and delete various contacts, log in to TrainingApp as Super User. The default URL for TrainingApp is <http://localhost:8880/ab/ContactManager.do>. The login/password for Super User is su/gw.

1. Create the ABContactSummary class

Acme has several integration points that need to make use of ABContact summaries. In general, it is highly desirable for integration points to transmit and work with only the data that is required.

In this exercise, you will create a class that represents an ABContact summary. Your ABContactSummary will contain a selected subset of information about a given ABContact. The small size of the object helps improve the performance of the integration points.

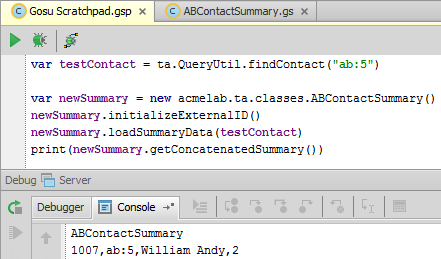
Tasks

1. In Guidewire Studio, create the class
2. If it does not already exsist, create a package called acmelab.ta.classes.
3. Create a Gosu class named ABContactSummary.
4. Create the following properties accessible through public getters and setters:
5. ExternalID (an integer).
6. ContactID (a String).
7. Name (a String).
8. NumCheckingAccounts (an integer).
9. Create the initializeExternalID() method that meets the following criteria:
10. If the summary's ExternalID is 0, set the ExternalID to a unique integer no less than 1000, otherwise, do nothing.
11. Create the loadSummaryData() method that meets the following criteria:
12. Takes an input parameter of the type entity.ABContact.
13. Set the summary's ContactID to the ABContact's public ID.
14. Set the summary's Name to the ABContact's DisplayName field.
15. Use ABContact's BankAccounts array's where() method to determine the number of accounts that are checking accounts.
16. Set the summary's NumCheckingAccounts to the number of accounts where the type is checking.
17. Create the getConcatenatedSummary() method that meets the following criteria:
18. Returns a String that contains the value of all four properties as a comma-delimited list.

Verification

You will verify the behavior of the instance of a class.

1. Create bank accounts for an ABContact
2. In Training App, navigate to a given contact (such as William Andy).
3. Note the ABContact's Public ID.
4. Navigate to the Detail screen's Bank Accounts card.
5. Add two checking accounts and one savings account for the contact.
6. Click Update.
7. Generate console output for ABContactSummary using Gosu Scratchpad
8. In Guidewire Studio, open GosuScratchpad.
9. Using ta.QueryUtil.findContact(), create a variable that references your contact.
10. Create a new ABContactSummary.
11. Execute the initializeExternalID() method.
12. Execute the loadSummaryData() method using the given contact.
13. Retrieve and print the output of the getConcatenatedSummary () method.
14. Verify the output
15. Verify that the output in the Debug Console has four delimited values.



|  |  |
| --- | --- |
|  | Stop and ask your instructor to review your completed lab. |