Lesson 16 Acknowledging Messages

This exercise requires that you use **TrainingApp**, **Guidewire Studio**, **ExternalApp**, and a supported web browser. Start **Guidewire Studio for TrainingApp**. Start the server as **Debug** 'Server'.

Start **ExternalApp** using the **Start ExternalApp** shortcut.

The default URL for **TrainingApp** is: http://localhost:8880/ab/ContactManager.do. Log in to **TrainingApp** as Super User whose login/password is **su/gw**.

Exercise 1: Configure synchronous acknowledgment

Exercise

Succeed Insurance must determine if a given contact has been involved with a previous act of insurance fraud. To implement their fraud prevention system, they must send a message to an external system for every new contact or for an existing contact whose tax ID is updated. The payload must be in XML format and must contain the contact's full name, tax ID, and a reference value generated by Guidewire to help identify the message. The external system must respond immediately to the fraud investigation request with a fraud report code.

External system information:

The WSDL URL to the external system is:

http://localhost:8890/ab/ws/externalapp/webservice/FraudReportAPI?WSDL

The authentication parameters are:

- Username: externalappuser
- Password: gw

Do not code the authentication parameters in the implementation code – use plugin parameters that are passed to the code.

API method checkForFraudReport requires the transformed payload as its argument.

The valid acknowledgment report codes are as follows:

- **1** – Request processed; no fraud report found
  - Acknowledge the message.
- **2** – Request processed; fraud report found!
  - Acknowledge the message.
- **4** – Request could not be processed (Payload Format Error)
  - Acknowledge the message with error using error category **Payload Format**.
- **5** – Request could not be processed (Database Unavailable)
  - Acknowledge the message with error using error category **Database Contention**.

- **Default** – Request could not be processed (Acknowledgment Code Invalid). If the error code returned from the external system is not valid, then acknowledge the message with error using a new error category called **Acknowledgement Code Invalid**.

For training purposes, output an acknowledgment message to console using the print statement.

16.1.1 Requirements

Spec 1 Process external system synchronous response code based on the following criteria:

- **1** – Request processed; no **fraud** report found
  - Acknowledge the message.
- **2** – Request processed; **fraud** report found!
  - Acknowledge the message.
- **4** – Request could not be processed (Payload Format Error)
  - Acknowledge the message with error using error category **Payload Format**.
- **5** – Request could **not** be processed (Database Unavailable)
  - Acknowledge the message with error using error category **Database Contention**.
- **Default** – Request could not be processed (Acknowledgment Code Invalid). If the error code returned from the external system is not valid, then acknowledge the message with error using a new error category called **Acknowledgement Code Invalid**.

Spec 2 Add a new **ErrorCategory** typecode named **Acknowledgment Code Invalid**.

Spec 3 Create a helper method called **processResponseCode** that processes the message based on its external system synchronous response code.

16.1.2 Tasks

7. **Configure the ErrorCategory typelist.**
8. **Configure the FraudCheckTransport class.**
9. **Deploy code changes.**
10. **Perform verification steps.**

16.1.3 Verification steps

The external system will take the first digit in the Tax ID field and return that number as the acknowledgment code. To properly test the code, enter Tax ID numbers that starts with 1, 2, 4, 5, and 8. This will completely test all the different acknowledgment scenarios.

1. **Launch TrainingApp.**
2. **Clear pending messages in Message Table screen.**

   b) Navigate to **Administration ➔ Training: Messaging ➔ Message Table**.

   c) Select all messages with **Pending acknowledged** status.

d) Click the **Skip Selected Message(s)** button.

e) Click **OK** to the popup window.

3. **Clear messages in MessageHistory Table screen.**

f) Navigate to **Administration ➔ Training: Messaging ➔ MessageHistory Table**.

g) Select all messages.

h) Click **Delete Selected Message Histories** button.

i) Click **OK** to the popup window.

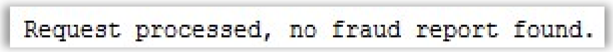
4. **Test response code 1.**

a) Search for **John Snow**.

b) Navigate to the **Details** screen.

c) Edit the **Tax ID** field.

- Enter **111-99-9999.**
- In **TrainingApp Studio** console verify the output is correct.



```
Request processed, no fraud report found.
```

- Verify the message was acknowledged in the **MessageHistory Table** screen.

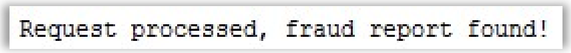
5. **Test response code 2.**

a) Search for **John Snow**.

b) Navigate to the **Details** screen.

c) Edit the **Tax ID** field.

- Enter **222-99-9999**.
- In **TrainingApp Studio** console verify the output is correct.



```
Request processed, fraud report found!
```

- Verify the message was acknowledged in the **MessageHistory** Table screen.

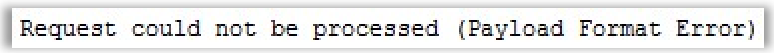
6. **Test response code 4.**

a) Search for **John Snow**.

b) Navigate to the **Details** screen.

c) Edit the **Tax ID** field.

- Enter **444-99-9999**.
- In **TrainingApp** Studio console verify the output is correct.



```
Request could not be processed (Payload Format Error)
```

- Verify the message is in the **Message Table** screen with a status of **Retryable error** and **Error Category** of **Payload Format**.
- Select the message and select **Skip Selected Message(s)** button.

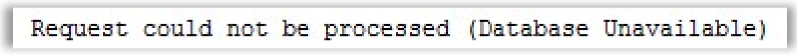
7. **Test response code 5.**

   a) Search for **John Snow**.

   b) Navigate to the **Details** screen.

   c) Edit the **Tax ID** field.

   - Enter **555-99-9999.**
   - In **TrainingApp Studio** console verify the output is correct.



```
Request could not be processed (Database Unavailable)
```

- Verify the message is in the **Message Table** screen with a status of **Retryable error** and **Error Category** of **Database Contention**.
- Select the message and select **Skip Selected Message(s)** button.

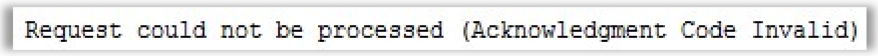
8. **Test invalid response code.**

   a) Search for **John Snow**.

   b) Navigate to the **Details** screen.

   c) Edit the **Tax ID** field.

   - Enter **999-99-9999**
   - In **TrainingApp Studio** console verify the output is correct.

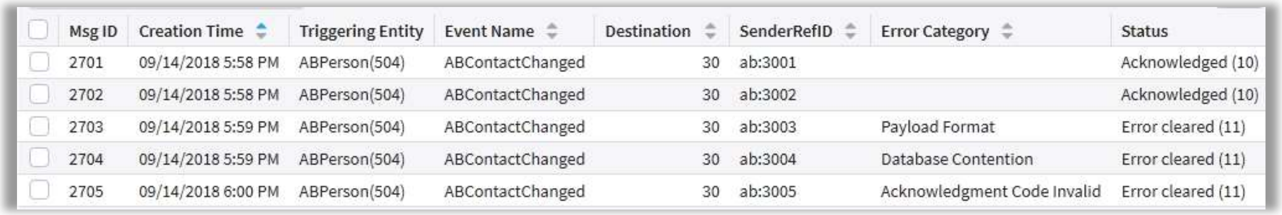


```
Request could not be processed (Acknowledgment Code Invalid)
```

- Verify the message is in the **Message Table** screen with a status of **Retryable error** and **Error Category** of **Acknowledgment Code Invalid**.
- Select the message and select **Skip Selected Message(s)** button.

9. **Verify MessageHistory table.**

The **MessageHistory** table should have two **Acknowledged** messages and three **Error cleared** messages.



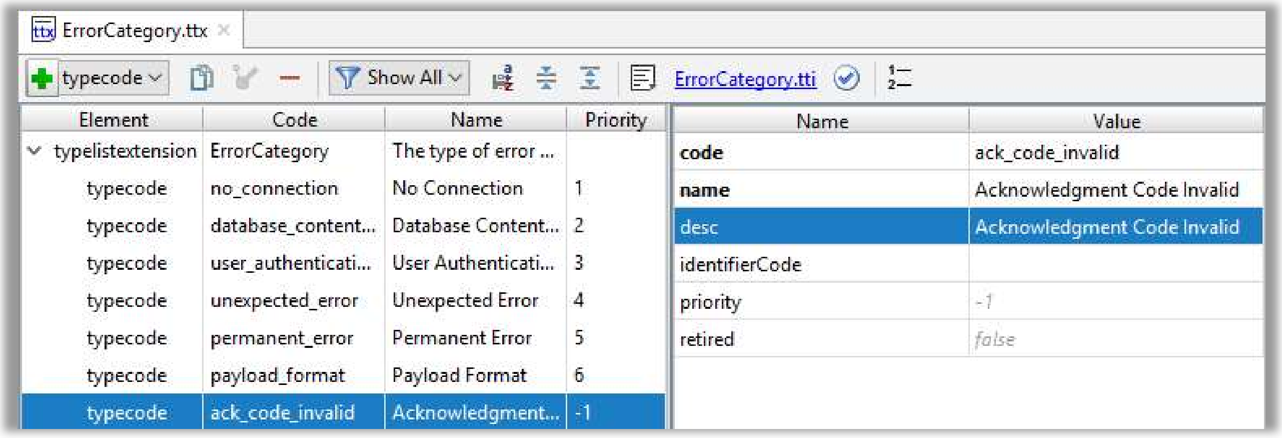
Msg ID	Creation Time	Triggering Entity	Event Name	Destination	SenderRefID	Error Category	Status
2701	09/14/2018 5:58 PM	ABPerson(504)	ABContactChanged	30	ab:3001		Acknowledged (10)
2702	09/14/2018 5:58 PM	ABPerson(504)	ABContactChanged	30	ab:3002		Acknowledged (10)
2703	09/14/2018 5:59 PM	ABPerson(504)	ABContactChanged	30	ab:3003	Payload Format	Error cleared (11)
2704	09/14/2018 5:59 PM	ABPerson(504)	ABContactChanged	30	ab:3004	Database Contention	Error cleared (11)
2705	09/14/2018 6:00 PM	ABPerson(504)	ABContactChanged	30	ab:3005	Acknowledgment Code Invalid	Error cleared (11)

Solution 1: Configure synchronous acknowledgment

1. **Configure the ErrorCategory typelist.**

   a) Open **ErrorCategory** typelist extension.

   b) Add new typecode.


ErrorCategory.ttx

Element	Code	Name	Priority
typelistextension	ErrorCategory	The type of error ...	
typecode	no_connection	No Connection	1
typecode	database_content...	Database Content...	2
typecode	user_authenticati...	User Authenticati...	3
typecode	unexpected_error	Unexpected Error	4
typecode	permanent_error	Permanent Error	5
typecode	payload_format	Payload Format	6
typecode	ack_code_invalid	Acknowledgment...	-1

Name	Value
code	ack_code_invalid
name	Acknowledgment Code Invalid
desc	Acknowledgment Code Invalid
identifierCode	
priority	-1
retired	false

2. **Configure the FraudCheckTransport class.**

```
package si.ta.messaging.fraudcheck

uses gw.plugin.InitializablePlugin
uses gw.plugin.messaging.MessageTransport
uses si.ta.messaging.fraudcheck.fraudcheckwsc.fraudreportapi.FraudReportAPI

class FraudCheckTransport implements MessageTransport, InitializablePlugin{
  // Plugin parameters
  private static final var USERNAME = "Username"
  private static final var PASSWORD = "Password"
  // API parameters
  private static var _username : String
  private static var _password : String

  private var _fraudReportApi: FraudReportAPI

  override function send(message : Message, transformedPayload : String) {
    var responseCode = _fraudReportApi.checkForFraudReport(transformedPayload)

    // Acknowledge message based on the response code
    var ackMessage = processResponseCode(message, responseCode)
    print(ackMessage)
  }

  override function shutdown() {  }
  override function suspend() {
    _fraudReportApi = null
  }

  override function resume() {
    _fraudReportApi = new FraudReportAPI()
    _fraudReportApi.Config.Http.Authentication.Basic.Username = _username
    _fraudReportApi.Config.Http.Authentication.Basic.Password = _password
  }

  override property set DestinationID(destinationID : int) {  }

  override property set Parameters(map : Map<Object, Object>) {
    _username = map.get(USERNAME) as String
    _password = map.get(PASSWORD) as String
  }

  // Helper method

  /** This method processes a message based on its external system synchronous response code.
   */
  @Param("aMessage", "The message instance")
  @Param("responseCode", "Response code from the external system")
  private function processResponseCode(message: Message, responseCode: Integer) : String {
    var output = ""

    // Acknowledge message based on response code
    switch (responseCode) {
      case 1:
        message.reportAck()
        output =("Request processed, no fraud report found.")
        break
      case 2:
        message.reportAck()
        output = ("Request processed, fraud report found!")
        break
      case 4:
        message.reportError(ErrorCategory.TC_PAYLOAD_FORMAT)
        output = ("Request could not be processed (Payload Format Error)")
        break
      case 5:
        message.reportError(ErrorCategory.TC_DATABASE_CONTENTION)
        output = ("Request could not be processed (Database Unavailable)")
        break
      default:
        message.reportError(ErrorCategory.TC_ACK_CODE_INVALID)
        output = ("Request could not be processed (Acknowledgment Code Invalid)")
        break
    }
    return output
  }

}
```

3. **Deploy code changes.**

   a) From the **Studio** menu, **Restart** the server.

4. **Perform verification steps.**