Acumatica Test SDK Common Patterns

Contents

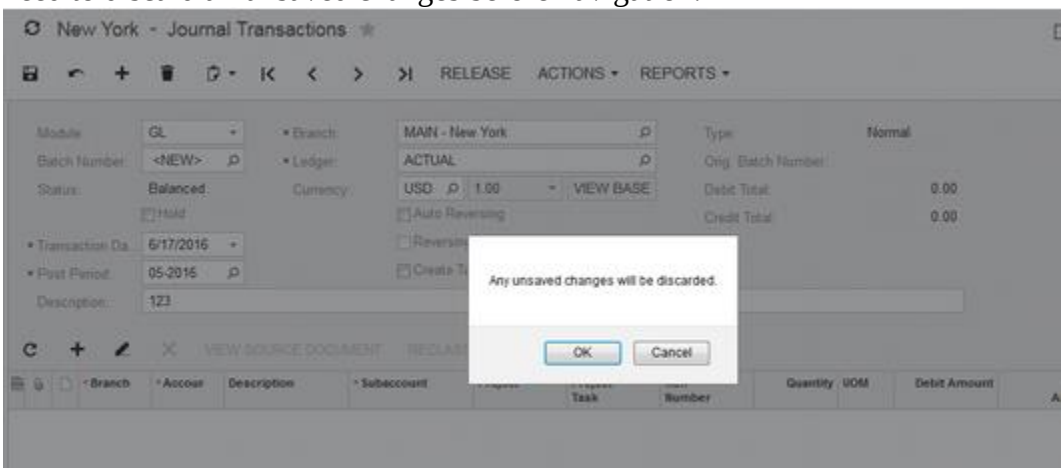
Windows Scale

ALWAYS use 100% scale on windows, or else the Tests will not recognize the element placements all of the time. To fix this, go to the Control Panel, select **All Control Panel Items > Display**, and **set the display scale to 100%**

How to Work with Alerts

Navigating Away from The Form Where Unsaved Changes Have Been Made

If you navigate away from the form where unsaved changes have been made, the Acumatica application throws the standard confirmation, which is shown in the following screenshot. You need to discard all unsaved changes before navigation.



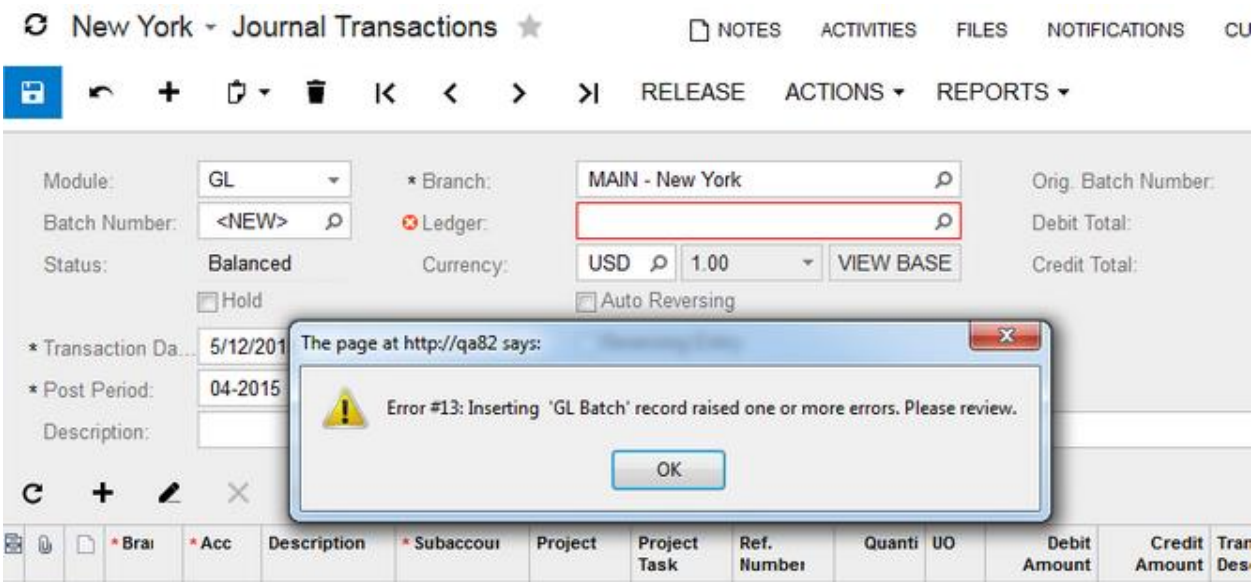
The code fragment below clicks **OK** in the alert and dismisses it without throwing an exception.

```
JournalEntry batch = new JournalEntry();
batch.OpenScreen();
batch.Insert();
batch.Summary.BranchID.Select("MAIN");
batch.Summary.LedgerID.Reset();
batch.Cancel();
JournalVouchers journalVouchers = new JournalVouchers();
journalVouchers.OpenScreen();
```

Verifying that the Alert with Exact Message Has Been Thrown

If an error message is displayed in your Acumatica application after the test application clicks a button and you need to verify the text of the alert, you can handle the error as described in this scenario.

An example of an error message, which appears if you click **Save** on the Journal Transactions form (GL301000; Finance > General Ledger > Enter) before filling in all required elements, is shown in the following screenshot.

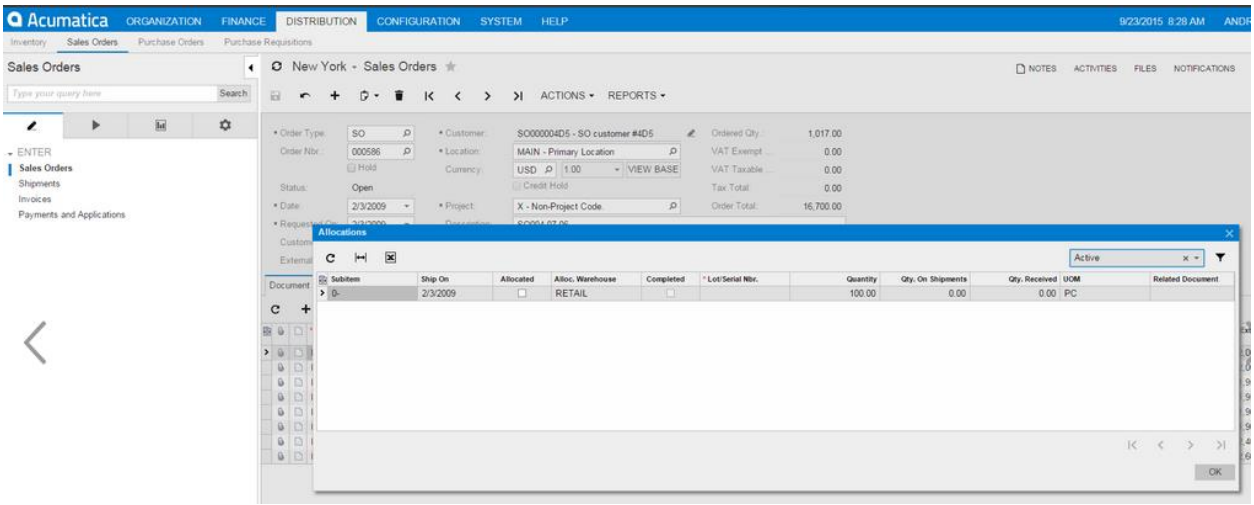


The code fragment below clicks Save on the Journal Transactions form (GL301000), validates that the alert exists, validates its text message, and clicks **OK** to dismiss the alert.

```
JournalEntry batch = new JournalEntry();
batch.OpenScreen();
batch.Insert();
batch.Summary.BranchID.Select("MAIN");
batch.Summary.LedgerID.Reset();
batch.VerifyAlert(batch.Save, "Error #13: Inserting 'GL Batch' record raised
```

How to Work with Pop-Up Panels

In this topic, you can find examples that show how to work with pop-up panels that appear on Acumatica forms by using Test SDK. An example of a pop-up panel is shown in the following screenshot.



You may need to perform the following task with pop-up panels:

Closing a Pop-up Panel

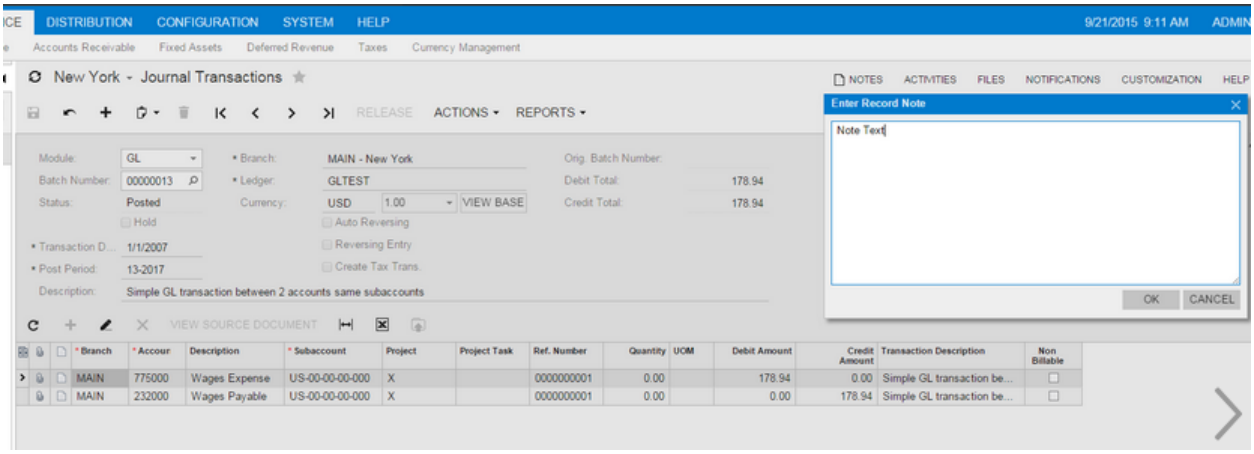
The following code shows how to close a pop-up panel through the example of the Allocations pop-up panel that can appear on the Sales Orders form (SO301000; Distribution > Sales Orders > Work Area > Enter).

```
OrderSo.OpenScreen();
OrderSo.Last();
OrderSo.DocumentDetails.Allocations();
OrderSo.AllocationsForm.Close();
```

How to Verify a Note on a Form

In this topic, you will see an example of the note text on a form being verified.

The following screenshot illustrates the verification of a note on the Journal Transactions form (GL301000).



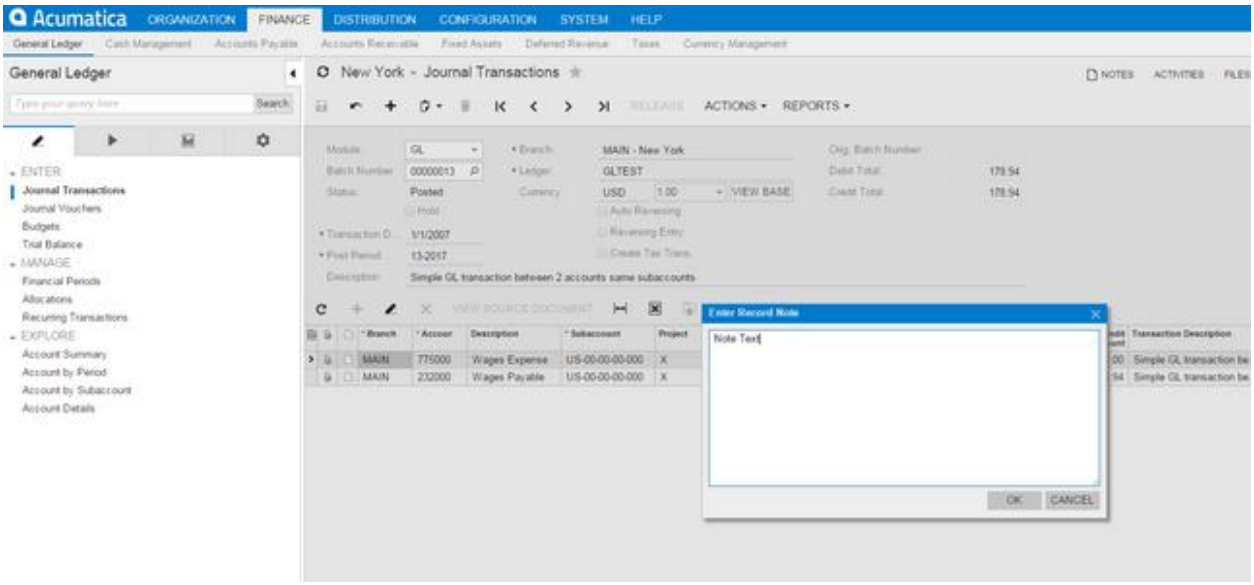
The code below shows how to verify the text of a note in your test.

```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Note();
JournalEntry.NotePanel.NoteEdit.VerifyEquals("Note Text")
JournalEntry.NotePanel.Cancel();
```

How to Add a Note to a Detail Line

In this topic, you will see an example of adding a note to a detail line of a document on an Acumatica ERP form.

The following screenshot illustrates a note being added to a detail line on the Journal Transactions form (GL301000).



The code below shows how to implement adding a note to a detail line in your test.

```
JournalEntry JournalEntry = new JournalEntry();

JournalEntry.OpenScreen();

JournalEntry.Details.SelectRow(1);

JournalEntry.Details.Row.Notes.Click();

JournalEntry.Details.NotePanel.Type("Note Text");

JournalEntry.Details.NotePanel.Ok();
```

How to Add a Note to an Acumatica Form

How to Upload a File

```
BankTransactionsImport BankTransactionsImport = new BankTransactionsImport();
BankTransactionsImport.OpenScreen();
BankTransactionsImport.UploadFile();
BankTransactionsImport.StatementFileUpload.SelectFile("C:\\share\\BankTransac
tions.dat");
BankTransactionsImport.StatementFileUpload.Upload();
```

How to Upload Data from an Excel File into a Detail Table

The following example shows how to upload data from an Excel file into a detail table on the Chart of Accounts form (GL202500; Finance > General Ledger > Configuration > Manage).

```
Account Account = new Account();
Account.OpenScreen();
Account.Upload();
Account.Details.UploadForm.SelectFile("C:\\share\\Accounts.xlsx");
Account.Details.UploadForm.Upload();

Account.ExcelImportSettings.NullValue.Type("NULL");
Account.ExcelImportSettings.Culture.Select("Norwegian");
Account.ExcelImportSettings.Mode.Select("Bypass Existing");
Account.ExcelImportSettings.Ok();

Account.Columns.Ok();
```

How to Attach a File to an Acumatica Form

The following code shows an example of a file being attached to a record on the Journal Transactions form (GL301000).

```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Last();
JournalEntry.FilesMenuShow();
JournalEntry.FilesUploadDialog.FilesAttached.VerifyRowsCount(0);
JournalEntry.FilesUploadDialog.FileUploader.SelectFile("C:\\share\\Test.txt")
;
JournalEntry.FilesUploadDialog.FileUploader.Upload();
JournalEntry.FilesUploadDialog.FilesAttached.VerifyRowsCount(1);
string fileName =
JournalEntry.FilesUploadDialog.FilesAttached.Row.FileName.GetValue();
JournalEntry.FilesUploadDialog.FilesAttached.Row.Comment.Type(
            string.Format("This file has been attached to Acumatica page:
{0}", fileName));
JournalEntry.FilesUploadDialog.Close();
```

How to Attach a File to a Detail Line

The following code shows how to attach a file to a detail line of a record on the Journal Transactions form (GL301000).

```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Last();
JournalEntry.Details.VerifyRowsCount(1);
```

```
JournalEntry.Details.SelectRow(1);
JournalEntry.Details.Row.Files.Click();
JournalEntry.Details.FilesUploadDialog.FilesAttached.VerifyRowsCount(0);
JournalEntry.Details.FilesUploadDialog.FileUploader.SelectFile("C:\\share\\Te
st.xlsx");
JournalEntry.Details.FilesUploadDialog.FileUploader.Upload();
JournalEntry.Details.FilesUploadDialog.FilesAttached.VerifyRowsCount(1);
string fileName =
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.FileName.GetValue();
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.Comment.Type(
                        string.Format("This file has been attached to a detail
line: {0}", fileName));
JournalEntry.Details.FilesUploadDialog.Close();
```

How to Remove an Attached File from an Acumatica Form

You can use one of the following approaches when removing an attached file from an Acumatica form:

- You can remove the file by using the File Maintenance form (SM202510).
- You can remove the file by using the Search in Files form (SM202520).

Removing an Attached File by Using the File Maintenance Form (SM202510)

The following code shows how to remove the attached file from the Journal Transactions form (GL301000) by using the File Maintenance form (SM202510).

```
JournalEntry JournalEntry = new JournalEntry();

JournalEntry.OpenScreen();

JournalEntry.Last();

JournalEntry.FilesMenuShow();

JournalEntry.FilesUploadDialog.FilesAttached.VerifyRowsCount(1);

JournalEntry.FilesUploadDialog.FilesAttached.SelectRow(1);

string fileName =
JournalEntry.FilesUploadDialog.FilesAttached.Row.FileName.GetValue();

JournalEntry.FilesUploadDialog.FilesAttached.Row.Comment.Type(

        string.Format("This file will be removed from the Acumatica ERP form:
{0}", fileName));

string linkText =
JournalEntry.FilesUploadDialog.FilesAttached.Row.Edit.GetValue();

JournalEntry.FilesUploadDialog.FilesAttached.Row.Edit.ClickLink(linkText);
```

```
FileMaintenance FileMaintenance = new FileMaintenance();

FileMaintenance.Delete();

FileMaintenance.CloseWindow();


JournalEntry.FilesUploadDialog.FilesAttached.Refresh();

JournalEntry.FilesUploadDialog.FilesAttached.VerifyRowsCount(0);

JournalEntry.FilesUploadDialog.Close();
```

Removing an Attached File by Using the Search in Files Form (SM202520)

The following code shows how to remove the attached file from the Journal Transactions form (GL301000) by using the Search in Files form (SM202520).

```
JournalEntry JournalEntry = new JournalEntry();

JournalEntry.OpenScreen();

JournalEntry.Last();

JournalEntry.FilesMenuShow();

JournalEntry.FilesUploadDialog.FilesAttached.VerifyRowsCount(1);

JournalEntry.FilesUploadDialog.FilesAttached.SelectRow(1);

string fileName =
JournalEntry.FilesUploadDialog.FilesAttached.Row.FileName.GetValue();

JournalEntry.FilesUploadDialog.FilesAttached.Row.Comment.Type(

            string.Format("This file will be removed from the Acumatica ERP
form: {0}", fileName));

JournalEntry.FilesUploadDialog.Close();


SearchInFiles SearchInFiles = new SearchInFiles();

SearchInFiles.OpenScreen();

SearchInFiles.Summary.DocName.Type(fileName);

SearchInFiles.Details.VerifyRowsCount(1);

SearchInFiles.Details.DeleteFile();

SearchInFiles.Details.VerifyRowsCount(0);


JournalEntry.OpenScreen();
```

```
JournalEntry.Last();

JournalEntry.FilesMenuShow();

JournalEntry.FilesUploadDialog.FilesAttached.VerifyRowsCount(0);

JournalEntry.FilesUploadDialog.Close();
```

How to Remove an Attached File from a Detail Line

You can use one of the two approaches to remove the file attached to a detail line:

- Remove the file by using the File Maintenance form (SM202510)
- Remove the file by using the Search in Files form (SM202520)

Removing the File Attached to a Detail Line by Using the File Maintenance form (SM202510)

The following code shows how to remove the file attached to a detail line on the Journal Transactions form (GL301000). The approach illustrated in this code uses the File Maintenance form (SM202510) to remove the attached file.

```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Last();
JournalEntry.Details.VerifyRowsCount(1);
JournalEntry.Details.SelectRow(1);
JournalEntry.Details.Row.Files.Click();
JournalEntry.Details.FilesUploadDialog.FilesAttached.VerifyRowsCount(1);
JournalEntry.Details.FilesUploadDialog.FilesAttached.SelectRow(1);
string fileName =
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.FileName.GetValue();
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.Comment.Type(
        string.Format("This file will be removed from a detail line of the
Acumatica ERP form: {0}", fileName));
string linkText =
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.Edit.GetValue();
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.Edit.ClickLink(linkT
ext);

FileMaintenance FileMaintenance = new FileMaintenance();
FileMaintenance.Delete();
FileMaintenance.CloseWindow();

JournalEntry.Details.FilesUploadDialog.FilesAttached.Refresh();
JournalEntry.Details.FilesUploadDialog.FilesAttached.VerifyRowsCount(0);
JournalEntry.Details.FilesUploadDialog.Close();
```

Removing the File Attached to a Detail Line by Using the Search in Files Form (SM202520)

The following code shows how to remove the file attached to a detail line on the Journal Transactions form (GL301000). The approach illustrated in this code uses the Search in Files form (SM202520) to remove the attached file.

```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Last();
JournalEntry.Details.VerifyRowsCount(1);
JournalEntry.Details.SelectRow(1);
JournalEntry.Details.Row.Files.Click();
JournalEntry.Details.FilesUploadDialog.FilesAttached.VerifyRowsCount(1);
JournalEntry.Details.FilesUploadDialog.FilesAttached.SelectRow(1);
string fileName =
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.FileName.GetValue();
JournalEntry.Details.FilesUploadDialog.FilesAttached.Row.Comment.Type(
            string.Format("This file will be removed from a detail line of
the Acumatica ERP form: {0}", fileName));
JournalEntry.Details.FilesUploadDialog.Close();

SearchInFiles SearchInFiles = new SearchInFiles();
SearchInFiles.OpenScreen();
SearchInFiles.Summary.DocName.Type(fileName);
SearchInFiles.Details.VerifyRowsCount(1);
SearchInFiles.Details.DeleteFile();
SearchInFiles.Details.VerifyRowsCount(0);
JournalEntry.OpenScreen();

JournalEntry.Last();
JournalEntry.Details.VerifyRowsCount(1);
JournalEntry.Details.SelectRow(1);
JournalEntry.Details.Row.Files.Click();
JournalEntry.Details.FilesUploadDialog.FilesAttached.VerifyRowsCount(0);
JournalEntry.Details.FilesUploadDialog.Close();
```

How to Publish Customization Projects

In your test application, you can upload a customization project and publish it by using any of the following approaches.

1. Define the following extension class for the page wrapper of the Customization Projects form (SM204505; System > Customization > Manage).

   ```
   using Controls.CompilationPanel;

     namespace Core

   {
   ```

```
    public class CustomizationProjects : SM204505_ProjectList
    {
        public c_projects_grid Details
        {
            get { return base.Projects_grid; }
        }
        public CompilationPanel CompilationPanel
        {
            get { return base.CompilationPanel; }
        }
    }
}
```

2. Upload and publish a customization project in your test as follows.

```
CustomizationProjects.OpenScreen();
CustomizationProjects.ActionImport();
CustomizationProjects.Details.UploadForm.SelectFile("C:\share\Customiza
tionProject.zip");
CustomizationProjects.Details.UploadForm.Upload();
CustomizationProjects.Details.Row.IsWorking.SetTrue();
CustomizationProjects.ActionPublish();
CustomizationProjects.CompilationPanel.Validate(true, "Validation
finished successfully.");
CustomizationProjects.CompilationPanel.Publish(true, "Website has been
updated.");
CustomizationProjects.CompilationPanel.Close();
```

How to Work with Smart Delays

Waiting for a Long-Running Operation to Complete

To wait for a long-running operation to complete, use the `WaitForLongOperationToComplete()` method of the `Wait` class.

By default, for all buttons that start long-running operations, waiting for the long-running operation to complete is already implemented in Test SDK. However, you may need to change the default wait action. The code example below shows how to change the default wait action.

```
...
public JournalEntry()
{
    ToolBar.Release.WaitAction = Wait.WaitForLongOperationToComplete;
}


...
JournalEntry journalEntry = new JournalEntry();
journalEntry.OpenScreen();
journalEntry.Insert();
journalEntry.Summary.BranchID.Select("MAIN");
journalEntry.Summary.LedgerID.Select("ACTUAL");
journalEntry.Summary.Description.Type("Test journal entry 1");
journalEntry.Details.New();
journalEntry.Details.Row.AccountID.Type("100000");
journalEntry.Details.Row.ProjectID.Select("X");
journalEntry.Details.Row.CuryDebitAmt.Type(100);
journalEntry.Details.New();
journalEntry.Details.Row.AccountID.Select("101000");
journalEntry.Details.Row.CuryCreditAmt.Type(100);
journalEntry.Release();
```

You can also make the test application wait for the specified result of the long-running operation (success or failure) and check the message on completion of the operation, as shown in the following code.

```
...
public JournalEntry()
{
    ToolBar.Release.WaitAction = () =>
Wait.WaitForLongOperationToComplete(false, "Operation failed.");
```

```
}


...
JournalEntry journalEntry = new JournalEntry();
journalEntry.OpenScreen();
journalEntry.Insert();
journalEntry.Summary.BranchID.Select("MAIN");
journalEntry.Summary.LedgerID.Select("ACTUAL");
journalEntry.Summary.Description.Type("Test journal entry 1");
journalEntry.Details.New();
journalEntry.Details.Row.AccountID.Type("100000");
journalEntry.Details.Row.ProjectID.Select("X");
journalEntry.Details.Row.CuryDebitAmt.Type(100);
journalEntry.Details.New();
journalEntry.Details.Row.AccountID.Select("101000");
journalEntry.Details.Row.CuryCreditAmt.Type(100);
journalEntry.Release();
```

Waiting For a New Window to Open

The code example below shows how to wait for a form to load in another window.

```
...


public JournalEntry()
{
    Details.ToolBar.ViewDocument.WaitAction = Wait.WaitForNewWindowToOpen;
}


...


JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
```

```
JournalEntry.Summary.Module.Select("AP");

JournalEntry.Last();

JournalEntry.Details.ViewDocument();


Bill Bill = new Bill();

Bill.Cancel();

Bill.CloseWindow();
```

Waiting for a Condition

To wait for a condition in your test application, use the `WaitForCondition()` method of the `Wait` class, as shown in the following code.

You do not need to check simple conditions, such as whether a box or a button is visible or available on the form, because these conditions are checked by Test SDK automatically when the test application clicks a button or changes data in a box. You may need to use waiting for a condition when you compose a complex reaction on some action.

```
Wait.WaitForCondition(JournalEntry.ToolBar.Save.IsEnabled, Wait.LongTimeOut);
```

How to Capture Screenshots

```
Log.Screenshot();
```

How to Get Datetime in the Current User's Timezone

You can get a datetime in the current user's timezone using the following property:

```
Browser.InstanceTime
```

How to use Comparator to compare .xml, .csv, .pdf, Excel files

Methods of a static class Core.Comparator can be used to verify data in a bulk

1. XML data (Comparator.Xml.Compare(...), is used inside Report.Compare(...))
2. PDF files (Comparator.Pdf.Compare(...))
3. Image files (Comparator.Image.Compare(...))
4. Table Data in different formats
    0. .xlsx
    1. .csv
    2. string[][]
    3. Grids in Acumatica UI

Usage:

1. Set up in tests: (for example you need to verify that exported .xlsx is correct)

```
#region Step 2. Export to Excel

using (TestExecution.CreateTestStepGroup("Export to Excel"))
{
    AccountDetails.Export(); //Note: Wait action should be set
ToolBar.Export.WaitAction = Wait.Wait.WaitForFileDownloadComplete;
    Comparator.Table.Compare("AD100_TC2_S2",
Core.Core.Browser.Browser.Downloads.GetLastFile());
}

#endregion
```

2. Get the baseline (from test results or by manually exporting the file)

3. Add the file to your Tests project (.csv format is recommended), set Copy To Output Directory file property to Copy if Newer (for old-style csproj)

```
<None Include="ERP\FINANCE\General Ledger\Account
Details\Data\AD100_TC2_S2.csv">
  <CopyToOutputDirectory>PreserveNewest</CopyToOutputDirectory>
</None>
```

How to Verify string/long/int/DateTime returned by a method

In this topic, you will see an example of verification of sting/long/int/DateTime returned by a custom method.

```
using PX.QA.Tools;
...
SomeMethod().VerifyEquals("test", "Method result");

long value = SomeMethod();
value.VerifyEquals(123, "Method result");

"test".VerifyEquals("test", "Some string");
```

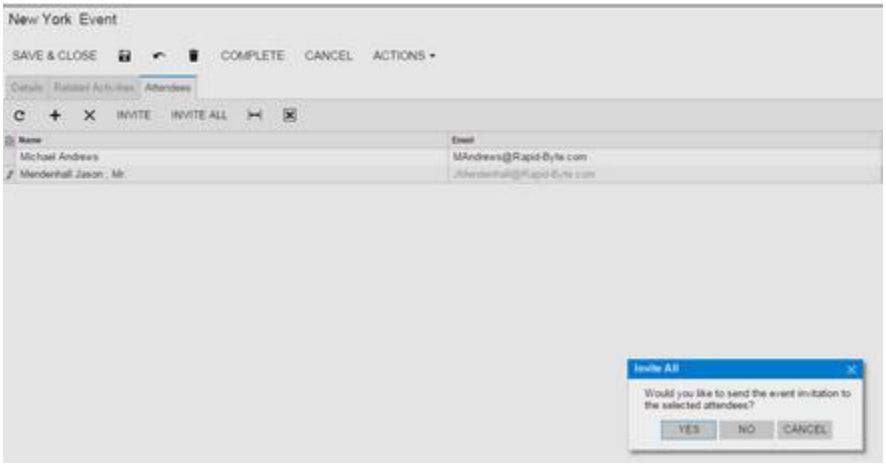
Pay attention: these Verify* methods have second argument (verifiableName), which allows you to specify what exactly you are verifying. Don't forget to set it in test.

How to work with dynamic controls

Test SDK provides ability to find a control (ToolBarButton, Button, CheckBox, Input, Selector, FormulaCombo, DropDown, DateSelector, Container, SmartPanel, GroupBox, QuickSearch, Grid) not generated in WG, e.g. created via customization project. There are two methods: DynamicControl and DynamicGrid (last one has been created just for convenience and it invokes DynamicControl inside with specific type parameter).

How to Work with Pop-Up Dialog Boxes

In this topic, you can find examples that show how to work with pop-up dialog boxes. An example of pop-up dialog box is shown in the following screenshot.



You may need to perform the following tasks with pop-up dialog boxes:

Clicking a Button in a Pop-Up Dialog Box

The following code shows how to click a button in a pop-up dialog box through the example of the **Invite All** dialog box, which can be invoked on the Event form (CR306030).

```
EventList EventList = new EventList();
EventList.OpenScreen();
EventList.New();
Event Event = new Event();
Event.Summary.Subject.Type("test");
Event.Attendees.New();
Event.Attendees.Row.Name.Select("Jason Mendenhall");
var messageText = Event.MessageBox.GetValue(); // get text message of dialog
Event.Save();//save, then close next.
Event.MessageBox.Close();
Event.MessageBox.Yes();//Click Yes
//Event.MessageBox.No()//or click No
//Event.MessageBox.Cancel()//or click Cancel
```

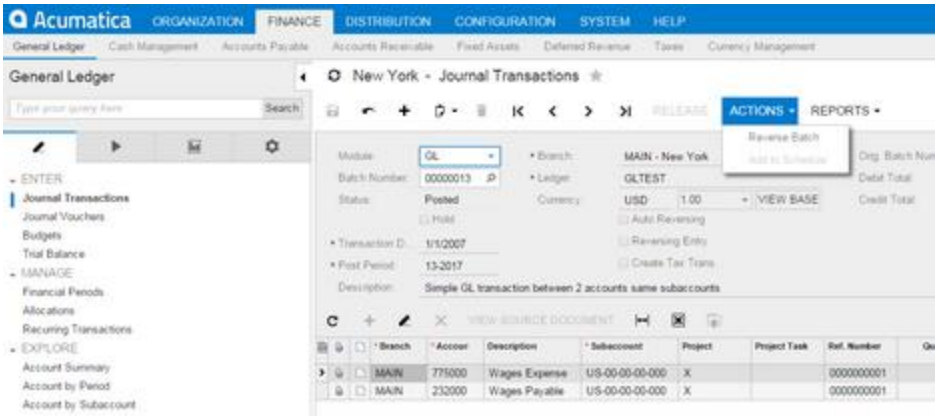
How to Work with Drop-Down Menus on Toolbars

In this topic, you can find examples that illustrate how to work with a drop-down menu on a toolbar by using Test SDK. The following screenshot shows the toolbar buttons that are located on the drop-down menu of the Journal Transactions form (GL301000; Finance > General Ledger > Work Area > Enter).

You may need to perform the following tasks with the drop-down menu of a toolbar:

Clicking a Toolbar Button That Is Located on a Drop-Down Menu

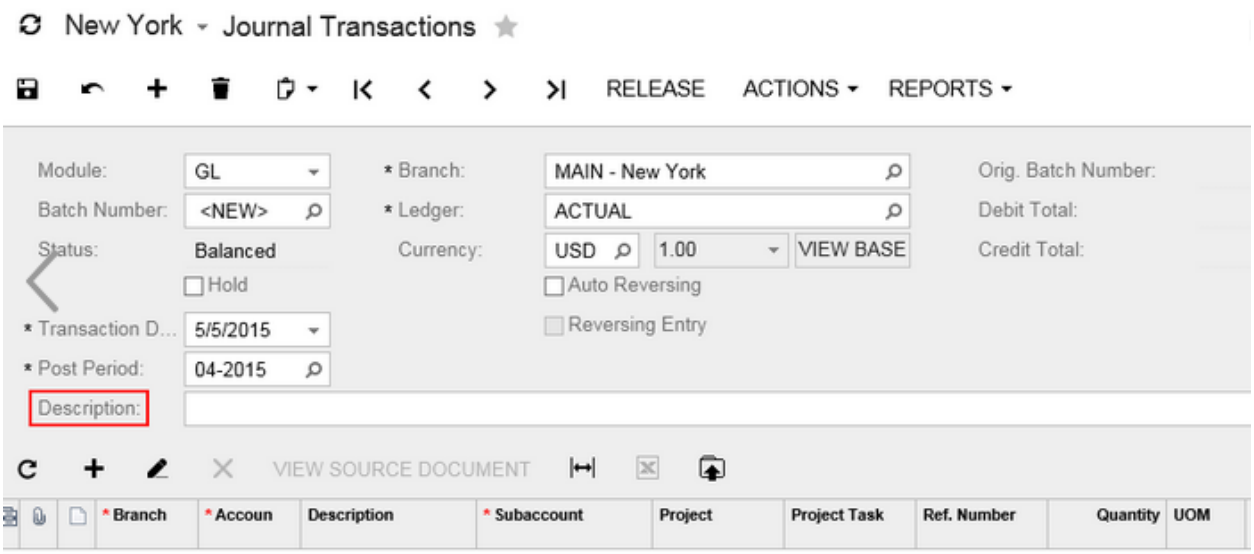
To click a toolbar button that is located on a drop-down menu, you need to invoke the `Click()` method of the button class, as shown in the following code. You do not need to open the menu first.




```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Next();
JournalEntry.ToolBar.ReverseBatch.IsEnabled().VerifyEquals(true); //check
that the toolbar action exists.
JournalEntry.ToolBar.ReverseBatch.Click();
```

How to Work with the Names of UI Elements

In this topic, you can find examples that illustrate how to work with the names of UI elements by using Test SDK. The name of the **Description** element on the Journal Transactions form (GL301000; Finance > General Ledger > Work Area > Enter) is highlighted in the following screenshot.




```
JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen();
JournalEntry.Summary.DescriptionLabel.GetValue();
JournalEntry.Summary.DescriptionLabel.GetValue().VerifyEquals("Description:")
; //you can verify the text of the label as well.
```

How to Work with Windows

On some Acumatica forms, pop-up forms can appear as you work with the form.

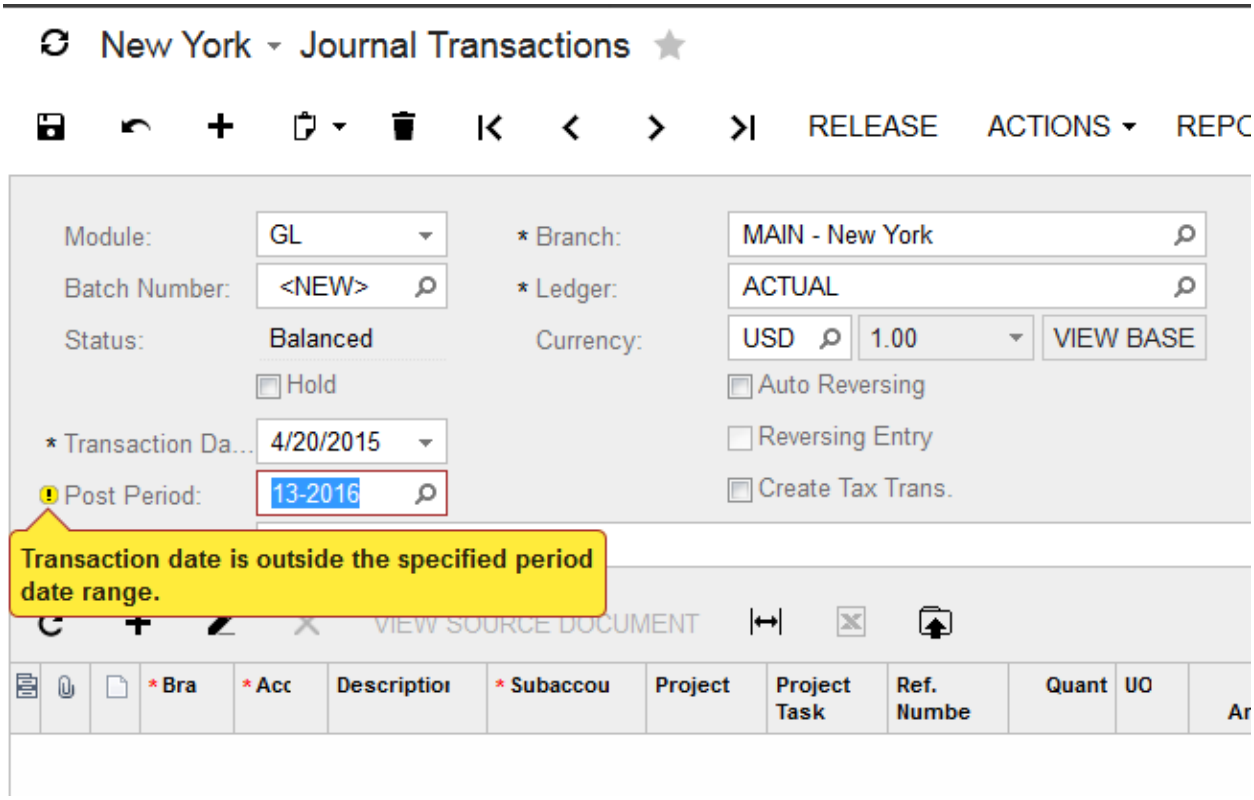
The following example shows how to select an active window when multiple Acumatica forms appear on the screen of your computer.

```
...

public JournalEntry()
{
    Details.ToolBar.ViewDocument.WaitAction = Wait.WaitForNewWindowToOpen;
}

...

JournalEntry JournalEntry = new JournalEntry();
JournalEntry.OpenScreen(); //The Journal Transactions form (GL301000) is the
first form that appears on the screen.
JournalEntry.Summary.Module.Select("AP");
JournalEntry.Last();
JournalEntry.Details.ViewDocument(); //The Bills and Adjustments form
(AP301000) is the second form that appears on the screen.

Bill Bill = new Bill();
Bill.Cancel();
Bill.CloseWindow(); //Closes currently selected window
```

```
JournalEntry.Summary.Module.VerifyEquals("AP");
```

How to Work with Warnings

Test SDK provides the `GetWarning()` method for working with the warnings that occur on the form or on a control, such as the warning shown in the following screenshot. By using the `GetWarning()` method, you can verify whether a warning appears on the form or control, and check the text of the warning message.



In your test application, you can verify a warning on the form or control by performing one of the following tasks:

Verifying the Number of Warnings That Appear on the Form

To check the number of warnings that appear on the form, use the `GetWarning()` method of the form class. The following example verifies that one error appears on the form.

```
ReceiptPo.OpenScreen();
ReceiptPo.Insert();
ReceiptPo.Summary.BranchID.Select("MAIN");
ReceiptPo.Summary.FinPeriodID.Type("13-2016");
ReceiptPo.GetWarnings().VerifyCount(1);
```

Verifying the Text of the Warning Message on the Form

To check that the correct text of the warning message is displayed on the form, use the `GetWarning()` method of the form class with the text of the warning as the parameter, as shown in the following example. You can specify a part of the warning message in the parameter of the method; in this case, the method checks whether the warning message on the form contains the specified text. You can also use wildcards in the parameter of the method

```
ReceiptPo.OpenScreen();
ReceiptPo.Insert();
ReceiptPo.Summary.BranchID.Select("MAIN");
ReceiptPo.Summary.FinPeriodID.Type("13-2016");
ReceiptPo.GetWarnings().VerifyContains("Transaction date is outside the
specified period date range");
```

Verifying Whether a Warning Appears on the Control

To check whether a warning appears on the control, use the `GetWarning()` method of the control class, as shown in the following example.

```
ReceiptPo.OpenScreen();
ReceiptPo.Insert();
ReceiptPo.Summary.BranchID.Select("MAIN");
ReceiptPo.Summary.FinPeriodID.Type("13-2016");
ReceiptPo.Summary.FinPeriodID.GetWarning().VerifyContains("Transaction date
is outside the specified period date range");
```

Verifying the Text of the Warning Message on the Cell of a Detail Table

To check whether a warning appears on the cell of a detail table, activate the needed row and use the `GetWarning()` method of the cell class, as shown in the following example.

```
ReceiptPo.OpenScreen();
ReceiptPo.Insert();
ReceiptPo.Summary.BranchID.Select("MAIN");
ReceiptPo.Details.New();
ReceiptPo.Details.Row.BranchID.Reset();
ReceiptPo.Details.SelectRow(1);
ReceiptPo.Details.Row.BranchID.GetWarning().VerifyContains("'Branch' may not
be empty.");
```

Verifying the Text of the Warning Message on the Row of a Detail Table

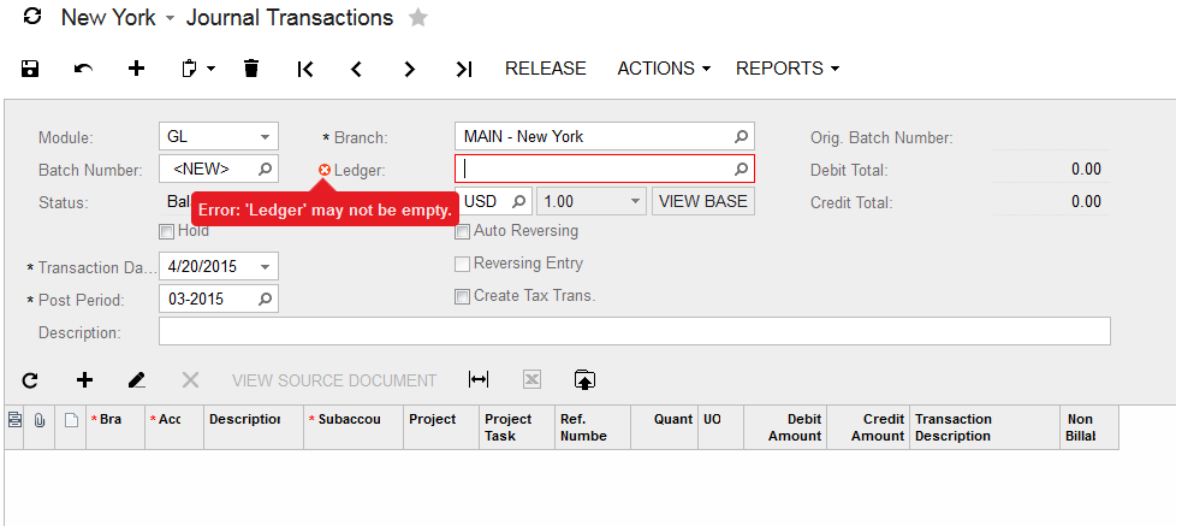
To check the text of the warning message that appears on the row of a detail table, activate the needed row and use the `GetWarning()` method of the detail table class, as shown in the following example.

```
ReceiptPo.OpenScreen();
```

```
ReceiptPo.Insert();
ReceiptPo.Summary.BranchID.Select("MAIN");
ReceiptPo.Details.New();
ReceiptPo.Details.Row.BranchID.Reset();
ReceiptPo.Details.SelectRow(1);
ReceiptPo.Details.Row.GetWarning().VerifyContains("'Branch' may not be
empty.");
```

How to Work with Errors

Test SDK provides the `GetError()` method for working with errors that occur on the form or on a control, such as the error in the following screenshot. By using the `GetError()` method, you can verify whether an error appears on the form or on the control, and check the text of the error message.



Verifying Whether an Error Appears on the Form

To check whether an error appears on the form, use the `GetError()` method of the form class, as shown in the following example.

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.Summary.LedgerID.Reset();

ReceiptPo.HasError().VerifyEquals(true);
```

Verifying that No Error Appears on the Form

To make sure that no error appears on the form, use the `GetError()` method of the form class with the parameter set to `False`, as shown in the following example.

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.HasError().VerifyEquals(false);
```

Verifying the Text of the Error Message on the Form

To check that the correct text of the error message is displayed on the form, use the `GetError()` method of the form class with the text of the error as the parameter, as shown in the following example. You can specify a part of the error message in the parameter of the method; in this case, the method checks whether the error message on the form contains the specified text. You can also use wildcards in the parameter of the method

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.Summary.LedgerID.Reset();

ReceiptPo.GetErrors().VerifyContains("'Ledger' may not be empty.");
```

Verifying Whether an Error Appears on the Control

To check whether an error appears on the particular control, use the `GetError()` method of the control class, as shown in the following example.

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.Summary.LedgerID.Reset();

ReceiptPo.Summary.LedgerID.HasError().VerifyEquals(true);
```

Verifying the Text of the Error Message on a Control

To check that the correct text of the error message is displayed on the control, use the `GetError()` method of the control class with the text of the error as the parameter, as shown in the following example. You can specify a part of the error message in the parameter of the method; in this case, the method checks whether the error message on the form contains

the specified text. You can also use wildcards in the parameter of the method (for details, see How to Use Wildcards When Verifying Errors and Warnings).

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.Summary.LedgerID.Reset();

ReceiptPo.Summary.LedgerID.GetError().VerifyContains("'Ledger' may not be
empty.");
```

Verifying the Text of the Error Message on the Cell of a Detail Table

To check whether an error appears on the particular cell of a detail table, activate the needed row and use the `GetError()` method of the cell class, as shown in the following example.

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.Details.New();

ReceiptPo.Details.Row.BranchID.Reset();

ReceiptPo.Details.SelectRow(1);

ReceiptPo.Details.Row.BranchID.GetError().VerifyContains("'Branch' may not be
empty.");
```

Verifying the Text of the Error Message on the Row of a Detail Table

To check the text of the error message that appears on the row of a detail table, activate the needed row and use the `GetError()` method of the detail table class, as shown in the following example.

```
ReceiptPo.OpenScreen();

ReceiptPo.Insert();

ReceiptPo.Summary.BranchID.Select("MAIN");

ReceiptPo.Details.New();

ReceiptPo.Details.Row.BranchID.Reset();

ReceiptPo.Details.SelectRow(1);

ReceiptPo.Details.Row.GetError().VerifyContains("'Branch' may not be
empty.");

,
```

How to Use Column Filters of a Detail Table

You can specify a filtering condition in every column of a detail table. The list of conditions and filters is available in the user interface of your Acumatica application and depends on the control type of the column (such as text, number, or date).

In this topic, you can find the descriptions of the following tasks:

Setting a Column Filter

To specify a column filter, use the `Columns` property. See below for examples of setting different column filters.

The following code shows an example of setting the *Starts With* filter, which is applicable for any text or selector field.

```
SalesPrice.Details.Columns.InventoryID.StartsWith("REQ");
```

The following code examples illustrate the setting of the *Equals* filter:

- For a date picker

  ```
  SalesPrice.Details.Columns.EffectiveDate.Equals(new
  System.DateTime(2011, 5, 16));
  ```

- For a box with text

  ```
  SalesPrice.Details.Columns.InventoryID_InventoryItem_Descr.Equals("RQ13
  ");
  ```

- For a box with a number

  ```
  SalesPrice.Details.Columns.SalesPrice.Equals(60);
  ```

The following code illustrates the setting of the *Is Between* and *Is Greater than or Equal to* filters, which are available for date pickers and boxes with numbers.

```
SalesPrice.Details.Columns.SalesPrice.IsBetween(50, 60);
SalesPrice.Details.RowsCount().VerifyEquals(2);
SalesPrice.Details.Columns.SalesPrice.IsGreaterThanOrEqualTo(60);
SalesPrice.Details.RowsCount().VerifyEquals(20);
```

The code below shows how to use the *True* and *False* filters, which are available for check box columns.

```
SalesPrice.Details.Columns.IsPromotionalPrice.IsTrue();
SalesPrice.Details.Columns.IsPromotionalPrice.IsFalse();
```

The following code shows how to select the filtering values for a drop-down column. To specify the values to be selected, you can use either the values visible in the UI or the internal IDs of the values.

```
//Using values visible in the UI
EntryType.Details.Columns.Module.SelectValues("CA", "AP");
//Using internal IDs of the values
FixedAsset.TransactionHistory.Columns.TranType.SelectValues("A+", "A-", "P+",
"P-");
```

Clearing a Filter

To clear the filter of a column, use the `ClearFilter()` method. This method is not available for drop-down columns.

```
EntryType.Details.Columns.EntryTypeId.ClearFilter();
```

To clear the filter for a drop-down column, call the `SelectAll()` method.

```
EntryType.Details.Columns.Module.SelectAll();
```

How to Navigate through the Rows of a Detail Table

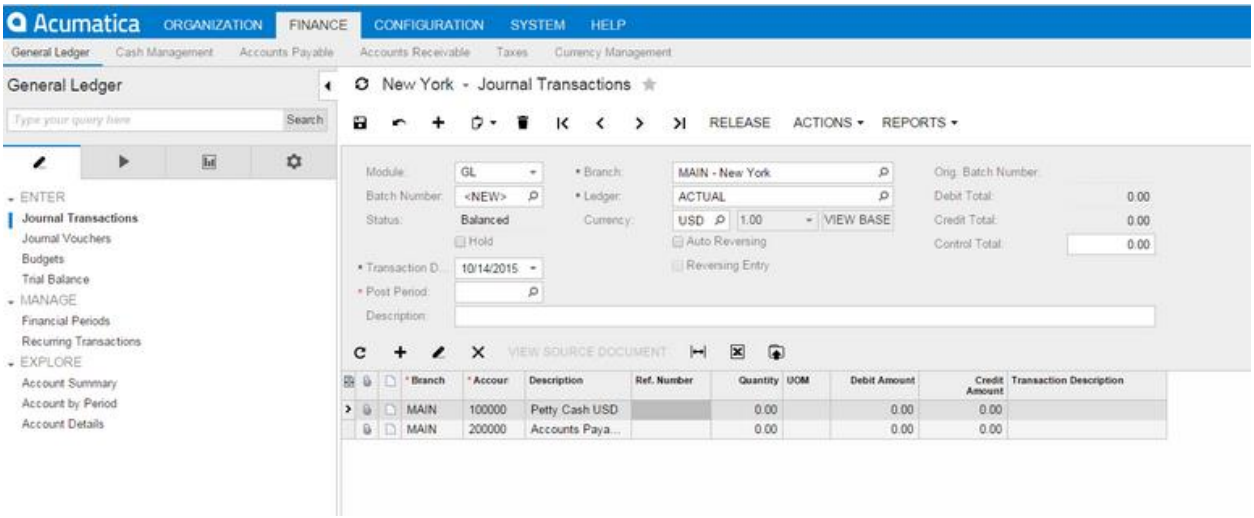
To navigate through the rows of a detail table, use the `SelectRow()` method. This method can select a record in the detail table by its row number or by a *<Column Name>:<Cell Value>* pair.

If the specified record cannot be activated (for example, a wrong index or nonexistent *<Column Name>:<Cell Value>* pair is specified as a parameter of the method), the `SelectRow()` method prints an error message in the log.

IMPORTANT NOTE

To select the very first row in a grid use index 1, e.g.: JournalEntry.Details.SelectRow(1);

The following screenshot illustrates the selection of the first row of a detail table on the Journal Transactions form (GL301000; Finance > General Ledger > Enter).



The code below shows how to select the first row of the detail table on the Journal Transactions form (GL301000) by row number:

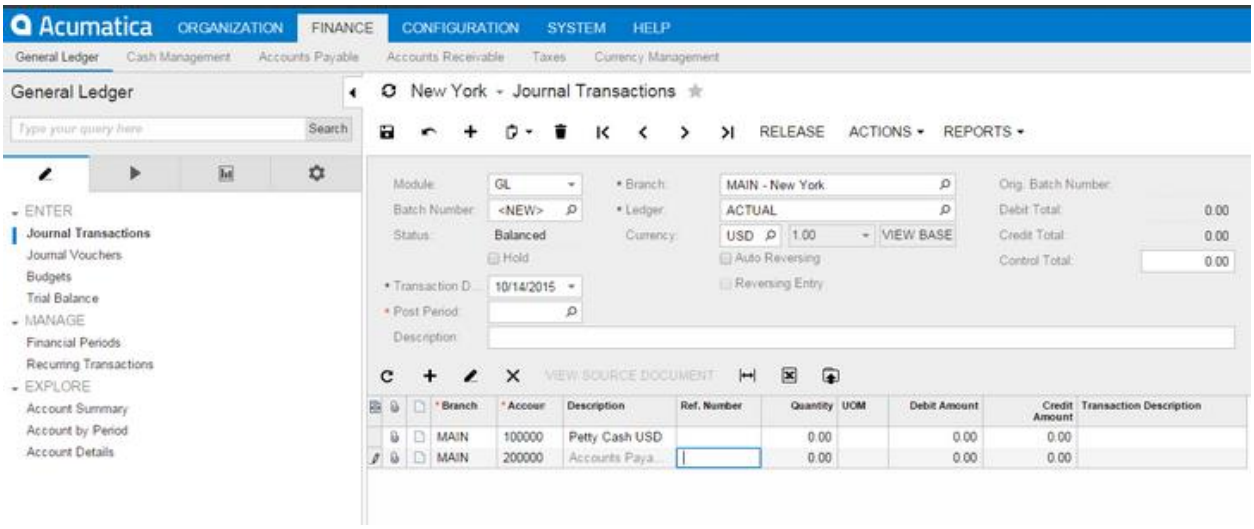
```
JournalEntry.OpenScreen();
JournalEntry.Insert();
JournalEntry.Details.New();
JournalEntry.Details.Row.AccountID.Select("100000");
JournalEntry.Details.New();
JournalEntry.Details.Row.AccountID.Select("200000");
JournalEntry.Details.SelectRow(1);
```

The code below shows how to select the first row of the detail table on the Journal Transactions form (GL301000) by *<Column Name>:<Cell Value>* pair:

```
JournalEntry.OpenScreen();
JournalEntry.Insert();
JournalEntry.Details.New();
JournalEntry.Details.Row.AccountID.Select("100000");
JournalEntry.Details.New();
JournalEntry.Details.Row.AccountID.Select("200000");
JournalEntry.Details.SelectRow(JournalEntry.Details.Columns.Description,
"Petty Cash USD");
```

How to Commit Changes in Rows of a Detail Table

To commit changes in the rows of a detail table, use the `CommitRows()` method. This method commits all changes made to the rows of the detail table and exits the table editing mode. The following screenshot shows two rows that are committed on the Journal Transactions form (GL301000; Finance > General Ledger > Enter).



The following code shows how to commit all changes made to the rows of the detail table on the Journal Transactions form (GL301000).

```
JournalEntry.OpenScreen();
JournalEntry.Insert();
JournalEntry.Details.New();
JournalEntry.Details.Row.AccountID.Select("100000");
JournalEntry.Details.New();
JournalEntry.Details.Row.AccountID.Select("200000");
JournalEntry.Details.CommitRows();
```