



# T280 Testing Business Logic with the Acumatica Unit Test Framework

Gabriel Paz Paiva

Software Developer Intern

# Timing and Agenda

---

**May 30, 2023 -10 AM -11.00 AM**

**Lesson 1: Creating a Test Project and a Test Class**

**Lesson 2: Creating a Test Method**

**May 31, 2023 -10 AM -11.00 AM**

**Lesson 3: Correctly Assigning Field Values in Tests**

**Lesson 4: Testing the Display of Errors and Warnings**



Creating a Test Project and a Test Class  
Creating a Test Method  
Correctly Assigning Field Values in Tests  
Testing the Display of Errors and Warnings

# Company Story

---

Company Story The Smart Fix company specializes in repairing cell phones of several types. The company provides the following services:

- Battery replacement: This service is provided on customer request and does not require any preliminary diagnostic checks.
- Repair of liquid damage: This service requires a preliminary diagnostic check and a prepayment.
- Screen repair: This service is provided on customer request and does not require any preliminary diagnostic checks.

To manage the list of devices serviced by the company and the list of services the company provides, two custom maintenance forms, Repair Services (RS201000) and Serviced Devices (RS202000), have been added to the company's Acumatica ERP instance. Another custom maintenance form, Services and Prices (RS203000), gives users the ability to define and maintain the price for each provided repair service. Also, the Stock Items (IN202500) form of Acumatica ERP has been customized to give users the ability to mark particular stock items as repair items—that is, the items (such as replacement screens and batteries) that are supplied to the customer as part of the repair services. The Repair Work Orders (RS301000) custom data entry form is used to create and manage work orders for repairs. On the Repair Work Order Preferences (RS101000) custom setup form, an administrative user specifies the company's preferences for the repair work orders.

# Company Story

---

## Mission Description

To keep the business logic consistent, you need to develop unit tests for the developed forms. Adding these unit tests to the customization library will ensure that the implemented behavior will persist even if changes are made later to the customization library.

Day 1

# Lesson 1: Creating a Test Project and a Test Class

---

## Learning Objectives

In this lesson, you will learn how to do the following:

- Create a project in Visual Studio
- Configure the project to be a test project that uses the xUnit.net library and the Acumatica Unit Test Framework
- Create the test class that will contain unit tests

## To Create a Test Class

---

You need to create a class for testing the Repair Services (RS201000) custom form, whose business logic is implemented in the `RSSVRepairServiceMaint` class.



## Figure: xunit package information

---



Version:

Options

### Description

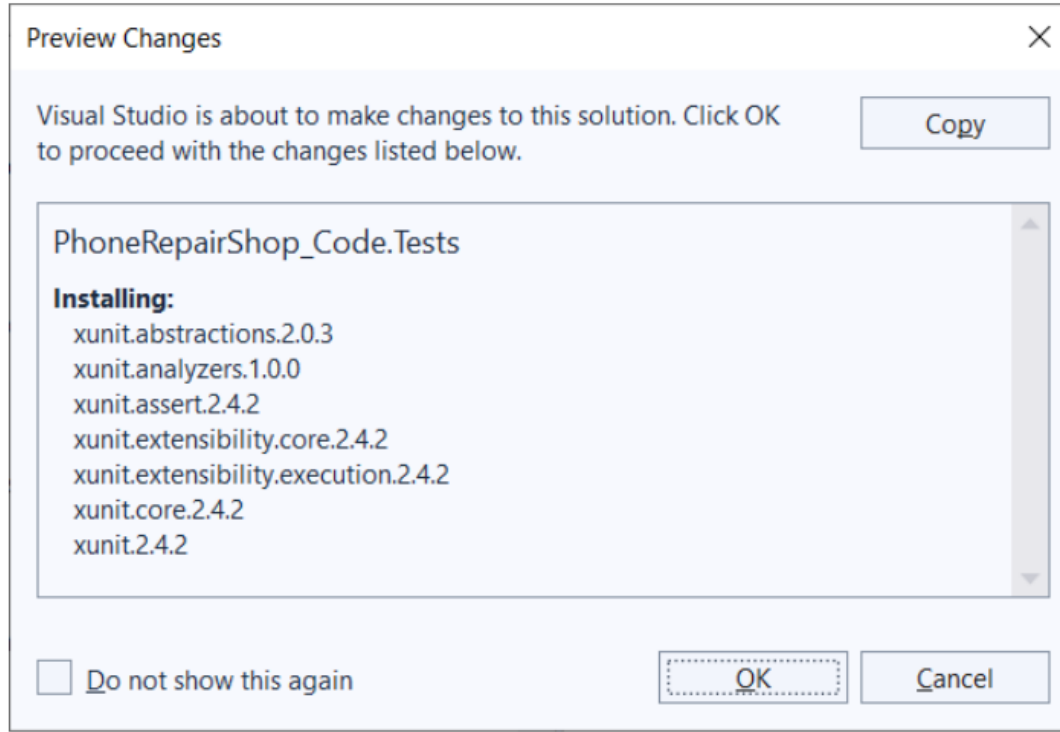
xUnit.net is a developer testing framework, built to support Test Driven Development, with a design goal of extreme simplicity and alignment with framework features.

Installing this package installs xunit.core, xunit.assert, and xunit.analyzers.

**Version:** 2.4.2  
**Author(s):** jnewkirk,bradwilson  
**License:** [Apache-2.0](#)  
**Downloads:** 277,512,152  
**Date published:** Tuesday, August 2, 2022 (8/2/2022)

## Figure: Preview Changes dialog box

---



# Lesson 2: Creating a Test Method

---

## Learning Objectives

In this lesson, you will learn how to do the following:

- Create test methods without parameters
- Create an instance of the tested graph
- Create and update objects in the graph cache
- Verify that the desired conditions are met for the values present in the method
- Run and debug test methods
- Create test methods with parameters
- Set values for the test parameters
- Create unit tests for graph extensions

## To Create a Test Method Without Parameters

---

Suppose that you want to make sure that the following behavior of the Repair Services (RS201000) custom form has not changed: The selection of the Walk-In Service check box and the selection of the Requires Preliminary Check check box are mutually exclusive. You need to create a test method that changes the state of one check box and checks the state of the other check box.

## Figure: The Test Explorer window with added tests

The screenshot shows the Test Explorer window with a toolbar at the top. The toolbar includes icons for running tests (play, stop, refresh, cancel), a summary bar showing 3 tests, 1 passed, 0 failed, and 2 not run, and a search bar labeled "Search Test Explorer".

Test	Duration
PhoneRepairShop_Code.Tests (3)	8,2 sec
PhoneRepairShop_Code.Tests (3)	8,2 sec
InventoryItemMaintTests (2)	
RepairItemTypeEnabled_WhenRepairItemSelec...	
RepairItemTypeEnabled_WhenRepairItemSel...	
RepairItemTypeEnabled_WhenRepairItemSel...	
RSSVRepairServiceMaintTests (1)	8,2 sec
PreliminaryCheckAndWalkInServiceFlags_AreO...	8,2 sec

**Group Summary**

PhoneRepairShop\_Code.Tests

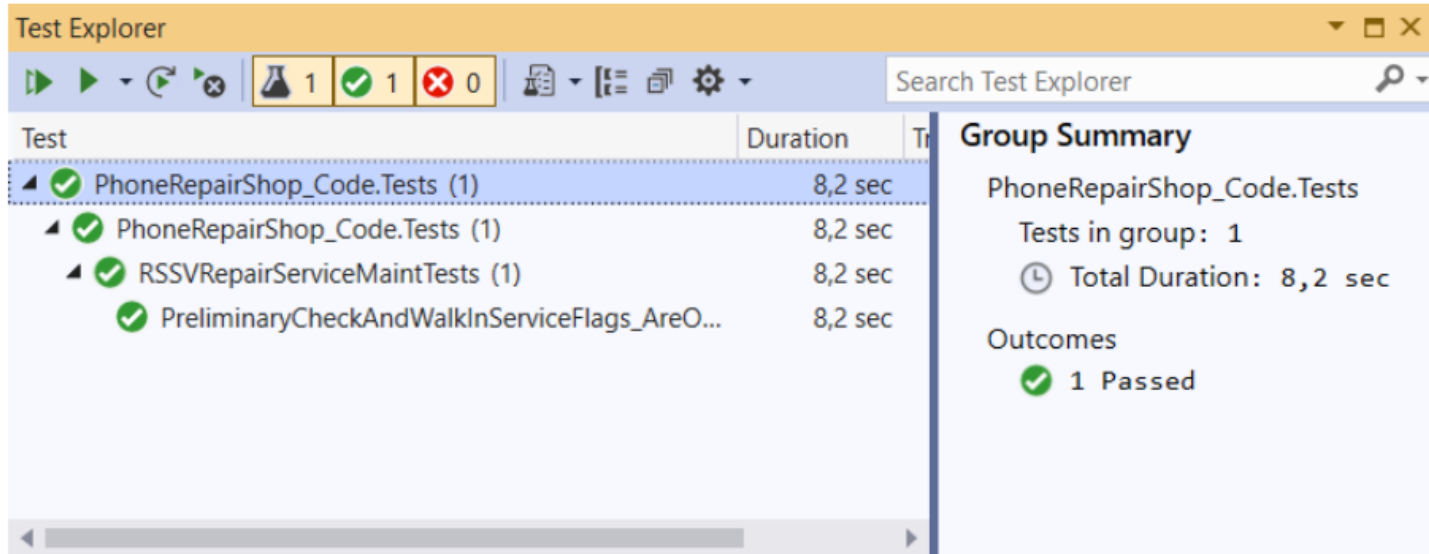
Tests in group: 3

⌚ Total Duration: 8,2 sec

**Outcomes**

- ⚠ 2 Not Run
- ✅ 1 Passed

## Figure: The Test Explorer window with one test



The screenshot shows the Test Explorer window with a toolbar at the top containing icons for running tests, a search bar, and a summary of test results (1 passed, 0 failed). The main area displays a tree view of test groups, with the selected group expanded to show its sub-items. The right-hand pane provides a detailed group summary, including the total duration and the number of passed tests.

Test	Duration	Tr
PhoneRepairShop_Code.Tests (1)	8,2 sec	
PhoneRepairShop_Code.Tests (1)	8,2 sec	
RSSVRepairServiceMaintTests (1)	8,2 sec	
PreliminaryCheckAndWalkInServiceFlags_AreO...	8,2 sec	

**Group Summary**

PhoneRepairShop\_Code.Tests

Tests in group: 1

🕒 Total Duration: 8,2 sec

Outcomes

✅ 1 Passed

## Day 2

# Lesson 2: Creating a Test Method

---

## Learning Objectives

In this lesson, you will learn how to do the following:

- Create test methods with parameters
- Set values for the test parameters
- Create unit tests for graph extensions



## To Create a Test Method with Parameters

---

Suppose that you want to make sure that the following behavior of the customized Stock Items (IN202500) form has not changed: The selection of the Repair Item check box causes the Repair Item Type drop-down list to be available, and the clearing of the Repair Item check box causes the Repair Item Type drop-down list to be unavailable. You need to create a test method that selects or clears the Repair Item check box and checks whether the Repair Item Type drop-down list is available or unavailable, respectively.

## To Register a Service

---

Suppose that running a test method leads to the generation of the `Autofac.Core.Registration.ComponentNotRegisteredException` exception. When this exception occurs, the error message specifies which component (interface) is not registered. You need to register the service that implements the specified interface.

# Lesson 3: Correctly Assigning Field Values in Tests

---

## Learning Objectives

In this lesson, you will learn how to do the following:

- Specify the values of key and non-key fields when an object is created or updated
- Use the default field values when an object is created

# To Test the Effect of Changing Field Values

---

Suppose that you want to make sure that the following behavior of the Services and Prices (RS203000) form has not changed: If changes are made to the state of the Required or Default check box (or both check boxes) of a row, these changes affect the states of these check boxes in other rows. Also, you want to make sure that the prices are calculated correctly.

You need to create a test method. In this method, you need to create at least three repair items of different types and a labor item, configure them, and make sure that the fields that correspond to the Required and Default check boxes have the correct states and that the boxes with prices have the correct values.

# Lesson 4: Testing the Display of Errors and Warnings

---

## Learning Objectives

In this lesson, you will learn how to do the following:

- Test the logic of a graph that has a setup DAC
- Test whether a proper error is attached to a field
- Test whether a proper warning is attached to a field
- Clear the cache of errors
- Find the proper DAC object in the cache

# To Test the Display of Errors and Warnings

---

Suppose that you want to make sure that the following behavior of the Repair Work Orders (RS301000) form has not changed:

- When a new work order is initialized, the system copies to it the repair items and labor items of the RSSVRepairPrice object that has the same DeviceID and ServiceID values as the work order being initialized.
- Negative values of labor quantities are prohibited.
- The values of labor quantities are not permitted to be less than the minimum values.
- In work orders for which the priority is Low, repair services that are marked as requiring preliminary checks are prohibited.

You need to create a test method. In this method, you need to initialize the settings of the graph.



Thank you!

Gabriel Paz Paiva