Unit Tests for Regular Business Logic

Andrew Boulanov

Cloud xRP Summit

Virtual Developer Conference

Acumatica

The Cloud ERP

Acumatica Unit Tests Framework Progress

- We introduced unit testing of reusable business objects concept a year ago
- While widely used in Acumatica, reusable business objects are still a new technology for partners
- Partners usually create customization around Acumatica regular business objects
- Sometimes partners create their own separate graphs
- All of this code will benefit from the unit tests framework
- The tests will become less dependent from the Acumatica code than any kind of integration tests involving user interface and database
- In the future we may consider ability to include unit tests into customization package to execute during validation process

Design and Operation Principles

- Unit test code is kept very similar to the code of Acumatica business logic, using same API
- Unit test initiates CRUD operations like conventional work via user interface
- CRUD operations raises all events connected to them in the common way
- Events invoke different parts of code connected to them
- Database is not accessible in any way
- Strictly required definitions like dimensions are provided via Service Locator
- Any other object that is needed for the operation, should be created by unit test code in advance
- The code may populate only desired object content leaving everything other in its default state
- Ones created objects are accessible in database queries

Access to Acumatica Unit Tests Framework

- Everything bellow is actual for Acumatica distributives starting 2019r1 update 5
- Base objects are include into PX.Data.Unit namespace what is a part of PX.Data.dll
- You should inherit your test class from TestBase, then it will be possible to user its protected methods like Setup, Tail, RegisterServices
- TestBase also provides default services registered with ServiceLocator, including Licensing mock, Dimensions, Access provider, etc.
- The same namespace contains basic classes for testing reusable business objects, DocumentMockBase, DetailMockBase, GraphMockBase
- PX.Objects.dll includes PX.Objects.Unit namespace that currently consists of 2 useful services: FinPeriodServiceMock required for most Acumatica graphs involving financials part, and CurrencyServiceMock required to test our new multi currency feature implementation

Configuring Unit Tests Solution

- Create a library and reference PX.Common.dll, PX.Data.dll and PX.Objects.dll from Acumatica installation, add any your own libraries with business logic
- Add Nuget packages, you may find list of them in the SampleTest solution
- Inherit from PX.Data.Unit.TestBase, write your tests
- Start them from the test explorer
- xUnit runner is also supported of course

Most important API Calls

- Override RegisterServices if you want to provide a mock via ServiceLocator, FinPeriodServiceMock is a good example
- Call Setup before graph creation to provide preferences to graph constructor
- Cache.Insert should be used to instantiate referenced objects, like accounts/sibs, customers/vendors, locations
- Be aware of basic classes used as references, for example, BAccountR should be used in most places instead of customer/vendor
- The framework will provide default empty content for right tables in case of database queries containing joins, use Tail method if you want to substitute with your own implementation

Acumatica

The Cloud ERP

Andrew Boulanov

aboulanov@acumatica.com