Module 2 Day 7

Integration Testing

What makes an application?

- **Program Data**
  - ✓ Variables & .NET Data Types
  - ✓ Arrays
  - ✓ More Collections (list, dictionary, stack, queue)
  - ✓ Classes and objects (OOP)

- **Program Logic**
  - ✓ Statements and expressions
  - ✓ Conditional logic (if)
  - ✓ Repeating logic (for, foreach, do, while)
  - ✓ Methods (functions / procedures)
  - ✓ Classes and objects (OOP)
  - ❑ Frameworks (MVC)

- **Input / Output**
  - **User**
    - ✓ Console read / write
    - ❑ HTML / CSS
    - ❑ Front-end frameworks (HTML / CSS / JavaScript)
  - **Storage**
    - ✓ File I/O
    - ❖ Relational database
    - ❑ APIs

DAO, continued

- AppSettings.json
- Nullable columns
- Inserting, updating and deleting rows
- @@Identity or Scope_Identity
- ExecuteNonQuery()
  - When there are no results to be returned
- ExecuteScalar()
  - When exactly one row and one column is expected
  - E.g., "Select Count(*) from table"

Let's Code

Types of Testing

- Unit Testing
  - Verifies parts of an application independently from other parts (isolated)
  - Narrow scope (focused)
  - Tests many cases (deep)
  - Fast!
- Integration Testing
  - Verifies interactions between multiple components
  - Broader scope
  - More difficult to test thoroughly
- End-to-end Testing (E2E)
  - Verifies interaction from end-user through a transaction and back to end-user
  - Broadest scope
  - Most difficult to test all possible combinations of scenarios
  - Slowest to run (for both setup and execution)

Faster

Broader scope

DAO Integration Testing

- DAO testing is an example of an Integration Test
  - DAO often relies on outside resources (DBMS), thus *integration*
- Any type of test should be:
  - *Repeatable:* If the test passes/fails on first execution, it should pass/fail on second execution if no code has changed.
  - *Independent:* A test should be able to be run on its own, independent of other tests.
  - *Obvious:* When a test fails, it should be as obvious as possible why it failed.

Testing DAOs

- Test Select, Insert, Update and Delete to verify the expected change
  - Existence or non-existence of rows
  - RowsAffected meets expectations
- Insert a set of test data so we know our starting state
- Next test should run with clean, known data
- This can be done in a few ways
  - We are going to use Transactions for easy cleanup

Let's Code