Writing Automated Tests for Exception Throwing Code

Jason Roberts
.NET DEVELOPER

@robertsjason dontcodetired.com

Overview

Testing exceptions with NUnit

Testing exceptions with xUnit.net

Testing exceptions with MSTest V2

Key course takeaways and resources

Summary

Testing exceptions with NUnit

- Assert.That(... , Throws.TypeOf<...>());
- Assert.That(... , Throws.InstanceOf<...>());

Testing exceptions with xUnit.net

- Assert.Throws<...>(...)
- Assert.ThrowsAny<...>(...)

Testing exceptions with MSTest V2

- Assert.ThrowsException

Key Takeaways

- Use exceptions not error return codes
- Thrown exceptions bubble up the call stack until they get caught by a catch block or crash to the operating system
- All exceptions inherit from System.Exception
- Exceptions can wrap other exceptions
- Use try & catch blocks to handle exceptions
- Use a finally block to execute clean-up code

Key Takeaways

- Do not add a catch block that does nothing or just rethrows - catching should add value
- Do not use exceptions for normal program flow logic
- Be careful to preserve the stack trace when rethrowing
- Only create custom exceptions when existing exception types are not suitable
- Avoid sensitive information in exceptions as exceptions could be logged

Resources and Further Learning

Documentation:

- C# programming guide to exceptions and exception handling (http://bit.ly/excsguide)
- Framework design guidelines for exceptions (http://bit.ly/exframe)

Related Pluralsight courses:

- "Automated Testing with MSTest V2"
- "Testing .NET Code with xUnit.net: Getting Started"
- "Introduction to .NET Testing with NUnit 3"