Automating Unit Tests with Command Line Utility

Paul D. Sheriff
Business/IT Consultant PDS Consulting

psheriff@pdsa.com www.pdsa.com

Goals

Using dotnet test command

Using loggers

Run specific tests using a filter

dotnet test Command

Run from developer command prompt

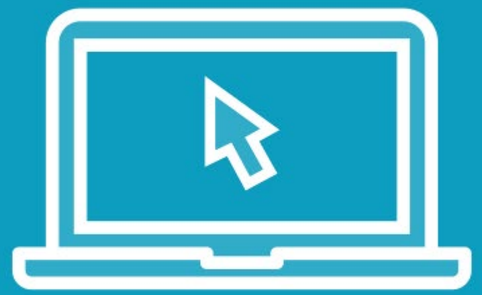
Log to console, .trx, or .html file

Run all tests

Run tests starting with...

Run tests filtered by Priority, Name or TestCategory

Demo



Show dotnet test command

Loggers

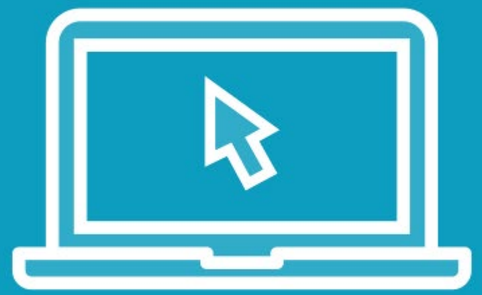
Console

(quiet, minimal, normal, detailed)

.trx file (for viewing in Visual Studio)

.html file

Demo



Log verbose information

Log to a .trx file

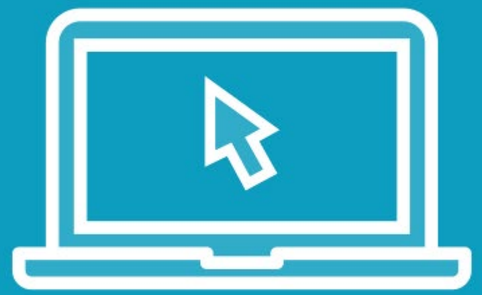
Log to a .html file

Run a Specific Test

Use --filter "Name=TestName"

Runs just that one test

Demo



Run one specific test

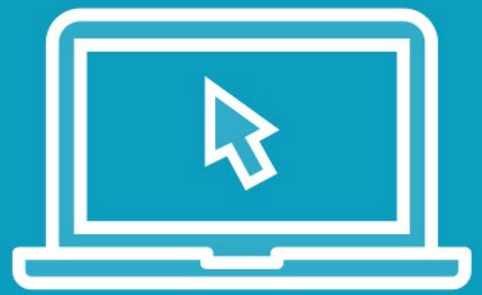
Run Matching Test Names

Use --filter "Name~TestName

Pass in partial test name

Matches on all tests that contain name

Demo



Run matching test names

Filter on Attributes

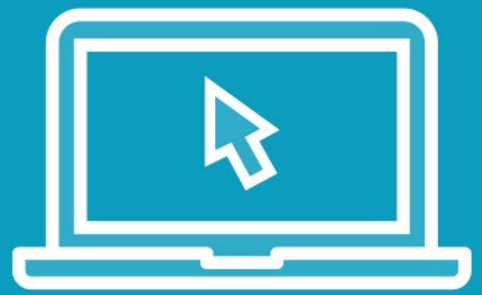
Use --filter argument

Pass in TestCategory, Priority, ClassName, Name, FullyQualifiedName

Example:

--filter "Priority=1"

Demo



Filter on attributes

Module Summary

- *dotnet test* helps you automate your tests
  - Run via a command file
  - Schedule with Task Scheduler
- Choose where to log test results
- Multiple ways to select test(s) to run
  - Run a single test name
  - Run all tests that match a partial name
  - Run all tests with a specified attribute
- Search for unit testing at www.pdsa.com/blog

Course Summary

- How unit testing improves your code
- How to build unit tests
- Using attributes
- Using different Assert classes
- Data-driven testing
- Command-line helps you automate tests

I hope you enjoyed this course!

Paul D. Sheriff

Business/Technology Consultant

psheriff@pdsa.com **www.pdsa.com**