Writing Basic SpecFlow Tests

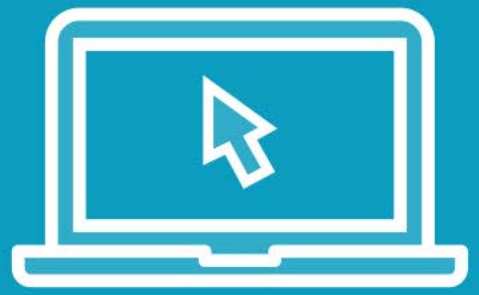
Jason Roberts
.NET MVP

@robertsjason dontcodetired.com

Overview

- Initial demo sample production code
- Create the first scenario
- Implement the test automation code
- Running and debugging test scenarios
- Adding additional scenarios
- Add some deliberate duplication
- Reviewing maintainability

Demo

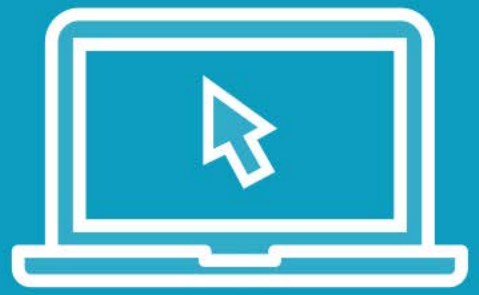


Demo Sample
Production Code

Show demo class

Add ref from test project

Demo



Creating the First Scenario

Write first scenario

Generate step definition file

Modify to remove parameters

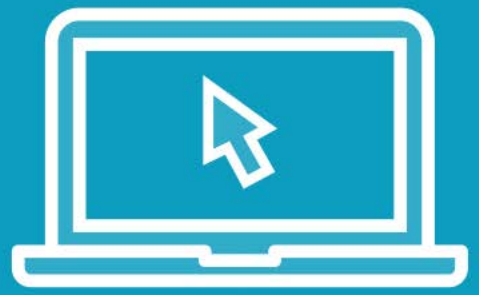
Build

Automation code not yet written

ScenarioContext.Current.Pending();

Rename GivenIMANewPlayer to GivenImANewPlayer

Demo



Writing Test Automation Code

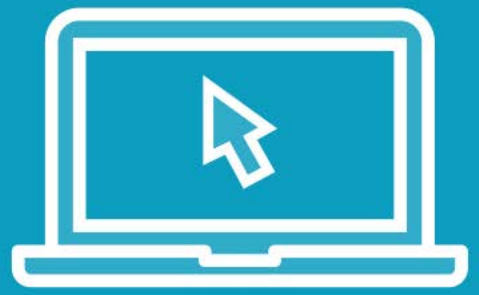
Implement Given, When, Then steps

```
private  PlayerCharacter _player;

_player.Hit(0);

Assert.Equal(100, _player.Health);
```

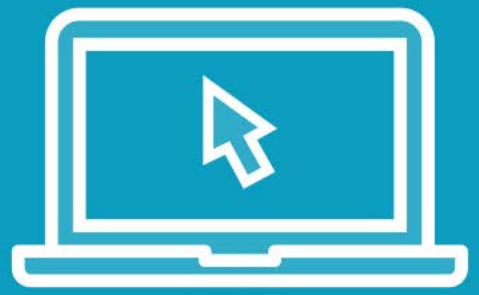
Demo



Running and Debugging Test Scenarios

- Run test with Test Explorer
- Adding breakpoint in When step
- Debugging from Test Explorer
- Using breakpoints in scenarios
- Step into code from scenario text

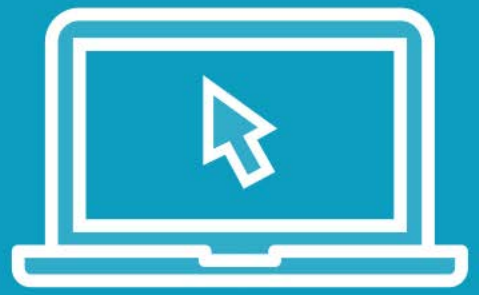
Demo



Adding Additional Scenarios

- Add new scenarios
- Starting health is reduced when hit
- Taking too much damage results in player death
- Copy steps to clipboard
- Remove parameters
- Hand write step definitions
- Add test automation code
- Run new tests

Demo



Reviewing Maintainability

- Duplication in scenarios
- Scenario steps differ only by data
- Duplication in test automation code
- Parameterized steps in next module

Summary

Saw initial PlayerCharacter class

"Taking no damage when hit doesn't affect health"

private PlayerCharacter _player;

Running and debugging test scenarios

Added two additional scenarios

Auto generation and manual creation

Reviewed maintainability

Duplicated scenario text

Duplicated automation code

Next:

Increasing Maintainability with Shared Steps, Parameters, and Data Tables