

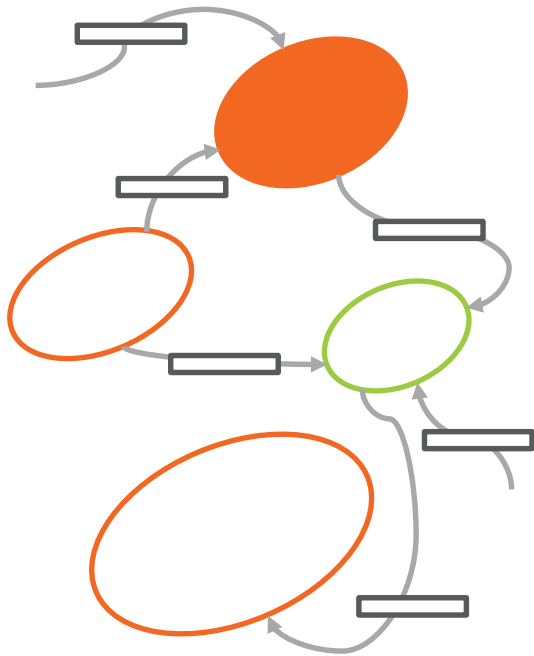
Exploring the Domain



Kevin Kuebler

@kevinkuebler

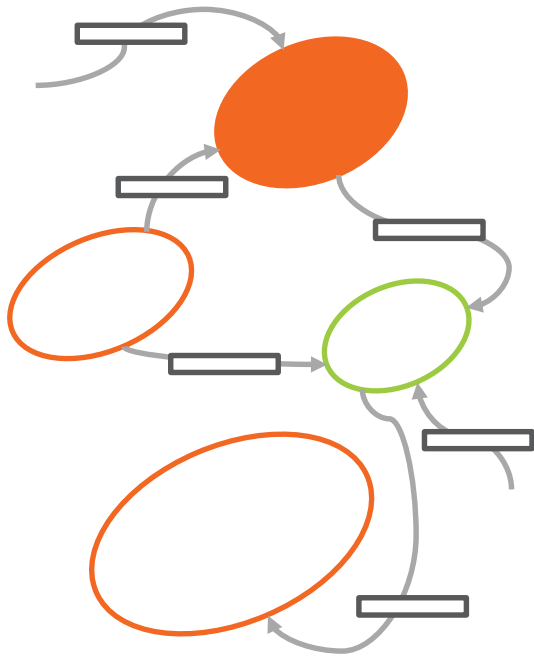
Sample Application



Domain-Driven Design Fundamentals

by Steve Smith and Julie Lerman

Sample Application

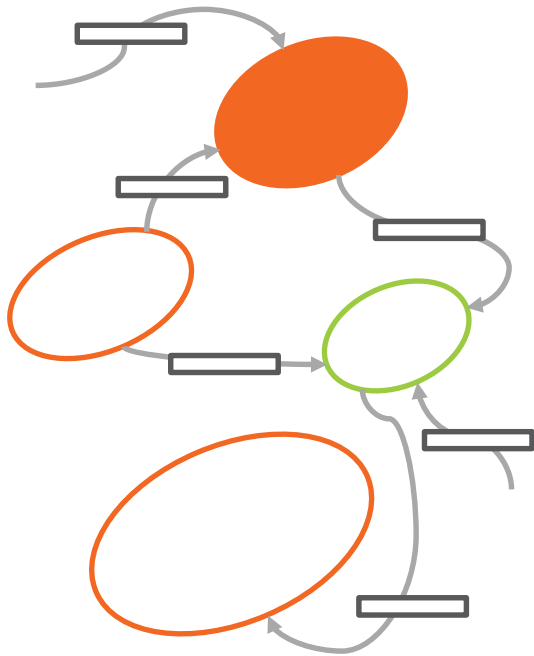


Domain-Driven Design Fundamentals

by Steve Smith and Julie Lerman

bit.ly/PS-DDD

Sample Application



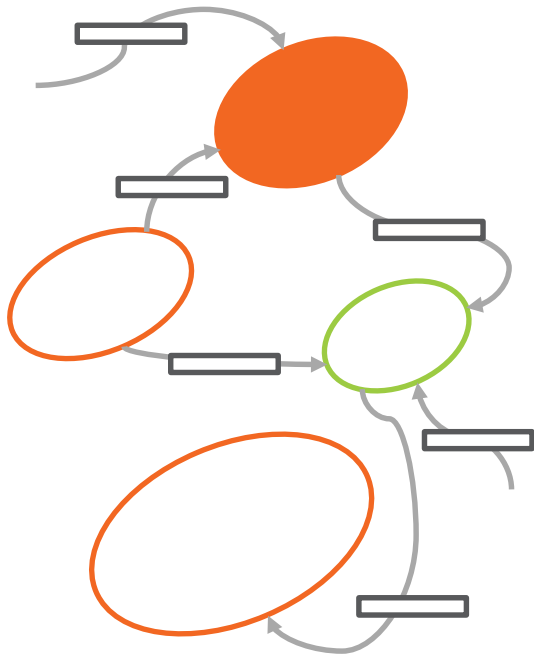
Domain-Driven Design Fundamentals

by Steve Smith and Julie Lerman

bit.ly/PS-DDD

Vet Manager
web application

Sample Application



Domain-Driven Design Fundamentals

by Steve Smith and Julie Lerman

bit.ly/PS-DDD

Vet Manager
web application

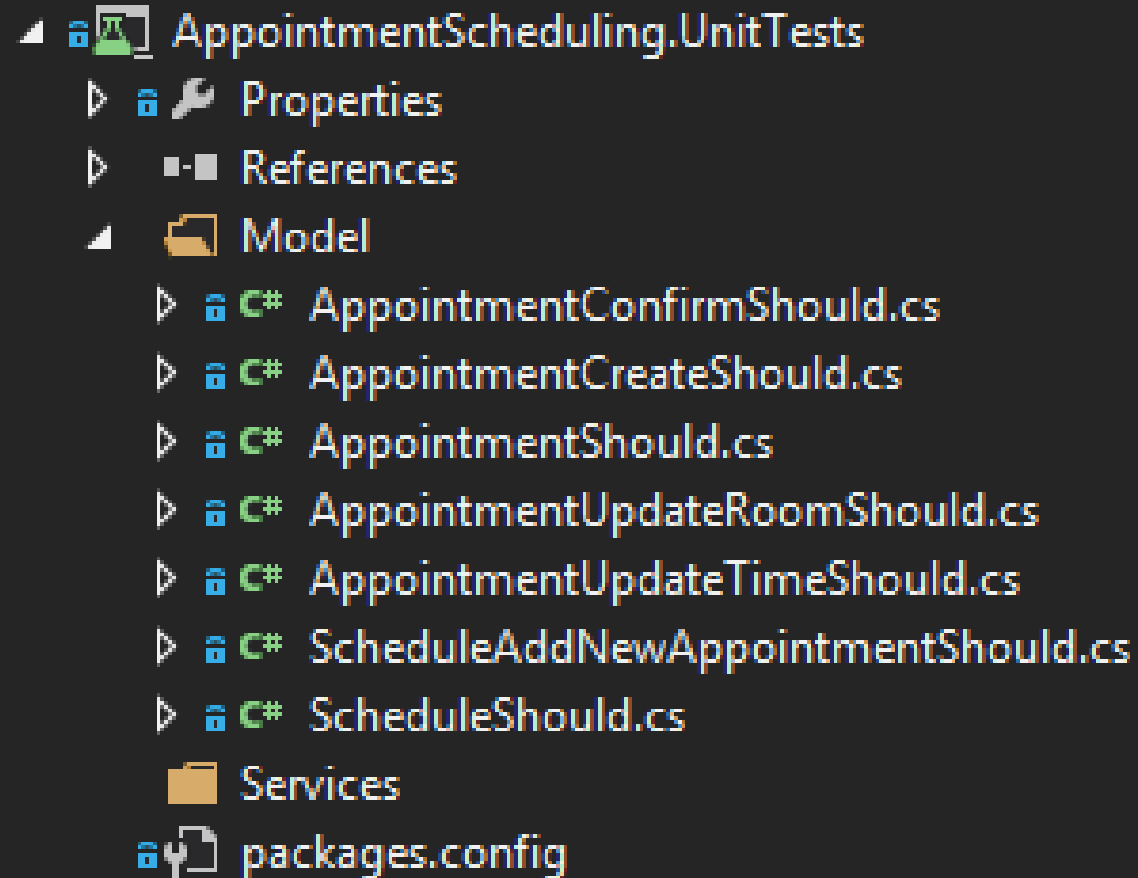
Client: Patient: Vet Manager Schedule Clients Doctors Rooms

Appointments

Today < > Monday, June 09, 2014

	Exam Room 1 Dr. Smith	Exam Room 2 Dr. Jones	Exam Room 3	Exam Room 4
	Mon 6/09	Mon 6/09	Mon 6/09	Mon 6/09
7:00 AM				
8:00 AM	(DE) Corde - Joe Eames	(DE) Willie - Tyler Young	(DE) Rocky - Brian Lagunas	(DE) Barney - Andrew Mallett
9:00 AM	(DE) Tinkerbell - Michael Perry (DE) Radar - Michael Perry	(DE) JoeFish - Tyler Young	(DE) Charlie - Jesse Liberty (DE) Allegra - Jesse Liberty	(DE) Tinkelbel - Reindert-Jan Ekker (DE) Ruske - Julie Yack (DE) Lizzie - Julie Yack
10:00 AM	(WE) Darwin - Steve Smith	(DE) Fabian - Tyler Young	(DE) Misty - Jesse Liberty	

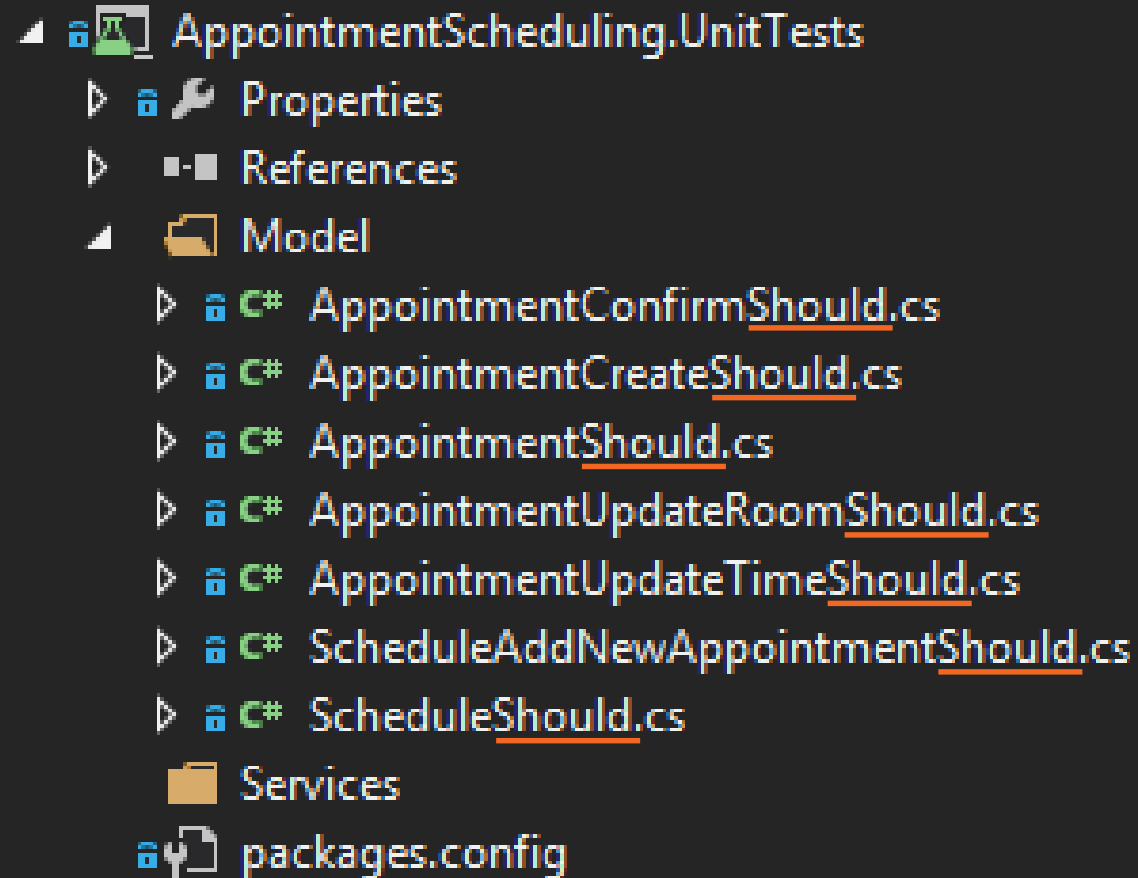
Existing Tests



```
AppointmentScheduling.UnitTests
├── Properties
├── References
├── Model
│   ├── AppointmentConfirmShould.cs
│   ├── AppointmentCreateShould.cs
│   ├── AppointmentShould.cs
│   ├── AppointmentUpdateRoomShould.cs
│   ├── AppointmentUpdateTimeShould.cs
│   ├── ScheduleAddNewAppointmentShould.cs
│   └── ScheduleShould.cs
├── Services
└── packages.config
```

The image shows a file explorer view of the `AppointmentScheduling.UnitTests` project. The project structure includes a `Properties` folder, a `References` folder, a `Model` folder containing seven test files, a `Services` folder, and a `packages.config` file. The test files in the `Model` folder are: `AppointmentConfirmShould.cs`, `AppointmentCreateShould.cs`, `AppointmentShould.cs`, `AppointmentUpdateRoomShould.cs`, `AppointmentUpdateTimeShould.cs`, `ScheduleAddNewAppointmentShould.cs`, and `ScheduleShould.cs`.

Existing Tests



```
AppointmentScheduling.UnitTests
├── Properties
├── References
├── Model
│   ├── AppointmentConfirmShould.cs
│   ├── AppointmentCreateShould.cs
│   ├── AppointmentShould.cs
│   ├── AppointmentUpdateRoomShould.cs
│   ├── AppointmentUpdateTimeShould.cs
│   ├── ScheduleAddNewAppointmentShould.cs
│   └── ScheduleShould.cs
├── Services
└── packages.config
```

The screenshot displays the 'AppointmentScheduling.UnitTests' project in a Visual Studio solution explorer. The project is expanded, showing its internal structure. It includes a 'Properties' folder, a 'References' folder, and a 'Model' folder. The 'Model' folder contains seven C# test files: 'AppointmentConfirmShould.cs', 'AppointmentCreateShould.cs', 'AppointmentShould.cs', 'AppointmentUpdateRoomShould.cs', 'AppointmentUpdateTimeShould.cs', 'ScheduleAddNewAppointmentShould.cs', and 'ScheduleShould.cs'. Additionally, there is a 'Services' folder and a 'packages.config' file at the root of the project.

Test Naming

Projects and Namespaces			
36	✓ 36	✗ 0	?
✓	<AppointmentScheduling> (36 tests)		Success
✓	<Infrastructure> (11 tests)		Success
✓	<AppointmentScheduling.UnitTests> (25 tests)		Success
✓	{ } AppointmentScheduling.UnitTests.Model (25 tests)		Success
✓	AppointmentConfirmShould (3 tests)		Success
✓	AppointmentCreateShould (7 tests)		Success
✓	AppointmentShould (2 tests)		Success
✓	AppointmentUpdateRoomShould (3 tests)		Success
✓	AppointmentUpdateTimeShould (3 tests)		Success
✓	ScheduleAddNewAppointmentShould (4 tests)		Success
✓	ScheduleShould (3 tests)		Success
✓	BeAnEntity		Success
✓	MarkConflictingAppointmentsUponCreation		Success
✓	UnmarkConflictingAppointmentsUponTheirUpdate		Success

Test Naming

Projects and Namespaces

36 36 0 0 0

- ✓ <AppointmentScheduling> (36 tests) Success
- ✓ <Infrastructure> (11 tests) Success
- ✓ <AppointmentScheduling.UnitTests> (25 tests) Success
 - ✓ AppointmentScheduling.UnitTests.Model (25 tests) Success
 - ✓ AppointmentConfirmShould (3 tests) Success
 - ✓ AppointmentCreateShould (7 tests) Success
 - ✓ AppointmentShould (2 tests) Success
 - ✓ AppointmentUpdateRoomShould (3 tests) Success
 - ✓ AppointmentUpdateTimeShould (3 tests) Success
 - ✓ ScheduleAddNewAppointmentShould (4 tests) Success
 - ✓ ScheduleShould (3 tests) Success
 - ✓ BeAnEntity Success
 - ✓ MarkConflictingAppointmentsUponCreation Success
 - ✓ UnmarkConflictingAppointmentsUponTheirUpdate Success

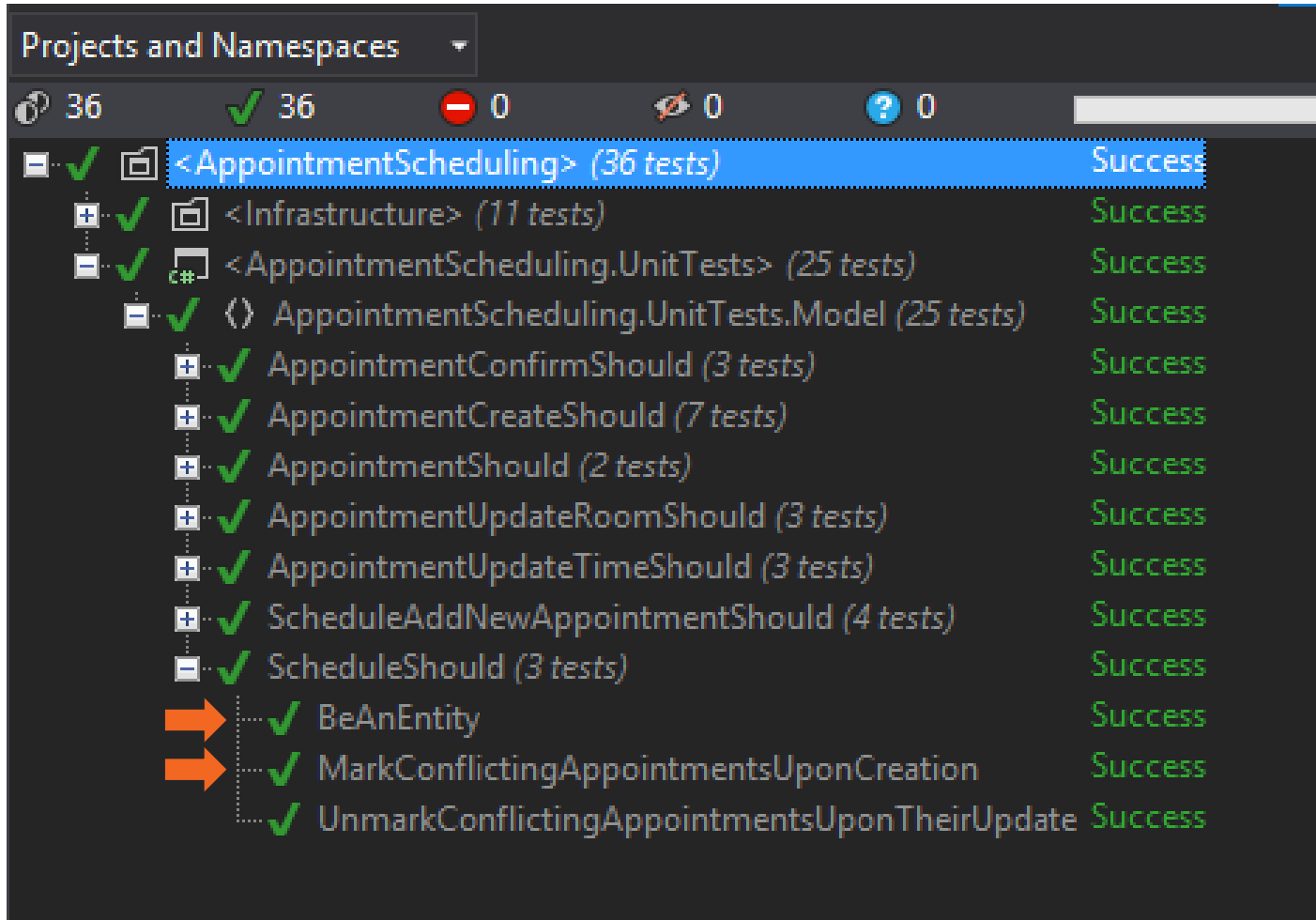
Test Naming

Projects and Namespaces

36 36 0 0 0

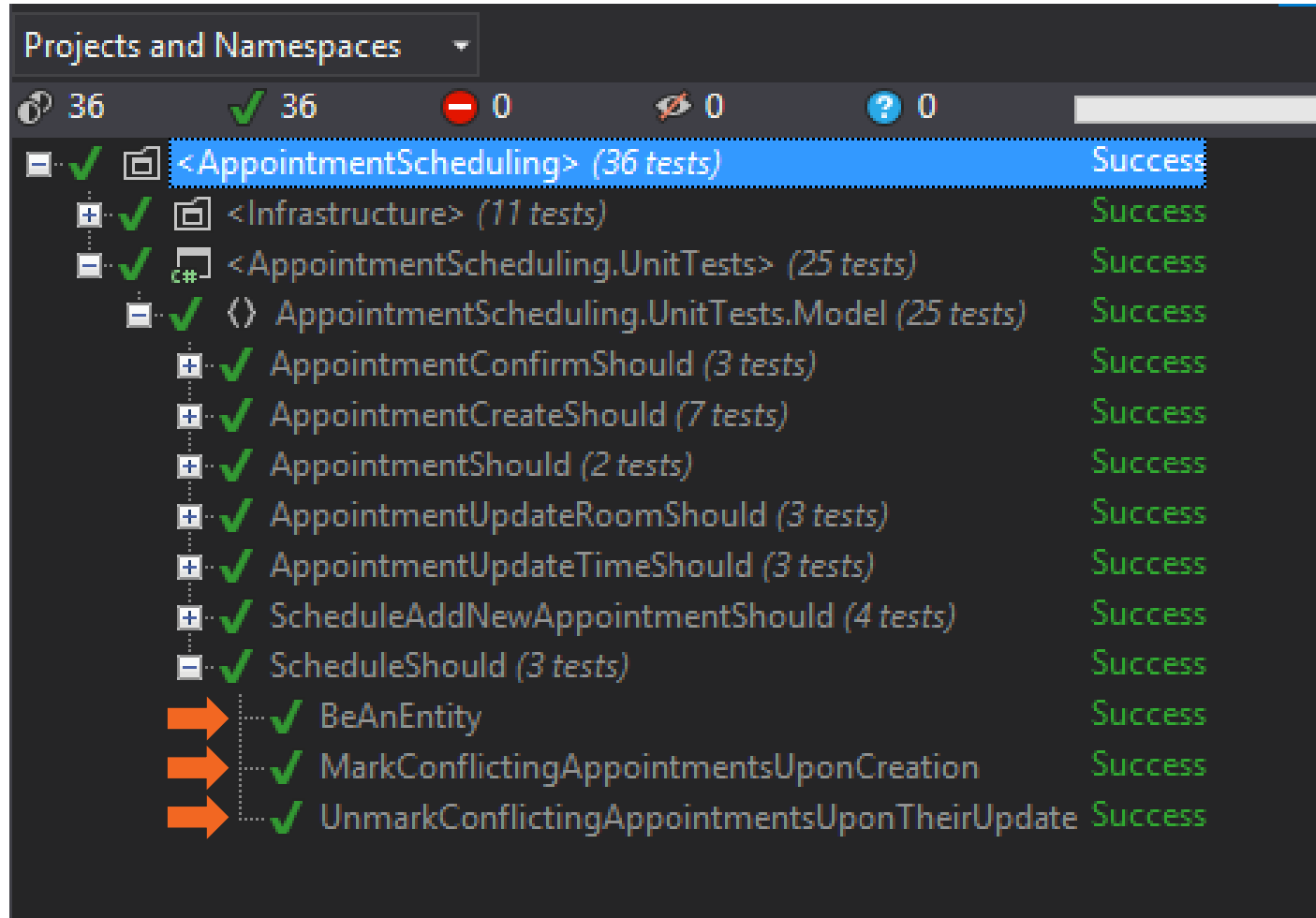
- ✓ <AppointmentScheduling> (36 tests) Success
- ✓ <Infrastructure> (11 tests) Success
- ✓ <AppointmentScheduling.UnitTests> (25 tests) Success
 - ✓ AppointmentScheduling.UnitTests.Model (25 tests) Success
 - ✓ AppointmentConfirmShould (3 tests) Success
 - ✓ AppointmentCreateShould (7 tests) Success
 - ✓ AppointmentShould (2 tests) Success
 - ✓ AppointmentUpdateRoomShould (3 tests) Success
 - ✓ AppointmentUpdateTimeShould (3 tests) Success
 - ✓ ScheduleAddNewAppointmentShould (4 tests) Success
 - ✓ ScheduleShould (3 tests) Success
 - ✓ BeAnEntity Success
 - ✓ MarkConflictingAppointmentsUponCreation Success
 - ✓ UnmarkConflictingAppointmentsUponTheirUpdate Success

Test Naming



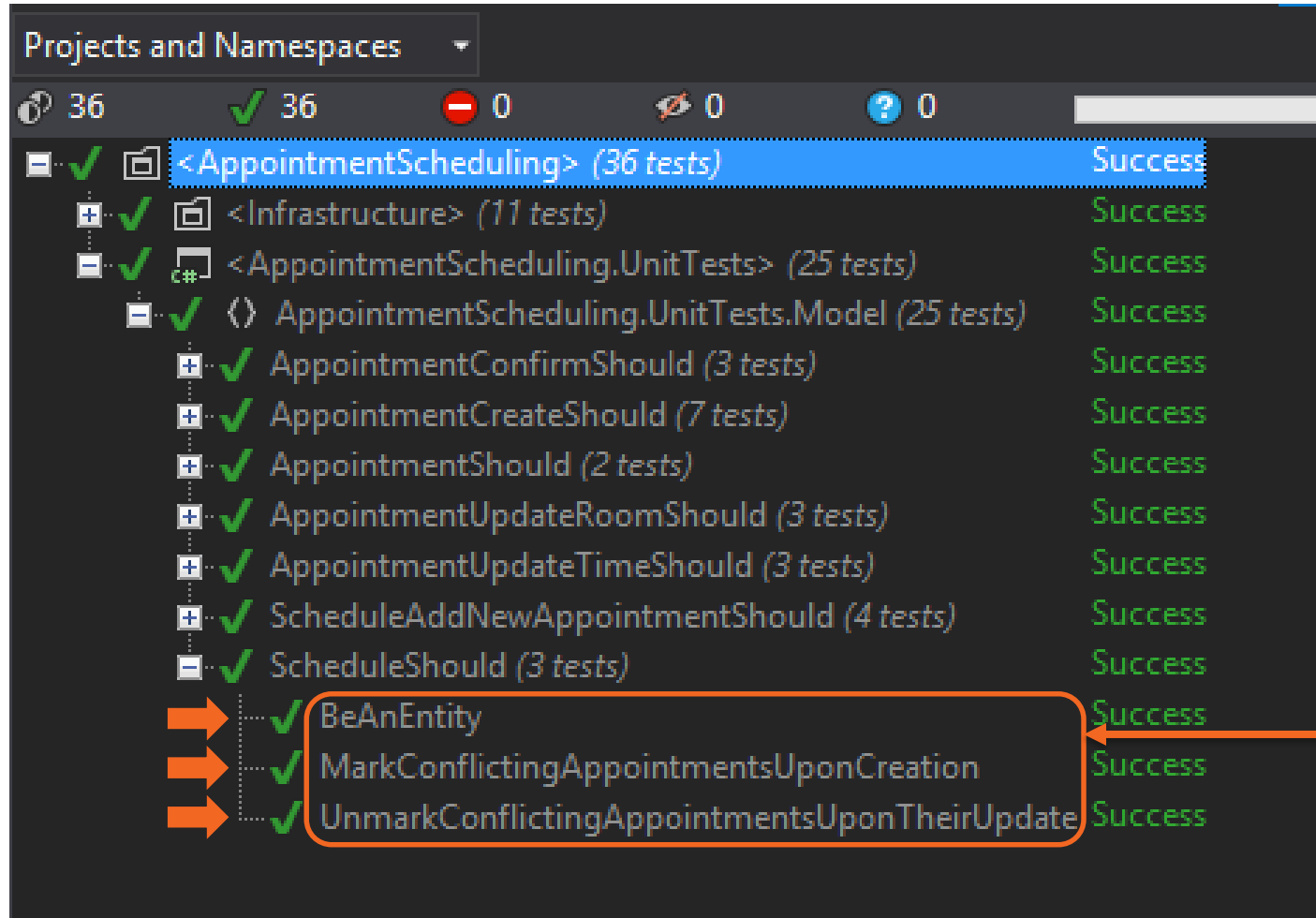
ScheduleTests
vs.
ScheduleShould

Test Naming



ScheduleTests
vs.
ScheduleShould

Test Naming



ScheduleTests
vs.
ScheduleShould

Attributes and behaviors that should be true about a Schedule

Test Naming

Projects and Namespaces										
	36		36		0		0		0	
			<AppointmentScheduling> (36 tests)				Success			
			<Infrastructure> (11 tests)				Success			
			<AppointmentScheduling.UnitTests> (25 tests)				Success			
			AppointmentScheduling.UnitTests.Model (25 tests)				Success			
			AppointmentConfirmShould (3 tests)				Success			
			AppointmentCreateShould (7 tests)				Success			
			AppointmentShould (2 tests)				Success			
			AppointmentUpdateRoomShould (3 tests)				Success			
			AppointmentUpdateTimeShould (3 tests)				Success			
			ScheduleAddNewAppointmentShould (4 tests)				Success			
			ScheduleShould (3 tests)				Success			

Test Naming

Appointment class

Projects and Namespaces					
36	✓ 36	✗ 0	⚡ 0	❓ 0	
✓	✓				<AppointmentScheduling> (36 tests) Success
+	✓				<Infrastructure> (11 tests) Success
✓	✓				<AppointmentScheduling.UnitTests> (25 tests) Success
✓	✓				() AppointmentScheduling.UnitTests.Model (25 tests) Success
+	✓				AppointmentConfirmShould (3 tests) Success
+	✓				AppointmentCreateShould (7 tests) Success
+	✓				<u>AppointmentShould</u> (2 tests) Success
+	✓				AppointmentUpdateRoomShould (3 tests) Success
+	✓				AppointmentUpdateTimeShould (3 tests) Success
+	✓				ScheduleAddNewAppointmentShould (4 tests) Success
+	✓				ScheduleShould (3 tests) Success

Test Naming

Projects and Namespaces

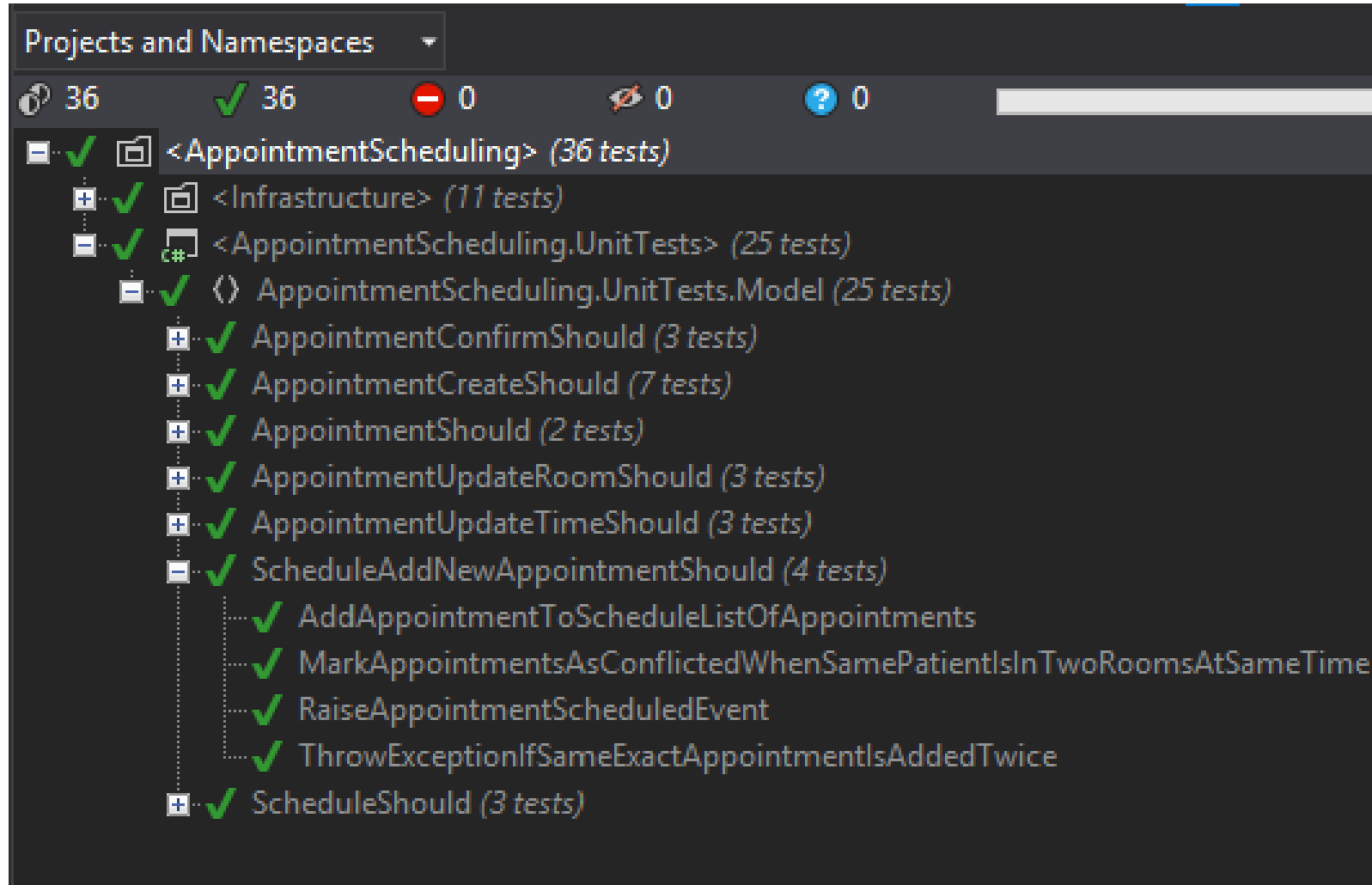
36 36 0 0 0

<AppointmentScheduling> (36 tests)	Success
<Infrastructure> (11 tests)	Success
<AppointmentScheduling.UnitTests> (25 tests)	Success
() AppointmentScheduling.UnitTests.Model (25 tests)	Success
Appointment <u>Confirm</u> Should (3 tests)	Success
AppointmentCreateShould (7 tests)	Success
Appointment <u>Should</u> (2 tests)	Success
AppointmentUpdateRoomShould (3 tests)	Success
AppointmentUpdateTimeShould (3 tests)	Success
ScheduleAddNewAppointmentShould (4 tests)	Success
ScheduleShould (3 tests)	Success

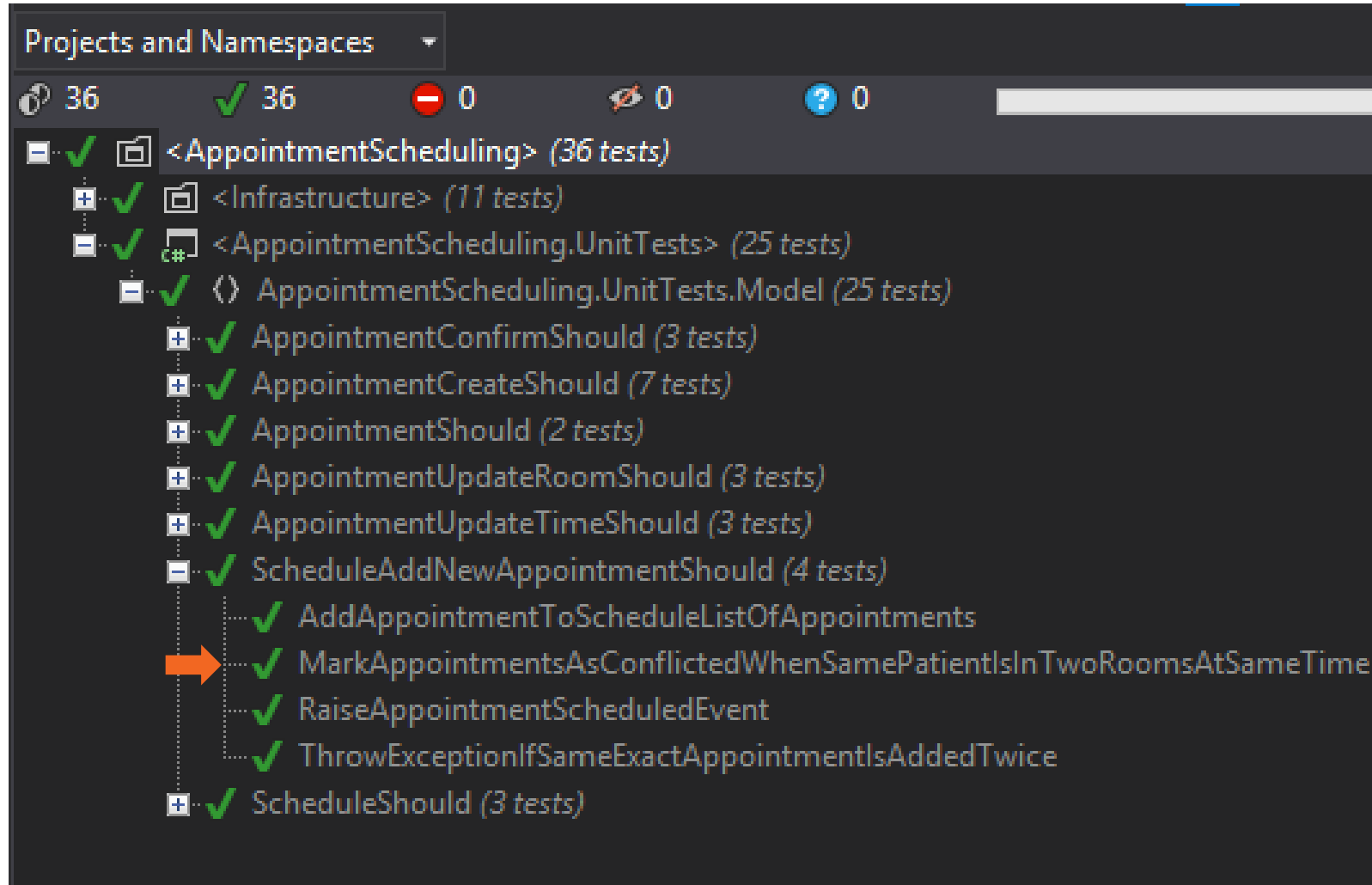
Confirm method on Appointment class

Appointment class

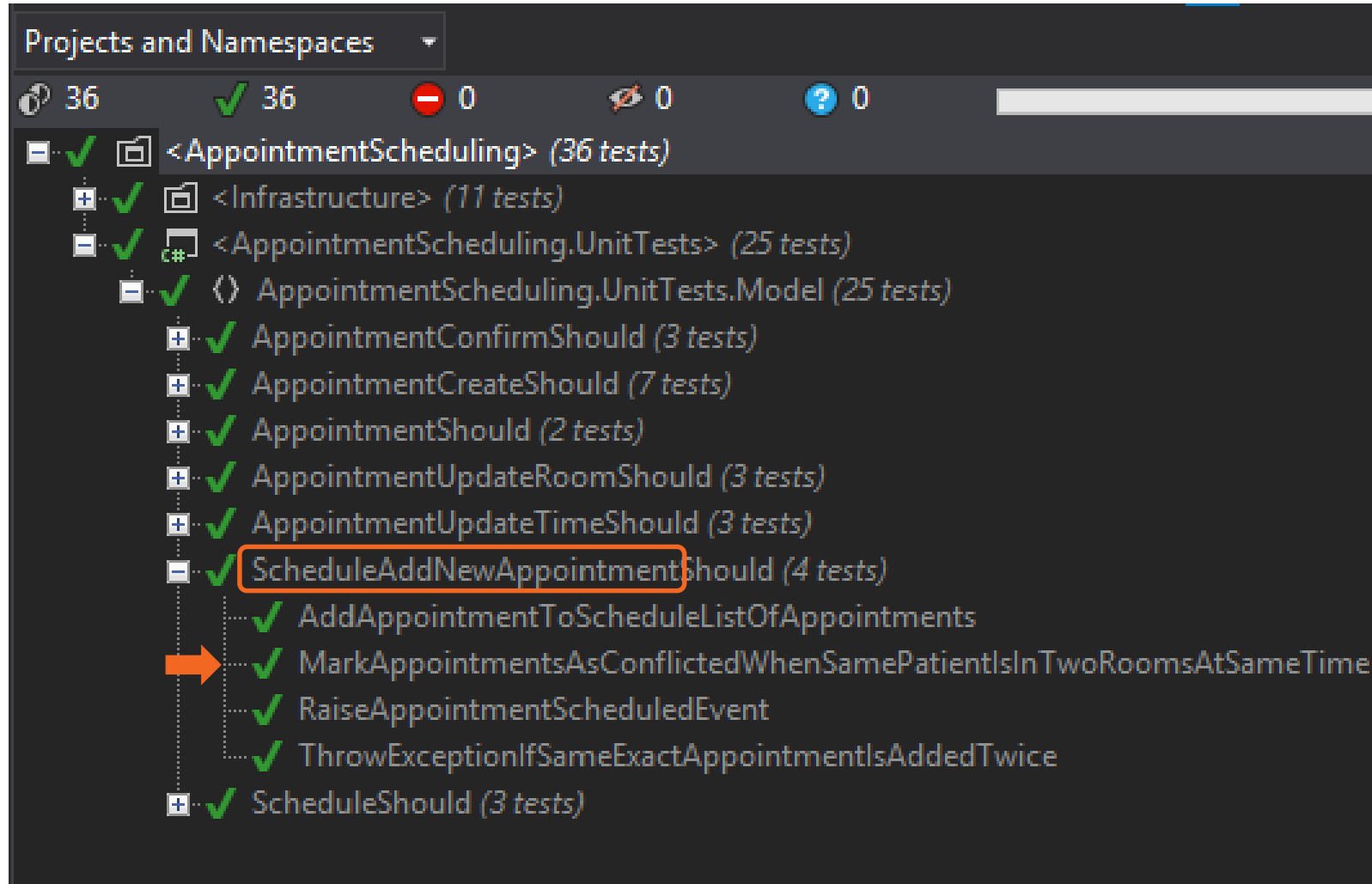
Test Naming



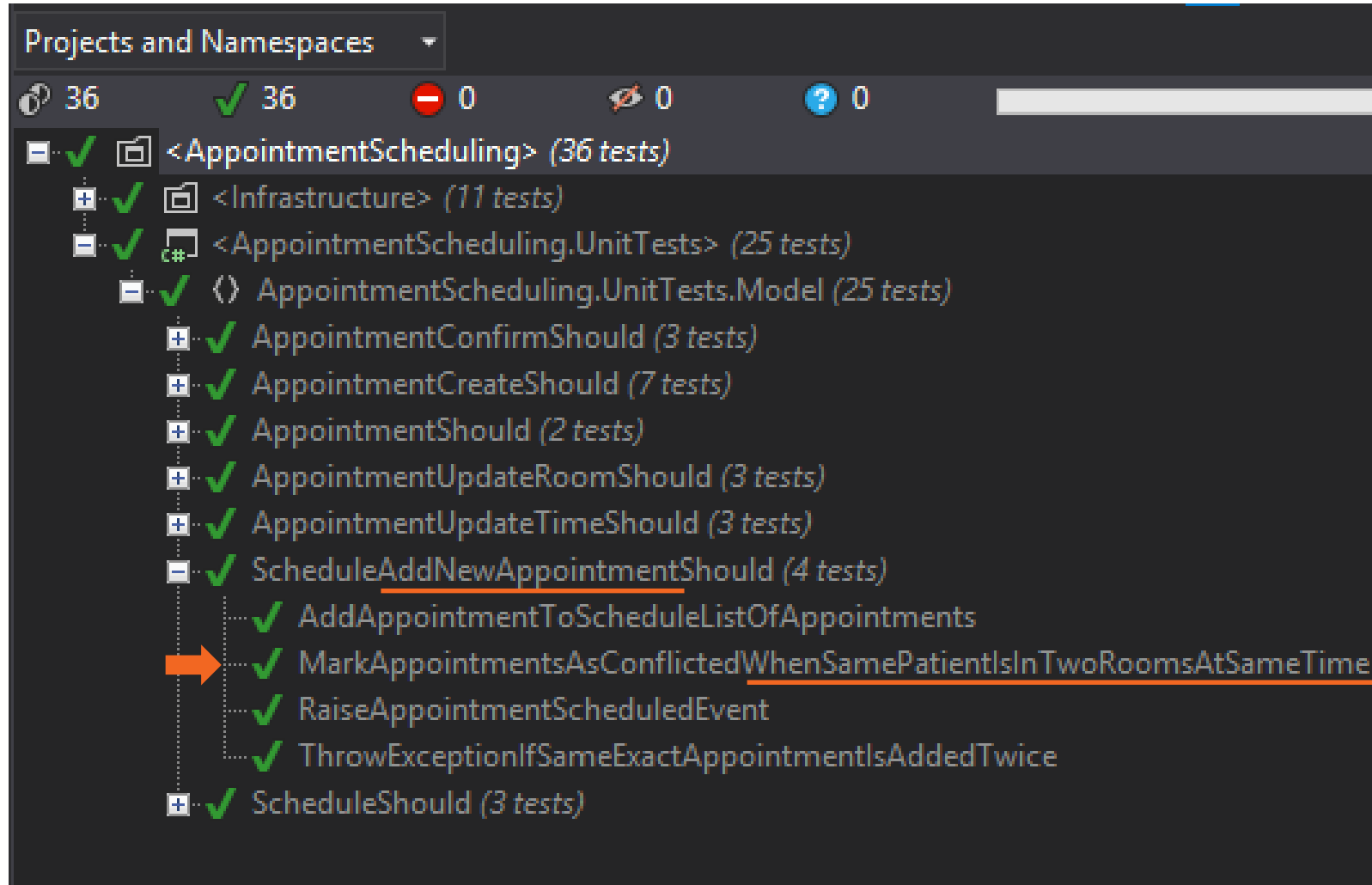
Test Naming



Test Naming

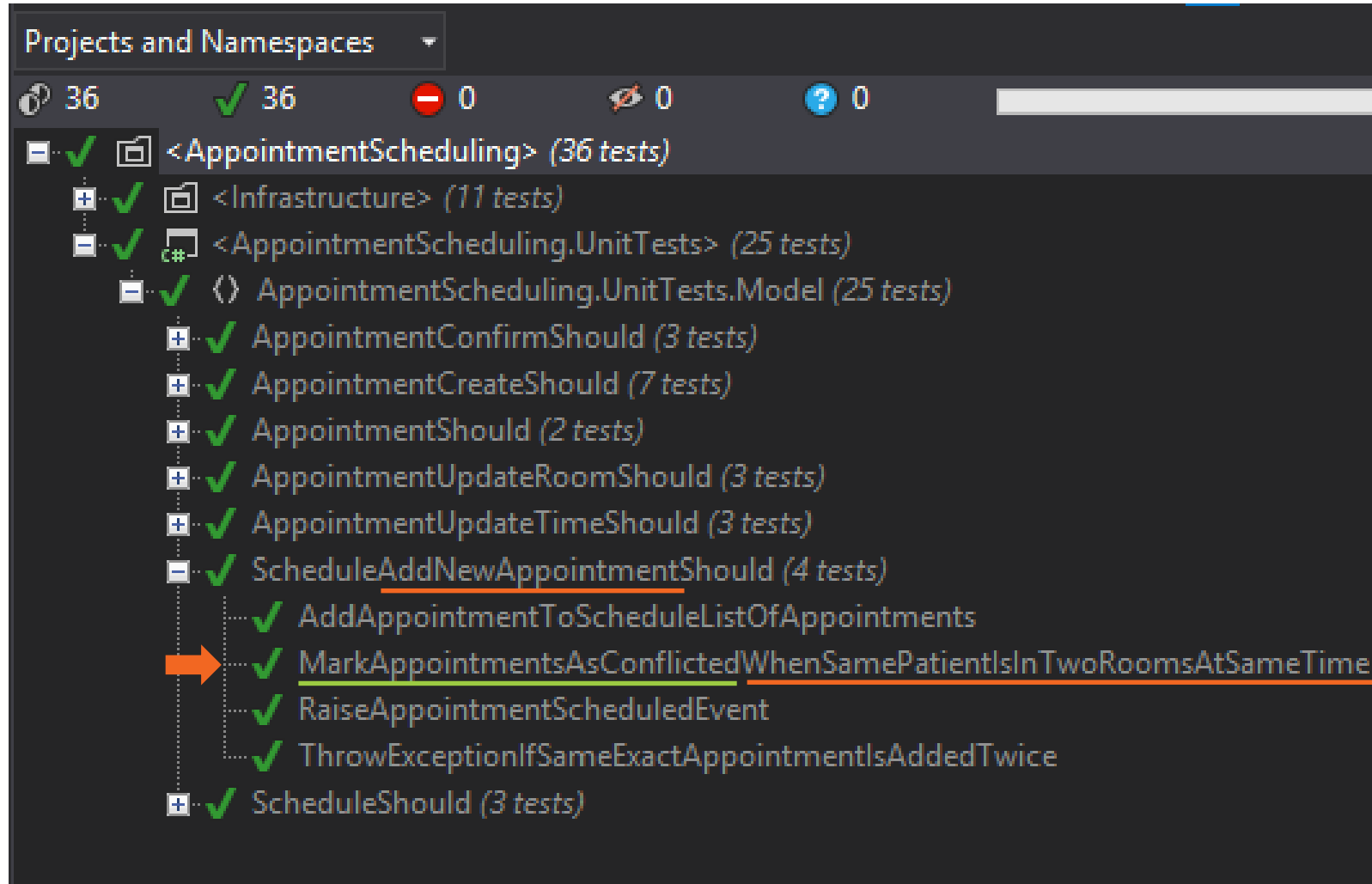


Test Naming



Context

Test Naming



Context

Specification

Summary

Summary

- Overview of *Vet Manager* sample application
 - bit.ly/PS-DDD

Summary

- Overview of *Vet Manager* sample application
 - bit.ly/PS-DDD
- Test naming and organization

Summary

- Overview of *Vet Manager* sample application
 - bit.ly/PS-DDD
- Test naming and organization
 - Avoid generic, unfocused test fixtures like “ScheduleTests”

Summary

- Overview of *Vet Manager* sample application
 - bit.ly/PS-DDD
- Test naming and organization
 - Avoid generic, unfocused test fixtures like “ScheduleTests”
 - “Should” suffix naming (e.g. “ScheduleShould”)
 - Promotes more cohesive statements about a piece of code
 - Tends to be code/implementation focused

Summary

- Overview of *Vet Manager* sample application
 - bit.ly/PS-DDD
- Test naming and organization
 - Avoid generic, unfocused test fixtures like “ScheduleTests”
 - “Should” suffix naming (e.g. “ScheduleShould”)
 - Promotes more cohesive statements about a piece of code
 - Tends to be code/implementation focused
 - Context/Specification – avoids coupling to specific implementations