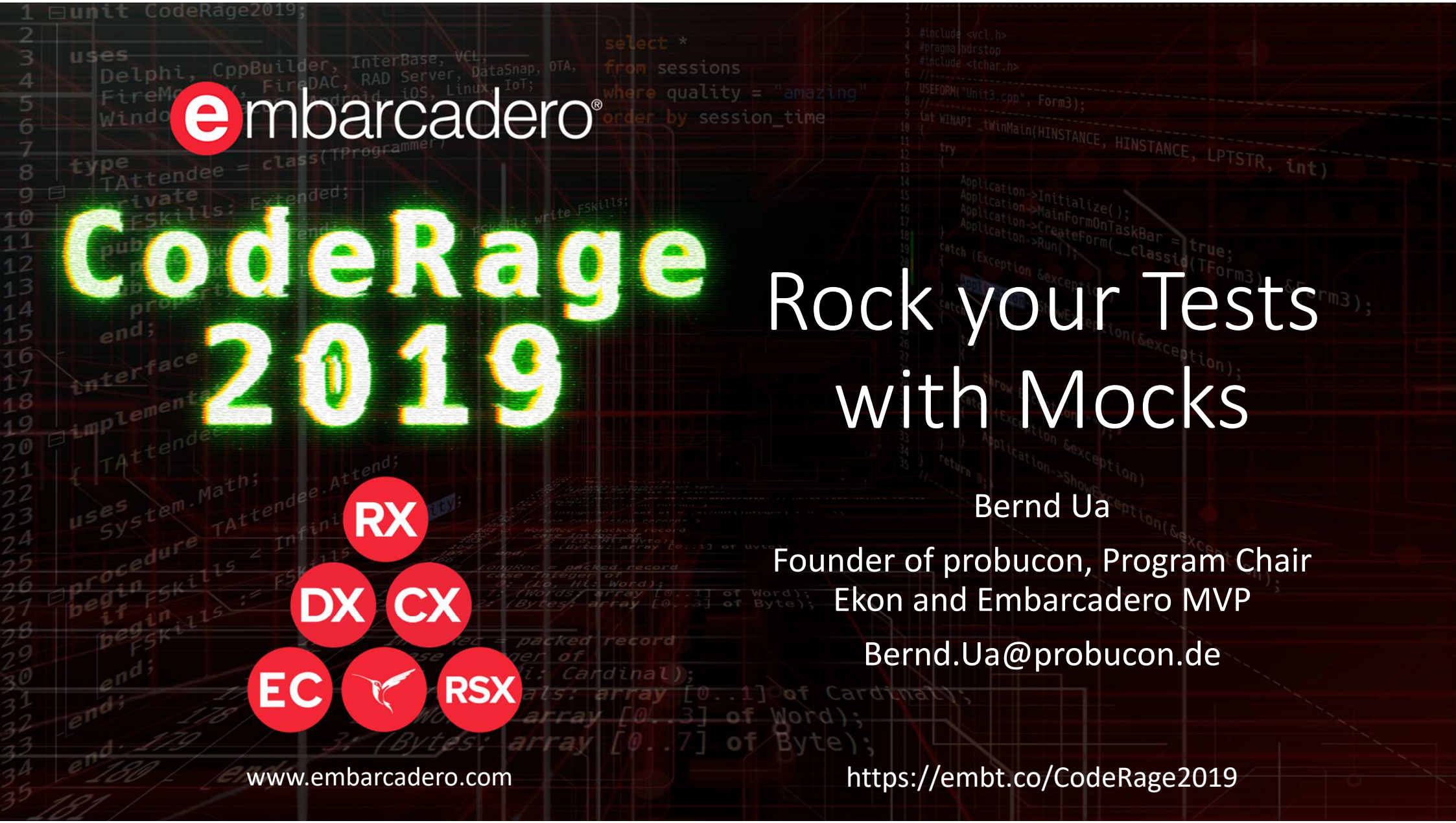


Rock your Tests with Mocks

Bernd Ua

Founder of probucon, Program Chair Ekon and Embarcadero MVP

Bernd.Ua@probucon.de

https://embt.co/CodeRage2019

Overview

- Spring4D is a great open source framework to leverage the power of Delphi. In this session we will take a look a the mocking framework contained in Spring4D. We will see how easy it is, to get started with mocking and mock out interfaces your classes under tests are using.

Bernd Ua

What is mocking ?“

- If you are mocking out something you replace a productive implementation with a special implementation for testing, the so called mock
- You can write Mocks manually or use ones that have been automatically created by frameworks
- The difference between mocks and stubs or dummies is some extra code to log and check calls made to the mock

What a Unittest should not do

- a Unit-Test should better not use
  - Databases
  - Network access
  - File Systems
  - External Ressources
  - A configured Environment
- If your unit test uses one of the above it is more likely an integration test

Using Mocks in your Tests

- You can replace concrete access to databases/resources with interfaces and mocks
- The number of classes that need to be tested with resources is minimized, thereby increasing test performance
- Mocks can also be used to test scenarios that are difficult to represent in reality

What characterizes a good unit test?

- It is clearly arranged
  - Probably follows AAA-scheme Arrange-Act-Assert
- It is executed quickly
- It helps to locate a problem quickly

Automatic Mocking

- Advantages
  - Setup of your Test and the definition of expectations can be found directly within the test and not somewhere else
  - Less work and less errors than manual programming
- Disadvantages
  - There might be limitations (IInvokable, Interfaces etc)

Mocking Frameworks for Delphi

- For Delphi before 2009/XE2 there is just PascalMock
  - PascalMock can check Calls and their Order and Parameters using Variant
- For actual Delphi Versions there are
  - DSharp Mocks
  - https://bitbucket.org/sglienke/dsharp/overview
  - DelphiMock
  - https://github.com/VSoftTechnologies/Delphi-Mocks
  - Spring4D Mocks
  - https://bitbucket.org/sglienke/spring4d/src/master/

A short Spring4D history

- Open Source Library für Delphi
- Lizenz Apache License 2.0
- Started in 2010 on Google Code and changed later on to BitBucket
- Actually Stefan Glienke is maintaining the framework and pushing it forward
- Plenty of stuff in it
  - Dependency injection container
  - generic interfaces for lists and collections
  - Multicast events, Nullable types etc
- Since Version 1.2 containing a Mocking Framework

Spring 4D Links

- GIT
- https://bitbucket.org/sglienke/spring4d.git
- WIKI
- https://bitbucket.org/sglienke/spring4d/wiki/Home
- API Documentation at DevJet Software
- http://www.devjetsoftware.com/docs/spring4d/

Simple installation

- Download from BitBucket
- Unzip to destination directory
- Run Build.exe and select Delphi IDEs for install
- If you select Update Registry your Library Path is extended for you

Mocking Framework in Spring 1.2

- Is realized with the help of generic records (Mock<T>) and Interfaces
- The mocks uses fluent interface technique for easy setup
- Simply include unit Spring.Mocking.pas in your uses clause
- Use the generic record Mock<T> for the interface you want to mock out
- Use Mock<T>.Setup to define the behaviour
- You can mock out Interfaces compiled with {$M+} or inheriting IInvokable instead of IInterface
- If you mock out Classes only virtual methods are being mocked out

Setting up a mock in the test

- Use the Setup-Interface to define the behavior
  - .Setup.Returns for functions and results
  - .Setup.Executes for procedures
  - .Setup.Raises to throw exceptions
- Use „When“ after Setup to define the conditions for the behavior
- In Strict-Mode you can only call defined methods
- Use generic Record TArg (or global var Arg) to express special parameter values (like IsAny, IsIn , IsNil etc)

Controlling Mock Behavior

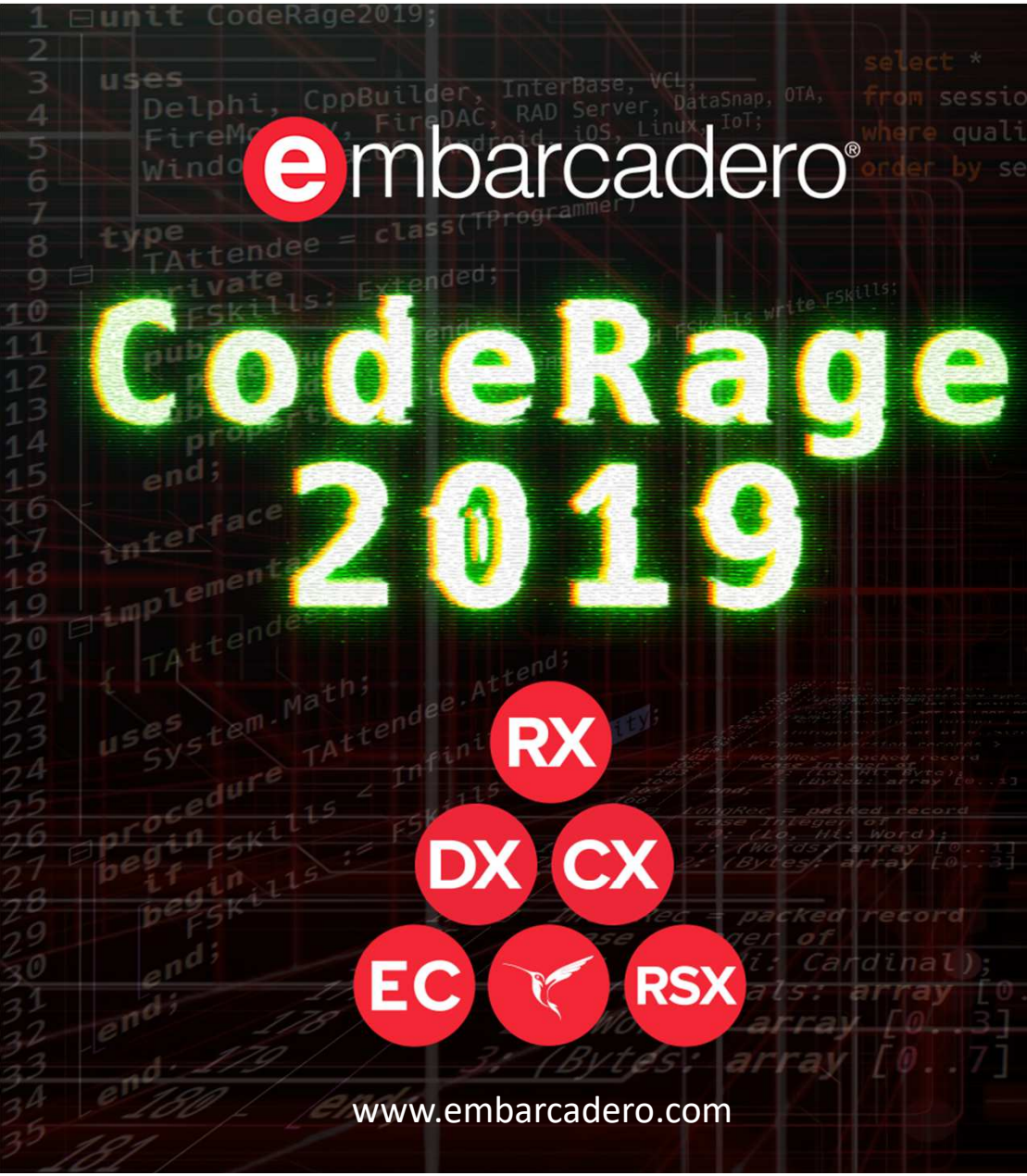
- TMockBehavior.Strict
  - Mock throws an Exception if any Method is called that has not been setup before
- TMockBehavior.Dynamic
  - Mock accepts every call
  - Returns default values for every call

Checking Results

- Most other mocking frameworks use a verify method to compare expectations with issued calls
- Spring4D uses a generic method Received for this purpose
- Received optionally accepts a Times argument to configure the number of expected calls (Once, Never, AtLeastOnce, etc)
- If the calls in Received differ from the actual calls an exception is raised

Checking the Order of Calls

- Spring4D ignores the order if you check or define behavior via Received or Setup
- If you want to check the order of calls, you have to use a MockSequence Record
- Define a local variable of type MockSequence and use it as a parameter for Setup
- If you are done with your test execution check the whole sequence with a call to MockSequence.Completed



More Information

- Landing page with Links to articles, slides and source for this video
- https://probucon.de/blog/2019/12/coderage-2019/
- Bernd Ua
- Bernd.Ua@probucon.de