How to live happily ever after with integration tests

FLUENT INTEGRATION TESTS

Disclaimer

Dear viewer,

Please be advised that opinions expressed in this presentation and on the following slides as well as comments made during the discussion aftet the presentation are solely those of the presenter and not necessarily those of Lumesse. Lumesse does not guarantee the accuracy or reliability of the information provided herein.

Thank you!

Agenda

- Overview
- The three challenges
- Fluent interface
- Page objects
- Reusable scenarios
- Reusable special functionality
- Clean resources

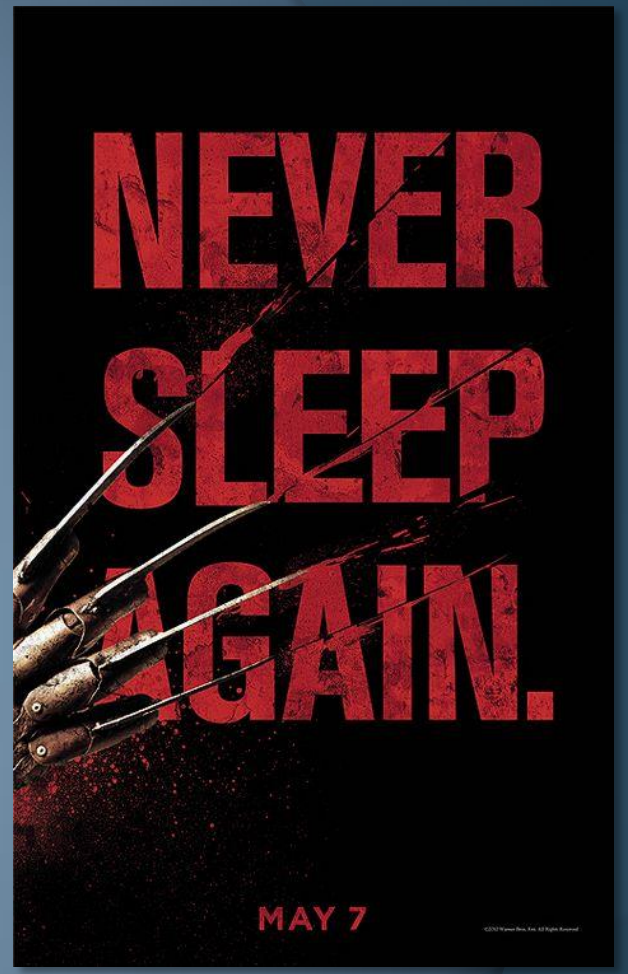
Overview

- Different types of tests
  - Manual tests
  - Unit tests
  - Performance tests
  - Acceptance tests
- Integration vs Unit tests
- Test code is important – it saves you

- Let’s get serious...

The three challenges

- Writing integration tests is hard
- Reading integration tests is even harder
- Maintaining integration tests is a nightmare



Fluent interface

In software engineering, a fluent interface (as first coined by Eric Evans and Martin Fowler) is an implementation of an object oriented API that aims to provide for more readable code.

A fluent interface is normally implemented by using method cascading (concretely method chaining) to relay the instruction context of a subsequent call (but a fluent interface entails more than just method chaining). Generally, the context is

- defined through the return value of a called method
- self-referential, where the new context is equivalent to the last context
- terminated through the return of a void context.

Page objects

Page Object is a Design Pattern which has become popular in test automation for enhancing test maintenance and reducing code duplication. A page object is an object-oriented class that serves as an interface to a page of your AUT. The tests then use the methods of this page object class whenever they need to interact with that page of the UI. The benefit is that if the UI changes for the page, the tests themselves don’t need to change, only the code within the page object needs to change. Subsequently all changes to support that new UI are located in one place.

Reusable scenarios

Code reuse, also called software reuse, is the use of existing software, or software knowledge, to build new software, following the reusability principles.

Reusable special functionality

In computer science and software engineering, reusability is the likelihood that a segment of source code can be used again to add new functionalities with *slight* or **no** modification

Clean resources

- Clean database with automatic *migrations* and injection of test data
- Initialization of other resources (files, folders..)


?
Q
&
A
!

Thank you!

http://padcom13.blogspot.com

Hope you didn’t feel like falling asleep ☺