DEV MAGIC FAKE

JUST ONE LINE OF CODE

Understanding Dev Magic Fake

By: M.Radwan

DEV MAGIC FAKE

TABLE OF CONTENTS

1. Abstract

Dev Magic Fake is a development framework, physically it's a .Net Class library that provides Fakeable operations for all needed activities for developers in which allow them to run the system features as if they complete its programming, so that the feature can be tested by QC for functional requirements and can be verified by the client against the acceptance criteria and if it meets its requirements or not.

After the feature get pass the QC test, and pass the client's verification against the acceptance criteria, the real development will start using TDD approach by creating all unit tests for the approved feature, of course all unit test will pass because of Dev Magic Fake, so we will start refactoring the code by replacing all fakeable code with the real one, it's better that we have integration test to verify that the test pass because of the real implementation not because Dev Magic Fake framework.

2. Introduction

Throughout my software development experience I have faced many problems regarding software requirements, the problem that the customer always said that, the produced software always not as I expected!! This always happen

As developers we didn't get the whole picture of the application when we start developing its feature neither the customer!, we and the customer will realize what the software should behave and perform throughout the developing of the application, the customer start to know what exactly he needs as soon as we finish parts of the application, and of course the developers start to understanding the application better with the feedback of the customer and how it should behave and perform, this make us be able to deliver what the business and customer really needs, this is absolutely what Agile is, But how to implement real agile development practices? This is the question that we faced all the time.

One of the main drivers to create this framework is the missed communication between software stockholders like software managers, business analyst and software developers, when we starting asking our software stockholders about the requirements, we tell them that we don't get the whole picture of the business, they try to model the Database Schema that describe the business, some of them start talking and use expressions that understand only by the narrator, they assume that these expressions are obvious and clear, they didn't know that everyone understand these expressions in a different way, they didn't get our point that we need to understand the business not the Schema, if we really understand the business we will implement the right feature, it doesn't matter what is the schema, the only matter is what is the business functionality, everyone in the stockholders start describe these functionality by different approach according to his experience and culture, this description of curse has many assumptions and assuming that lead to different understanding and different implementation of the business functionality from what actually the customer needs

The existing approaches for implanting Agile best practices are not bad but not as we really need, we spending more time developing some real parts of the application without real understand of the application and after we realize and understand the application, we will find that most of our development output will be throw away, even the TDD (Test Driven Development) or BDD (Behavior Driven Development) doesn't

work as well, because as we mention before, in real world the customer really didn't know what exactly he needs until seeing the application running and interact with its interface and features, and then he starts realize how the application should perform and work and start asking for modification and enhancement because the current system is not what he is looking for

Dev Magic Fake not just a code framework, Dev Magic Fake is an idea and software delivery approach that make us focus on understanding, realizing, verifying and maybe even testing the application without completing the underline layers, this approach and code working by developing only the interface needed and ignoring the underline layers until the whole feature or user story is completely understood, verified and tested

Dev Magic Fake not a sliver bullet, it will also require some maintainability to remove the faking parts of the code and replace it with the real one, so if we don't have good code management it will be a mess, in the future work I am looking for how to eliminate or reduce the maintainability of the faking code by using DI (Dependency Injection) and IoC (Inversion of Control), but right now we focus on how to provide faking framework so we can know what are the behaviors and response of the application without any focus on the code of the underline layers until complete, test and approve the application features

Consideration:

It's better that we create integration test for all faked features which will show us if there is a faking code or not, because the unit test will pass anyway and we will not be able to know if it passing because of Dev Magic Fake or real implementation , but implementing the integration test will expose any faking code

3. Dev Magic Fake Feature List

- Easy implementation and usage without any prerequisite, we only need 2 steps to work with Dev Magic Fake:
  - Reference the Dev Magic Fake assembly
  - Create an integer Id property for all classes that needs to be faked
- Save any simple instance (has only primitive data types) of any class **(with just one line of code)**
- Retrieve any instance of any type by Id **(with just one line of code)**
- Save any complex instance (container of other instances) and retrieve its nested instance from Dev Magic Fake to link the container or the main instance to the nested instances **(with just one line of code)**
- Save any complex instance and save its nested instances to Dev Magic Fake **(with just one line of code)**
- Save any complex instance (container of other instances) that has collection and retrieve it's items from Dev Magic Fake to link the container or the main instance to these nested instances **(with just one line of code)**
- Save any complex instance that has a collection and save the collection items also to the Dev Magic Fake **(with just one line of code)**
- Get all saved instances of any class **(with just one line of code)**
- Generate instances for all classes in an assembly and generate it's data **(with just one line of code)**
- Generate instances for all classes that saved in the Dev Magic Fake and generate it's data**(with just one line of code)**
- Use the default data generation mechanism **(without any configuration)**
- Control the data generation mechanism using range of values like using specific class and property or for specific property for any class. **(Configured using app setting)**
- Control the data generation mechanism using data types like string and integer should be generated with specific length and characters or numbers **(Configured using app setting)**
- Control the data generation that just included in specific assembly **(Configured using app setting)**
- Control the data generation that just included in specific namespace **(Configured using app setting)**

- Control the data generation that just marked with Fakeable attribute only **(Configured using app setting)**
- Control the data generation to eliminate any type marked with NotFakeable attribute **(Configured using app setting)**
- Control the data generation for the depth of the object graph **(Configured using app setting)**
- Create any instance of any type and generate its data **(with just one line of code)**
- Create list of instance of any type and generate its data **(with just one line of code)**
- Provide permanent saved data for all saved instances through the ability to save the whole object graph to HDD and retrieve it to its original state later
- Easily query Dev Magic Fake using LINQ

4. Dev Magic Fake Road Map and Upcoming Features

- Support data generation based on data annotation
- Support data generation based on regular expression
- Support data generation based on database tables
- Support many to many types in an effective and better way
- Support custom collection
- Support random enumeration
- Using configuration section designer instead of app settings
- Using UI for configuration
- Eliminate or reduce the maintainability of the faking code by using DI (Dependency Injection) and IoC (Inversion of Control).

5. Summary

Dev Magic Fake is a faking framework that gives us the ability to focus on how to complete, verify and test the application behaviors and response without focus on coding or developing the underline layers until the application features finished, tested and approved

So the main goals of the Dev Magic Fake are to give us the ability to:

- Focus on what rather than focus on how
- Provide real implementation of abstraction in software development
- Implementing “Develop By Feature” approach by Agile methodology
- Complete the feature without coding the underline layers
- Give the ability to creating a passed successful unit testing to test the behavior and response of the application without completing the underline layers
- Give the ability to creating a passed successful UI test to test the behavior and response of the UI without complete the underline layers
- Create faking code with no effort
- Create faking code in no time
- Permanent data storage
- Minimum effort for replacing the faking code with the real one

6. About the Author

M.Radwan is a Lead Architect, Configuration Manager and Build Engineer, with more than 9 years of software architecture, design, development, and management experience, specializing in Microsoft technologies and Agile methodology. Consulting and coaching clients in Egypt, KSA, Libya and Kuwait.

M.Radwan Focus on :C# / .NET, ASP.NET, MVC, JQuery, Test-Driven Development, MS Build, TFS, MS Team build, Application Architectures, Agile, Process Automation And Improvement, Configuration Management, Automation all tasks related to software development activities, this include but not limited for Development, Build, Configuration, Deployment, Test, etc.

M.Radwan is M.Sc. of computer sciences and information technology in Agile Methodology

M.Radwan holds number of Microsoft certifications including MCT, MCPD, MCITP in EPM, MCTS (7), MCSD, MCAD and CIW

M.Radwan believes that we have to learn from our mistakes and this what we called experience and the only way for productivity is to automate this experience.

http://social.msdn.microsoft.com/profile/M.Radwan

http://www.codeplex.com/site/users/view/mradwan

http://www.linkedin.com/in/mohamedahmedradwan

http://twitter.com/#!/mradwan06

http://stackoverflow.com/users/386323/m-radwan

http://www.youtube.com/user/MRadwanMSF

http://www.facebook.com/M.Radwan.TFS

7. Feedback

I would love to hear about what I do well and how I can improve, if there is anything you don't like or have an idea or enhancement, please email me on mradwan.automationplanet@gmail.com or contribute to the CodePlex discussions page for Dev Magic Fake, if you would like to contribute to this project as a developer, technical writer, or any other role please let me know

Thanks

M.Radwan