**Instituto Superior Engenharia de Lisboa**

Relatório

Série 3

Ambientes Virtuais de Execução

Licenciatura em Engenharia Informática e Computadores

Semestre de Inverno 2016/2017

Lisboa, 15 de Janeiro de 2017

**Docente:**

Fernando Miguel Carvalho

## Introdução

Na sequência do trabalho desenvolvido nas séries anteriores, foi pedido para complementar a *API (Application Programming Interface)* de forma a ser possível indicar quais os campos a serem ignorados pela *API* bem como a utilizaçao de funções para a inicialização dos mesmos. Também foi pedido para fazer várias verificações sobre o tipo de dados utilizado de forma a fornecer robustez à aplicação.

## Implementação

A *API* foi desenvolvida tendo como base as séries anteriores. Assim sendo adicionou-se um novo projecto à solução denominado AutoFixture. Manteve-se o código separado em duas classes, **AutoFixture** e **Fixture**.

Em AutoFixture adicionou-se um método com o nome Ignore que retorna a interface *IFixture .* Este método permite à interface, via reflexão, fazer a chamada ao método ignore, e por isso a assinatura tem que ser correspondente ao método que se vai implementar. Assim sendo o método ignore definido não recebe -nao era suposto ter ali um object ou qq merda?

Na classe Fixture definiram-se novos métodos ou alteraram-se os existentes de forma a suportar novos requisitos. Segue uma breve explicação da forma como cada um dos requisitos foi realizado

##### **Não inicialização dos campos­­­ pelo nome**

Definiu-se um método ignore que adiciona a uma lista o o nome da propriedade. Esta lista vai ser consultada aquando a realização da atribuição do valor quer a propriedades quer a campos, e em caso de o nome existir na lista, a atribuição não é realizada.

#### Não inicialização através de anotações

Implementou-se o método Ignore<A> where A : Attribute. Este método procura as propriedades da classe e verifica o atributo. No caso do atributo existir para essa propriedade, ou seja, a propriedade está anotada com esse tipo, entao adiciona à mesma lista do método anterior o nome da propriedade.

#### utilização de Funcções na inicialização dos campos ou propriedades

Implementou-se um método member que recebe um nome e uma função que retorna <R>.   
De forma a fornecer robustez à aplicação é feita uma verificação de que o tipo de retorno da função utilizada para inicializar o campo ou propriedade é compatível com o tipo esperado.

De seguida é adicionado ao dicionário o nome da propriedade e a função. Desta forma ao inicializar a propriedade é retirada a função, chamado o DynamicInvoke e afectada a propriedade ou o campo.

##### Implementação para propriedades do tipo []R ou IEnumerable<R>

Foi alterada a assinatura do método T[] Fill para IEnumerable<T> Fill. No método New faz-se a verificação para ver se a klass a instanciar é do tipo ICollection em vez de ver se é do tipo array, desta forma suporta todo o tipo de Collections.

## Conclusão

Este trabalho serviu para consolidar e aplicar os conceitos dados nas aulas (reflexão, lambdas,

delegates, custom attributes) . Foi interessante desenvolver uma API mais fluida e perceber o quanto é importante o uso de Interfaces e genéricos para ter um programa modular e adaptável.