eCOM avec le canevas Spring .NET
eCOM avec le canevas Spring .NET

Rédaction initiale: Olivier Ducas

Contributeurs:

L'objectif de cette extension est de réaliser une implémentation de l'application eCOM au moyen du canevas Spring .NET et du langage C#.

---



---

## *Spring et Spring .NET : Qu'est ce que c'est ?*

Spring est un canevas de développement d'applications Java d'entreprise (JavaEE) qui "simplifie" notanment leurs tests. Une présentation rapide de Spring est disponible à la section consacrée à eCom avec Spring (Java).

Spring .Net est le portage du canevas Spring dans l'environnement .Net en utilisant la technologie Microsoft .NET. Le coeur de Spring.Net (et Spring) est la mise en oeuvre du design pattern « Inversion Of Control (IOC) » par « Injection de Dépendance » décrit par Martin Fowler. Le canevas Spring.Net prend en charge la création et la mise en relation d'objets par l'intermédiaire d'un fichier de configuration. Ce fichier de configuration décrit les objets à fabriquer et les relations de dépendances entre ces objets.

Outre cette espèce de super fabrique d'objets, cette déclinaison de la version .NET permet principalement :

- le developpement des applications WEB avec ASP.NET
- la prise en chage de .Net Remoting, des composants services d'entreprises (COM+) et des services Web créés en .NET
- lutilisation d'ADO.NET et/ou des canevas comme NHibernate pour les accès aux données
- la gestion des transactions declaratives
- la programmation par aspect (AOP) basé principalement sur l'instrumentation de code (dynamic proxy in java, instrumentation de code)
- la prise en charge des ressources (fichiers, images, etc.), la validation (pages web ou objets metiers) sous forme de canevas, le multithread et les tests unitaires

Motivations

Fournir un canevas leger, modulaire et non intrusive pour créer des applications d'entreprises. Spring.NET peut être intégré dans toutes les couches de l'application (présentation, services, domaines, données) et contribue à l'améliotation de la productivité, de la qualité et de la Performance des codes. Sa modularité permet de n'utiliser, par exemple, que le conteneur d'injection de dépendances pour l'application tout en restant sur ADO.NET pour acceder aux données. Principes

Spring.NET est un conteneur dit « léger ». C'est une infrastructure similaire à un serveur d'application J2EE. Son principal avantage, par rapport aux serveurs d'application, est que les classes n'ont pas besoin d'implémenter une quelconque interface pour être prises en charge par le canevas Spring.NET. En effet, le developpeur crée des objets .NET ordinaires (PONOs : Plain Old .NET Object) et le conteneur prend en charge la création et la gestion de dépendances entre les objets :

- les composants ne créent pas eux-même leurs dépendances avec le reste du système. Leurs dépendances sont injectées par leur environnement d'exécution
- les composants ne manipulent pas directement des implémentations, ils manipulent des contrats
- couplage faible (Facile de remplacer des composants par d'autres, Maintenance simplifiée)
- implémentations indépendantes des contextes d'utilisation (Composants réutilisables)

L'injection de dépendance, au niveau du canevas Spring.NET, se fait par fichier de Configuration sous format XML et peut se décliner sous deux formes qui sont :

- l'injection par propriétés et,
- l'injection par constructeur

Exemple de PONOs : Account

```
using System;namespace Ecom.Springnet.Domains { public class Account { private String _iban;
private String _owner; private double _balance; public Account() { } public String Iban { get {
return _iban; } set { _iban = value; } } public String Owner { get { return _owner; } set {
_owner = value; } } public double Balance { get { return _balance; } set { _balance = value; } }
} }
```

```
<?xml version="1.0" encoding="UTF-8"?><configuration> <configSections> <sectionGroup
name="spring"> <section name="context" type="Spring.Context.Support.ContextHandler,
Spring.Core"/> <section name="objects" type="Spring.Context.Support.DefaultSectionHandler,
Spring.Core"/> <section name="parsers" type="Spring.Context.Support.ConfigParsersSectionHandler,
Spring.Core"/> </sectionGroup> </configSections> <spring> <context> <resource
uri="config://spring/objects"/> </context> <objects xmlns="http://www.springframework.net">
<object id="Account" type="Ecom.Springnet.Domains.Account, Domains" singleton="false"/>
</objects> </spring> </configuration>
```

Exemple d'injection de dépendances par constructeur :

Exemple de PONOs : Account

```
using System;namespace Ecom.Springnet.Domains { public class Account { private String _iban;
private String _owner; private double _balance; public Account() { } public Account(String iban,
String owner, double balance) { this._iban = iban; this._owner = owner; this._balance = balance;
} public String Iban { get { return _iban; } set { _iban = value; } } public String Owner { get
{ return _owner; } set { _owner = value; } } public double Balance { get { return _balance; }
set { _balance = value; } } } }
```

Les valeurs sont injectées par Spring dans le constructeur à la construction de l'objet Account.

```
<?xml version="1.0" encoding="UTF-8"?><configuration> <configSections> <sectionGroup
name="spring"> <section name="context" type="Spring.Context.Support.ContextHandler,
Spring.Core"/> <section name="objects" type="Spring.Context.Support.DefaultSectionHandler,
Spring.Core"/> <section name="parsers" type="Spring.Context.Support.ConfigParsersSectionHandler,
Spring.Core"/> </sectionGroup> </configSections> <spring> <context> <resource
uri="config://spring/objects"/> </context> <objects xmlns="http://www.springframework.net">
<object id="Account" type="Ecom.Springnet.Domains.Account, Domains" singleton="false"/>
<constructor-arg type="String" value="66 666 66666 66"/> <constructor-arg type="String"
```

```
"Olivier"/> <constructor-arg type="double" value="1000"/> </object> </objects> </spring>
</configuration>
```

Exemple d'injection de dépendances par propriétés :

Exemple de PONOs : Account

```
using System;namespace Ecom.Springnet.Domains { public class Account { private String _iban;
private String _owner; private double _balance; public Account() { } public String Iban { get {
return _iban; } set { _iban = value; } } public String Owner { get { return _owner; } set {
_owner = value; } } public double Balance { get { return _balance; } set { _balance = value; } }
} }
```

Les valeurs sont injectées par Spring dans les propriétés à la construction de l'objet Account.

```
<?xml version="1.0" encoding="UTF-8"?><configuration> <configSections> <sectionGroup
name="spring"> <section name="context" type="Spring.Context.Support.ContextHandler,
Spring.Core"/> <section name="objects" type="Spring.Context.Support.DefaultSectionHandler,
Spring.Core"/> <section name="parsers" type="Spring.Context.Support.ConfigParsersSectionHandler,
Spring.Core"/> </sectionGroup> </configSections> <spring> <context> <resource
uri="config://spring/objects"/> </context> <objects xmlns="http://www.springframework.net">
<object id="Account" type="Ecom.Springnet.Domains.Account, Domains" singleton="false"/>
<property name="Iban" value="66 666 66666 66"/> <property name="Owner" value="Olivier"/>
<property name="Balance" value="1000"/> </object> </objects> </spring> </configuration>
```

Modules

Les modules utiles à la réalisation d'eCOM NHibernate Web Extensions (ASP.NET AJAX) Logging (Log4NET) Interop (IIOP) Messaging (NMS & Tibco EMS) Scheduling (Quartz) Transaction WebFlow Web Services Messaging

## *Réalisation*

Installation

Récupérez la dernière release stable du canevas Spring .NET depuis http://www.springframework.net/download.html

(la dernière version des sources peut être récupérée depuis le dépôt CVS de SourceForge).

Dézippez le fichier dans le répertoire %SPRINGDOTNET_HOME%

Récupérez et installez également les dernières releases stables de NAnt et NUnit.

L'installation ...

Le répertoire %SPRINGDOTNET_HOME%examplesSpring contient un certain nombre d'exemples bien connus conçus avec Spring : ...

Récupérez l'embryon de projet au moyen d'un checkout de la forge eCOM

La commande est la suivante : **svn checkout svn://svn.forge.objectweb.org/svnroot/ecom/ecom-springdotnet** Les beans d'eCOM

- Account

- ProductStore
- Product
- Cart
- EuroConvertor
- ...

## *Bibliographie*


- Le site de Spring .NET Framework http://www.springframework.net
- La documentation de référence de Spring .NET http://www.springframework.net/documentation.html
- http://fr.wikipedia.org/wiki/Spring_framework
- Introduction à Spring http://ego.developpez.com/spring/
- Quelques tutoriels sur Spring et Spring.Net http://tahe.developpez.com/
    - http://tahe.developpez.com/dotnet/springioc/
    - http://tahe.developpez.com/dotnet/m2vc-aspnet/
    - http://springframework.net/documentation.html#Presentations
- Inversion de Controle
    - http://www.dotnetguru.org/articles/dossiers/ioc/Fowler/IoC.htm
    - http://msdn.microsoft.com/msdnmag/issues/05/09/DesignPatterns/


eCOM avec le canevas Spring .NET (fr)
Creator: xwiki:XWiki.ducas  Date: 2007/12/27 08:16
Last Author: xwiki:XWiki.kjosey_wales  Date: 2008/07/15 19:50