Seasar.NET で DI と AOP を使ってみよう

杉本 和也 （Seasar.NETリーダ ）

Seasar.NET の主なプロダクト

- S2Container.NET
  - AOP をサポートしたDIコンテナ
- S2Dao.NET
  - O/R マッピングフレームワーク

Seasar.NET プロジェクト

- .NET を対象としたオープンソースソフトウェアを開発するプロジェクト (Seasar プロジェクト所属)

Seasarファウンデーション
（国内オープンソースソフトウェア開発コミュニティの運営支援）

OSS 開発コミュニティ

Seasar プロジェクト

im-OSSC

The Ashikunep Kotan

escafe.org

OSS 開発 TOP プロジェクト

Seasar.NET プロジェクト

OSS 開発プロジェクト

S2Container.NET

S2Dao.NET

DI を使わない場合

自分で EmployeeDao を new (インスタンス化) して使う

```
public class EmployeeLogic
{
    public Employee GetEmployeeByEmpID(int empID)
    {
        IEmployeeDao empDao = new EmployeeDao();
        Employee emp = empDao.GetByEmpID(empID);
        return emp;
    }
}
```

DI を使う場合

DI コンテナが new (インスタンス化) してフィールドにセットしてくれる

```
public class EmployeeLogic
{
    protected IEmployeeDao empDao;

    public Employee GetEmployeeByEmpID(int empID)
    {
        Employee emp = empDao.GetByEmpID(empID);
        return emp;
    }
}
```

DI（Dependency Injection）：依存注入

- 実装クラスの差し替えが可能（インターフェースを介して利用する）
  - 設定ファイルや属性で実装クラスを変更可能
- DIコンテナが代わりにインスタンス化してインスタンスを管理する
- **DIは単体テストに有効**

S2Container.NET の２つの DI コンテナ

- S2Container (Java から移植)
  - AOP をサポートした DI コンテナ
  - XMLファイルにコンポーネントをクラス等をコンポーネントとして登録する
- Quill （.NET 向けに新しく実装）
  - 業務ロジックに利用することに特化した機能限定の AOP をサポートした DI コンテナ
  - 属性ベースで DI, AOP を指定する

Quill による DI のデモ

Aspect Oriented Programming（AOP）

- アスペクト指向プログラミング
  - AOPで横断的関心事をモジュール化することができる
    - たとえばメソッドの開始と終了部分にログを出力する機能を作ります。これを複数メソッドで必要とします。一般に共通機能はメソッドに抽出しますが、このようにメソッドの開始・終了部分にログを出力するといった機能はメソッド化することが難しいです。こういう機能を横断的関心事と表現します。
  - 用途はロギング、トランザクション、認証、S2Dao.NETのようなフレームワーク等

AOP を使わない場合

ロギング処理をコード中に埋め込む

```
public class CulcLogic
{
    public int Plus(int x, int y)
    {
        Console.WriteLine("BEGIN CulcLogic#Plus(" + x + "," + y + ")");
        int result = x + y;
        Console.WriteLine("END CulcLogic#Plus(" + x + "," + y + ") : " + result);
        return result;
    }
}
```

コンソール

```
BEGIN CulcLogic#Plus(3,4)
END CulcLogic#Plus(3,4) : 7
```

AOP を使った場合

ロギング処理を属性に指定

```
public class CulcLogic
{
    [Aspect(typeof(TraceInterceptor))]
    public int Plus(int x, int y)
    {
        int result = x + y;
        return result;
    }
}
```

コンソール

```
BEGIN CulcLogic#Plus(3,4)
END CulcLogic#Plus(3,4) : 7
```

AOP のポイント

- 共通化（メソッド化）が難しい機能をモジュール化できる
- コードがすっきりする
- ロギング等を埋め込む際のバグの混入を軽減できる
- S2Dao.NET のようなフレームワークを構築できる
- ただし多用は厳禁！（デバッグしにくい）

Quill による AOP のデモ

S2Dao.NET

- 規約ベースの O/R マッピングフレームワーク
  - SQLの実行とオブジェクトのマッピングをサポート
  - データアクセスに関する詳細を隠蔽
  - 利用者は ADO.NET に関する知識が不要
  - マッピングを定義する XML ファイルが不要
  - 実行した SQL がそのままログ出力される
  - SQLのテストが簡単
  - データプロバイダに依存しない

S2Dao.NET のデモ

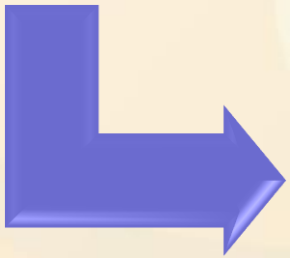
S2Dao.NET - 利用手順

1. Entity クラスの作成
2. Dao インターフェースの作成
3. 必要に応じて Dao インターフェースにメソッドを追加
4. Dao をロジッククラスに DI して使う

S2Dao.NET - Entity クラス

- 基本的にテーブルと1対1で作成
  - 基本的にテーブル名と同じ名前のクラスを作成
  - 基本的にカラム名と同じ名前のプロパティを作成

(例)Employeeテーブル	
ID(PK)	数値型
NAME	文字列型
JOB	文字列型
SALARY	数値型



```
public class Employee{
  private int id;
  private string name;
  private string job;
  private int salary;

  public int ID{
    set { id = value; }
    get { return id; }
  }
・・・省略
```

S2Dao.NET - Entity クラスの属性

- 特別な設定があれば次の属性を設定可能

クラス用の属性	用途
Table	テーブル名とクラス名が異なる場合
NoPersistentProps	カラムとマッピングしないプロパティ名
VersionNoProperty	排他制御 (Int32)のプロパティ名
TimestampProperty	排他制御 (DateTime)のプロパティ名

プロパティ用の属性	用途
Column	カラム名とプロパティ名が異なる場合
ID	IDの自動生成方法
Relno／Relkeys	別テーブルとの結合

S2Dao.NET - Dao インターフェース

- Entity クラスと1対1でインターフェースを作成
- 実行させたいSQLに対して1対1でメソッドを作成

```
[Bean(typeof(Employee))]
public interface IEmployeeDao
{
  int Insert(Employee emp);
  int Update(Employee emp);
  int Delete(Employee emp);
  Employee[] GetAll();
}
```

S2Dao.NET - Dao インターフェースの決まり事

- 実行したいSQLに応じてメソッド名・引数・戻り値に決まり事（規約）がある

実行するSQL	メソッド名	引数	戻り値
INSERT	Insert～ Add～ Create～	Entityクラス (SQL自動生成の場合)	Int32(更新件数) Void
UPDATE	Update～ Modify～ Store～	Entityクラス (SQL自動生成の場合)	Int32(更新件数) Void
DELETE	Delete～ Remove～	Entityクラス (SQL自動生成の場合)	Int32(更新件数) Void
SELECT	上記以外の メソッド名の 場合	任意(Where句等で必要なパラメータ)	Entityクラス(配列も可) IList (Genericも可) String, Int32等

S2Dao.NET - メソッドを追加

- カスタマイズしたSQLを実行したい場合
  - Sql 属性で SQL を記述
  - Query属性で WHERE 句 / ORDER BY 句を記述
  - 外部ファイルにSQLを記述
    - インターフェース名_メソッド名.sql

```
//SQL文を指定
[Sql("SELECT SALARY FROM EMPLOYEE WHERE ID=/*id*/3")]
int GetSalaryByID(int id);
//WHERE句を指定
[Query("JOB = /*job*/ ORDER BY ID ASC")]
Employee[] GetByJob(string job);
//IEmployeeDao_GetBetweenSalary.sqlファイルのSQLを実行
Employee[] GetBetweenSalary(int low, int high);
```

S2Dao.NET - Dao インターフェースの属性

- 特別な設定があれば次の属性を設定可能

メソッド用の属性	用途
Query	WHERE句、ORDER BY句
Sql	SQL文
Procedure	ストアドプロシージャ名
NoPersistentProps	UPDATE/INSERTに含めたくない列
PersistentProps	UPDATE/INSERTに含めたい列

S2Dao.NET - SQL について

- /*～*/ 構文を利用したパラメータのバインド
  - ？や @ の代わりにSQLコメントをしよう
- DB ツール（SQL Server Management Studio 等）とプログラムの両方で実行できる「2WaySQL」

```
SELECT * FROM EMPLOYEE WHERE JOB = /*job*/'SE'
```

  - DB ツールでは次のように解釈

```
SELECT * FROM EMPLOYEE WHERE JOB = 'SE'
```

  - プログラム (S2Dao.NET) では次のように解釈

```
SELECT * FROM EMPLOYEE WHERE JOB = @job
```

S2Dao.NET - Dicon ファイルに登録

- AOP を使って Dao インターフェースに O/R マッピング機能を織り込む
  - ※ S2Dao.NET は現状 Quill だけで利用できないので S2Container と連携する必要あり

```
<!-- DaoInterceptor-->
<component name="DaoInterceptor"
  class="Seasar.Dao.Interceptors.S2DaoInterceptor"/>

<!-- 社員Dao -->
<component class="IEmployeeDao">
  <aspect>DaoInterceptor</aspect>
</component>
```

おさらい

- Seasar.NET プロジェクトとは？
- S2Container.NET (AOP をサポートしたDI コンテナ)
  - Dependency Injection (DI)
  - DI のデモ
  - Aspect Oriented Programming (AOP)
  - AOP のデモ
- S2Dao.NET (O/R マッピングフレームワーク)
  - S2Dao.NET のデモ
  - S2Dao.NET の説明

Seasar.NET プロジェクトの歴史

- [2004年3月] Seasar2 リリース (Java)
- [2005年4月] Seasar2を.NETに移植開始
- [2005年11月] S2.NET 1.0 リリース
- [2007年10月] S2Container.NET 1.3.2

DI のまとめ

- メリット
  - 呼び出すクラスに依存しないコーディングが可能になる（インターフェースを介して利用する）
  - DI は単体テストに有効
- 注意点
  - 依存しているクラスが1度にインスタンス化されるので利用ケース・利用方法を考慮する
  - 設定・属性が間違っていると実行時エラーになる

AOP のまとめ

- メリット
  - コードがすっきりする
  - ロギング等を埋め込んだり解除したりする際のバグの混入を軽減できる
  - S2Dao.NET のようなフレームワークを構築できる
- 注意点
  - 多用禁止
    - デバッグがしにくい

S2Dao.NET のまとめ

- メリット
  - データアクセス部のコード量を削減
  - ADO.NETの使い方のバグを軽減
  - S2WaySQL によるテスト効率アップ
  - SQL のチューニングが容易
  - 急なデータプロバイダの変更にも対応
- サポートツール「DBFlute」
  - C#のコードを自動生成
  - S2Dao.NETの機能補完

リソース

- Seasar.NET プロジェクト
  - http://s2container.net.seasar.org/ja/seasarnet.html
- S2Container.NET
  - http://s2container.net.seasar.org/
- S2Dao.NET
  - http://s2dao.net.seasar.org/

- sugimotokazuyaの日記
  - http://d.hatena.ne.jp/sugimotokazuya/

ありがとうございました

- Seasar.NET を今後ともよろしくお願いいたします。
- Seasar.NET への参加もお待ちしております。