Haute École Léonard de Vinci
SCIENCES ET TECHNIQUES

BINV314A
.NET Outils et Concepts d'Application d'Entreprise

Pattern Repository & UnitOfWork

Sommaire

- Quelques rappels
- Pattern Repository
- Pattern UnitOfWork

Différentes Approches

- Database First
  - Génération des POCO et du modèle à partir de la DB
- Model First
  - Créer UML
- Code First
  - Annotations [Key], [Foreign Key], …

Classes Entities: Propriétés de navigation

```
from p in ctx.Persons
join c in ctx.Cities
on p.BornIn equals c.CityID
select new
{
    p.FirstName,
    c.Name
};
```

```
from p in ctx.Persons
select new
{
    p.FirstName,
    p.BornInCity.Name
};
```

Entity Framework

Lakers	.NET Framework Components
Frontend	Winforms, WPF MVVM, ASP.NET MVC, ConsoleApplication
Backend – Service (UCC)	ASP.NET Web API, WCF
Backend – Business Logic	C# Classes
Backend – DAL – Repository / UnitOfWork	Pattern Repository / UnitOfWork
Backend – DAL	LINQ To Entities – Entity Framework
Database	SQL Server

Pattern Repository

- Objectifs
  - Isoler la couche d'accès aux données de la couche métier pour …
    1. Permettre des tests unitaires
    2. Permettre une automatisation des tests
    3. Eviter le code redondant
    4. Centraliser l'accès aux données et donc sa sécurité
    5. Lisibilité du code
    6. …

Pattern Repository

- Comment ?
  - Généricité par utilisation de types génériques (T)
  - Un repository par Entité/Table
  - Une interface IRepository
  - Une classe de base implémentant l'interface et pouvant être étendue au besoin

Pattern Repository

- Comment ?

```
public interface IRepository<T>
{
    void Insert(T entity);
    void Delete(T entity);
    IList<T> SearchFor(Expression<Func<T, bool>> predicate);
    // sauve l'entité si l'élément n'existe pas déjà ->
       l'existence se base sur le prédicat
    bool Save(T entity,Expression<Func<T, bool>> predicate);
    IList<T> GetAll();
    T GetById(int id);
}
```

Pattern Repository

- Comment ?

```
public class BaseRepositorySQL<TEntity> : IRepository<TEntity>
where TEntity : class
    {

        protected readonly LegumesContext _dbContext;
        public BaseRepositorySQL(LegumesContext dbContext)
        {
            _dbContext = dbContext;
        }

        public void Insert(TEntity entity)
        {

       ...
```

Pattern Repository

- Généricité -> quelques changements

```
_dbContext.CourseSet.Add(course);

→
_dbContext.Set<TEntity>().Add(entity);
```

UnitOfWork

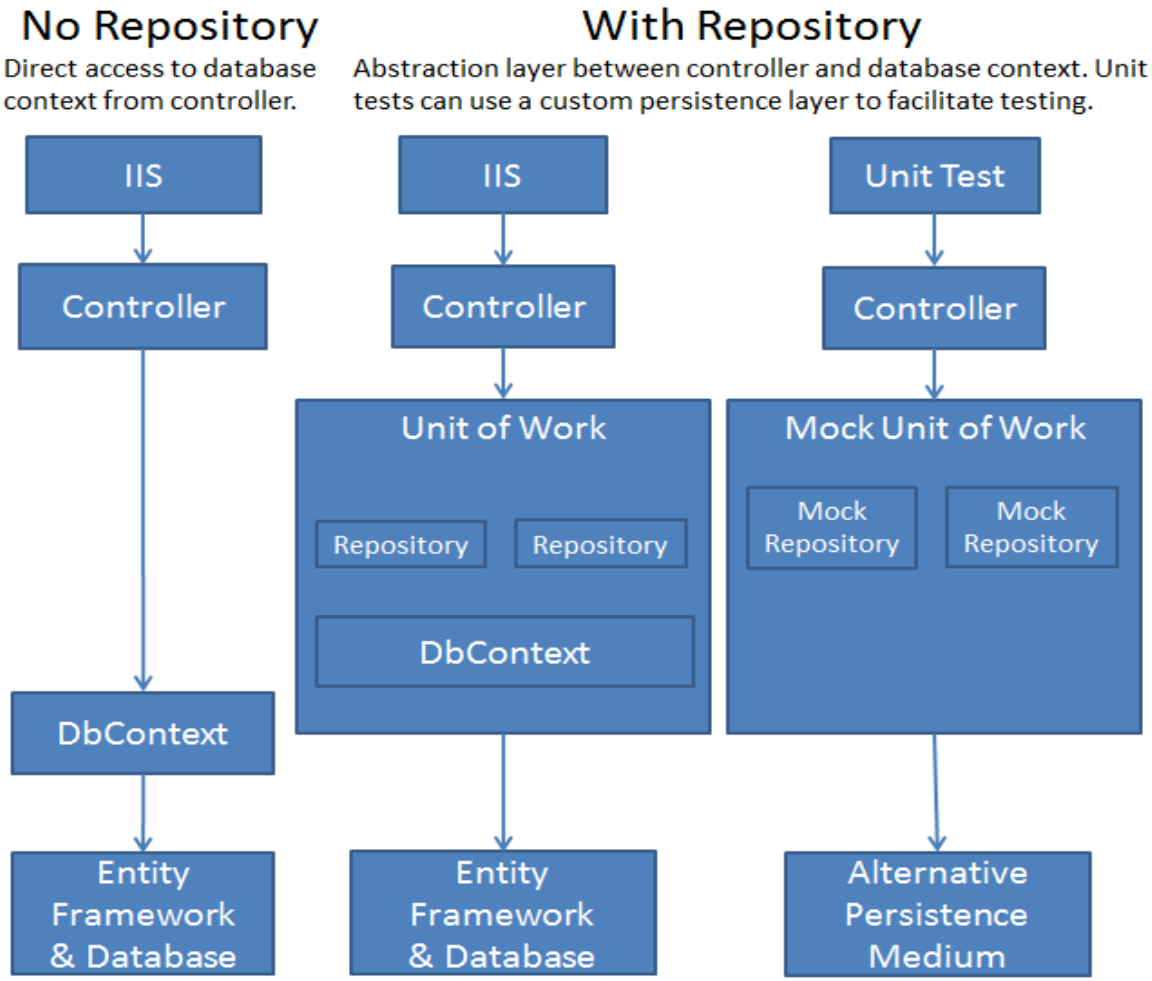

Image issue de https://docs.microsoft.com

UnitOfWork

- Objectifs
  - Encapsuler les repository
    1. Permettre des tests unitaires ++
    2. Permettre une automatisation des tests ++
    3. Eviter le code redondant
    4. Lisibilité du code
    5. Changer facilement le stockage des données (SQL, …)
    6. ….

Démo Légume

- Le code de la démo est disponible sur MooVin et vous pouvez évidemment vous en inspirez.