# ADO.NET

ADO.NET est un ensemble de types fournis par le framework .NET dont l'objectif commun est la manipulation de bases de données. Cette brique, fournie depuis les débuts de .NET, est la fondation de l'accès aux données avec C#.

## 1. Présentation

Les types qui composent ADO.NET sont répartis en deux catégories, définies de telle sorte qu'il est très simple de séparer le requêtage de la base de données et la manipulation des données dans l'application.

Les traitements qui doivent être effectués en lien direct avec une base de données sont implémentés par des types spécifiques à un fournisseur de données. Il est possible, en utilisant ces types, de récupérer des jeux de données ou de supprimer des enregistrements par exemple.

Les jeux de données récupérés peuvent aussi être stockés localement et manipulés totalement indépendamment de leur source grâce à certains types dédiés. Ce mode de fonctionnement est appelé **mode déconnecté**. Il permet notamment de produire des portions de code communes à tous les types de bases de données, ce qui améliore la fiabilité et la maintenabilité des applications en réduisant le nombre de lignes de code et, par conséquent, le nombre de sources potentielles d'erreurs.

## 2. Les fournisseurs de données

Les fournisseurs de données sont les éléments permettant le dialogue avec les différents types de bases de données. Chacun de ces fournisseurs est conçu pour autoriser le fonctionnement avec un type de base de données.

Quatre fournisseurs sont livrés avec le framework .NET. Ils permettent d'interagir avec les sources de données suivantes :

- SQL Server
- Oracle
- OLE DB
- ODBC

Les fonctionnalités présentées par chacun des fournisseurs sont comparables. En effet, les classes qu'ils contiennent héritent de types communs permettant notamment l'établissement de connexions, l'exécution de requêtes ou la lecture de données. Les principaux types de base implémentés par ces fournisseurs sont les suivants :

- La classe `Connection` permet d'établir une connexion à une base de données.
- La classe `DbCommand` permet d'exécuter une ou plusieurs requêtes SQL.
- La classe `DbParameter` représente un paramètre de commande.
- La classe `DbDataReader` fournit un accès séquentiel (en avant uniquement) et en lecture seule à des jeux de données.
- La classe `DbDataAdapter` est la passerelle entre le mode connecté et déconnecté. Elle assure la mise en cache local des données provenant de l'exécution de requêtes SQL et met à jour les données de la base en fonction de l'état des données locales de travail.
- La classe `DbTransaction` encapsule la gestion des transactions pour l'exécution de requêtes SQL multiples.

Ce mode de structuration offre la possibilité d'étendre les sources de données supportées par une application en développant des fournisseurs pour d'autres systèmes de bases de données. À ce jour, il existe des fournisseurs pour MySQL, SQLite, PostgreSQL, Firebird et bien d'autres encore.

### a. SQL Server

Le fournisseur de données pour SQL Server utilise son propre protocole pour communiquer avec le serveur de base de données. Il est par conséquent **peu consommateur** de ressources et **performant** puisqu'il n'utilise pas de couche logicielle supplémentaire comme le font OLE DB et ODBC. Il est compatible avec les versions 7.0 (sortie en 1998) et supérieures de SQL Server.

Les classes spécifiques à ce fournisseur se trouvent dans l'espace de noms `System.Data.SqlClient`. Leur nom est systématiquement préfixé par *Sql* : `SqlConnection`, `SqlDataReader`, etc.

C'est ce fournisseur de données qui est utilisé dans les explications et exemples de cet ouvrage. Il est possible de les transposer aux autres fournisseurs en utilisant les classes appropriées.

### b. Oracle

Le fournisseur de données pour Oracle permet la communication avec les bases de données Oracle 8.1.7 et supérieures à travers la couche logicielle cliente Oracle. Il est donc nécessaire d'installer les **composants Oracle** appropriés pour utiliser ce fournisseur.

Les types proposés par ce fournisseur se situent dans l'espace de noms `System.Data.OracleClient`, lui-même déclaré dans l'assembly *System.Data.OracleClient.dll*. Ceci implique d'ajouter une référence à cet assembly pour utiliser ce fournisseur. Le nom de chacun de ces types est préfixé par Oracle.

Le fournisseur Oracle est livré avec le framework .NET de la version 2.0 à la version 4.6. Dans cette dernière version, il est considéré comme **déprécié**, ce qui signifie qu'il est supporté mais qu'il est susceptible d'être supprimé du framework dès la prochaine version. Microsoft conseille de s'appuyer sur des fournisseurs de données tiers pour communiquer avec les bases de données Oracle.

### c. OLE DB

Ce fournisseur utilise la couche logicielle OLE DB pour communiquer avec les serveurs de bases de données. Il est intéressant lorsqu'aucun fournisseur spécifique à une base de données n'existe mais qu'un pilote natif compatible OLE DB est disponible. Dans ce cas, le fournisseur OLE DB va agir comme un **intermédiaire entre l'application et le pilote**, et ce dernier s'occupera de la communication avec la base de données.

Les classes spécifiques à ce fournisseur de base de données se trouvent dans l'espace de noms `System.Data.OleDb` et leur nom est préfixé par *OleDb*.

Pour pouvoir fonctionner, ce fournisseur nécessite la présence sur la machine des composants MDAC (*Microsoft Data Access Components*) en version 2.6 ou supérieure.

### d. ODBC

Le fournisseur ODBC utilise le même principe de fonctionnement que le fournisseur OLE DB. Il nécessite donc qu'un pilote natif compatible ODBC existe pour la source de données visée afin de dialoguer avec celui-ci. C'est ensuite le pilote natif qui communiquera avec la base de données.

Ce fournisseur de données est implémenté dans l'espace de noms `System.Data.Odbc`, et chacun de ses types a pour préfixe de noms *Odbc*.

Pour pouvoir fonctionner, ce fournisseur nécessite la présence sur la machine des composants MDAC (*Microsoft Data Access Components*) en version 2.6 ou supérieure.