

## Technische documentatie

### Datatoegang

Om de data vanuit de database beschikbaar te maken in ons programma maken we gebruik van ADO.NET Data Adapters. Ze worden ook wel Data Providers genoemd, omdat ze, zoals de naam doet vermoeden, data aanbieden. In ons geval gebruiken we de Data Adapter om data uit een database-tabel te halen. In dit geval mogen we dit ook een Table Adapter noemen, zoals Microsoft dit aanraadt om onderscheid te maken.

Data Adapters zijn standaard ingebakken in de .NET omgeving, waar ze een deel zijn van de “ADO.NET managed providers” waar ze de verzameling objecten zijn die gebruikt worden om te communiceren tussen een databron en een dataset. In deze “managed providers” zitten ook de objecten voor connecties op te zetten, data te lezen en besturingsobjecten.

In het algemeen wordt er vaak data gelezen en weggeschreven in een database. Dit proces is op zich niet heel moeilijk. Maar voor elke databron en dataset moet er een aparte functie geschreven worden die de data omzet naar het formaat dat de database wenst. Wanneer er meerdere databronnen zijn kan dit een tijdrovende klus zijn. De oplossing hiervoor zijn de Data Adapters, of in dit geval, Table Adapters. Deze Adapters vormen vanzelf de data om naar het formaat dat de database wenst. Met andere woorden: we moeten maar één functie aanroepen om verschillende databronnen in onze database te laten lezen of schrijven. Dit biedt niet alleen minder werk, maar ook minder zorgen, omdat we zeker zijn dat het formaat van de database gehandhaafd worden, als het eenmaal goed geconfigureerd en getest is. Ook het later toevoegen van nieuwe databronnen is geen probleem omdat steeds dezelfde toegang gebruikt wordt.

Er zijn vier verschillende handelingen mogelijk met de Data Adapters:

- SelectCommand: rijen uit de database halen
- InsertCommand: rijen invoegen in de database
- UpdateCommand: rijen aanpassen in de database
- DeleteCommand: rijen verwijderen in de database

Met behulp van de grafische tools (“Data Adapter Configuration Wizard”) van Visual Studio, kan op een interactieve manier datakoppelingen gemaakt worden in de database. Men selecteert in de wizard de gewenste tabellen, hierin selecteert men de kolommen die men wil gebruiken en maakt de individuele koppelingen. Visual Studio zal zelf de achterliggende SQL-commando’s genereren (Select, Insert, Update, Delete). Wanneer men dit wenst, kan men echter zelf de SQL statements schrijven of de aangeboden statements wijzigen.

### Voorbeeld:

We hebben een Table Adapter “CourseAdapter” aangemaakt waarin een InsertCommand is aangemaakt om data in de database-tabel `Course` in te voegen. Het enige wat we nu moeten doen is de “functie” aanroepen met de gewenste parameters.

```
using ASPNETDBTableAdapters;
```

```
string guid = "2b389083-d627-4a21-847f-9ee30aa9abfe";
System.Guid g = new Guid(guid);

//read vars from page
int _courseID = Convert.ToInt32(txID.Text);
string _courseName = txName.Text;
string _description = txDesc.Text;

//insert into our table
CourseAdapter.InsertNewCourse(_courseID, g, _courseName, _description);
```

Wanneer de Table Adapter nadien wordt aangepast (bijvoorbeeld de betreffende database-tabel) hoeft er verder niets worden aangepast in de code.

Omdat de wizard zelf de andere Commands heeft aangemaakt, kan het verwijderen van een tabel zo gemakkelijk zijn als:

```
using ASPNETDBTableAdapters;

//read vars from page
int _courseID = Convert.ToInt32(txID.Text);

CourseAdapter.RemoveCourse(_courseID);
```

Dit toont de kracht aan van de Table Adapters.

Ook kunnen de Table Adapters gekoppeld worden aan andere objecten in de .NET omgeving, zoals bijv. de datagrid. Een simpel voorbeeld hierbij is de manier waarop de agenda gevuld wordt in ons project.

Deze Table Adapter is de standaard "Fill" Command die de Wizard vanzelf aanreikt wanneer men geen specifieke variabelen selecteert (het SQL equivalent is hier `SELECT * FROM DBTable`).

De code is als volgt om een datagrid te vullen:

```
using ASPNETDBTableAdapters;

CourseTableAdapter CourseAdapter = new CourseTableAdapter();
gridCourse.DataSource =
CourseAdapter.GetCourseByDateAndUser(calThisM.SelectedDate,
User.Identity.Name);
gridCourse.DataBind();
```

De conclusie is dat het gebruik Data Adapters een snelle, gemakkelijke en gebruiksvriendelijke manier is om data te manipuleren in een database.