

Module 1: Data-Centric Applications and ADO.NET

Overview

- Design of Data-Centric Applications
- ADO.NET Architecture
- ADO.NET and XML

Lesson: Design of Data-Centric Applications

- Data Storage
- What Is a Connected Environment?
- What Is a Disconnected Environment?
- Data Access Application Models

Data Storage

ADO.NET supports the following types of data storage:

- Unstructured
- Structured, non-hierarchical data
 - Comma Separated Value (CSV) files, Microsoft Excel spreadsheets, Microsoft Exchange files, Active Directory files, and others
- Hierarchical
 - XML documents and others
- Relational database
 - SQL Server, Oracle, Access, and others

What Is a Connected Environment?

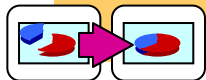
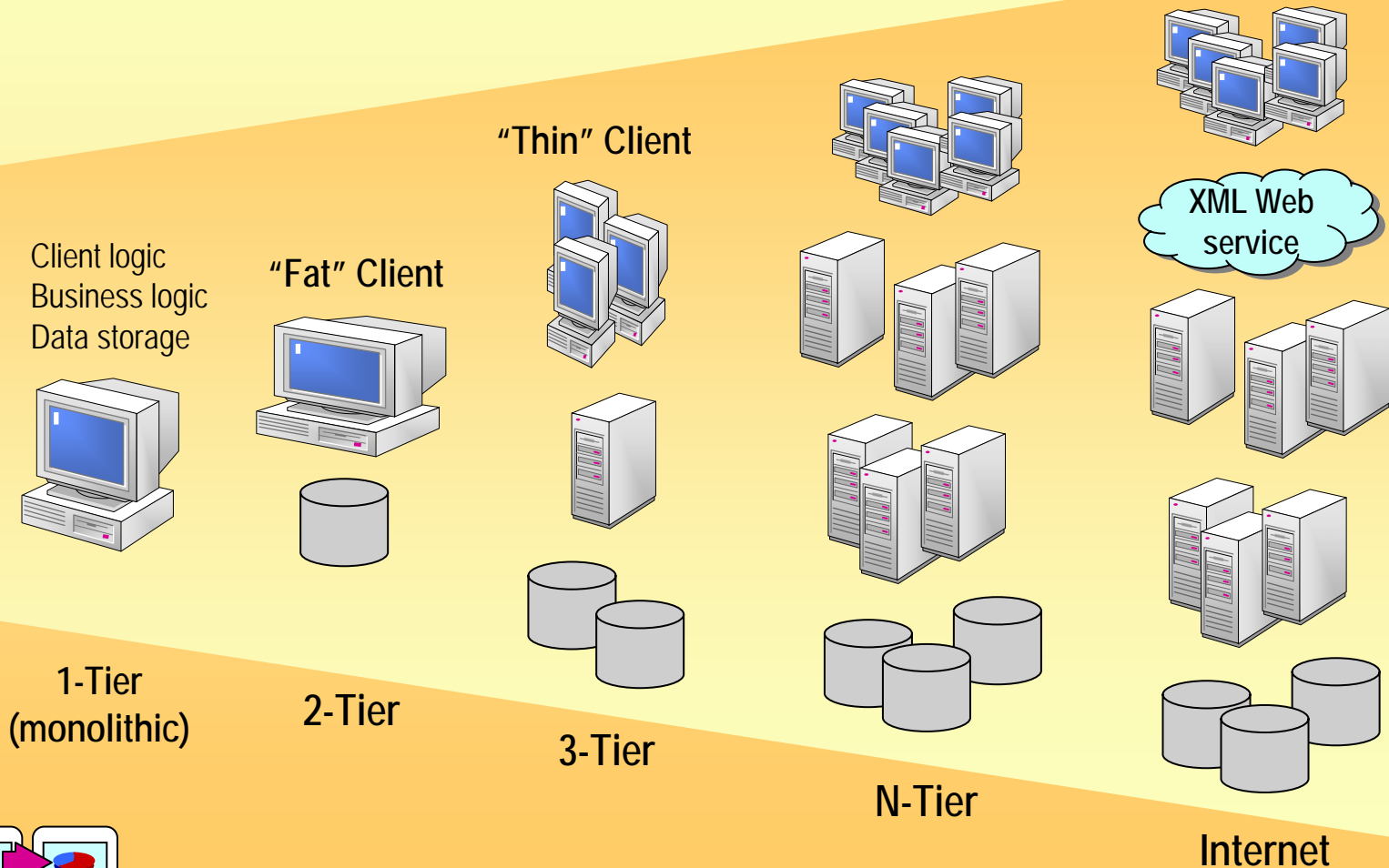
- A connected environment is one in which users are constantly connected to a data source
- Advantages:
 - Environment is easier to secure
 - Concurrency is more easily controlled
 - Data is more likely to be current than in other scenarios
- Disadvantages:
 - Must have a constant network connection
 - Scalability

What Is a Disconnected Environment?

- In a disconnected environment, a subset of data from a central data store can be copied and modified independently, and the changes merged back into the central data store
- Advantages
 - You can work at any time that is convenient for you, and can connect to a data source at any time to process requests
 - Other users can use the connection
 - A disconnected environment improves the scalability and performance of applications
- Disadvantages
 - Data is not always up to date
 - Change conflicts can occur and must be resolved

Data Access Application Models

Evolution of data access



Lesson: ADO.NET Architecture

- What Is ADO.NET?
- What Are the Data-Related Namespaces?
- Evolution of ADO to ADO.NET
- The ADO.NET Object Model
- Using ADO.NET Classes in a Connected Scenario
- Using ADO.NET Classes in a Disconnected Scenario

What Is ADO .NET?

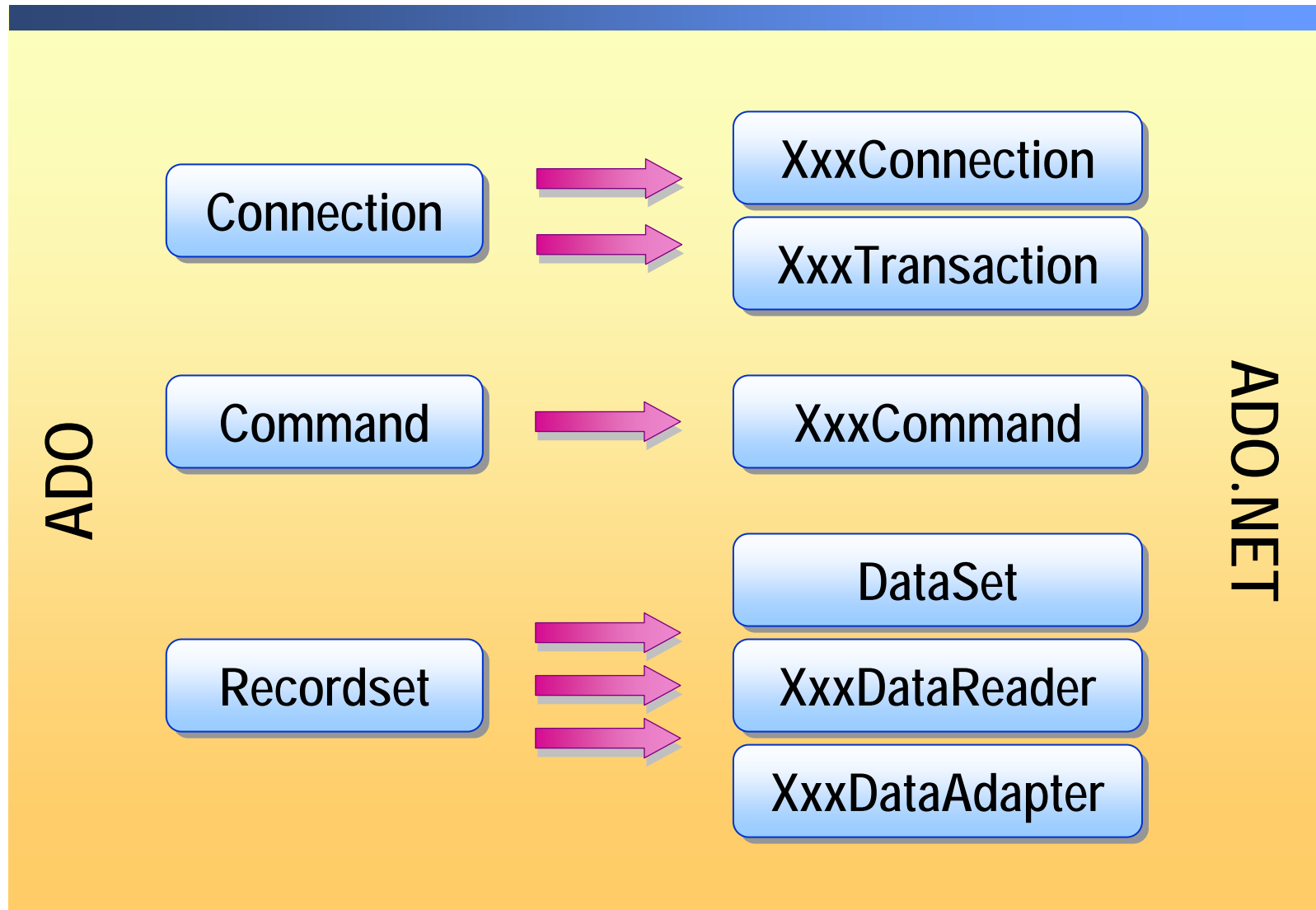
ADO.NET is a set of classes for working with data.
It provides:

- An evolutionary, more flexible successor to ADO
- A system designed for disconnected environments
- A programming model with advanced XML support
- A set of classes, interfaces, structures, and enumerations that manage data access from within the .NET Framework

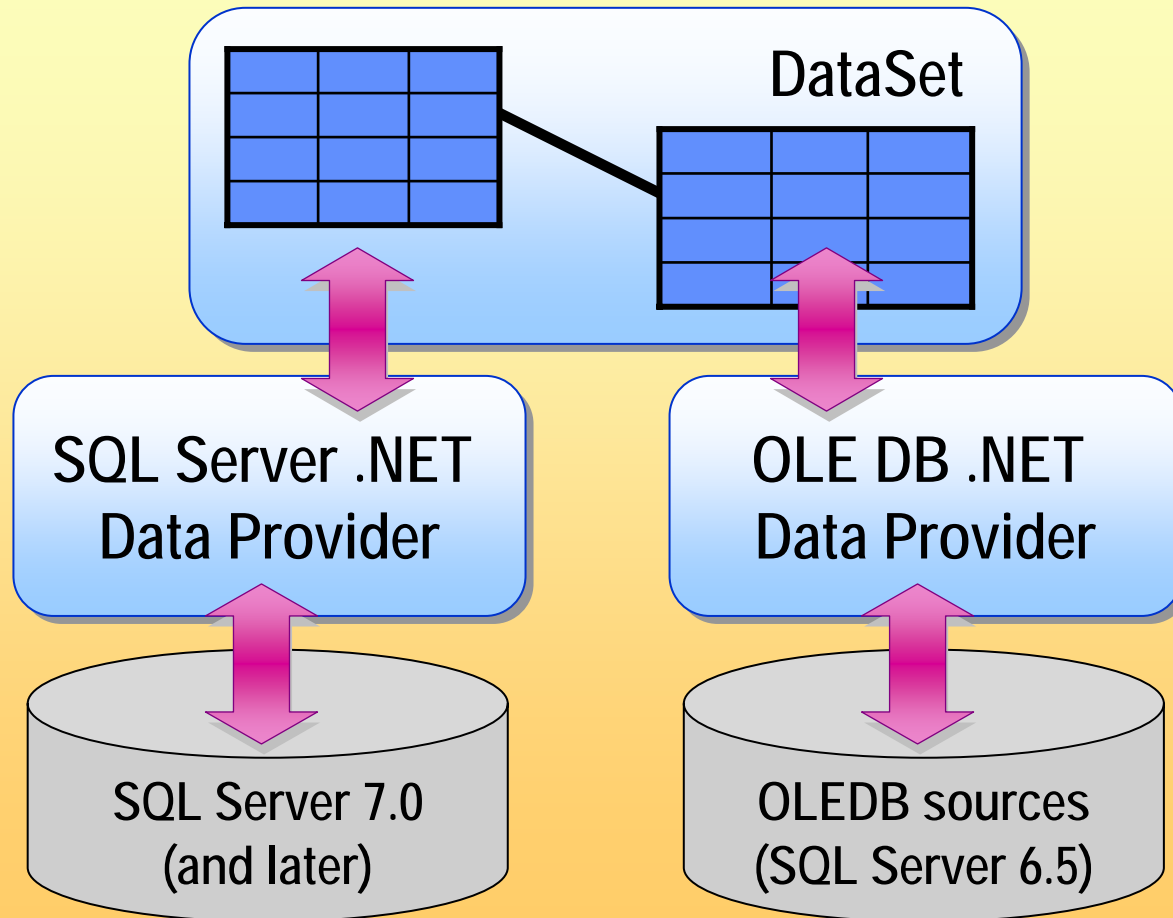
What Are the Data-Related Namespaces?

- The data-related namespaces include:
 - System.Data
 - System.Data.Common
 - System.Data.SqlClient
 - System.Data.OleDb
 - System.Data.SqlTypes
 - System.Xml
- Practice

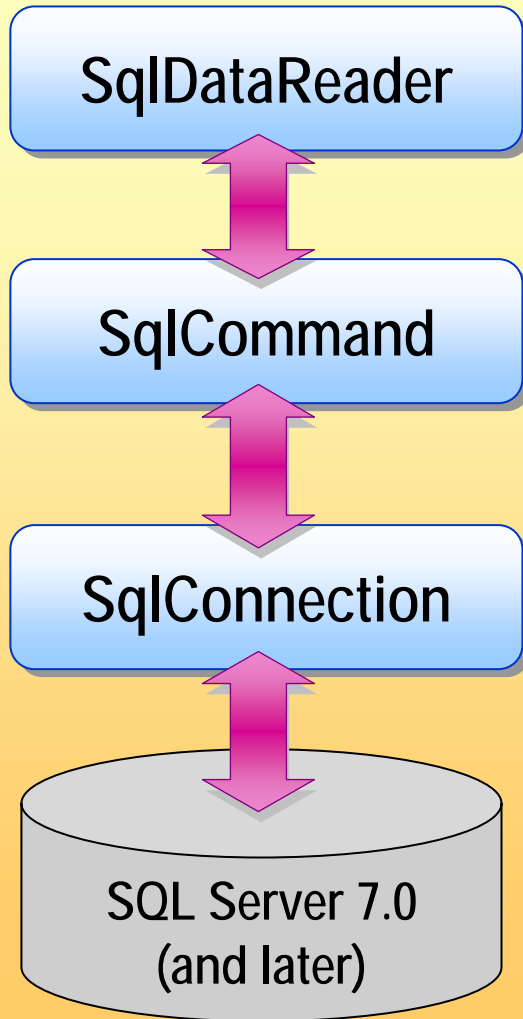
Evolution of ADO to ADO.NET



The ADO .NET Object Model



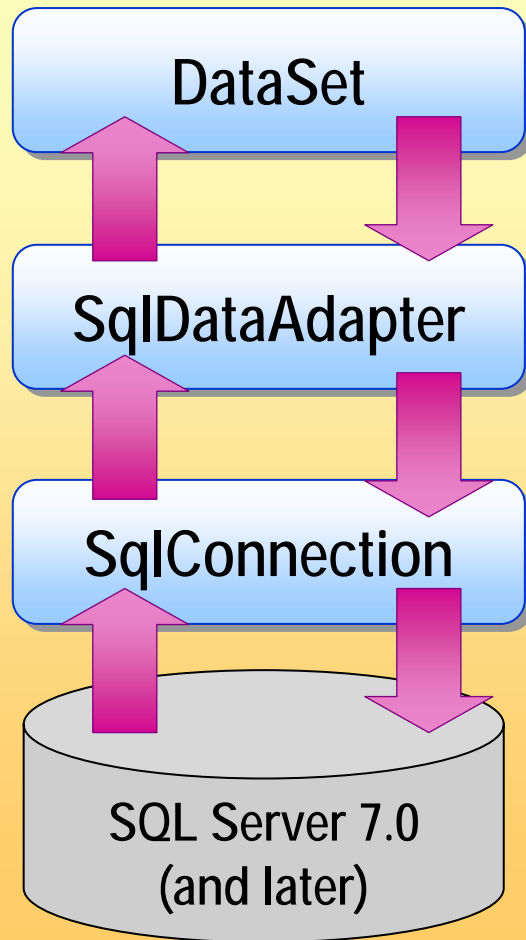
Using ADO.NET Classes in a Connected Scenario



- In a connected scenario, resources are held on the server until the connection is closed

1. Open connection
2. Execute command
3. Process rows in reader
4. Close reader
5. Close connection

Using ADO.NET Classes in a Disconnected Scenario



- In a disconnected scenario, resources are not held on the server while the data is processed

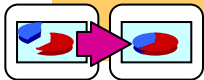
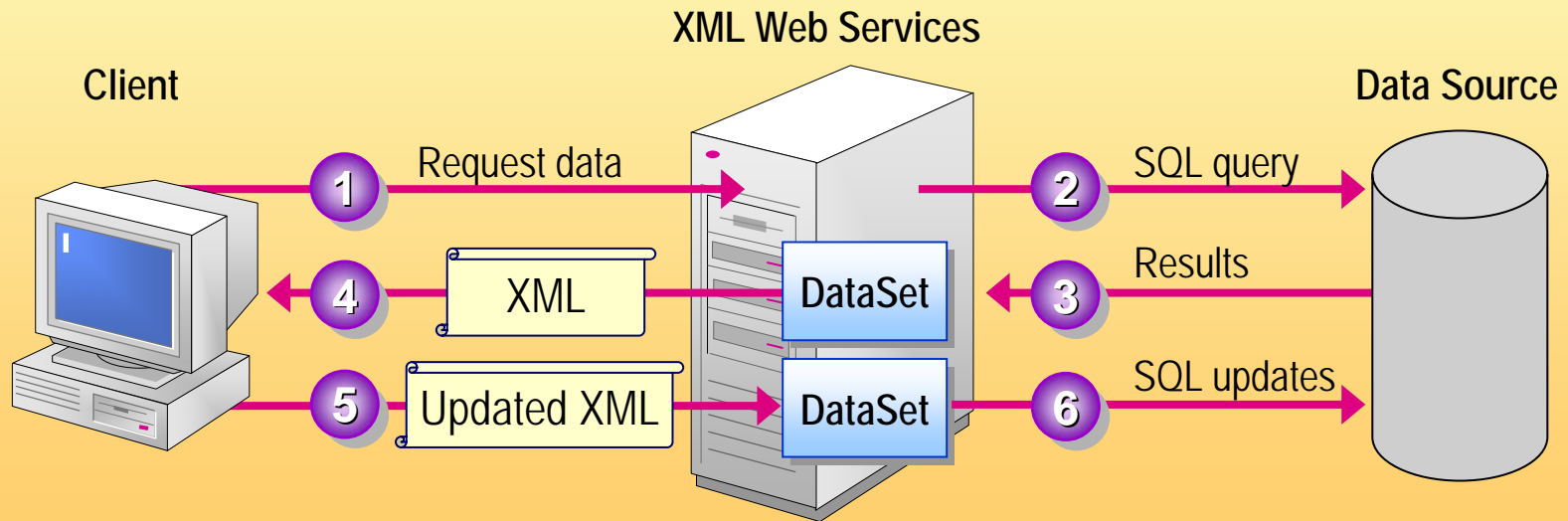
1. Open connection
2. Fill the DataSet
3. Close connection
4. Process the DataSet
5. Open connection
6. Update the data source
7. Close connection

Lesson: ADO.NET and XML

- ADO.NET and XML
- Multimedia
- Demonstration

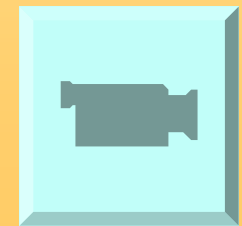
ADO.NET and XML

- ADO.NET is tightly integrated with XML
- Using XML in a disconnected ADO.NET application



Multimedia: How an XML Web Service Uses ADO.NET

- This animation describes how an XML Web service uses ADO.NET



Demonstration

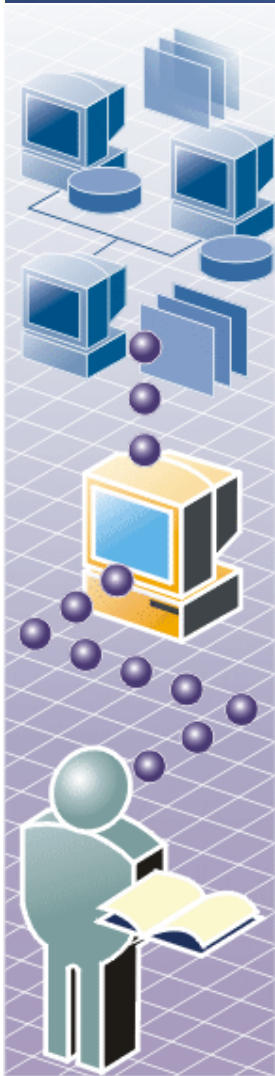


- Solution to Lab 7.1
- Includes two projects:
 - An XML Web service that uses ADO.NET to generate a DataSet
 - A Windows application that uses the XML Web service

Review

- Design of Data-Centric Applications
- ADO.NET Architecture
- ADO.NET and XML

Lab 1: Data-Centric Applications and ADO .NET



- Exercise 1: Adding ADO .NET Objects to an ASP .NET Web Application
- Exercise 2: Executing a Query and Displaying the Results