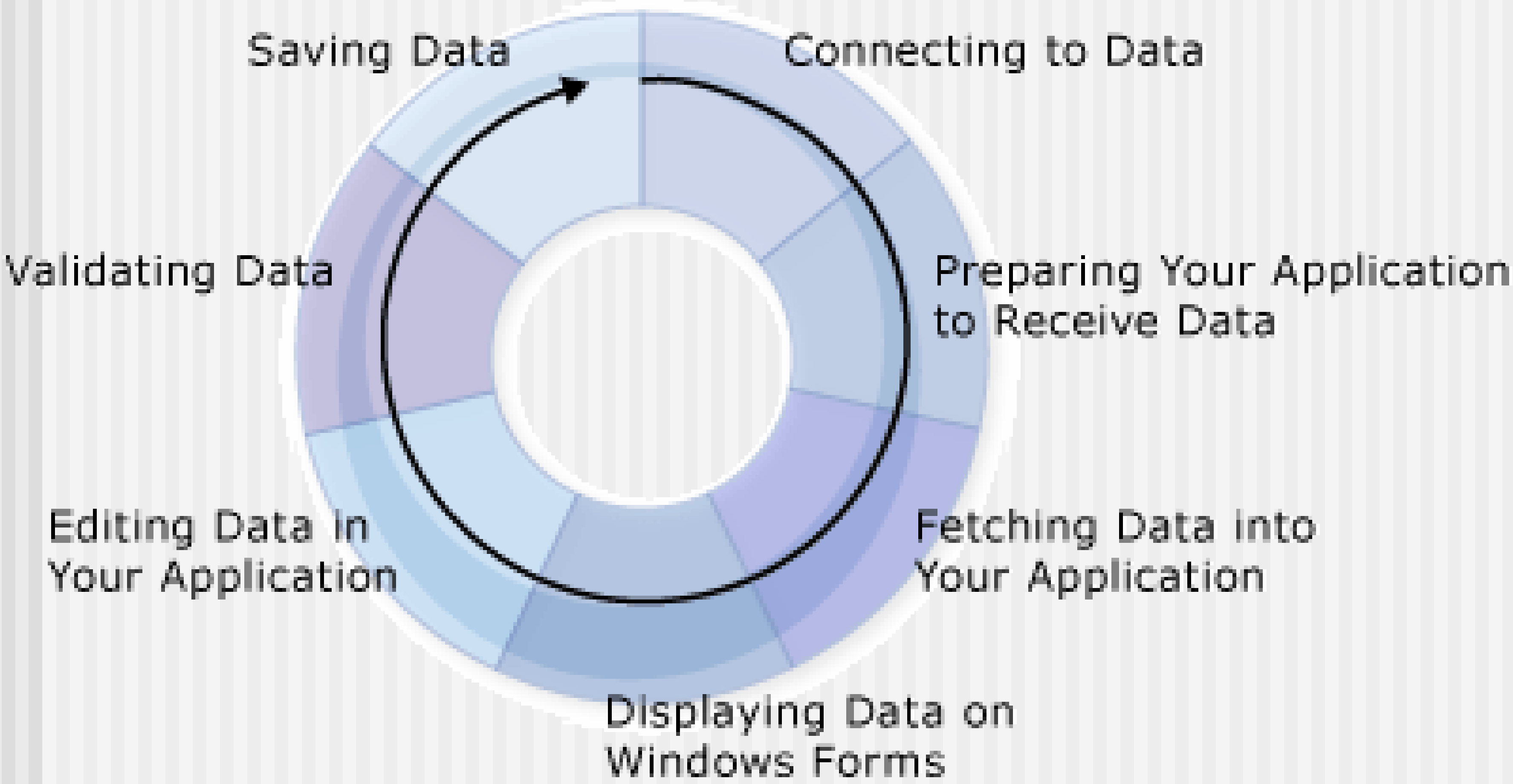
Seminar 1

ADO.NET

The Data Cycle


Connecting to Data
Preparing Your Application to Receive Data
Fetching Data into Your Application
Displaying Data on Windows Forms
Editing Data in Your Application
Validating Data
Saving Data

Connecting to Data

- **Connecting to Data**: establishes a two-way communication between application and database server (TableAdapter, ObjectContext)
- **Preparing App to Receive Data**: when using a disconnected data model there are specific objects (*datasets, entities, LINQ* to SQL objects) which temporarily store data
- **Fetching Data:** execute queries and stored procedures (TableAdapters, LINQ to Entities, direct connection between entities and stored procedures)
- **Displaying Data:** data-bound controls
- **Editing and Validating Data:** add/modify/delete records and verify that these new values are acceptable for the requirements of application
- **Saving Data:** *TableAdapterManager, SaveChanges*

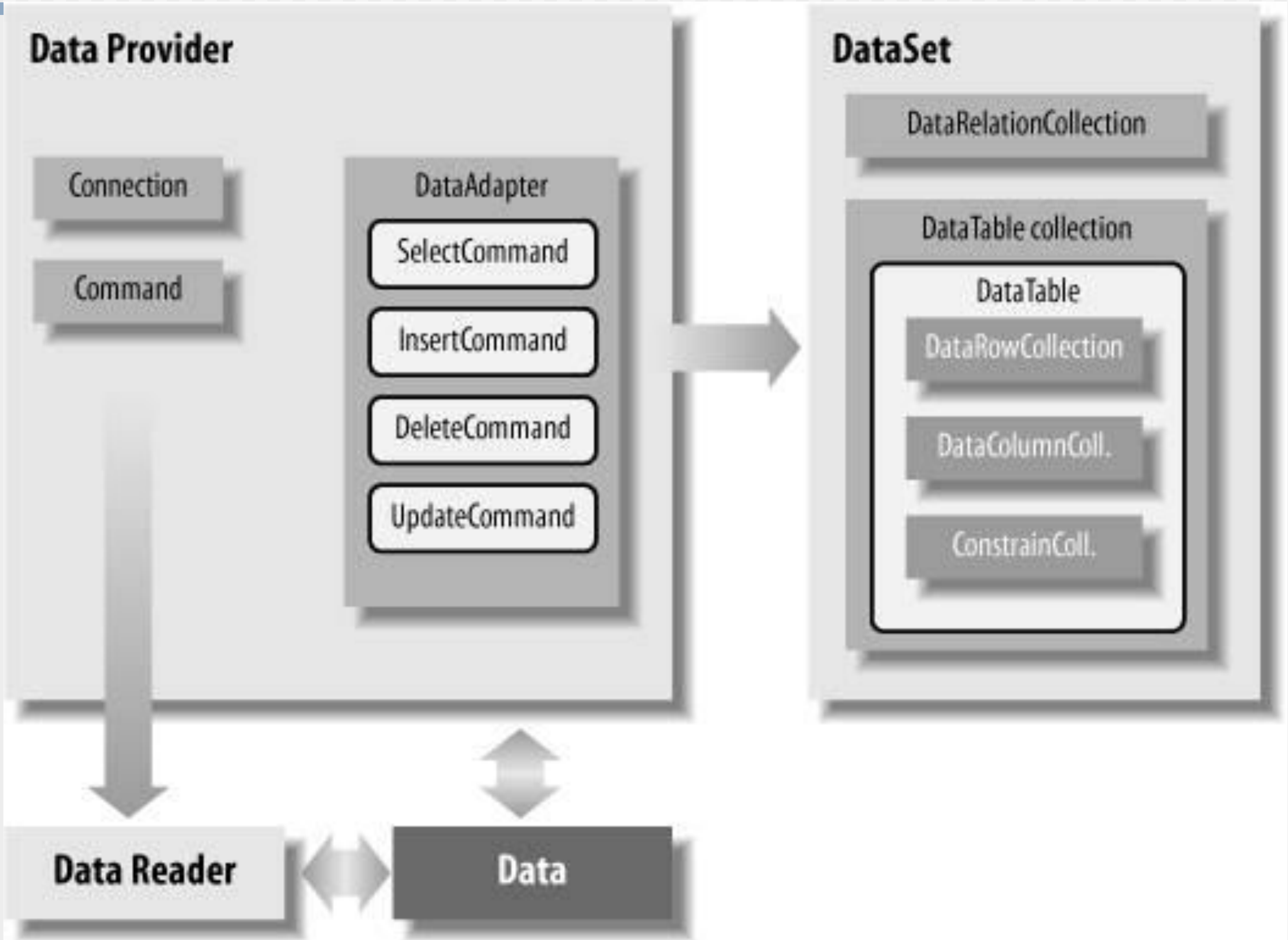
Data Models

- Types/untyped datasets
- Conceptual model based on Entity Data Model. It can be used by Entity Framework or WCF Data Services
- LINQ to SQL classes

Typed/Untyped Datasets

- Datasets are objects that contain data tables where you can temporarily store the data for use in application
- Provides a local in-memory cache
- Works even if application becomes disconnected from the database
- The structure of a ***DataSet*** is similar to that of a relational database (has tables, rows, columns, constraints, and relationships)

ADO.NET


Data Provider
Connection
Command
DataAdapter
SelectCommand
InsertCommand
DeleteCommand
UpdateCommand
DataSet
DataRelationCollection
DataTable collection
DataTable
DataRowCollection
DataColumnColl.
ConstrainColl.
Data Reader
Data

Console

- It represents the standard input, output and error streams for console applications.

- Properties:
  - *WindowLeft, WindowTop, WindowHeight, WindowWidth, BackgroundColor, Title* etc

- Methods:
  - Write(….), WriteLine(….)
  - Read(), ReadLine(), ReadKey()

SQLConnection

- Represents an open connection to a database
- Cannot be inherited
- If the **SqlConnection** goes out of scope, it is not closed → must explicitly close the connection (Close).
- Properties:
  - ***ConnectionString*** - string used to open a SQL Server database (see http://www.connectionstrings.com/)
  - ***ConnectionTimeout*** - time to wait to establish a connection before terminating the attempt and generating an error
- Methods:
  - Open()
  - Close()
- If ***SqlException*** is generated, the ***SqlConnection*** remains open when the severity level < 19

SQLCommand

- Represents a Transact-SQL statement or stored procedure to execute against a SQL Server database
- Properties
  - CommandText
  - CommandTimeout
- Methods
  - *ExecuteNonQuery* – return number of rows affected
  - *ExecuteScalar* – return value of first column of first row
  - *ExecuteReader* – build an SQLDataReader

SQLDataReader

- reads a forward-only stream of rows from a SQL Server database

SQLDataAdapter

- Bridge between DataSet and SQL Server for retriving and saving data
- Properties:
  - ***UpdateCommand*** - statement or stored procedure used to update records in the data source.
  - ***InsertCommand*** - statement or stored procedure to insert records
  - ***DeleteCommand*** - statement or stored procedure to delete records from the data set.
- Methods:
  - Fill(DataSet, String) - adds or refreshes rows in the DataSet object to match those in data source

Data binding

- ***DataBindings*** – property of a control which gives the possibility to bind any property of that control to the property of an object

- ***Binding*** class - represents the simple binding between the property value of an object and the property value of a control.
  - **Binding (String propertyName, Object dataSource, String dataMember)**

DataSet

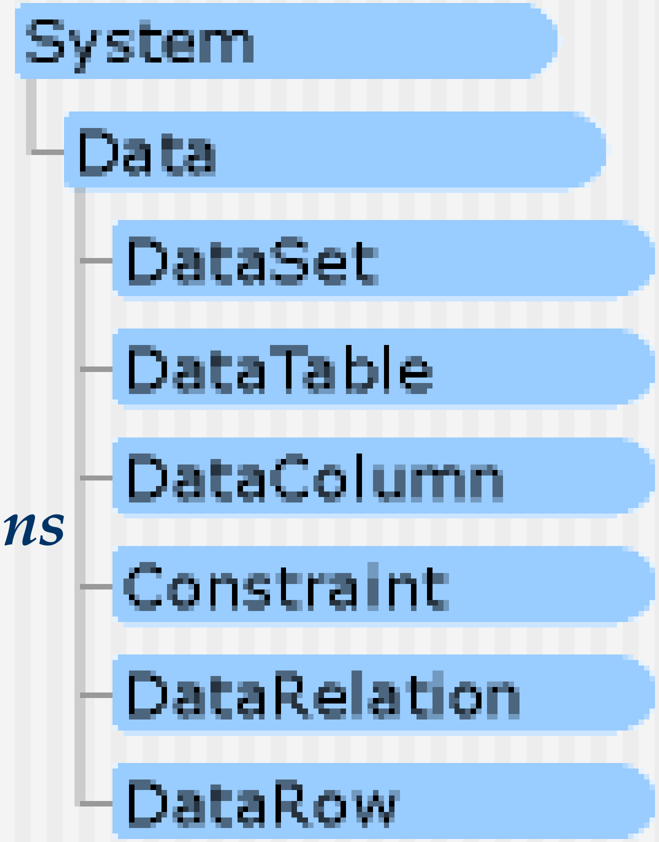
- Represents an in-memory cache of data
- Properties:
  - ***Tables*** – collection of table contained in DataSet
  - ***Relations*** – collection of relations (child/parent tables)
- Methods:
  - ***Clear*()** – clears all rows in all tables
  - ***HasChanges*()** – indicates if there are new, deleted, or modified rows

Datasets

- The *DataSet* class includes the *DataTableCollection* and *DataRelationCollection*

- The *DataTable* class includes *DataRowCollection, DataColumnCollection, ChildRelations* and *ParentRelations*.

- The *DataRow* class includes the *RowState* property (possible values: *Deleted, Modified, Added* and *Unchanged*)



System
Data
DataSet
DataTable
DataColumn
Constraint
DataRelation
DataRow

Populating Datasets with Data

- A dataset contains no actual data by default

- Data tables filled by executing ***TableAdapter*** queries, or executing data adapter commands (***SqlDataAdapter***)

```
aTableAdapter.Fill(aDataSet.TableName);
```

- Saving data

```
aTableAdapter.Update(aDataSet.TableName)
```

Update method examines the value of the ***RowState*** property to determine which records need to be saved and what specific database command (*InsertCommand, UpdateCommand,* and *DeleteCommand*) should be invoked.

Navigating Records

- each table exposes a collection of rows.

- like any collection, rows are accessed by means of the collection's index or using collection-specific statements in host programming language.
- typed dataset

```
 TextBox1.Text = aDataSet.TableName(3).aField
```

- untyped dataset

```
string IDValue = (string)
dataset1.Tables[“aTable"].Rows[0][“aField"];
```