Разработка приложений на платформе .NET

Лекция 21.

ADO.NET

Базы данных

- Сервера баз данных: MS SQL Server, Oracle, MySQL, DB2, Foxpro, FireBird, PostgreSQL, …

- Хранят данные в виде таблиц
- Хранят отношение между таблицами
- Обеспечивают целостность данных, отказоустойчивость
  - Поддерживают механизм транзакций
- Обеспечивают доступ к данным посредством языка SQL

Язык SQL

- Стандарты SQL 86, SQL 89, SQL 92, … SQL 2008
- Содержит команды в текстовом виде
- Запросы компилируются самим сервером
- Не все сервера баз данных поддерживают стандарты в полном объеме или имеют некоторые модификации команд описанных в стандартах
- Модификации языка
  - T-SQL (transact SQL, Microsoft)
  - PL-SQL (Oracle)
  - MySQL

Язык SQL

- Операторы определения данных (Data Definition Language, DDL): CREATE, ALTER, DROP
- Операторы манипуляции данными (Data Manipulation Language, DML): SELECT, INSERT, UPDATE, DELETE
- Операторы определения доступа к данным (Data Control Language, DCL): GRANT, REVOKE, DENY
- Операторы управления транзакциями (Transaction Control Language, TCL): COMMIT, ROLLBACK, SAVEPOINT

Data Definition Language, DDL

- CREATE – создать объект ()
  - CREATE TABLE, CREATE DATABASE, CREATE TRIGGER, CREATE PROCEDURE
  - CREATE TABLE tablename (colname1 type1 constr1, … colnameN typeN constrN, table constraints)

```
CREATE TABLE Books(
    id int IDENTITY(1,1) NOT NULL,
    name nvarchar(max) NOT NULL,
    author nvarchar(max) NOT NULL,
    abstract ntext NULL,
    price money NOT NULL,
    CONSTRAINT PK_books1 PRIMARY KEY CLUSTERED (id ASC))
```

- DROP – удаление объекта
  - DROP TABLE Books
- ALTER – изменение объекта

Data Manipulation Language, DML

- SELECT - считывает данные, удовлетворяющие заданным условиям
- INSERT - добавляет новые данные
- UPDATE - изменяет существующие данные
- DELETE - удаляет данные

SELECT

- SELECT col1, …, colN
  FROM table1, …, tableM
  [WHERE constraints]
  [ORDER BY order]
  [GROUP BY constraints] – выбор данных
- FROM – задает таблицы для выбора данных
- WHERE – условия на строки, которые будут выбраны
- ORDER BY – упорядочение результата
- GROUP BY – группировка
  SELECT Author, Names, Cost FROM Books WHERE Cost>50
  ORDER BY Author

INSERT, UPDATE, DELETE

- INSERT INTO tablename(ColName1, …, ColNameN) VALUES(val1, …, valN) – добавление строки
  - Значения всех полей должны быть заданы
  - Нельзя задавать IDENTITY поля
  - Можно опускать столбцы допускающие NULL или имеющие значение по умолчанию
  - Значение неизвестно – NULL
- UPDATE tablename SET col1=val1, …, colN=valN WHERE [constraints]
  - Изменяется только часть полей во всех строках, соответствующих условию
- DELETE FROM tablename WHERE [constraints]
  - Удаляет все строки, удовлетворяющие условию

Работа с SQL

Демонстрация

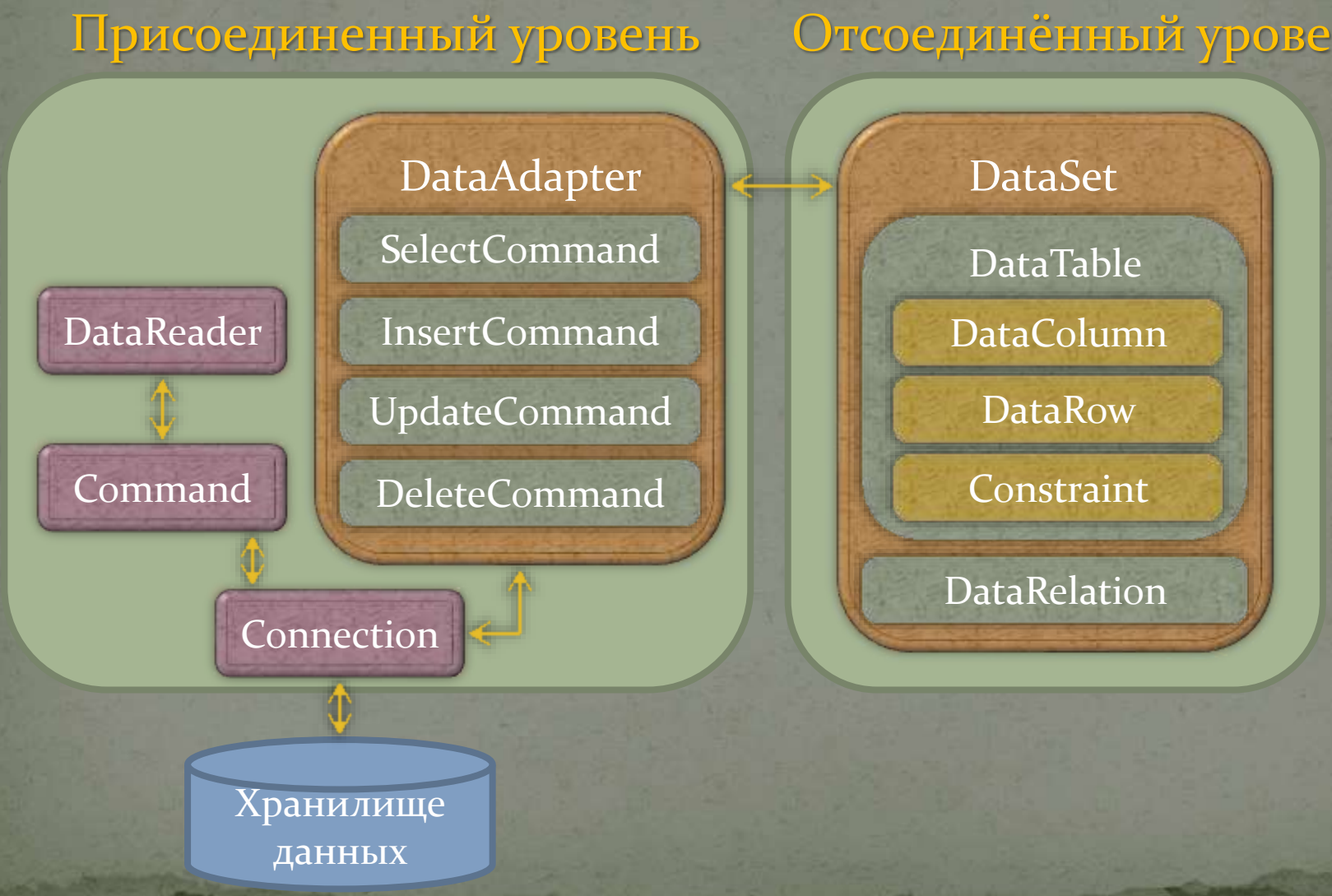
ADO.NET

- Предоставляет доступ к локальным и удаленным реляционным базам данных
- Предоставляет единую модель доступа к различным базам данных
- Доступ к базам данных осуществляют провайдеры данных характерные (“заточенные”) для данного сервера баз данных
  - Повышает производительность
  - Позволяет использовать специфические особенности сервера баз данных

Модели доступа

- Присоединенная
  - Открывается соединение с БД
  - Соединение остается на весь период работы
- Отсоединенная
  - Данные загружаются из БД. Сохраняется локальная копия данных. Соединение закрывается
  - Данные модифицируются пользователем (при отсутствии соединения с БД)
  - Данные сохраняются в БД (кратковременно открывая соединение)
- Entity Framework – ORM (Object-relational mapping)
  - Предоставляет связь между объектами в коде и записями в реляционной базе данных.

ADO.NET


Присоединенный уровень
Отсоединённый уровень
DataAdapter
SelectCommand
InsertCommand
UpdateCommand
DeleteCommand
DataReader
Command
Connection
DataSet
DataTable
DataColumn
DataRow
Constraint
DataRelation
Хранилище данных

ADO .NET

- Единая модель доступа к реляционным данным
  - Разные базы – схожие модели
- Провайдеры данных
  - Обеспечивает набор классов для доступа к конкретной БД
  - Провайдеры в .NET:
    - SqlClient – MS SQL Server 2000/2005/2008/2008 R2/2012
    - Odbc - ODBC
    - OleDb – OLE DB
    - OracleClient – Oracle
  - Также доступны провайдеры для
    - FireBird, DB2, MySQL, PostgreSQL, …
- Пространства имен
  - System.Data – основное
  - System.Data.OleDb – Компоненты OLE DB провайдера
  - System.Data.SqlClient – Компоненты MS SQL Server провайдера
  - System.Data.SqlClientCe – Компоненты MS SQL Server Mobile провайдера
  - System.Data.Odbc– Компоненты ODBC провайдера
  - System.Data.OracleClient– Компоненты Oracle провайдера

Классы

- Компоненты модели
  - XXXConnection – соединение с БД
  - XXXCommand - команда
  - XXXParameter – параметр команды
  - XXXDataReader – чтение результата запроса
  - XXXTransaction – транзакция
  - XXXTableAdapter – адаптер данных
- Где XXX название провайдера
- Например:
  - SqlConnection – соединение с SQL Server
  - OdbcCommand – команда для ODBC провайдера
  - OracleParameter – параметр команды для Oracle
  - OleDbDataReader – reader для OLE DB Провайдера
  - SqlTransaction – транзакция для SQL Server
  - SqlTableAdapter – адаптер данных для SQL Server

Класс SqlConnection

- Класс соединения с БД MS SQL Server
- Содержит параметры соединения (строку соединения)
- Open() – открывает соединение с базой данных
- Close() – закрывает соединение
- Необходимо явное закрытие соединения
  - Необходимо вызывать Close()
  - Реализует IDisposable. При этом закрывает соединение (Close())
- Пример:

```
string cs =
        "Data Source=(local);" +
        "Initial Catalog=AdventureWorks;" +
        "Integrated Security=SSPI;";
using(SqlConnection sc =
        new SqlConnection(cs)) {
        sc.Open();
        // do some work
}  // end of using()
```

Строка соединения

- Основные параметры:
  - Data Source – сетевое имя сервера
  - Encrypt – шифрованное соединение
  - Initial Catalog – имя БД на сервере
  - Integrated Security – способ аутентификации
    - True – текущий пользователь Windows
    - False – SQL Server-аутентификация
  - UserID, Password
- Класс SqlConnectionStringBuilder
  - Программное построение строки

SqlCommand

- Команда для работы с БД
- Основные свойства
  - CommandText – SQL-код, составляющий тело команды
  - CommandType – команда, таблица, хранимая процедура
  - Parameters – параметры команды

SqlCommand - методы

- Выполнение команды на SQL сервере
  - ExecuteReader() – создает курсор на чтение из БД (SqlDataReader)
  - ExecuteScalar() – позволяет получить скалярное значение
  - ExecuteNonQuery() – команды, не требующие чтения данных
- Prepare() – оптимизация команды (занимает время)
- Асинхронные версии команд

SqlDataReader

- Последовательное чтение результатов запроса
- **bool** Read() – следующая строка
- **object this**[**int** i], **object this**[**string** name] – получение значения поля
- **int** GetOrdinal(**string** name) – номер столбца по имени
- GetXXX(**int** i) – получение типизированного значения столбца (осторожно с null). XXX – название простого типа
  - int reader.GetInt32(int order)
- Close(), Dispose() – закрытие DataReader

Пример

```
SqlConnection connection = new SqlConnection(“Data Source=(local);Initial Catalog=Northwind;Integrated Security=True”);
connection.Open();
SqlCommand command = new SqlCommand(@"SELECT OrderID, OrderDate, Freight, ShipAddress FROM Orders, connection);
        List<Order> orderList = new List<Order>();
        using (SqlDataReader reader = command.ExecuteReader())
        {
            while (reader.Read())
            {
                Order order = new Order();
                order.Id = reader.GetInt32(0);
                order.OrderDate = (reader["OrderDate"] is DBNull) ? null : (DateTime?)reader["OrderDate"];
                order.Freight = (reader["Freight"] is DBNull) ? null : (double?)Convert.ToDouble(reader["Freight"]);
                order.ShipAddress = Convert.ToString(reader["ShipAddress"]);
                orderList.Add(order);
            }
connection.Close();
```

Получение данных из БД

Демонстрация

Параметры

- Задает параметр команды
- SqlCommand.Parameters – коллекция параметров команды
  - SqlCommand.Parameters[name] – получение параметра по имени
- Имя параметра в команде начинается с @ (в MS SQL Server):
  - SELECT name, author FROM books WHERE author=@author

SqlParameter

- Представляет именованный параметр
- Конструкторы
  - SqlParameter(string, object) – со значением
  - SqlParameter(string, SqlDbType) – с заданным типом
- Свойства
  - object SqlParameter.Value – значение параметра
- Пример:
  - SqlParameter author = new SqlParameter(“@author”, “Lenin”);
  - SqlCommand.Parameters.Add(author);

Использование параметров

Демонстрация

Параметры vs. Текст

- Динамическое формирование:
  - cmd.CommandText = “SELECT * FROM books” + “WHERE author=\”” + author + “\””;
- Недостатки
  - Только для простых типов параметров – не работает с image, text, binary
  - Меньше возможностей оптимизации
  - Подверженность атакам с внедрением SQL-кода (SQL Injection)

SQL Injection

Демонстрация

Транзакции

- Обеспечиваю поддержку транзакций. ACID
  - Атомарность
    - Если в операцию вовлечены несколько порций данных, то фиксируются либо все, либо ни одна
  - Согласованность (Целостность)
    - Данные после завершения транзакции находятся в согласованном состоянии
  - Изоляция
    - Транзакция оперирует изолированными данными. Пока транзакция не завершена частичные данные никому не видны
  - Длительность
    - После завершения транзакции данные сохраняются. Даже в случае возникновения сбоев, сохраненные данные восстановятся

Транзакции

- Пространство имен System.Transactions
- Создание транзакции

  using (TransactionScope scope = new TransactionScope()) { ….}
- Если внутри скопа произошло исключение, транзакция будет откатана
- Для комита транзакции необходимо вызвать метод Complete()
- При обращении к нескольким серверам транзакция автоматически будет изменена на распределенную транзакцию и передана на управления в координатор распределенных транзакций (DTC)

```
using (TransactionScope scope = new TransactionScope())
        {
            SqlConnection connection1 = new SqlConnection(connectionString1);
            ….
            SqlConnection connection2 = new SqlConnection(connectionString2);
            …
            scope.Complete();
        }
```

Транзакция

Демонстрация

Строка подключения в конфигурационном файле

- XML файл настроек приложения
- Позволяет менять настройки без перекомпиляции приложения
- В Visual Studio – файл App.config
- При компиляции переименовывается в файл виде ИмяПрилржения.exe.config

- Пример:

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <connectionStrings>
    <add name="sqlProvider" connectionString="Data Source=192.168.1.1;Initial
        Catalog=Northwind;Integrated Security=True"/>
  </connectionStrings>
</configuration>
```

- Доступ к строке соединения из кода:

```
ConnectionStringSettings connectionString =
        ConfigurationManager.ConnectionStrings["sqlProvider"];
SqlConnection connection =
        new SqlConnection(connectionString.ConnectionString);
```

Строка подключения

Демонстрация