Buổi 6

Lớp DTO – Đối tượng vận chuyển dữ liệu

Trong khuôn khổ tài liệu buổi 6, chúng ta sử dụng CSDL WebBanHang:

- File *[WebBanHang] Mo ta CSDL.png*: Lược đồ mô tả cấu trúc bảng, khóa ngoại.
- File *[WebBanHang] CSDL.sql*: Phát sinh CSDL.

I. Lớp DTO

1. Khái niệm

Trong quá trình xây dựng ứng dụng với mô hình 3 lớp, việc trao đổi dữ liệu giữa các lớp xảy ra rất thường xuyên thông qua việc gọi phương thức, truyền tham số và trả về giá trị, ví dụ:

- Khi người dùng đăng kí tài khoản, GUI phải gửi xuống BUS toàn bộ thông tin của tài khoản, gồm có Tên tài khoản, Mật khẩu, Email, SĐT, Địa chỉ, Họ tên… Sau đó, BUS lại phải gửi xuống DAO toàn bộ những thông tin này.
- Khi cần lấy thông tin của 1 tài khoản nào đó, DAO phải gửi toàn bộ thông tin lên BUS và BUS sẽ gửi lên GUI dưới dạng giá trị trả về của phương thức. Do phương thức chỉ có thể trả về 1 giá trị nên hiện tại chúng ta phải trả về ở dạng `DataRow`.

Việc truyền nhiều tham số cho phương thức sẽ khiến cho mã nguồn trở nên dài dòng và gây khó khăn, bất tiện cho lập trình viên. Việc cần trả về nhiều giá trị mà phải trả về ở dạng DataRow sẽ gây bất tiện trong việc đọc hiểu code.

Để giải quyết vấn đề này, chúng ta sẽ sử dụng lớp DTO.

Lớp DTO – *Data Transfer Object* chịu trách nhiệm vận chuyển dữ liệu giữa các lớp trong mô hình 3 lớp. Bản thân lớp DTO không phải là 1 thành phần của mô hình 3 lớp, nó chỉ có nhiệm vụ lưu trữ nhiều dữ liệu thành 1 đối tượng chứa nhiều thuộc tính.

2. Xây dựng lớp DTO

Tạo project DTO với kiểu *Class Library*. Sau đó, trong cả 3 lớp GUI, BUS và DAO, cần thêm reference tới lớp DTO, vì cả 3 lớp này đều có thể dùng đến lớp DTO.

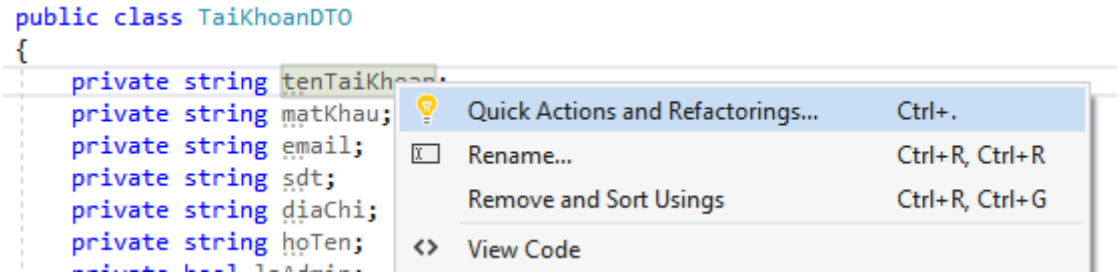
Lớp DTO bao gồm nhiều lớp đối tượng khác nhau, mỗi lớp đối tượng mô phỏng lại cấu trúc của 1 bảng trong CSDL, dùng để lưu trữ thông tin của 1 dòng trong bảng đó.

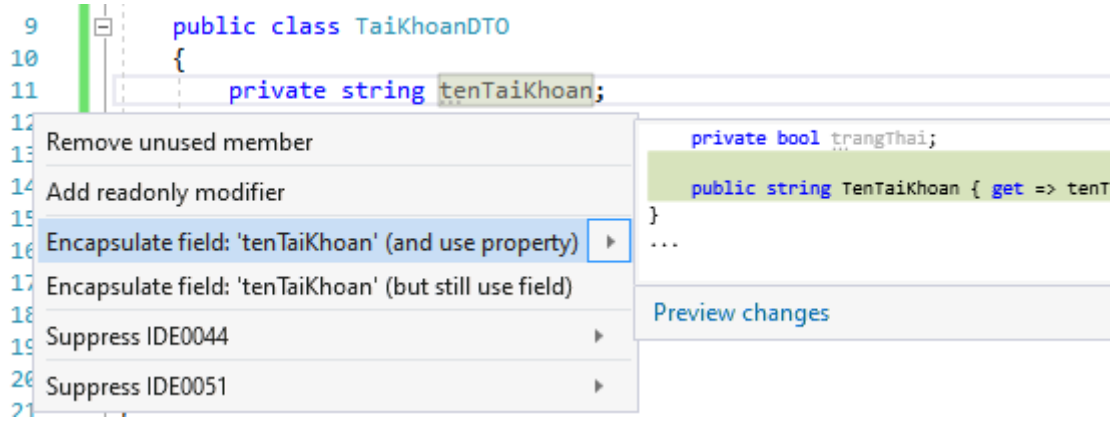
Ví dụ: Lớp đối tượng `TaiKhoanDTO` mô phỏng lại cấu trúc bảng TaiKhoan:

```
public class TaiKhoanDTO
{
    private string tenTaiKhoan;
    private string matKhau;
    private string email;
    private string sdt;
    private string diaChi;
    private string hoTen;
    private bool laAdmin;
    private string anhDaiDien;
    private bool trangThai;
}
```

Lưu ý: Tất cả các thuộc tính đặt tên bắt đầu bằng chữ viết thường, tầm vực là `private`. **Khuyến cáo** đặt tên thuộc tính của lớp DTO giống như tên thuộc tính của bảng tương ứng.

Do thuộc tính chỉ có tầm vực là `private` nên cần tạo property với tầm vực `public` để có thể truy cập vào các thuộc tính khi ở bên ngoài lớp. Để tạo property cho 1 thuộc tính, click chuột phải vào tên từng thuộc tính, chọn *Quick Actions and Refactorings…* → *Encapsulate field: '…' (and use property)* (phím tắt *Ctrl + R, E*). Có thể chọn nhiều thuộc tính cùng lúc để tiết kiệm thời gian:





Cuối cùng, cài đặt các phương thức khởi tạo (nếu cần). Trong ví dụ này, chúng ta sẽ cài đặt phương thức khởi tạo mặc định để quy định một số giá trị mặc định cho đối tượng `TaiKhoanDTO`:

```
public TaiKhoanDTO()
{
        LaAdmin = false;
        AnhDaiDien = "";
        TrangThai = true;
}
```

II. Sử dụng lớp DTO

Sau khi cài đặt lớp đối tượng `TaiKhoanDTO`, nếu cần thao tác với bảng TaiKhoan và trao đổi dữ liệu giữa các lớp trong mô hình 3 lớp, chúng ta sẽ sử dụng lớp đối tượng DTO này.

1. Chức năng đăng kí

Code mẫu cho chức năng đăng kí đã có ở buổi 4. Chúng ta sẽ thay đổi lại mã nguồn để sử dụng thêm lớp DTO.

Tại lớp DAO, sửa lại phương thức `ThemTK()` như sau:

```
public static bool ThemTK(TaiKhoanDTO tk)
{
        string query = "INSERT INTO TaiKhoan (TenTaiKhoan, MatKhau, Email,
    SDT, DiaChi, HoTen, LaAdmin, AnhDaiDien, TrangThai) VALUES
    (@TenTaiKhoan, @MatKhau, @Email, @SDT, @DiaChi, @HoTen, @LaAdmin,
    @AnhDaiDien, @TrangThai)";
        SqlParameter[] param = new SqlParameter[9];
        param[0] = new SqlParameter("@TenTaiKhoan", tk.TenTaiKhoan);
```

```
            param[1] = new SqlParameter("@MatKhau", tk.MatKhau);
            param[2] = new SqlParameter("@Email", tk.Email);
            param[3] = new SqlParameter("@SDT", tk.SDT);
            param[4] = new SqlParameter("@DiaChi", tk.DiaChi);
            param[5] = new SqlParameter("@HoTen", tk.HoTen);
            param[6] = new SqlParameter("@LaAdmin", tk.LaAdmin);
            param[7] = new SqlParameter("@AnhDaiDien", tk.AnhDaiDien);
            param[8] = new SqlParameter("@TrangThai", tk.TrangThai);
            return DataProvider.ExecuteInsertQuery(query, param) == 1;
      }
```

Tại lớp BUS, sửa lại phương thức `DangKi()` như sau:

```
      public static bool DangKi(TaiKhoanDTO tk)
      {
            if (TaiKhoanDAO.KTTKTonTai(tk.TenTaiKhoan))
            {
                  return false;
            }
            else
            {
                  return TaiKhoanDAO.ThemTK(tk);
            }
      }
```

Ở lớp GUI, sửa lại sự kiện `Click` của nút Đăng kí như sau:

```
      protected void btnDangKi_Click(object sender, EventArgs e)
      {
            TaiKhoanDTO tk = new TaiKhoanDTO();
            tk.TenTaiKhoan = txtTenTK.Text;
            tk.MatKhau = txtMK.Text;
            tk.Email = txtEmail.Text;
            tk.SDT = txtSDT.Text;
            tk.DiaChi = txtDiaChi.Text;
            tk.HoTen = txtHoTen.Text;
            if (TaiKhoanBUS.DangKi(tk))
            {
                  // Đăng kí thành công
            }
            else
            {
                  // Đăng kí thất bại
            }
      }
```

2. Chức năng load danh sách tài khoản

Bên cạnh việc nhận nguồn dữ liệu từ `DataTable`, GridView còn có thể nhận dữ liệu từ 1 danh sách các đối tượng thuộc lớp DTO, tức là kiểu `List<…DTO>`[1].

[1] Xem thêm về List<> tại link: https://www.tutorialsteacher.com/csharp/csharp-list

Tại lớp DAO, cài đặt phương thức LayDSTaiKhoan() như sau:

```
public static List<TaiKhoanDTO> LayDSTaiKhoan()
{
    string query = "SELECT * FROM TaiKhoan";
    SqlParameter[] param = new SqlParameter[0];
    DataTable dtbTaiKhoan = DataProvider.ExecuteSelectQuery(query, param);
    List<TaiKhoanDTO> lstTaiKhoan = new List<TaiKhoanDTO>();
    foreach (DataRow dr in dtbTaiKhoan.Rows)
    {
        lstTaiKhoan.Add(ConvertToDTO(dr));
    }
    return lstTaiKhoan;
}
```

Do phương thức ExecuteSelectQuery() của lớp đối tượng DataProvider trả về kết quả ở dạng DataTable, nên để chuyển sang List<TaiKhoanDTO> chúng ta cần có 1 phương thức chuyển đổi 1 DataRow từ bảng TaiKhoan thành 1 đối tượng TaiKhoanDTO:

```
public static TaiKhoanDTO ConvertToDTO(DataRow dr)
{
    TaiKhoanDTO tk = new TaiKhoanDTO();
    tk.TenTaiKhoan = dr["TenTaiKhoan"].ToString();
    tk.MatKhau = dr["MatKhau"].ToString();
    tk.Email = dr["Email"].ToString();
    tk.SDT = dr["SDT"].ToString();
    tk.DiaChi = dr["DiaChi"].ToString();
    tk.HoTen = dr["HoTen"].ToString();
    tk.LaAdmin = Convert.ToBoolean(dr["LaAdmin"]);
    tk.AnhDaiDien = dr["AnhDaiDien"].ToString();
    tk.TrangThai = Convert.ToBoolean(dr["TrangThai"]);
    return tk;
}
```

Tại lớp BUS, cài đặt phương thức LayDSTaiKhoan() như sau:

```
public static List<TaiKhoanDTO> LayDSTaiKhoan()
{
    return TaiKhoanDAO.LayDSTaiKhoan();
}
```

Tại lớp GUI, cài đặt sự kiện Page_Load như sau:

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!Page.IsPostBack)
    {
        grvDSTaiKhoan.DataSource = TaiKhoanBUS.LayDSTaiKhoan();
        grvDSTaiKhoan.DataBind();
    }
}
```

3. Chức năng load thông tin 1 tài khoản

Với GridView, khi muốn sửa thông tin của 1 tài khoản, cần chọn tài khoản đó, sau đó load thông tin lên form.

Tại lớp DAO, cài đặt phương thức LayThongTinTaiKhoan() như sau:

```
public static TaiKhoanDTO LayThongTinTaiKhoan(string tenTK)
{
    string query = "SELECT * FROM TaiKhoan WHERE TenTaiKhoan =
  @TenTaiKhoan";
    SqlParameter[] param = new SqlParameter[1];
    param[0] = new SqlParameter("@TenTaiKhoan", tenTK);
    return ConvertToDTO(DataProvider.ExecuteSelectQuery(query,
  param).Rows[0]);
}
```

Tại lớp BUS, cài đặt phương thức LayThongTinTaiKhoan() như sau:

```
public static TaiKhoanDTO LayThongTinTaiKhoan(string tenTK)
{
    if (!TaiKhoanDAO.KTTKTonTai(tenTK))
    {
        return null;
    }
    else
    {
        return TaiKhoanDAO.LayThongTinTaiKhoan(tenTK);
    }
}
```

Tại lớp GUI, cài đặt sự kiện RowCommand cho GridView như sau:

```
protected void grvDSTaiKhoan_RowCommand(object sender,
  GridViewCommandEventArgs e)
{
    if (e.CommandName == "ChonTK")
    {
        string tenTK = e.CommandArgument.ToString();

        TaiKhoanDTO tk = TaiKhoanBUS.LayThongTinTaiKhoan(tenTK);
        if (tk != null)
        {
            txtTenTaiKhoan.Text = tk.TenTaiKhoan;
            txtMatKhau.Text = tk.MatKhau;
            txtEmail.Text = tk.Email;
            txtSDT.Text = tk.SDT;
            txtDiaChi.Text = tk.DiaChi;
            txtHoTen.Text = tk.HoTen;
            chkLaAdmin.Checked = tk.LaAdmin;
            txtAnhDaiDien.Text = tk.AnhDaiDien;
            chkTrangThai.Checked = tk.TrangThai;
        }
    }
}
```

III. Bài tập

Hoàn thành trang QLTaiKhoan.aspx với các chức năng Xem danh sách tài khoản, Thêm, Sửa, Xóa tài khoản.