

Baik 👉 berikut penjelasan **baris demi baris dan per bagian** dari program

`DragDropManager.cs`.

Program ini adalah bagian dari proyek **MiniPhotoshop**, dan berfungsi untuk **mengelola fitur drag-and-drop (seret dan lepas)** antara gambar utama (*main canvas*) dan kumpulan *thumbnail*.

🔗 Bagian 1: Import Library

```
using MiniPhotoshop.Logic;
using System.Drawing;
using System.Windows.Forms;
using System.Linq;
using System.Collections.Generic;
```

- `using MiniPhotoshop.Logic;` → Mengimpor logika utama aplikasi seperti `ImageEditorService` dan `UIManager`.
 - `using System.Drawing;` → Menyediakan tipe data gambar (`Bitmap`, `Image`, `Point`, `dsb`).
 - `using System.Windows.Forms;` → Menyediakan kontrol GUI seperti `PictureBox`, `ToolStripMenuItem`, `MessageBox`, dan event mouse.
 - `using System.Linq;` → (Diperbaiki) menyediakan fungsi LINQ seperti `Count()`, `Select()`, dll.
 - `using System.Collections.Generic;` → Diperlukan untuk penggunaan `List<>`.
-

🔗 Bagian 2: Deklarasi Kelas

```
namespace MiniPhotoshop.Helpers
{
    public class DragDropManager
```

- `namespace MiniPhotoshop.Helpers` → Menunjukkan bahwa kelas ini berfungsi sebagai *helper utility* dalam aplikasi `MiniPhotoshop`.
 - `public class DragDropManager` → Kelas publik yang bertugas mengatur mekanisme **drag and drop gambar** di antarmuka aplikasi.
-

🔗 Bagian 3: Variabel Kelas (Field)

```
private readonly PictureBox _mainCanvas;
private readonly List<PictureBox> _thumbnails;
private readonly PictureBox _histogram;
private readonly ToolStripMenuItem _viewMenu;

private readonly ImageEditorService _editorService;
private readonly UIManager _toolManager;
```

- `readonly` berarti nilai hanya dapat diinisialisasi melalui konstruktor, tidak bisa diubah lagi setelah itu.

- `_mainCanvas` → area utama untuk menampilkan gambar besar (gambar aktif).
- `_thumbnails` → daftar empat `PictureBox` kecil yang menampilkan gambar mini.
- `_histogram` → area grafik histogram dari gambar (menunjukkan distribusi warna).
- `_viewMenu` → menu tampilan yang diaktifkan ketika gambar utama sudah dimuat.
- `_editorService` → layanan logika editor untuk memanipulasi gambar.
- `_toolManager` → pengelola UI alat editing seperti crop, rotate, dsb.

Bagian 4: Konstruktor

```
public DragDropManager(
    PictureBox mainCanvas, List<PictureBox> thumbnails, PictureBox histogram,
    ImageEditorService editorService, UIManager toolManager,
    ToolStripMenuItem viewMenu)
```

- Konstruktor menerima semua objek GUI dan logika yang dibutuhkan, lalu menyimpannya ke variabel kelas.
- Tujuannya agar kelas ini dapat mengakses dan mengontrol komponen-komponen tersebut dari satu tempat.

Bagian 5: Registrasi Event

```
public void RegisterDragDropEvents()
{
    _mainCanvas.DragEnter += OnCanvas_DragEnter;
    _mainCanvas.DragDrop += OnCanvas_DragDrop;
    _mainCanvas.MouseDown += OnCanvas_MouseDown;

    foreach (var thumbnail in _thumbnails)
    {
        thumbnail.MouseDown += OnThumbnail_MouseDown;
        thumbnail.DragEnter += OnThumbnail_DragEnter;
        thumbnail.DragDrop += OnThumbnail_DragDrop;
    }
}
```

- Metode ini mendaftarkan semua *event handler* untuk `PictureBox` utama dan daftar *thumbnail*.
- `+=` berarti menambahkan fungsi event handler untuk merespons aksi pengguna:
 - `DragEnter` → saat objek diseret ke atas area.
 - `DragDrop` → saat objek dilepas di area tersebut.
 - `MouseDown` → saat tombol mouse ditekan (memulai drag).

Bagian 6: Fungsi Menambah Gambar ke Thumbnail

```
public void LoadImageToThumbnail(Bitmap image)
```

- Menambahkan gambar baru ke daftar *thumbnail*.
 - Jika slot penuh (sudah 4 gambar), menampilkan pesan peringatan:
 - `if (_thumbnails[3].Image != null)`
 - `{`
 - `MessageBox.Show("Slot thumbnail sudah penuh ...");`
 - `return;`
 - `}`
 - Jika belum penuh, gambar-gambar lama digeser satu posisi ke kanan:
 - `for (int i = _thumbnails.Count - 2; i >= 0; i--)`
 - `{`
 - `_thumbnails[i + 1].Image = _thumbnails[i].Image;`
 - `}`
 - Gambar baru dimasukkan di posisi pertama (kiri atas):
 - `_thumbnails[0].Image = image;`
-

Bagian 7: Event Handler Drag & Drop

Bagian ini menangani seluruh proses **klik, seret, dan lepas gambar** antara *thumbnail* dan *main canvas*.



OnThumbnail_MouseDown

```
private void OnThumbnail_MouseDown(object sender, MouseEventArgs e)
```

- Dijalankan saat pengguna mengklik *thumbnail*.
 - Jika gambar kosong atau klik bukan kiri → keluar.
 - Membuat objek `DataObject` yang membawa dua data:
 - Gambar bitmap (`DataFormats.Bitmap`)
 - Sumber asalnya (`"SourceThumbnail"`)
 - Lalu memulai operasi drag:
 - `clickedThumbnail.DoDragDrop(dragData, DragDropEffects.Copy);`
-



OnCanvas_MouseDown

```
private void OnCanvas_MouseDown(object sender, MouseEventArgs e)
```

- Sama seperti di atas tapi berlaku untuk area utama.
 - Mengecek apakah gambar utama ada dan bukan dalam mode menggambar (`Cursor == Cursors.Cross`).
 - Membuat salinan gambar dan memulai proses `DoDragDrop()`.
-



OnCanvas_DragEnter

```
private void OnCanvas_DragEnter(object sender, DragEventArgs e)
```

- Mengecek apakah data yang diseret adalah gambar (`DataFormats.Bitmap`).
 - Jika iya, ubah kursor menjadi `Copy` (boleh dilepas).
 - Jika tidak, ubah efek menjadi `None`.
-



OnCanvas_DragDrop

```
private void OnCanvas_DragDrop(object sender, DragEventArgs e)
```

- Dijalankan saat gambar dilepas di area utama.
 - Mengambil gambar yang dijatuhkan:
 - `Bitmap droppedImage = (Bitmap)e.Data.GetData(DataFormats.Bitmap);`
 - Memanggil layanan editor untuk menyiapkan gambar (`_editorService.InitializeImage()`).
 - Menampilkan gambar di canvas dan mengaktifkan semua alat edit.
 - Jika gambar berasal dari thumbnail, maka thumbnail sumber dihapus:
 - `if (e.Data.GetDataPresent("SourceThumbnail"))`
 - `{`
 - `var sourceThumbnail =`
 - `(PictureBox)e.Data.GetData("SourceThumbnail");`
 - `sourceThumbnail.Image = null;`
 - `}`
-



OnThumbnail_DragEnter

```
private void OnThumbnail_DragEnter(object sender, DragEventArgs e)
```

- Sama seperti `OnCanvas_DragEnter`, tapi berlaku untuk thumbnail.
 - Mengecek apakah yang diseret adalah gambar dan memberikan efek `Copy`.
-



OnThumbnail_DragDrop

```
private void OnThumbnail_DragDrop(object sender, DragEventArgs e)
```

- Terjadi ketika gambar dilepas di atas thumbnail.
- Mengambil gambar dari data `e.Data`.
- Jika sumber gambar berasal dari *thumbnail lain*:
 - Gambar lama dihapus dari sumber.

- Thumbnail tujuan diisi dengan gambar baru.
 - Jika sumbernya dari *canvas utama*:
 - Thumbnail diisi gambar.
 - Canvas, histogram, dan alat edit dikosongkan karena tidak ada gambar aktif.
-

Bagian 8: Fungsi Tambahan — Hitung Thumbnail Aktif

```
public int GetLoadedThumbnailCount()
{
    int count = 0;
    foreach (var tb in _thumbnails)
    {
        if (tb.Image != null)
        {
            count++;
        }
    }
    return count;
}
```

- Menghitung berapa banyak *thumbnail* yang saat ini berisi gambar (tidak kosong).
 - Berguna untuk validasi atau pembatasan jumlah gambar.
-

Kesimpulan Umum

Kelas **DragDropManager** ini:

- Mengatur interaksi *drag and drop* antara **gambar utama** dan **daftar thumbnail**.
 - Memungkinkan pengguna:
 - Menyeret gambar dari thumbnail ke canvas (untuk diedit).
 - Menyeret gambar dari canvas ke thumbnail (untuk disimpan/arsip).
 - Memindahkan gambar antar-thumbnail.
 - Juga mengelola keadaan aplikasi seperti *histogram*, *tool manager*, dan *menu tampilan* agar selalu sinkron.
-