

Berikut adalah **penjelasan lengkap baris demi baris** dari program `HistogramDrawer.cs` 📌

```
using System.Drawing.Drawing2D;
```

➡ Mengimpor namespace `System.Drawing.Drawing2D` yang berisi kelas-kelas untuk menggambar grafik 2D tingkat lanjut (misalnya `SmoothingMode`, `GraphicsPath`, dll.).

```
using System.Drawing;
```

➡ Mengimpor namespace `System.Drawing` yang digunakan untuk memanipulasi gambar, warna, font, dan grafik seperti `Bitmap`, `Graphics`, `Pen`, dan `Brush`.

```
using System.Linq;
```

➡ Mengimpor namespace `System.Linq` agar bisa menggunakan metode LINQ seperti `.Max()` untuk mencari nilai maksimum dalam array histogram.

```
namespace MiniPhotoshop.Logic.Histogram
```

➡ Menetapkan namespace tempat kelas ini berada, yaitu `MiniPhotoshop.Logic.Histogram`, yang menandakan bahwa kelas ini merupakan bagian dari logika perhitungan dan visualisasi histogram pada aplikasi `MiniPhotoshop`.

```
public static class HistogramDrawer
```

➡ Mendeklarasikan kelas `HistogramDrawer` sebagai **static**, artinya tidak perlu membuat objek untuk menggunakannya. Kelas ini berfungsi untuk **menggambar (menampilkan)** histogram dalam bentuk gambar (`Bitmap`).

```
// Menggambar histogram ke Bitmap baru
```

➡ Komentar yang menjelaskan bahwa metode di bawah akan menggambar histogram ke dalam objek gambar (`Bitmap`).

```
public static Bitmap Draw(int[] hist, int channel, int width, int height)
```

➡ Mendefinisikan metode `Draw()` yang:

- Menerima array `hist` berisi data histogram (panjang 256),
 - Parameter `channel` untuk menentukan warna garis histogram (merah, hijau, biru, atau putih),
 - `width` dan `height` sebagai ukuran gambar yang akan dihasilkan.
- Metode ini mengembalikan objek `Bitmap` yang berisi hasil visualisasi histogram.

```
if (hist == null || hist.Length != 256) return new Bitmap(width, height);
```

➡ Mengecek apakah array `hist` kosong (`null`) atau panjangnya tidak sama dengan 256. Jika iya, maka fungsi langsung mengembalikan gambar kosong berukuran `width × height`.

```
Bitmap histBmp = new Bitmap(width, height);
```

➡ Membuat objek gambar baru (`Bitmap`) dengan ukuran sesuai parameter untuk menggambar histogram di dalamnya.

```
using (Graphics g = Graphics.FromImage(histBmp))
```

➡ Membuat objek `Graphics` dari `histBmp` untuk memungkinkan proses menggambar di dalam gambar bitmap tersebut. `using` memastikan resource dilepaskan otomatis setelah digunakan.

```
g.Clear(Color.Black);
```

➡ Mengisi seluruh area gambar dengan warna hitam sebagai latar belakang histogram.

```
g.SmoothingMode = SmoothingMode.AntiAlias;
```

➡ Mengaktifkan **anti-aliasing** agar garis dan teks yang digambar terlihat halus dan tidak bergerigi.

```
Pen axisPen = new Pen(Color.White, 1);  
Brush textBrush = Brushes.White;  
Font font = new Font("Arial", 8);
```

➡ Mendefinisikan alat gambar:

- `axisPen`: pena putih untuk menggambar garis sumbu,
- `textBrush`: kuas putih untuk menulis teks label,
- `font`: font “Arial” berukuran kecil (8pt) untuk menulis angka label pada sumbu.

```
int marginLeft = 60;  
int marginBottom = 30;
```

➡ Menentukan jarak tepi kiri dan bawah dari area gambar agar ada ruang untuk label dan sumbu.

```
g.DrawLine(axisPen, marginLeft, height - marginBottom, width - 10, height -  
marginBottom); // Sumbu X  
g.DrawLine(axisPen, marginLeft, 10, marginLeft, height - marginBottom); //  
Sumbu Y
```

➡ Menggambar sumbu X dan Y (garis putih):

- Sumbu X berada di bawah,
- Sumbu Y berada di sisi kiri.

```
int xAxisLength = width - marginLeft - 10;
```

➡ Menghitung panjang sumbu X dari margin kiri hingga tepi kanan (dikurangi margin kecil).

```
int labelCountX = 5;  
for (int i = 0; i <= labelCountX; i++)  
{  
    int val = i * 255 / labelCountX;  
    float xPos = marginLeft + (val / 255f) * xAxisLength;  
    float yPos = height - marginBottom;  
  
    g.DrawLine(Pens.White, xPos, yPos, xPos, yPos + 5);  
  
    string label = val.ToString();  
    SizeF size = g.MeasureString(label, font);  
    g.DrawString(label, font, textBrush, xPos - size.Width / 2, yPos + 5);  
}
```

➡ Bagian ini menggambar **label pada sumbu X (nilai intensitas 0–255)**:

- Loop 6 kali (0–5) agar label terbagi rata (misal 0, 51, 102, 153, 204, 255),
- Menggambar tanda kecil (tick) dan menuliskan angka di bawahnya.

```
int max = hist.Max();
if (max == 0) max = 1;
```

➡ Menentukan nilai tertinggi dari histogram untuk digunakan dalam **skala vertikal (Y)**.
Jika semua nilai nol, diset menjadi 1 agar tidak terjadi pembagian nol.

```
int labelCountY = 5;
int labelMarginRight = 10;
for (int i = 0; i <= labelCountY; i++)
{
    int val = i * max / labelCountY;
    float yPos = height - marginBottom - (val / (float)max) * (height - 40);

    g.DrawLine(Pens.White, marginLeft, yPos, marginLeft - 5, yPos);

    string label = val.ToString();
    SizeF size = g.MeasureString(label, font);
    g.DrawString(label, font, textBrush, marginLeft - labelMarginRight -
size.Width, yPos - size.Height / 2);
}
```

➡ Bagian ini menggambar **label pada sumbu Y (frekuensi histogram)**:

- Menentukan 5 titik label,
 - Menggambar garis kecil di tiap titik (tick),
 - Menulis nilai jumlah kemunculan piksel (0 hingga nilai maksimum).
-

```
Pen pen;
switch (channel)
{
    case 0: pen = Pens.Red; break;
    case 1: pen = Pens.Lime; break;
    case 2: pen = Pens.Blue; break;
    case 3: pen = Pens.White; break;
    default: pen = Pens.Gray; break;
}
```

➡ Menentukan warna garis histogram berdasarkan channel yang dipilih:

- Merah = channel R
- Hijau = channel G
- Biru = channel B
- Putih = grayscale
- Abu-abu = default

```
PointF[] points = new PointF[256];
for (int i = 0; i < 256; i++)
{
    float xPos = marginLeft + (i / 255f) * xAxisLength;
    float yPos = height - marginBottom - ((hist[i] / (float)max) * (height - 40));
    points[i] = new PointF(xPos, yPos);
}
```

➡ Membuat array titik (`PointF[]`) untuk menggambar kurva histogram:

- `xPos` dihitung dari nilai intensitas (0–255),
- `yPos` dihitung dari proporsi nilai histogram (`hist[i]`) terhadap nilai maksimum (`max`),
- Titik-titik ini akan dihubungkan menjadi garis histogram.

```
g.DrawLine(pen, points);
```

➡ Menggambar garis histogram berdasarkan titik-titik (`points`) yang telah dihitung dengan warna sesuai channel.

```
return histBmp;
```

➡ Mengembalikan objek `Bitmap` hasil gambar histogram agar bisa ditampilkan di GUI (misalnya di `PictureBox`).

✿ Kesimpulan

Kelas `HistogramDrawer` berfungsi untuk **mengonversi data histogram numerik menjadi gambar visual**.

Program ini:

1. Membuat bitmap kosong.
2. Menggambar sumbu X dan Y lengkap dengan label.
3. Menentukan warna histogram berdasarkan channel.
4. Menggambar kurva histogram sesuai data `hist`.

Hasil akhirnya adalah citra histogram berwarna yang merepresentasikan distribusi intensitas warna dari suatu gambar.