IlmuKomputer.Com
Ikatlah ilmu dengan menuliskannya

BackgroundWorker pada Windows Aplikasi Menggunakan C#

Junindar, ST, MCPD, MOS, MCT, MVP .NET
junindar@gmail.com
http://junindar.blogspot.com

Abstrak

BackgroundWorker merupakan kelas yang dibuat untuk menjalankan instruksi-instruksi pada Thread yang berbeda. *BackgroundWorker* dirancang untuk memudahkan dalam kebutuhan *user interface* seperti pada *Windows Form* maupun *Windows Presentation Foundation* (WPF). Pada *BackgroundWorker* terdapat beberapa *event handler* sehingga programmer dapat meletakkan baris kode kedalam event handler tersebut. Sehingga mendapatkan output yang diinginkan.

Pendahuluan

BackgroundWorker merupakan kelas yang dibuat untuk menjalankan instruksi-instruksi pada Thread yang berbeda. *BackgroundWorker* dirancang untuk memudahkan dalam kebutuhan *user interface* seperti pada *Windows Form* maupun *Windows Presentation Foundation* (WPF). Pada *BackgroundWorker* terdapat beberapa *event handler* sehingga programmer dapat meletakkan baris kode kedalam event handler tersebut. Sehingga mendapatkan output yang diinginkan.

Sebagai contoh *BackgroundWorker* biasa digunakan untuk operasi-operasi yang memerlukan waktu relatif lama. Dan bisa juga kita kombinasikasi dengan control progressbar untuk mendapatkan progress dari operasi yang lagi dilaksanakan. Karena operasi yang dijalankan menggunakan thread yang berbeda, maka user interface tidak akan *freeze* sehingga kita masih dapat memberikan intruksi lain melalui user interface kedalam program. Sebagai contoh, pada saat kita sedang menunggu operasi yang telah berjalan, kita dapat melakukan pembatalan operasi (*cancel*) dengan menggunakan button yang telah kita sediakan.

Isi

Pada latihan ini kita akan membuat sebuah aplikasi untuk mengambil data dari textfile dan akan ditampilkan kedalam ListView. Pada form akan terdapat dua buah button (Start dan Stop), ListView dan ProgressBar.

Untuk memudahkan memahami isi dari artikel ini, kita akan membuat sebuah project latihan, dimana kita akan menggunakan *BackgroundWorker*. Ikuti langkah-langkah dibawah ini.

1. Buat sebuah project dengan nama “Latihan BackgroundWorker”.
2. Ganti properties pada Form1 menjadi seperti berikut.

   Text = Form BackgroundWorker

   StartPosition = CenterScreen
3. Tambahkan control BackgroundWorker pada form, dan ganti properties nya seperti dibawah.

   WorkerReportProgress = true

   WorkerSupportCancellation = true
4. Tambah sebuah ProgressBar dan Label. Ganti text pada label manjadi “In progress, please wait...”.
5. Selanjutnya tambahkan dua buah button dan 1 buah ListView. Ganti properties kedua button tersebut seperti dibawah.

   Button 1

   Name = btnLoad

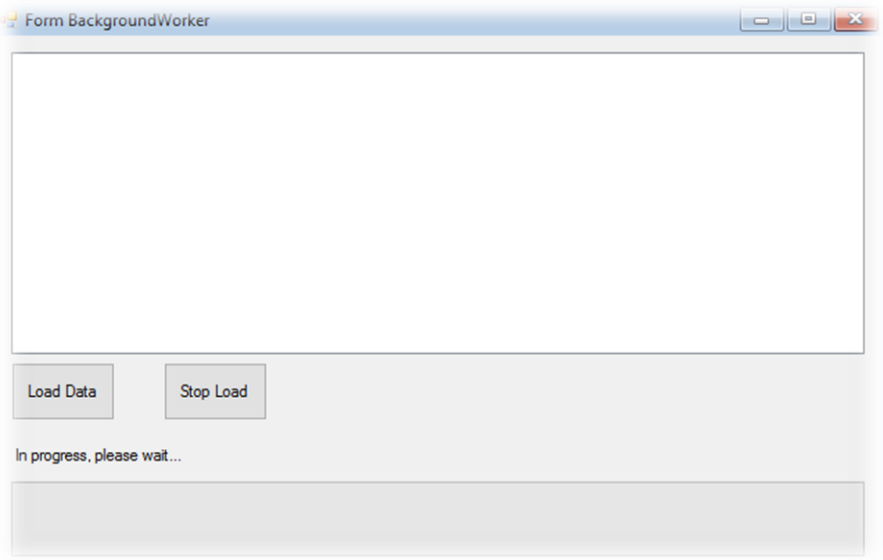
   Text = Load Data

   Button 2

   Name = btnStop

   Text = Stop Load

   Susun control-control tersebut seperti gambar dibawah.



Sebelum kita akan memulai coding, buat terlebih dahulu file txt, dengan format isinya sebagai berikut.

Junindar;junindar@gmail.com;099788888

Arya;arya@yahoo.com;8770900

Simpan file txt tersebut dengan nama data.txt didalam folder debug (Latihan BackGroundWorker\Latihan BackGroundWorker\bin\Debug).

Buka jendela code, lalu ketikkan sintaks dibawah

```
private void SettingListView()
{
        lsvList.View = View.Details;
        lsvList.GridLines = true;
        lsvList.FullRowSelect = true;
        lsvList.Columns.Add("No", 50, HorizontalAlignment.Left);
        lsvList.Columns.Add("Nama", 210, HorizontalAlignment.Left);
        lsvList.Columns.Add("Email", 200, HorizontalAlignment.Left);
        lsvList.Columns.Add("No Telp", 150, HorizontalAlignment.Left);
}
```

Method diatas digunakan untuk melakukan pengaturan pada ListView. Didalam ListView terdapat 4 kolom, dengan View nya diubah menjadi Detail. Lalu panggil method diatas pada event Form_Load seperti code dibawah.

```
private void Form1_Load(object sender, EventArgs e)
{
        SettingListView();
}
```

Selajutnya klik ganda button Load pada form. Button ini berfungsi untuk memulai proses load data dan ditampilkan di dalam ListView.

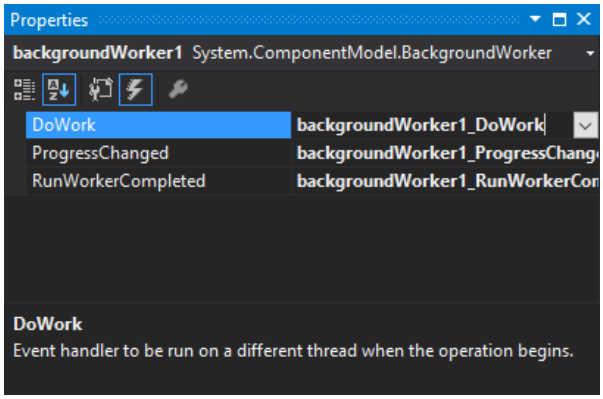
```
        try
        {
           if (backgroundWorker1.IsBusy != true)
           {
              lsvList.Items.Clear();
              backgroundWorker1.RunWorkerAsync();
           }
        }
        catch (Exception ex)
        {

           MessageBox.Show(ex.Message);
        }
```

Dapat dilihat dari sintaks diatas, jika BackgroundWorker tidak “busy” , maka pertama-tama akan membersihkan ListView dari data-data yang ada sebelumnya (lsvList.Items.Clear();) dan selanjutnya dengan dengan menjalankan BackgroundWorker (secara asynchronous).

Perlu diketahui setelah kita menjalankan BackgroundWorker (backgroundWorker1.RunWorkerAsync();), maka proses yang akan dijalankan berada pada event handler “DoWork”.

Selanjutnya buka Events DoWork pada BackgrounWorker. Pada properties BackgroundWorker klik button Events, selanjutnya pada DoWork klik ganda pada cell kosong disebelah DoWork.



Sehingga kita akan masuk kedalam jendela Code dan secara otomatis akan membuat Events DoWork.

```
 private void backgroundWorker1_DoWork(object sender, DoWorkEventArgs e)
{

 }
```

Tambahkan sintaks dibawah pada events diatas.

```
BackgroundWorker worker = sender as BackgroundWorker;
var lineCount = File.ReadLines(Application.StartupPath + ("\\data.txt")).Count();

Invoke(new MethodInvoker(delegate ()
{
    progressBar1.Maximum = lineCount;
}));
```

Setelah mendeklarasikan BackgroundWorker dengan nama “Worker”, langkah selanjutnya adalah mengambil jumlah baris pada file txt yang telah kita buat sebelumnya diatas disimpan pada “lineCount”. Variable lineCount ini berfungsi untuk mengganti value dari properties Maximum pada progressbar. Dikarenakan proses ini dijalankan menggunakan thread yang berbeda maka untuk proses assign value progressbar harus menggunakan MethodInvoker Delegate.

Lanjutkan penambahan sintaks seperti dibawah.

```
var sr = new StreamReader(Application.StartupPath + ("\\data.txt"));
var i = 1;
 while (!sr.EndOfStream)
 {
    if (worker.CancellationPending)
    {
       e.Cancel = true;
       break;
    }
    var lineText = sr.ReadLine().Split(';');
    Invoke(new MethodInvoker(delegate ()
    {
       ListViewItem itm = new ListViewItem(i.ToString());
       itm.SubItems.Add(lineText[0]);
       itm.SubItems.Add(lineText[1]);
       itm.SubItems.Add(lineText[2]);
       lsvList.Items.Add(itm);

    }));
    worker.ReportProgress(i);
    i++;
 }
 sr.Close();
```

Selanjutnya dengan menggunakan StreamReader kita akan mengambil data pada file txt. Selanjutnya variable “i” akan digunakan untuk menampung nilai dari jumlah baris pada file.

Dengan menggunakan fungsi while kita akan melakukan pengulangan untuk membaca file dengan cara membaca baris perbaris (while (!sr.EndOfStream)). Dan sebelum membuat sintaks membaca baris (sr.ReadLine()) terlebih dahulu kita akan melakukan pengecekan apakah BackgroundWorker telah di lakukan proses Cancel, jika iya maka proses akan berhenti. Dan jika tidak akan di lanjutkan proses membaca file perbaris.

Pada file txt yang dibuat sebelumnya dapat kita lihat kita menggunakan “;” (titik koma) sebagai pemisah antara data. Jadi untuk mendapatkan data-data tersebut dan dimasukkan kedalam listview sesuai dengan kolom-kolom nya maka kita akan menggunakan fungsi split. Yang nanti nya output yang dihasilkkan berupa array. Dan

dengan menggunakan MethodInvoker Delegate kita akan memasukkan data-data tersebut kedalam ListView.

worker.ReportProgress(i); digunakan untuk mengantarkan value yang nanti nya akan kita gunakan didalam events ProgressChanged.

Selanjutnya buat events ProgressChanged pada BackgroundWorker. Dan ketikkan sintaks dibawah.

```
labelMessage.Text = @"In progress, please wait... ";
progressBar1.Value = e.ProgressPercentage;
```

Nilai “i” pada ReportProgress (worker.ReportProgress(i);) digunakan untuk mengisi value dari progressbar. Sehingga progressbar akan selalu menampilkan progress terbaru pada antara muka.

Buat events RunWorkerCompleted pada BackgroundWorker. Dan ketikkan sintaks dibawah.

```
if (e.Cancelled)
{
   labelMessage.Text = @"Canceled!";
}
else if (e.Error != null)
{
   MessageBox.Show(@"Error: " + e.Error.Message);
}
else
{
  labelMessage.Text = @"Done!";
}
```

Dapat dilihat ada beberapa kondisi pada sintaks diatas yang hasilnya akan kita tampilkan pada Label. Events ini berfungsi untuk mendapatkan hasil dari BackgroundWorker setelah selesai di eksekusi.

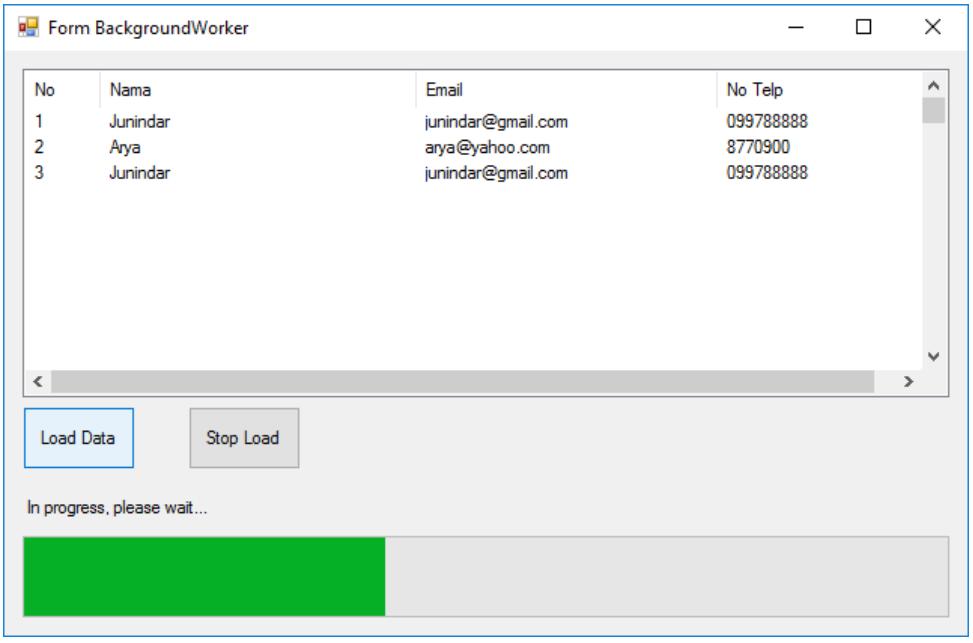
Dan yang terakhir adalah dengan membuat sintaks pada button Stop Load. Dimana button ini untuk melakukan pembatalan proses load yang sebelumnya dijalankan. Dikarenakan BackgroundWorker menggunakan Thread yang berbeda maka kita dapat

menjalankan proses lain selagi proses sebelumnya berjalan. Klik ganda button Stop Load dan ketikkan sintaks dibawah ini.

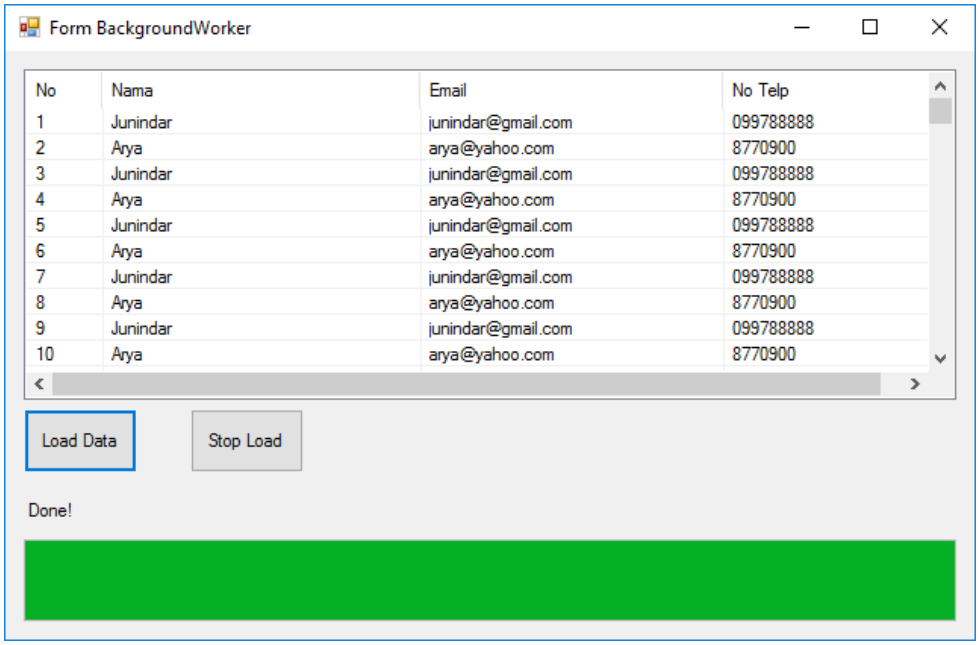
```
if (backgroundWorker1.WorkerSupportsCancellation)
{
      backgroundWorker1.CancelAsync();
 }
```

Pada saat kita menambahkan control BackgroundWorker pada form kita telah mengganti properties BackgroundWorker (WorkerReportsProgress dan WorkerSupportsCancellation). Karena sebelumnya WorkerSupportsCancellation telah kita ubah menjadi true, maka BackgroundWorker ini dapat melakukan pembatan proses yang telah dijalankan dengan menggunakan sintaks ini backgroundWorker1.CancelAsync();

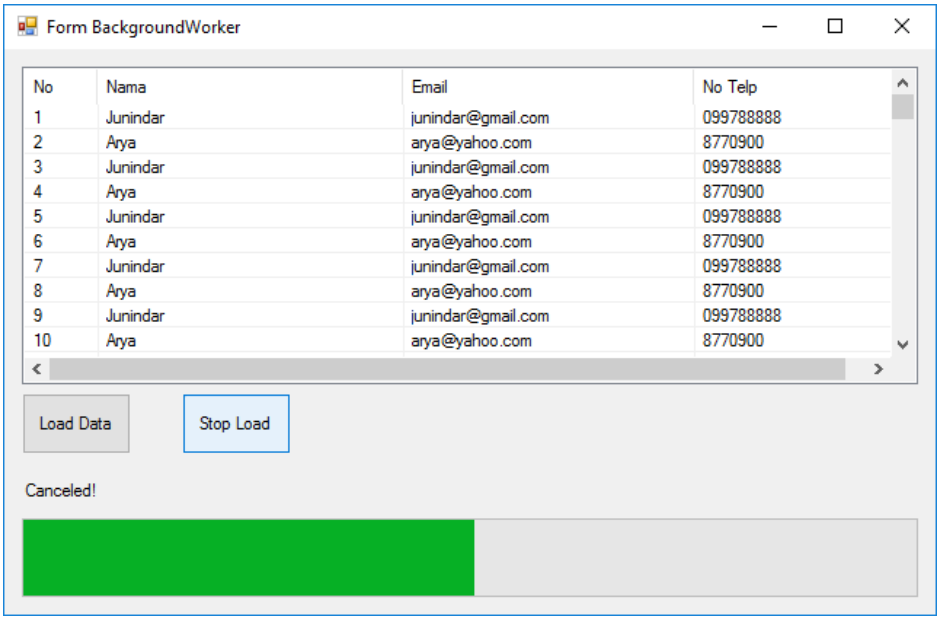
Jalankan Program, lalu klik button Load, sedangkan untuk membatalkan klik button Stop.



Pada saat proses berjalan.



Pada saat proses selesai



Pada saat pembatalan proses.

Penutup

Pada artikel ini telah dijelaskan bagaimana menggunakan BackgroundWorker pada sebuah aplikasi.

Sedangkan untuk memudahkan dalam memahami isi artikel, maka penulis juga menyertakan dengan full source code project latihan ini, dan dapat di download disini

http://junindar.blogspot.co.id/2016/12/backgroundworker-pada-windows-aplikasi.html

Referensi

1. www.msdn.microsoft.com
2. www.planetsourcecode.com
3. www.codeproject.com
4. www.aspnet.com

Masih banyak lagi referensi yang ada di Intenet. Anda tinggal cari di www.Google.com. Dengan kata kunci "**tutorial VB.Net"**

Biografi Penulis.

Junindar Lahir di Tanjung Pinang, 21 Juni 1982. Menyelesaikan Program S1 pada jurusan Teknik Informatika di Sekolah Tinggi Sains dan Teknologi Indonesia (ST-INTEN-Bandung). Junindar mendapatkan Award Microsoft MVP VB pertanggal 1 oktober 2009 hingga saat ini. Senang mengutak-atik computer yang berkaitan dengan bahasa pemrograman. Keahlian, sedikit mengerti beberapa bahasa pemrograman seperti : VB.Net, C#, SharePoint, ASP.NET, VBA. Reporting: Crystal Report dan Report Builder. Database: MS Access, MY SQL dan SQL Server. Simulation / Modeling Packages: Visio Enterprise, Rational Rose dan Power Designer. Dan senang bermain gitar, karena untuk bisa menjadi pemain gitar dan seorang programmer sama-sama membutuhkan seni. Pada saat ini bekerja di salah satu Perusahaan Consulting dan Project Management di Malaysia sebagai Senior Consultant. Memiliki beberapa sertifikasi dari Microsoft yaitu Microsoft Certified Professional Developer (MCPD – SharePoint 2010), MOS (Microsoft Office Specialist) dan MCT (Microsoft Certified Trainer) Mempunyai moto hidup: "**Jauh lebih baik menjadi Orang Bodoh yang giat belajar, dari pada orang Pintar yang tidak pernah mengimplementasikan ilmunya".**