آشنایی با APPSETTINGS

نویسنده:
محمد صافدل
(ققنوس)

مقدمه

AppSetting یکی از امکانات بسیار جالب VS 2005 است. کمتر برنامه ای سراغ داریم که نیاز به ذخیره اطلاعات مورد نیاز برای اجرای مجدد را نداشته باشد. در حقیقت حتی برنامه های ساده ای که می نویسیم هم باید چنین امکانی را داشته باشند. این امکان را Setting می گویند. راههای مختلفی برای این کار وجود دارد که بستگی به نوع برنامه و مسائل دیگری دارد شرایط برنامه آن را تعیین می کند. مثلا یکی از راههای ساده ای که در برنامه های Dos از آن استفاده می شد، استفاده از یک فایل binary با ساختار مشخص بود. در ویندوز اولین راهی که پیشنهاد شد استفاده از فایلهای ini بود و بعد از آن هم که سر و کله Registry پیدا شد! و برای ذخیره کردن اطلاعات برنامه می توانستید از آن استفاده نمایید. وقتی که .NET عرضه شد سر و کله فایلهای XML پیدا شد که پتانسیل زیادی برای ذخیره Setting داشتند. مایکروسافت در نسخه 2003 (.NET Framework 1.1) امکانی به نام App.Config را به پروژه های .NET اضافه کرد که یک فایل XML با ساختار استاندارد برای ذخیره Setting برنامه ها در .NET بشمار می رفت. این کار جالب مایکروسافت با امکاناتی که در .NET 2.0 (VS 2005) به آن اضافه نمود تکمیل شد. این امکانات را در این مقاله بررسی خواهیم کرد.

Settings

در VS 2005 وقتی که یک پروژه جدید از نوع Widows می سازید یک فولدر بطور پیش فرض در پنجره Solution Explorer نمایش داده می شود به نام Properties. اگر این فولدر را باز کنید زیرمجموعه های آن را خواهید دید که عبارتند از AssemblyInfo.cs و Resources.resx و Settings.settings که فعلا با دو مورد اول کاری نداریم و به سراغ Settings.Settings خواهیم رفت.

اگر روی Settings.Settings دبل کلیک کنید Editor مربوط به آن باز خواهد شد. این Editor فقط یک جدول است که مشخصات Setting مورد نظرتان را در آن وارد می نمایید. به عنوان مثال فیلدهای زیر را در آن وارد کنید:

Name: ForeColor

Type: System.Drawing.Color

Scope: User (می تونید Apllication هم انتخاب کنید. در ادامه بحث به تفاوت بین آنها اشاره می کنیم)

Value: Maroon (نکته جالب اینجاست که وقتی شما در قسمت Type، نوع داده مورد نظرتان را انتخاب می کنید قسمت Value متناسب با آن تغییر می کند مثلا در این مثال به محض انتخاب نوع System.Drawing.Color قسمت Value به نوعی تغییر می نماید که شما می توانید رنگ مورد نظرتان را انتخاب کنید.)

هر تغییری که شما در این Editor بدهید معادل آن در فایل Settings.Designer.cs کد مربوطه تولید خواهد شد و همینطور در فایل App.Config هم تغییرات لازم اعمال می شود.

حالا به فرم Form1.cs که بطور پیش فرض به پروژه اضافه شده رفته و یک Label روی آن قرار دهید. متن Label را مطابق نظرتان تغییر دهید. حالا در رویداد Form1_Load کد زیر را بنویسید:

```
label1.ForeColor=Properties.Settings.Default.ForeColor;
```

Namespace :Properties

Settings: کلاس که در پروژه به همین نام ساخته شده است

Default: یک Property از کلاس Settings

ForeColor: همان آیتمی که ما به عنوان تنظیمات به بخش Settings اضافه کردیم حالا از طریق این Property قابل استفاده می باشد.

Scope

اگر شما Scope را روی Application تنظیم کرده باشید هیچ فرقی ندارد که برنامه را در چه User اجرا می کنید ولی اگر مقدار آن را به User تنظیم کرده باشید قضیه کمی فرق می کند یعنی شما می توانید کاری کنید که تنظیمات انجام شده برای هر User بطور مجزا ذخیره شود و هر User که برنامه شما را اجرا کرد با تنظیمات مورد نظر همان کاربر اجرا شود. فقط برای این کار باید قبل از خروج از برنامه تنظیمات جدید را برای کاربر ذخیره نمایید. این کار هم به این شکل قابل انجام می باشد:

```
Properties.Settings.Default.Save();
```

حالا تصور کنید که یک فرم Setting دارید که کاربر در این فرم تنظیمات مورد نظرش را انجام داده و ذخیره می کند بنابراین به ازای هر کاربر تنظیمات ذخیره خواهد شد. ساختن فرم Setting و نوشتن کدهای آن را به عهده خودتان میگذارم.

یکی از فواید مهم استفاده از این روش این است که شما لازم نیست مقادیر ذخیره شده را به چیزی که می خواهید Cast کنید چون این کار در کلاس Settings بطور خودکار و با توجه به نوع داده تعیین شده در Type انجام خواهد شد.