- شماره مقاله : ۲
- سطح مقاله : مبتدي
- عنوان : استفاده از Enumeration ها در سي شارپ
- توسط : Mojgan110 ، http://mojgan110.wordpress.com
- تاريخ : ۲۹ دي ۱۳۸۵
- حرف امروز : به كمك امكاناتي كه داشتي ، از يه سوءتفاهمي كه برات توضيحش دادم بودم و هر آدم عاقلي كه عناد نداشت هم قبولش ميكرد ، يك دروغ بزرگ ساختي (دروغ هرچي بزرگتر ، باورش آسونتر) و واسش مدارك جعل كردي و به خورد تصديق كنندگانت دادي ! باشه ! من تو و حسابت را به خدا واگذار ميكنم ، مرور زمان همه چيز را ثابت ميكنه .

Enumeration ها ، از نوعهای Value ی دات نت هستند و معمولا وقتی که میخواهیم از نامهای سمبولیك برای مقادیر عددی استفاده کنیم ، از enum ها میتوانیم کمك بگیریم.

مثلا این میتونه تعریف یك enum باشه :

```
enum Colorz
{
    Red,
    Green,
    Blue,
    Magneta
};
```

در حالت پیش فرض که در مثال روبرو هم دیده میشه ، مقادیر عددی enum ها از صفر شروع میشه ، یعنی Red مقدار عددی معادلش صفر هست و Green مقدارش یك هست و اینا.
ولی لزومی ندارد که حتما از صفر شروع کنیم. میتوانیم از عدد دلخواه خودمان هم شروع کنیم و همچنین لزومی دارد که این مقادیر ، پشت سر همدیگر باشند بلکه هر عضو enum میتواند عدد خاص خود داشته باشه . مثلا به enum ی که در زیر تعریف شده نگاه کنید :

```
enum Colorz
{
    Red   = 88,
    Green = 77,
    Blue  = 8,
    Magneta
};
```

همینطور که میبینیم ، اعدادی که به هر عضو enum نسبت داده ایم ، به طور دلخواه بوده و عضو Magneta حالا مقدار عددی اش چند میشود ؟ میشود 9 ، چون به طور جداگانه به آن مقداری نسبت نداده بودیم و مقدار عضو قبلی هم که Blue بوده ، 8 هست ، پس Magneta که عضو بعدی هست میشه 9 ! (دهه ! اینها که پنج رقم اول شماره تلفن خونه ما بودند !)

در حالت پیش فرض ، این اعداد صحیحی که enum ها میپذیرند از نوع System.Int32 هستند ولی میتوانیم در جاهایی مثل Pocket PC ها یا مثلا این موبایلهایی که دات نت هم هست روشون و ممکنه که با مشکل کمبود حافظه روبرو بشویم ، از انواع عددی دیگری که کوچکتر هست و فضای کمتری میگیرند استفاده کنیم ، مثلا byte ، در اینصورت تعریف enum را به این صورت تغییر میدهیم :

```
enum Colorz : byte
{
    Red   = 88,
    Green = 77,
    Blue  = 8,
    Magneta
};
```

نحوه فراخوانی و استفاده از عناصر enum ها در برنامه ، به این صورت هست که حتما باید قبل از نام عنصر ، نام enum هم نوشته شود ، مثلا اینطوری : Colorz.Red

Enum ها بنکل از فضای **System.Enum** مشتق میشوند ، پس بهتر هست که نگاهی به System.Enum بیندازیم و با متدهای مهم آن آشنا شویم.

: System.Enum.GetNames , System.Enum.GetName

متد GetName ، معادل عددی اعضای enum را میگیرد و نام سمبولیك آن را بصورت یك متغیر string برمی گرداند.

متد GetNames هم مشابه همین کار قبلی را انجام میدهد با این تفاوت که نام enum را میگیرد و اسامی سمبولیك همه اعضای آن enum را بصورت یك آرایه رشته ای []string برمیگرداند.

: System.Enum.GetValues

این متد ، کارش بر عکس متد GetNames هست ، یعنی وقتی نام enum را بهش بدهیم ، بجای اینکه اسامی سمبولیك را برگرداند ، مقادیر عددی عناصر را برمیگرداند در یك آرایه.
نکته ای که باید به آن توجه کرد این هستش که خروجی این تابع بصورت صعودی مرتب شده است.
یعنی اگر آن را بر روی enum ی که مثال زده ایم اجرا کنیم ، ابتدا 8 را برمیگرداند ، سپس 9 را و سپس 77 و 88 را

: System.Enum.IsDefined

این متد ، یك متغییر رشته ای میگیرد و درصورتیکه آن متغییر ، یکی از نامهای سمبولیك موجود در enum مورد نظر باشد ، مقدار true برمیگرداند.

: System.Enum.Parse

کاری که Parse انجام میدهد این هست که بین معادلهای عددی_ نامهای سمبولیکی که به آن ارسال میکنیم ، OR میگیرد و نتیجه را بصورت Object برگشت میدهد.(اگر فقط یك نام به آن ارسال کنیم ، آنرا با عدد صفر ، OR میگیرد) . حال این آبجکت را میتوانیم به مثلا enum تبدیل (cast) کنیم تا enum جدیدی داشته باشیم ویا مثلا آنرا به اعداد صحیح cast کنیم تا یك مقدار_عددی داشته باشیم .
بدیهی است که اگر فقط یك نام سمبولیك ارسال کنیم و خروجی را هم به اعداد صحیح cast کنیم ، با صفر OR میشود و در واقع مقدار عددی_ آن نام سمبولیك را خواهیم داشت .

نکته دیگر اینکه ،میتوانیم با عناصر enum بصورت فیلدهای بیتی رفتار کنیم ، و از طرفی هم خاصیت [FlagsAttribute] چنین وظیفه ای دارد ولی من هنوز نمیدونم که چرا اگر به enum خاصیت مذکور را ندهیم ، باز هم میتوانیم بصورت بیتی با آن کار کنیم !!!!

نکته دیگر اینکه میتوان اعمالی مثل مقایسه را هم بین عناصر enum انجام داد که در واقع باعث مقایسه مقدار عددی آن عناصر میشود .

در شکل زیر ، مثالهایی از همه چیزهایی که گفته شد و نحوه نوشتن آنها را میبینیم.

```
Console.WriteLine("Name of Folan Onsor: {0}", Enum.GetName(typeof(Colorz), 8));

foreach (string symbolic in System.Enum.GetNames(typeof(Colorz)))
    Console.WriteLine(symbolic);

foreach (byte numeric in System.Enum.GetValues(typeof(Colorz)))
    Console.WriteLine(numeric);

if (System.Enum.IsDefined(typeof(Colorz), "Tintoretto"))
   Console.WriteLine("Tintoretto is defined in Enum");

Colorz color = (Colorz)System.Enum.Parse(typeof(Colorz), "Red, Green");
byte bite = (byte)Enum.Parse(typeof(Colorz), "Red,Green");
Console.WriteLine(color.ToString());
Console.WriteLine(bite.ToString());

Colorz shirt = Colorz.Red;
Colorz shoe = Colorz.Magneta;
Colorz OROfThem = shirt | shoe;
Console.WriteLine(OROfThem);    // 88 OR 9 = 89

if (shirt > shoe)
    Console.WriteLine("Shirt has greater value");
```