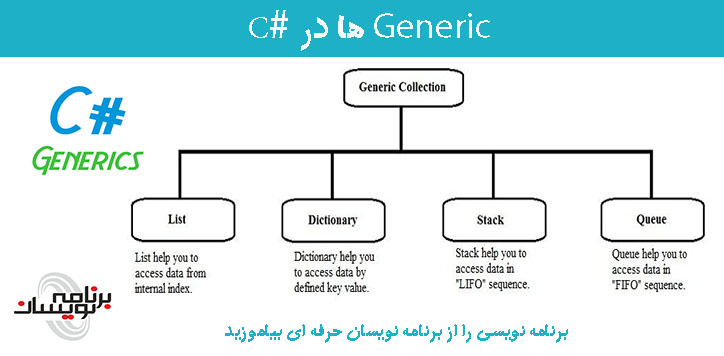
در این مقاله مفهوم Generic ها در سی شارپ ، نحوه ی به کارگیری آن ها، مزایای استفاده از آن ها و نوآوری هایی که می توانیم با استفاده از این نوع داده ها ایجاد کنیم (مانند constrain ها ، متدهای Generic و ارث بری های generic ) را بررسی خواهیم کرد. همچنین درباره ی بهینه سازی Generic ها توسط NET Framework. مطالبی ارائه خواهیم کرد.



Genericها چیستند؟

Generic ها به شما اجازه می دهند ساختار داده های امن بدون وابستگی به داده های حقیقی تعریف کنید. با استفاده و رعایت این موارد، سرعت برنامه و همچنین کیفیت کدها به طرز قابل توجهی افزایش پیدا می کند. زیرا شما الگوریتم های پردازشی مربوط به استفاده مجدد از داده ها را به کار می گیرید بدون این که چندین بار مجبور باشید کدنویسی انجام بدهید.

Generic ها شبیه قالب های C++ هستند اما تفاوت های قابل توجهی در نحوه ی استفاده  و همچنین قابلیت ها دارند.  این مقاله در مورد مشکل فضای آدرس دهی generic ها، به کارگیری آن ها، مزایای مدل های برنامه نویسی و همچنین ابتکاراتی که می توانیم با این نوع داده ها ایجاد کنیم ، بحث می کند. شما همچنین مطالبی در مورد نحوه ی بهینه سازی Generic ها در زمینه های مختلف  .NET Framework (مانند آرایه ها، reflection ، مجموعه ها، سریال سازی ، remoting و ....) نیز خواهید آموخت.

Generic Assembly

Generices  یک مجموعه است که در فضای نام زیر قابل استفاده است.

Generic ها به شما اجازه می دهند تا نوع داده ی استفاده شده در یک متد و یا یک کلاس را زمانی مشخص کنید که آن داده در زمان اجرای برنامه مورد نیاز باشد. (یعنی اجباری به تعریف نوع داده در ابتدای نوشتن کدها نیست و تا زمانی که داده ، وارد روند اجرای پروژه نشده است، امکان تعریف آن برای برنامه نویس وجود دارد.)به عبارت دیگر Generic ها به شما اجازه می دهند کلاس و یا متدی بنویسید که می تواند خودش را با هر نوع داده ای انطباق بدهد.

شما تعریف یک کلاس و یا یک متد را به همراه پارامترهای مورد نیاز برای داده ها می نویسید، زمانی که کامپایلر با یک سازنده  (constructor) مواجه می شود که مربوط به یک کلاس و یا یک تابع فراخوانی است، کد مورد نیاز برای مدیریت داده های موجود در آن ها را تولید می کند. یک مثال ساده برای درک مفهوم این مطلب در زیر آورده شده است:

ایجاد Generic List<T>

Generic List collection مانند یک آرایه است.شما یک لیست تعریف می کنید ، داده های آن را مقدار دهی و پر می کنید و سپس با انجام این امور می توانید به اعضای موجود در آن دسترسی پیدا کنید.

در این قسمت یک نمونه کد از نحوه استفاده از یک لیست قرار داده شده است

Sample Generic List <int>  
List<int> myValue = new List<int>();  
Sample Generic List <string>  
List<string> myValue = new List<string>();

در مثال بالا هر دو کد میتوانند با genericsجایگزین هم شوند.

List<T> myValue = new List<T>();

با استفاده از[T] میتوان به صورت "int" ,"string"یا هر نوع دیگر  ، داده ها را بر اساس نیاز نرم افزار استفاده کرد.

ویژگی Generics

Generic ها ویژگی های بسیار مفیدی دارند که می توانند با استفاده از این ویژگی ها عملکرد عملیات های موجود در برنامه را بهبود ببخشند و قابلیت هایی مانند استفاده مجدد از کد، امنیت و کارایی را برای شما فراهم کنند.

شما میتوانید مجموعه از کلاس های Genericsشده ایجاد کنید.کتابخانه .Netframework  شامل مجموعه از کلاس های Generic  شده است که در فضای نام  System.Collections.Generic قرار دارد  که قبلا راجب آن صحبت کرد ایم.

شما میتوان از مجموعه کلاس های  Generivc شده به جای  System.Collectionsاستفاده کنید.

شما میتوانید خودتان یک Generic interfaces که از  class ,  methods ,  eventsتشکیل شده ایجاد کنید.

شما میتوانید  Generic classes constrained ایجاد کنید که قادر به درسترسی متد های به انواع داده می باشد.

شما می توانید اطلاعات مربوط به انواع داده های generic را در زمان اجرا به وسیله ی reflection   به دست بیاورید.

مزایای استفاده از Generics

Generics در .NET به شما اجازه می دهند چندین بار از کدی که نوشته اید، استفاده کنید. انواع و داده های داخلی می توانند بدون مشکلی در ساختار کد ها تغییر کنند و هیچ وابستگی ای به مقادیر و یا منابع شما نیز نخواهند داشت. شما می توانید یک بار برنامه را بنوسید،آن را تست کنید و آن را با هر نوع داده دیگری که دوست دارید در دفعات بعدی مورد استفاده قرار بدهید. و همه این موارد از امنیت و پشتیبانی کامپایلر نیز برخودار هستند.

کدهای generic شما را مجبور به فشرده سازی کدها و مقادیر و منابع و مقادیر نمی کنند و کارایی را به صورت قابل توجهی افزایش می دهند.

با این نوع داده ها کارایی برنامه ها بسیار بالا رفته است و با استفاده از reference type ها نیز این مقدار چندین برابر می شود.

کدی که در این مقاله برای شما ضمیمه شده است شامل یک برنامه ی micro-benchmark ، که یک پشته را در یک حلقه ی کوتاه اجرا می کند. این برنامه به شما امکان تجربه کار با مقادیر و انواع reference ها را می دهد، همچنین می توانید تعداد دفعات تکرار حلقه را نیزبا توجه به کارایی مورد نیازتان تغییر بدهید.