**C# پیشرفته :**

در این سند نحوه شروع کار با تمرینات پیشرفته C# را شرح خواهیم داد.

سری LINQ :

Language Integrated Query ، LINQ، یک نحو منسجم برای پرس و جو از هر منبع داده ای را فراهم می کند. بیشتر از آن، کامپایلر و کتابخانه‌ها آن نحو سازگار را به پرس‌و‌جوهای بومی برای هر منبع داده متفاوت ترجمه می‌کنند. شما LINQ را یاد می گیرید و می توانید به طور موثر به داده ها از پایگاه داده ها، ذخیره سازی ابری، اشیاء موجود در حافظه، منابع داده JSON یا XML دسترسی داشته باشید. در این مجموعه، نحوه نوشتن پرس و جوهای LINQ را یاد خواهید گرفت. همچنین با مکانیسم‌هایی آشنا می‌شوید که LINQ برای ترجمه کد C# شما به پرس‌و‌جوهای بومی برای منابع داده خاص استفاده می‌کند.

سری Async :

برنامه های مدرن نیاز به پردازش ناهمزمان دارند. می توانید با سرویس ها در فضای ابری یا ماشین های دیگر تماس شبکه برقرار کنید. ممکن است فرآیندهای طولانی مدت را برای تجزیه و تحلیل مجموعه داده های بزرگ شروع کنید. زبان C# دارای ویژگی هایی است که استدلال و درک برنامه های همگام را آسان تر می کند. در این مجموعه، شما با async و wiat و کلاس Task آشنا خواهید شد. شما یاد خواهید گرفت که چگونه این ویژگی ها با هم کار می کنند تا به شما در نوشتن برنامه های ناهمزمان کمک کنند.

این داکیومنت شامل 8 قسمت می باشد که همراه با نمونه کد آن را بررسی می نماییم.

1. مقدمه ای بر (LINQ) :

LINQ مخفف Language-Integrated Query به معنای زبان پرس و جوی یکپارچه است که در دات نت 3.5 معرفی شد و به برنامه نویس اجازه می دهد که داده ها را از هر نوع منبع داده ای بدون نیاز به دانستن یک زبان دیگر، پرس و جو کند. پرس و جو، فرایند به دست آوردن داده از منبع داده است. LINQ پرس و جوی داده ای از منابع داده ای مختلف را بسیار راحت کرده است.

در این آموزش مرحله به مرحله با اصول LINQ آشنا می شوید.

نگاهی کلی بر عملگرهای LINQ :

* عملگرهای محدود کننده کلمه کلیدی where (Restriction operators) : این عملگرها توالی ورودی را برای تولید یک دنباله خروجی محدود یا فیلتر می کنند.
* عملگرهای انتخاب کلمه کلیدی select(Projection operators) : این عملگرها عناصر توالی خروجی را از عناصر ورودی انتخاب می کنند. این عناصر خروجی می توانند یکسان و یا انواع مختلف باشند.
* عملگرهای پارتیشن بندی (Partition operators) : از متدهای Take، Skip، TakeWhile و SkipWhile برای پارتیشن بندی دنباله ورودی می توانید استفاده کنید. می توانید قسمتی از دنباله ورودی را به عنوان دنباله خروجی دریافت کنید.
* عملگرهای مرتب سازی (Ordering operators) : کلمات کلیدی orderby ( به تنهایی برای مرتب سازی نزولی استفاده می شود) و کلمات کلیدی OrderBy, ThenBy, OrderbyDescending,ThenByDescending برای مرتب سازی خروجی داده ها استفاده می شود.
* عملگرهای گروه بندی (Grouping operators) : عملگرهای GroupBy و into یک دنباله را دسته بندی می کنند.
* عملگرهای مجموعه (Set Operators) : عملکردی را برای مقایسه چندین مجموعه داده ارائه می کنند. می توانید تقاطع، اتحاد، همه عناصر متمایز و تفاوت بین مجموعه ها را پیدا کنید. مانند کلمه کلیدی distinct
* عملگرهای تبدیل (Conversion operators) : اگر بخواهید یک مجموعه از نتیجه پرس و جو را به مجموعه دیگری تبدیل کنید از این عملگرها استفاده کنید.
* عملگرهای عنصر (Element operators) : در صورتی که بخواهید عناصر خروجی را بر اساس موقعیت آن عنصر در دنباله دریافت کنید می توانید از این عملگرها استفاده کنید. مانند متد های First, FirstOrDefault, Last, LastOrDefault, ElementAt
* ایجاد توالی (Generate sequences) : متدهای Range و Repeat دنباله هایی از اعداد صحیح را ایجاد می کنند. این ها می توانند دنباله های منبع پرس و جوها باشند.
* تعیین کمیت اعضا (Quantifying members) : متدهای Any و All برای عناصری که با یک شرط مطابقت دارند استفاده می شوند.
* عملگرهای جمع کننده (Aggregator operators) : تعدادی متد وجود دارد که محاسبات مقادیر را در یک دنباله انجام می دهد. مانند متد count و sum و... توجه داشته باشید که برخی از این روش ها مستلزم این هستند که دنباله ورودی از نوع عددی باشد.
* عملگرهای توالی (Sequence operations) : این متدها به جای عناصر یک دنباله، بر روی کل دنباله ها عمل می کنند. آنها توالی ها را مقایسه می کنند یا از ترکیب همه عناصر دنباله های جدیدی ایجاد می کنند. مانند متدهای SequenceEqual, Concat, Combine
* عملیات پیوند (Join operations) : این عملگرها، عملکردهای مشابه عملگرهای SQL Join را انجام می دهند. این ها با هر منبع داده LINQ کار می کنند. مانند متد join

1. فراگیری دستورات LINQ :

در این قسمت تمامی عملگرهای قید شده در قسمت مقدمه به صورت عملی در پروژه CsharpAdvanced اجرا می شود.

1. ترکیب و اجرای پرس و جوی در LINQ :

در این قسمت نیز، ادامه مبحث عملگرهای قید شده در قسمت مقدمه را به صورت عملی در پروژه CsharpAdvanced پیاده سازی کرده ایم.

1. Entity Framework و اتصال به منبع داده از راه دور در LINQ :

در این قسمت نحوه کار LINQ با Entity Framework را برای جستجو در منابع داده SQL می آموزیم.

* ایجاد پروژه
* نصب پکیج Microsoft.EntityFrameworkCore
* نصب پکیج Microsoft.EntityFrameworkCore.SqlServer
* نصب پکیج Microsoft.EntityFrameworkCore.Tools
* ساخت یک فولدر برای مدل های پروژه به اسم Models
* ساخت کلاس های مورد نیاز و تعریف پارامترها .
* ساخت کلاس EFContext که ارث بری شده از کلاس DBContext (این کلاس دارای متدهای تعریف کانکشن و و کوئری هاست) برای نحوه ساختار این کلاس به پروژه معرفی شده مراجعه کنید. ( connectionString به درستی تعریف شود.)
* اجرا دستور Add-Migration “NewDatabase” در قسمت package console manager .
* بعد از اجرا دستور بالا پوشه Migrations ایجاد می شود.
* مرحله بعدی ایجاد پایگاه داده با استفاده از Migrations مرحله قبل است.
* مجدد در قسمت package console manager دستور update-database –verbose اجرا می کنیم. دیتابیس ایجاد می شود.
* برای عملیات CRUD باید ابتدا یک instance از کلاس DBContext ایجاد کنیم.
* برای درج ردیف در پایگاه داده ایجاد شده و جدول مشخص باید از متد SaveChanges استفاده نماییم.
* برای حذف یک ردیف از پایگاه داده ایجاد شده و جدول مشخص باید از متد SaveChanges استفاده نماییم.
* برای ویرایش یک ردیف در پایگاه داده ایجاد شده و جدول مشخص باید از متد SaveChanges استفاده نماییم.
* برای Select از دیتا در پایگاه داده ایجاد شده و جدول کافیست از دیت مورد نظر یک لیست ایجاد کنید.
* تمامی مثال های عملیات CRUD در کلاس Program ، پروژه معرفی شده قابل مشاهده است.

1. مقدمه ای بر Async، Await و Tasks
2. Asynchronous Task Composition
3. رسیدگی به استثنائات ناهمزمان (Handling Asynchronous Exceptions)
4. تشخیص ناهمزمان و چند نخی (Distinguish Asynchronous And Multi-Threading)

در پروژه نمونه CsharpAdvanced تمامی این عملگرهای قید شده در این سند را با فایل های کلاسی هم نام با نوع عملگر و کلمه کلیدی می توانید مشاهده کنید.

<https://github.com/BaharehSmaili/CsharpAdvanced>