|  |
| --- |
| [Nhung vd ve linq va C#](http://forum.aspvn.net/fb.aspx?m=20021) (Phiên bản đầy đủ) |

|  |
| --- |
| [**Diễn Đàn Chính**](http://forum.aspvn.net/default.aspx) **>>** [**[Lập trình - Phát triển dự án với Microsoft .NET]**](http://forum.aspvn.net/tt.aspx?forumid=3) **>>** [**Microsoft .NET**](http://forum.aspvn.net/tt.aspx?forumid=42) |

|  |  |
| --- | --- |
| |  | | --- | | Thân bài | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **dcttrung1975** -> *Nhung vd ve linq va C#* (2/7/2010 11:02:32 AM) | | Trong suot qua trinh lam viec voi LINQ. Toi thay rang neu ko dung la mot dieu rat dang tiec. LINQ  co cac uu diem ve toc do thuc thi ko the nao che vao dau.   static string[] presidents =         {             "Adams", "Arthur", "Buchanan", "Bush", "Carter", "Cleveland",             "Clinton", "Coolidge", "Eisenhower", "Fillmore", "Ford", "Garfield",             "Grant", "Harding", "Harrison", "Hayes", "Hoover", "Jackson",             "Jefferson", "Johnson", "Kennedy", "Lincoln", "Madison", "McKinley",             "Monroe", "Nixon", "Pierce", "Polk", "Reagan", "Roosevelt", "Taft",             "Taylor", "Truman", "Tyler", "Van Buren", "Washington", "Wilson"         };         static string[] strings = { "one", "two", null, "three" };         static string[] numbers = { "0042", "010", "9", "27" };          static void Main(string[] args)         {             //All Prototype Where Every Element Causes the Predicate to Return True             bool all = presidents.All(s => s.Length > 3);             Console.WriteLine(all);              //First Contains Prototype Where No Element Matches the Specified Value             bool contains = presidents.Contains("Rattz");             Console.WriteLine(contains);              //The First Count Prototype             int fcount = presidents.Count();             Console.WriteLine(fcount);              //The Second Count Prototype             int scount = presidents.Count(s => s.StartsWith("J"));             Console.WriteLine(scount);              //An Example of the Second Min Prototype             string minName = presidents.Min();             Console.WriteLine(minName);              //An Example of the Second Max Prototype             string maxName = presidents.Max();             Console.WriteLine(maxName);              //A Query Using the Standard Dot Notation Syntax             IEnumerable<string> sequence = presidents.Where(n => n.Length < 6).Select(n => n);             foreach (string name in sequence)                             Console.WriteLine("{0}", name);                          //The Equivalent Query Using the Query Expression Syntax             IEnumerable<string> sequence1 = from n in presidents where n.Length < 6 select n;             foreach (string name in sequence1)                             Console.WriteLine("{0}", name);              //A Simple LINQ to Objects Query             string president = presidents.Where(p => p.StartsWith("Lin")).First();             Console.WriteLine(president);              //A Trivial Sample Query             IEnumerable<string> items = presidents.Where(p => p.StartsWith("A"));             foreach (string item in items)                 Console.WriteLine(item);              //A Trivial Sample Query with an Intentionally Introduced Exception             IEnumerable<string> items1 = presidents.Where(s => Char.IsLower(s[4]));             Console.WriteLine("After the query.");             foreach (string item in items1)                 Console.WriteLine(item);              // Create an array of ints.             int[] ints = new int[] { 1, 2, 3, 4, 5, 6 };             // Declare our delegate.             Func<int, bool> GreaterThanTwo = i => i > 2;             // Perform the query ... not really. Don't forget about deferred queries!!!             IEnumerable<int> intsGreaterThanTwo = ints.Where(GreaterThanTwo);             // Display the results.             foreach (int i in intsGreaterThanTwo)                 Console.WriteLine(i);              //Another Example of the First Select Prototype             var nameObjs = presidents.Select(p => new { p, p.Length });             foreach (var item in nameObjs)                 Console.WriteLine(item);              //A Third Example of the First Select Prototype             var nameObjs1 = presidents.Select(p => new { LastName = p, Length = p.Length });             foreach (var item in nameObjs1)                 Console.WriteLine("{0} is {1} characters long.", item.LastName, item.Length);              //An Example of the Second Select Prototype             var nameObjs2 = presidents.Select((p, i) => new { Index = i, LastName = p });             foreach (var item in nameObjs2)                 Console.WriteLine("{0}. {1}", item.Index + 1, item.LastName);              //An Example of the First SelectMany Prototype             IEnumerable<char> chars = presidents.SelectMany(p => p.ToArray());             foreach (char ch in chars)                 Console.WriteLine(ch);              //An Example of the Second SelectMany Prototype             IEnumerable<char> chars1 = presidents.SelectMany((p, i) => i < 5 ? p.ToArray() : new char[] { });             foreach (char ch in chars1)                 Console.WriteLine(ch);              //A More Complex Example of the First SelectMany Prototype             Employee[] employees = Employee.GetEmployeesArray();             EmployeeOptionEntry[] empOptions = EmployeeOptionEntry.GetEmployeeOptionEntries();             var employeeOptions = employees.SelectMany(e => empOptions.Where(eo => eo.id == e.id).Select(eo => new             {                 id = eo.id,                 optionsCount = eo.optionsCount             }));             foreach (var item in employeeOptions)                 Console.WriteLine(item);                        //An Example of the Only Take Prototype             IEnumerable<string> items2 = presidents.Take(5);             foreach (string item in items2)                 Console.WriteLine(item);              //Another Example of the Take Prototype             IEnumerable<char> chars2 = presidents.Take(5).SelectMany(s => s.ToArray());             foreach (char ch in chars2)                 Console.WriteLine(ch);              //An Example of Calling the First TakeWhile Prototype             IEnumerable<string> items3 = presidents.TakeWhile(s => s.Length < 10);             foreach (string item in items3)                 Console.WriteLine(item);              //An Example of Calling the Second TakeWhile Prototype             IEnumerable<string> items4 = presidents.TakeWhile((s, i) => s.Length < 10 && i < 5);             foreach (string item in items4)                 Console.WriteLine(item);              //An Example of the Only Skip Prototype             IEnumerable<string> items5 = presidents.Skip(1);             foreach (string item in items5)                 Console.WriteLine(item);              //An Example Calling the First SkipWhile Prototype             IEnumerable<string> items6 = presidents.SkipWhile(s => s.StartsWith("A"));             foreach (string item in items6)                 Console.WriteLine(item);              //An Example of Calling the Second SkipWhile Prototype             IEnumerable<string> items7 = presidents.SkipWhile((s, i) => s.Length > 4 && i < 10);             foreach (string item in items7)                 Console.WriteLine(item);              //An Example Calling the Only Concat Prototype             IEnumerable<string> items8 = presidents.Take(5).Concat(presidents.Skip(5));             foreach (string item in items8)                 Console.WriteLine(item);              //An Example Performing Concatention with an Alternative to Using the Concat Operator             IEnumerable<string> items9 = new[] {presidents.Take(5),presidents.Skip(5)}.SelectMany(s => s);             foreach (string item in items9)                 Console.WriteLine(item);              //An Example Calling the First ThenBy Prototype             IEnumerable<string> items10 = presidents.OrderBy(s => s.Length).ThenBy(s => s);             foreach (string item in items10)                 Console.WriteLine(item);              /\*//An Example Calling the Second OrderBy Prototype             MyVowelToConsonantRatioComparer myComp = new MyVowelToConsonantRatioComparer();             IEnumerable<string> namesByVToCRatio = presidents.OrderBy((s => s), myComp);             foreach (string item in namesByVToCRatio)             {                 int vCount = 0;                 int cCount = 0;                  myComp.GetVowelConsonantCount(item, ref vCount, ref cCount);                 double dRatio = (double)vCount / (double)cCount;                 Console.WriteLine(item + " - " + dRatio + " - " + vCount + ":" + cCount);             }\*/            }          //Converting an Array of Strings to Integers         private static void ConvertingStringToIntegers()         {                        //Converting an Array of Strings to Integers             int[] nums = numbers.Select(s => Int32.Parse(s)).ToArray();              //Converting an Array of Strings to Integers and Sorting It             int[] numsort = numbers.Select(s => Int32.Parse(s)).OrderBy(s => s).ToArray();         }          //Query with Intentional Exception Deferred Until Enumeration      }   Nhung vd tren la nhung vd co ban trong LINQ, neu ban muon co nhung phuc tap hon ... chung ta cung tham khao code ben duoi   public static void TestSortMatrix()         {             List<string> source1 = new List<string>() { "aa5", "aa6","aa1", "aa2", "aa3", "aa4"  };             List<string> source2 = new List<string>() { "bb1", "bb2", "bb3", "bb4", };             List<string> source3 = new List<string>() { "cc1", "cc2", "cc3", "cc4", "cc5" };             List<string> source4 = new List<string>() { "dd1", "dd2", "dd5", "dd6", "dd3", "dd4"  };             List<List<string>> source = new List<List<string>>();             source.Add(source1);             source.Add(source2);             source.Add(source3);             source.Add(source4);              List<List<string>> listAll = source.SelectMany(                 (it) => it.Select((t, i) => new { t, i })).GroupBy(z => z.i).Select(                 g => g.Select(z => z.t).ToList()).ToList();          }   Con rat nhieu ...rat nhieu dieu hay ve LINQ khi ban xu li ve database mi`nh se noi lan sau. Chuc cac ban vui ve. | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **hoangminh** -> *RE: Nhung vd ve linq va C#* (3/8/2010 7:58:34 AM) | | Bó tay, bác lấy vd của "nước ngoài" mà ko một lời giải thích, bác tự làm 1 cái vd "VN" thật đơn giản (chỉ cần vài dòng thui) và nói rỏ cái hay của nó thì mọi ng mới bít chứ, đâu phải ai nhìn vô code diều hiểu hết đâu, thiên tài nhiều khi còn phải nặng óc ra suy nghĩ mà còn ko ra nửa chứ đừng nói chi là thường dân. hic. | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **itshare277** -> *RE: Nhung vd ve linq va C#* (3/18/2010 1:23:19 AM) | | Cũng may là do biết về Lambda, nếu không thì nhìn vào mình cũng tịt thật. Hic hic. | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **bobolove223** -> *RE: Nhung vd ve linq va C#* (4/2/2010 2:21:43 PM) | | Đúng là không dùng LINQ sẽ rất đáng tiếc, nhưng những ví dụ thế này người mới học cũng khó mà hiểu được. Có lẽ làm những cái đơn giản, dễ mà hiểu thì tốt hơn. | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **hoangedward** -> *RE: Nhung vd ve linq va C#* (1/13/2011 1:38:24 PM) | | 101 LINQ Samples: cực kỳ dễ hiểu cho các bạn muốn làm quen và làm việc với LinQ một cách hiệu quả hơn!  <http://msdn.microsoft.com/en-us/vcsharp/aa336746> | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **hngiang84** -> *RE: Nhung vd ve linq va C#* (12/2/2011 10:29:44 AM) | | uh tôi cũng thấy nhiều anh em chuyển qua dùng linq , nhưng chưa thấy có frame dùng linq dạng mô hình 3 lớp, anh em nào phát triển được rồi thì public cho mọi người cùng học hỏi nhé | |  | | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | |  | | --- | | **diepviet** -> *RE: Nhung vd ve linq va C#* (12/26/2011 11:55:20 PM) | | Cảm ơn bạn đã up bài này lên nhé, rất hay.  Theo ý kiến của mình là: SỰ HẤP DẪN Ở CHỖ TÌM TÒI VÀ CỐ GẮNG ĐÓ. | |  | | |

|  |  |
| --- | --- |
| |  | | --- | | Trang: **[1]** | |

|  |
| --- |
| http://forum.aspvn.net/image/footerline.jpg |
| © 2002-2008 Vietnamese ASP.NET. | [**Trang chủ**](http://aspvn.net) | [Diễn đàn](http://forum.aspvn.net/) | [Đăng ký](http://forum.aspvn.net/register.aspx) | [Đăng Nhập](http://forum.aspvn.net/loginflat.aspx) | [Thắc mắc thường gặp?](http://forum.aspvn.net/faq.aspx) Sử dụng phần mềm Diễn đàn [ASPPlayground.NET](http://www.aspplayground.net) phiên bản **Advanced Edition** 2.5.5 Unicode Skin phát triển bởi **ASPVN.NET** phiên bản 1.0.0. Thời gian tải trang: 0.024 Nội dung được xây dựng bởi **Ban quản trị** và các **Thành viên** của diễn đàn Domain & Hosting được tài trợ bởi [ViTechNet.,JSC](http://www.host.com.vn) | [Microsoft Vietnam](http://www.microsoft.com/vietnam) |