LinqPad

최흥배

Twitter : @jacking75

http://jacking.tistory.com/

Over 1 million downloads

LINQPad

- Home / Download
- Purchase
- FAQ
- Beta + Special Builds
- LINQPad as a Code Scratchpad
- LINQPad + Entity Framework
- Why LINQ Beats SQL
- The LINQPad Challenge
- How LINQPad Works
- Feedback / Contact / Subscribe

Tired of querying in antiquated SQL?

Well, you don't have to! **LINQPad** lets you interactively query databases in a *modern query language*: **LINQ.** Kiss goodbye to SQL Management Studio!

LINQPad supports everything in C# 5.0 and Framework 4.x:

- LINQ to Objects
- LINQ to SQL, Entity Framework
- LINQ to XML
- Parallel LINQ
- OData / WCF Data Services, SharePoint, and Windows DataMarket
- Microsoft's ubercool Reactive Extensions

And that's not all - there's also specific support for:

- SQL Azure, SQL Table Storage, Oracle, SQLite and MySQL
- Microsoft StreamInsight
- Microsoft Dynamics CRM
- Third-party ORMs including Mindscape LightSpeed, DevArt's LinqConnect, LLBLGen, DevExpress eXpress Persistent Objects and DevForce
- (Even old-fashioned SQL!)

LINQPad is also a **great way to learn LINQ**: it comes loaded with 500 examples from the book, C# 5.0 in a Nutshell. There's no better way to experience the coolness of LINQ and *functional programming*.

Download now

- ✓ Super-lightweight setup
- ✓ Targets .NET Framework 4.x
- ✓ **FREE** (with no expiry)

Version: 4.51.03

Prerequisites | License | FAQ

More download options

Hate installers?
Download standalone executable

Need to run massive queries?
Download AnyCPU build

Still running .NET Framework 3.5?
Download LINQPad 2.x

Want the latest cool features?
Go to LINQPad Beta page

Purchase Premium Edition

http://www.linqpad.net/

컴파일러 없이 C#/VB 코드를 실행할 수 있으며, 데이터베이스 툴로도 사용할 수 있다.

기본적으로 Linq 작업에 최적화 되어 있다.

- LINQPad를 이용하여 LINQ 편집하기 http://www.yunsobi.com/blog/466
- LINQ 작성을 위한 툴 linqpad http://jojaebi.tistory.com/12

Purchase Premium Features

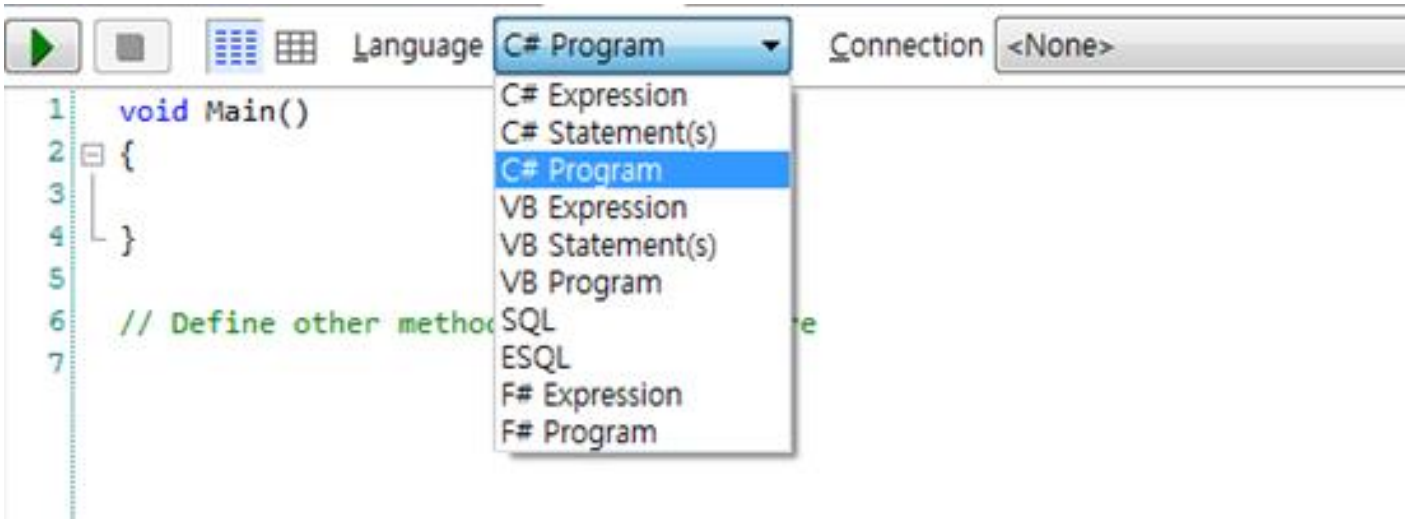
Purchase an activation code and unlock a wealth of extra features, including autocompletion, code snippets, cross-database querying, NuGet integration, direct editing of SQL data, and a fully integrated debugger!

No further downloads are required. Be up and running in less than a minute.

	Free Edition	Pro Edition	Developer Edition	Premium Edition
Run LINQ queries - or any C#/VB/F# expression/program	✓	✓	✓	✓
Results to rich text or data grids	✓	✓	✓	✓
Automatic translation to SQL, fluent-lambda and IL	✓	✓	✓	✓
Reference your own assemblies, plus the entire .NET Framework	✓	✓	✓	✓
Use custom extensions, visualizers and data source providers	✓	✓	✓	✓
Full C# autocompletion with list filtering and power-tick		✓	✓	✓
Smart tags for importing additional namespaces and references		✓	✓	✓
Quick info, code outlining and .NET Reflector integration		✓	✓	✓
Built-in and custom code snippets			✓	✓
Cross-database querying for SQL Server			✓	✓
Directly edit SQL data in grids and save changes back to database			✓	✓
Full NuGet integration			✓	✓
Integrated debugger, with call stack, threads, local variables/watch windows, breakpoints and single-stepping				✓

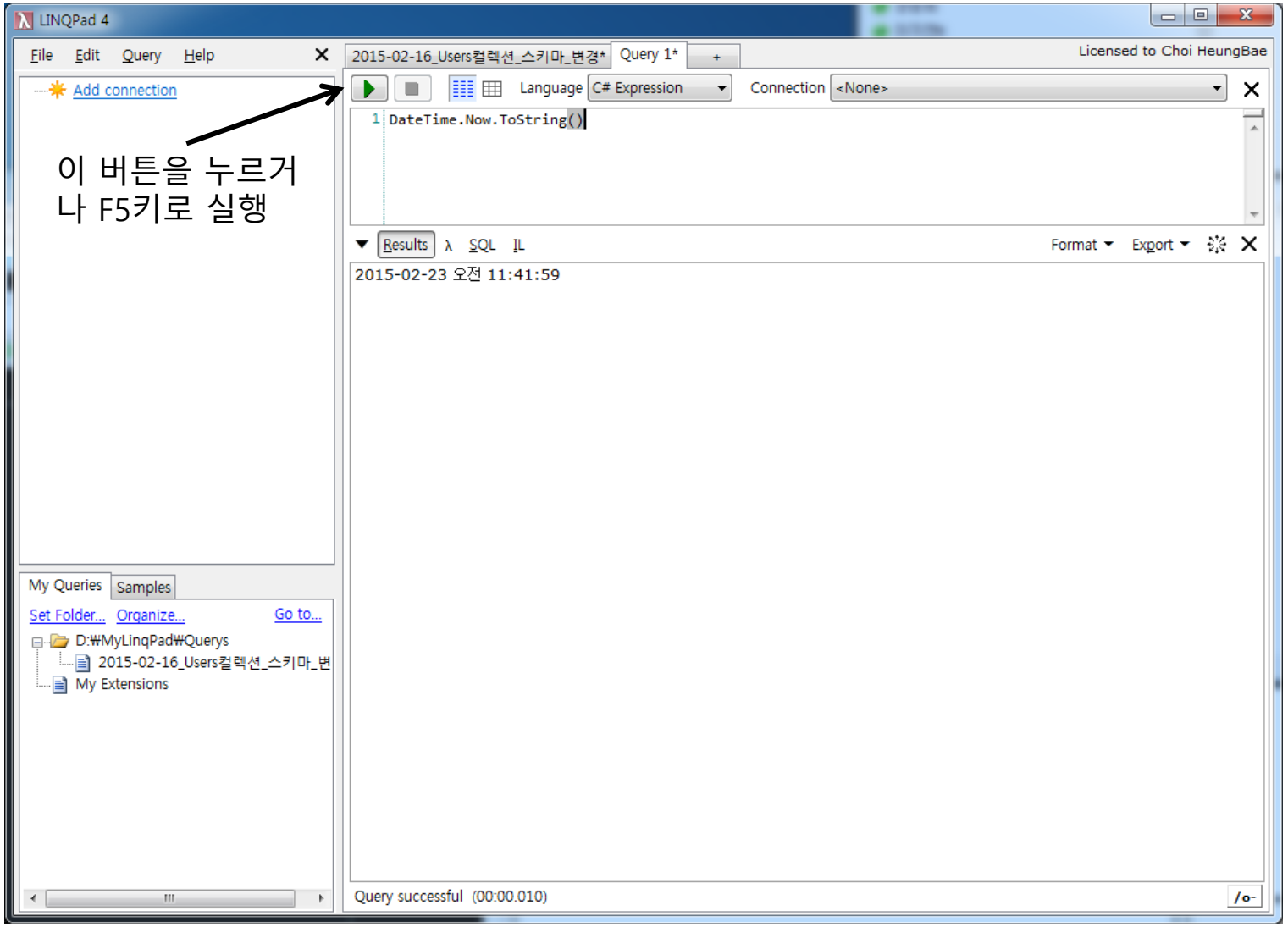
기본은 무료로 왠만한 기능을 문제 없이 사용할 수 있다.
디버깅은 프리미엄 버전만 지원.

C#/VB 코드 실행



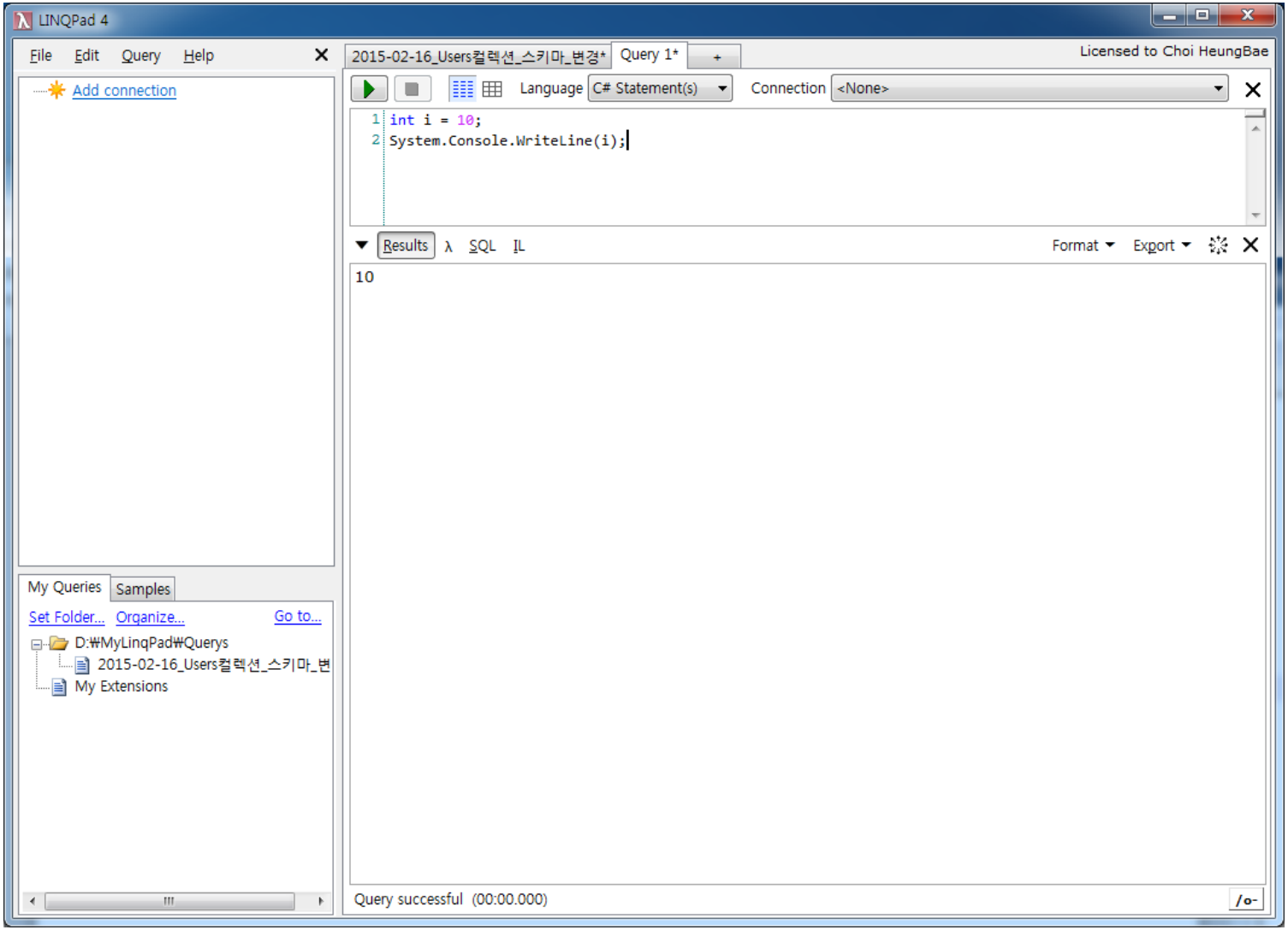
사용할 언어와 형식을 선택 후 코딩한 후 실행한다.

C#/VB 코드 실행: Expression



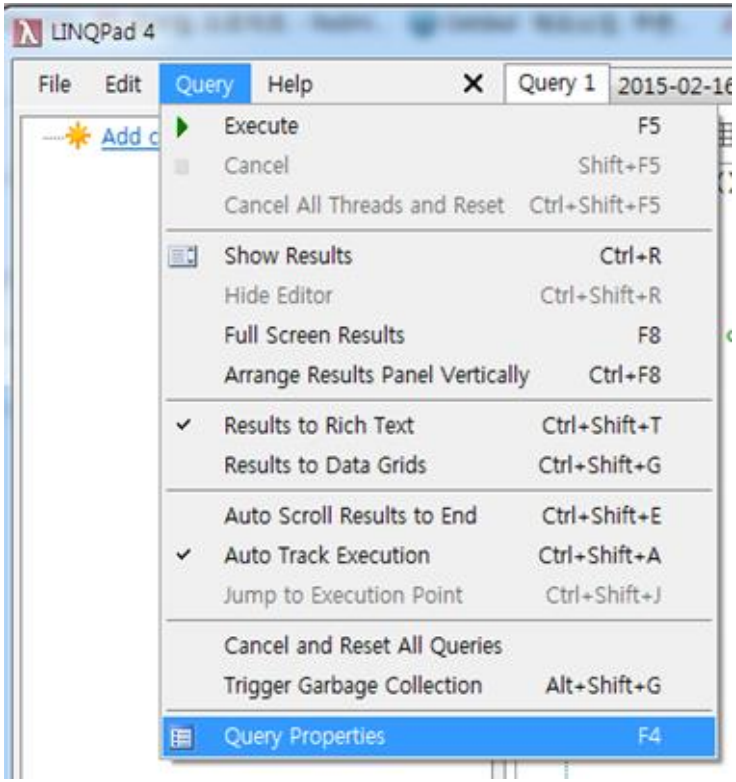
1 라인으로 된 코드만 실행

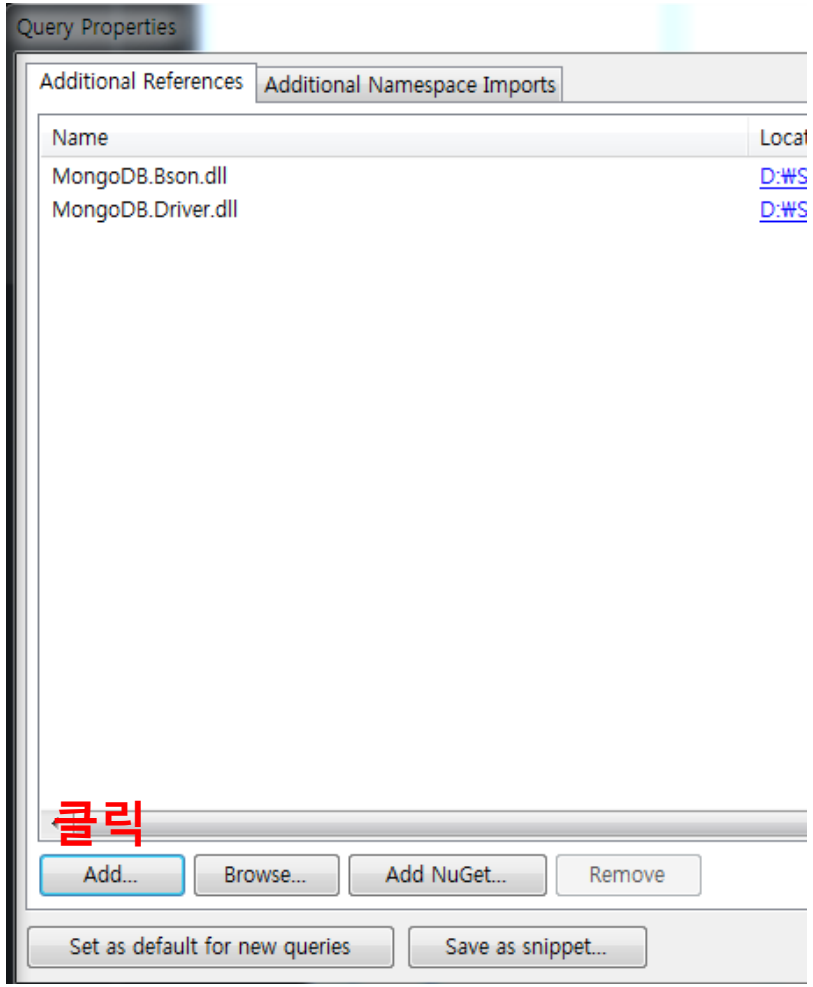
C#/VB 코드 실행: Statement



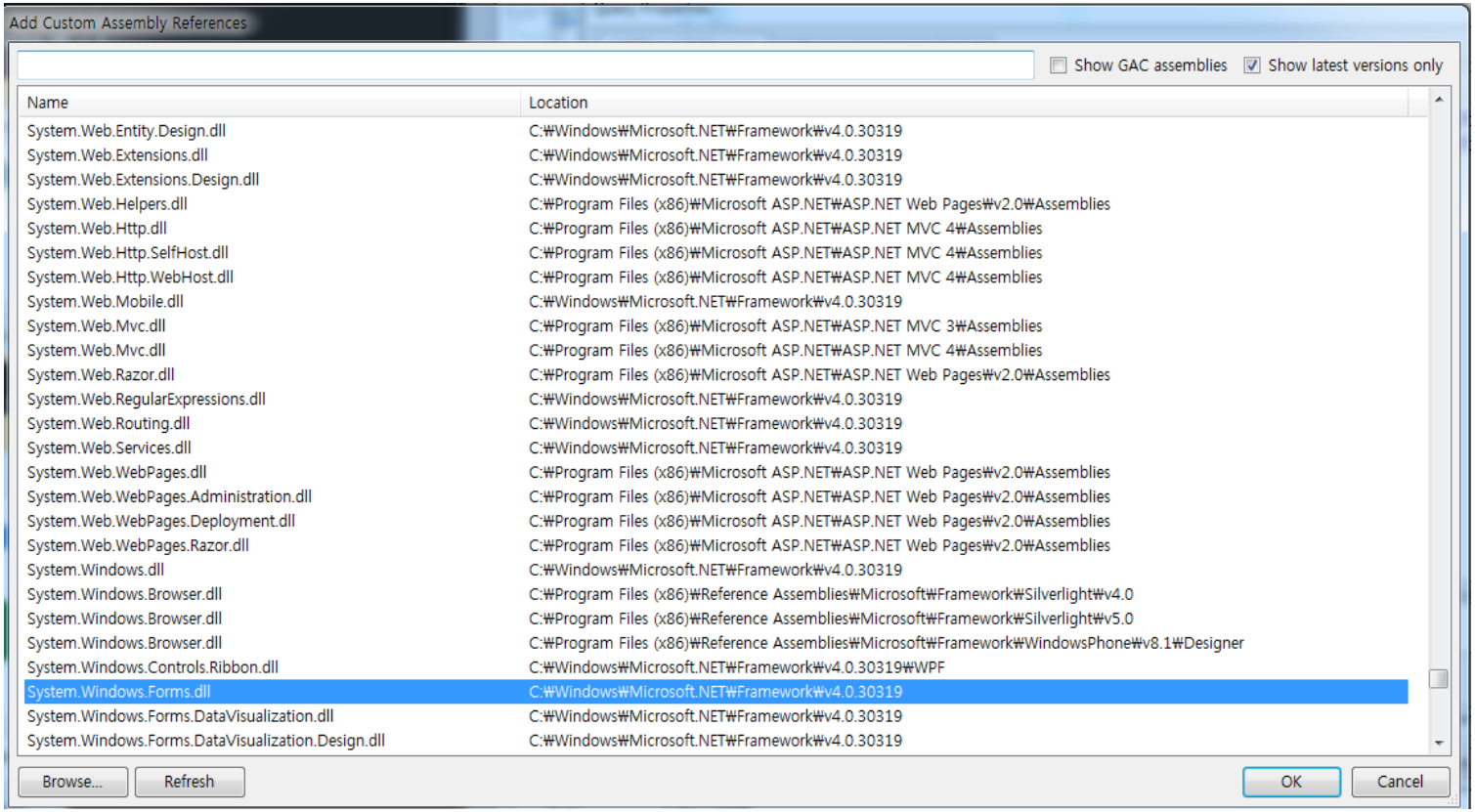
1 라인 이상의 코드 실행(class나 enum 정의 등은 못한다).
특정 부분만 실행하고 싶은 경우 마우스 드래그로 선택하면 된다.

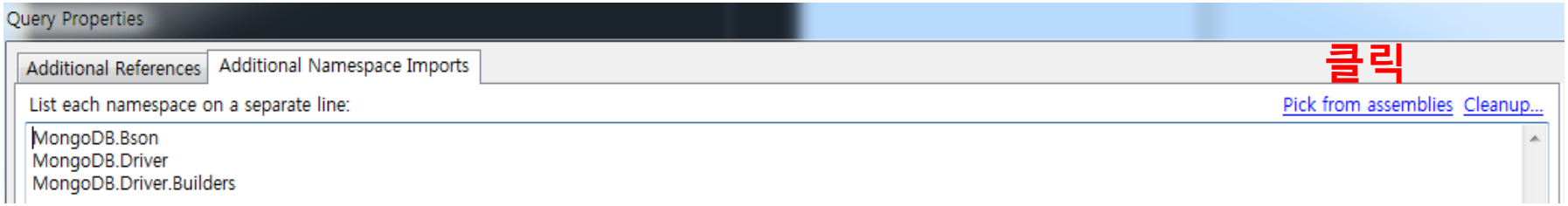
C#/VB 코드 실행: GUI

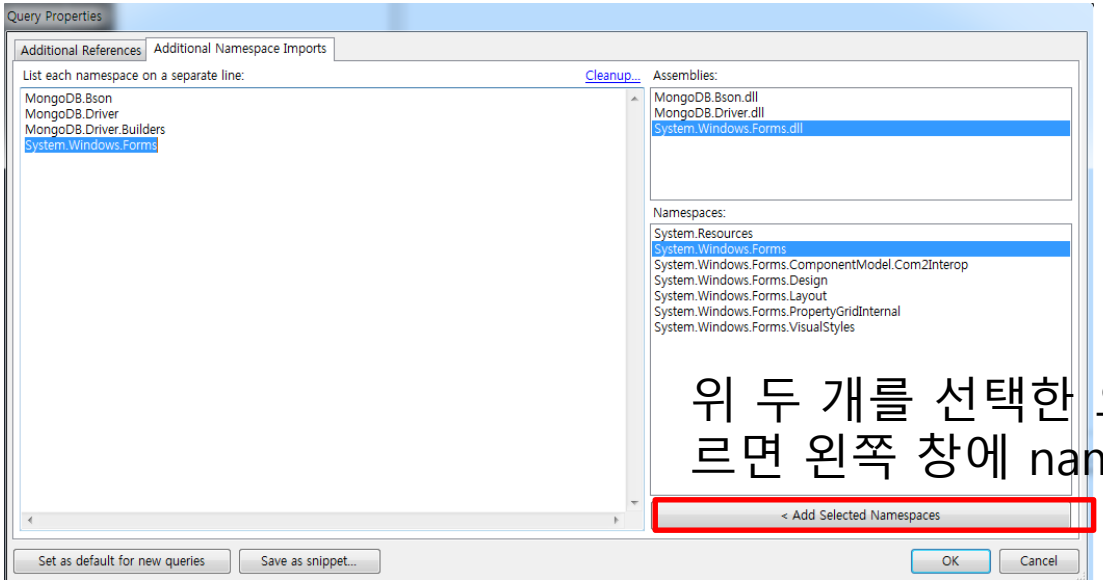




WinForm을 실행하기 위해서 관련 dll과 네임스페이스를 추가한다(편집창에서는 네임스페이스 선언이 불가능하다).

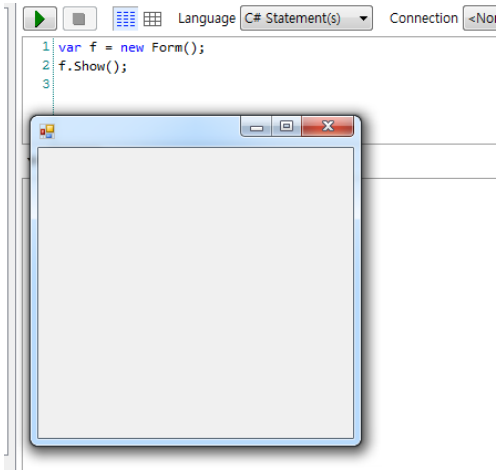

Add Custom Assembly References
Show GAC assemblies
Show latest versions only
Name
Location
System.Web.Entity.Design.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.Extensions.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.Extensions.Design.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.Helpers.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET Web Pages₩v2.0₩Assemblies
System.Web.Http.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET MVC 4₩Assemblies
System.Web.Http.SelfHost.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET MVC 4₩Assemblies
System.Web.Http.WebHost.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET MVC 4₩Assemblies
System.Web.Mobile.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.Mvc.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET MVC 3₩Assemblies
System.Web.Mvc.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET MVC 4₩Assemblies
System.Web.Razor.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET Web Pages₩v2.0₩Assemblies
System.Web.RegularExpressions.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.Routing.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.Services.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Web.WebPages.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET Web Pages₩v2.0₩Assemblies
System.Web.WebPages.Administration.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET Web Pages₩v2.0₩Assemblies
System.Web.WebPages.Deployment.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET Web Pages₩v2.0₩Assemblies
System.Web.WebPages.Razor.dll C:₩Program Files (x86)₩Microsoft ASP.NET₩ASP.NET Web Pages₩v2.0₩Assemblies
System.Windows.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Windows.Browser.dll C:₩Program Files (x86)₩Reference Assemblies₩Microsoft₩Framework₩Silverlight₩v4.0
System.Windows.Browser.dll C:₩Program Files (x86)₩Reference Assemblies₩Microsoft₩Framework₩Silverlight₩v5.0
System.Windows.Browser.dll C:₩Program Files (x86)₩Reference Assemblies₩Microsoft₩Framework₩WindowsPhone₩v8.1₩Designer
System.Windows.Controls.Ribbon.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319₩WPF
System.Windows.Forms.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Windows.Forms.DataVisualization.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
System.Windows.Forms.DataVisualization.Design.dll C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
Browse...
Refresh
OK
Cancel


Query Properties
Additional References
Additional Namespace Imports
클릭
List each namespace on a separate line:
Pick from assemblies
Cleanup...
MongoDB.Bson
MongoDB.Driver
MongoDB.Driver.Builders



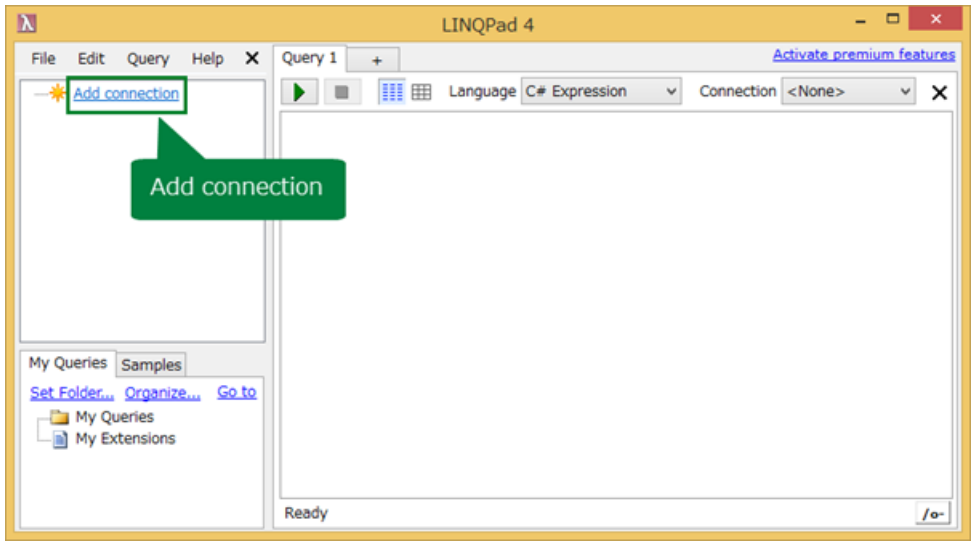
위 두 개를 선택한 오른쪽 아래 버튼을 누르면 왼쪽 창에 namespace가 추가된다

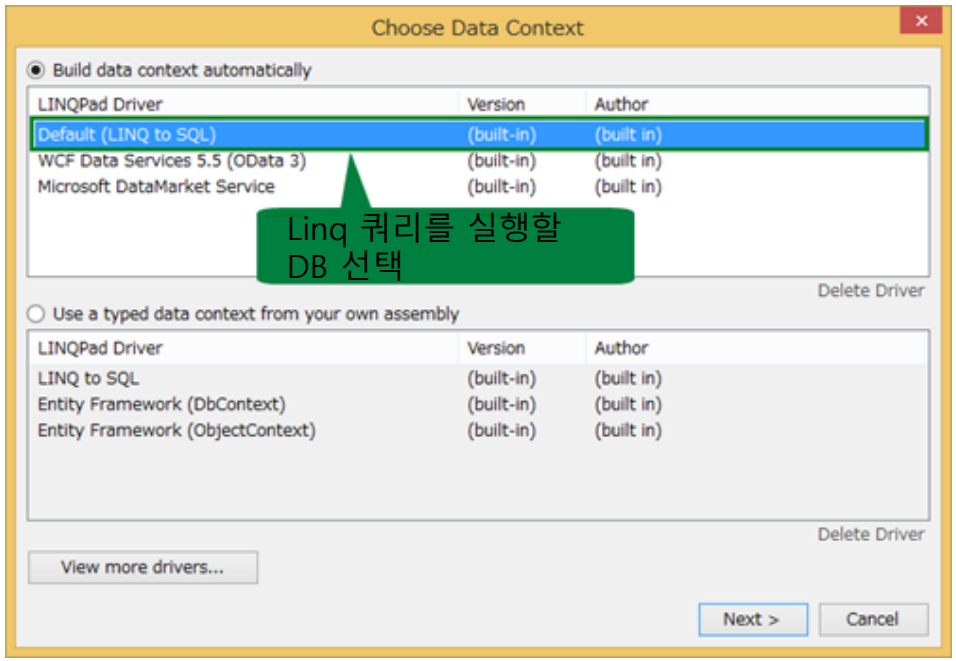
[Save as snippet...] 기능을 사용하면 참조나 네임스페이스 선언을 쉽게 재이용 가능.



sample 라는 snippet로 저장한 후 에디터 상에서 sample를 입력한 후 tab 키를 누르면 된다.

DB에서 Linq 쿼리 실행

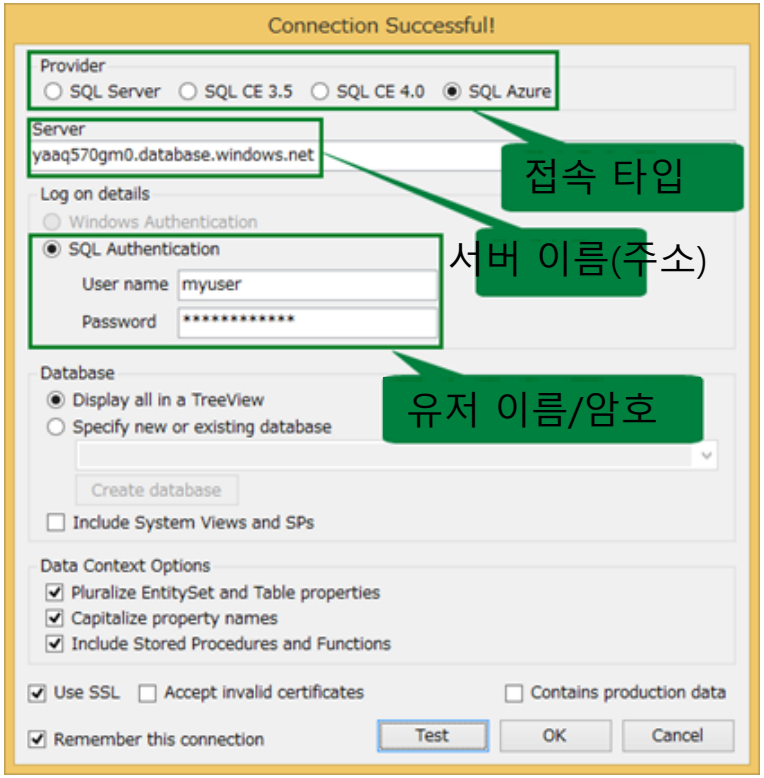


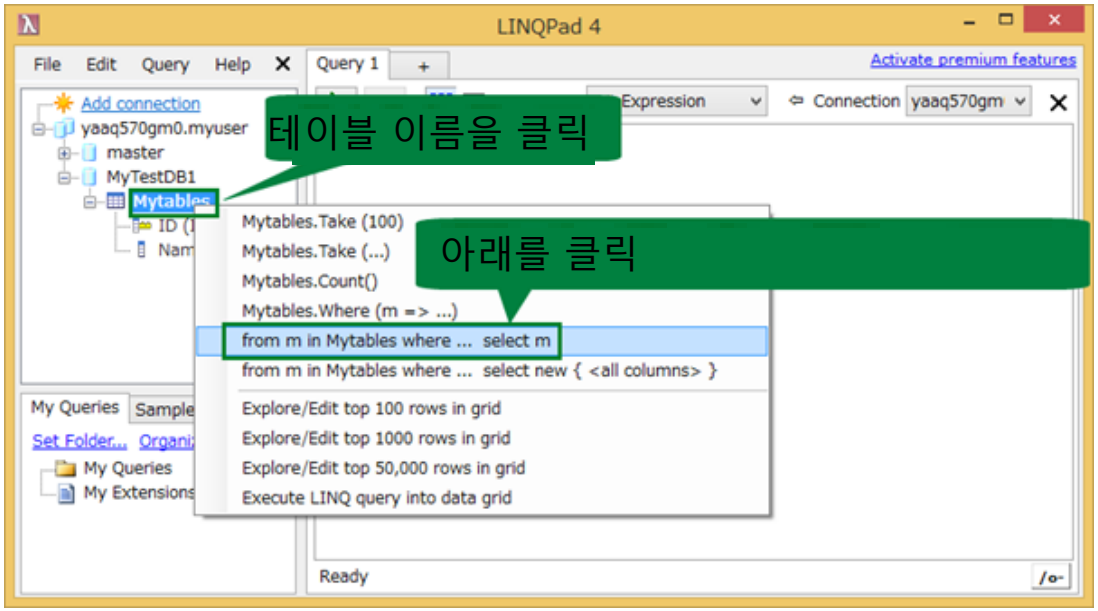


Azure SQL도 'Default'를 선택하면 된다.

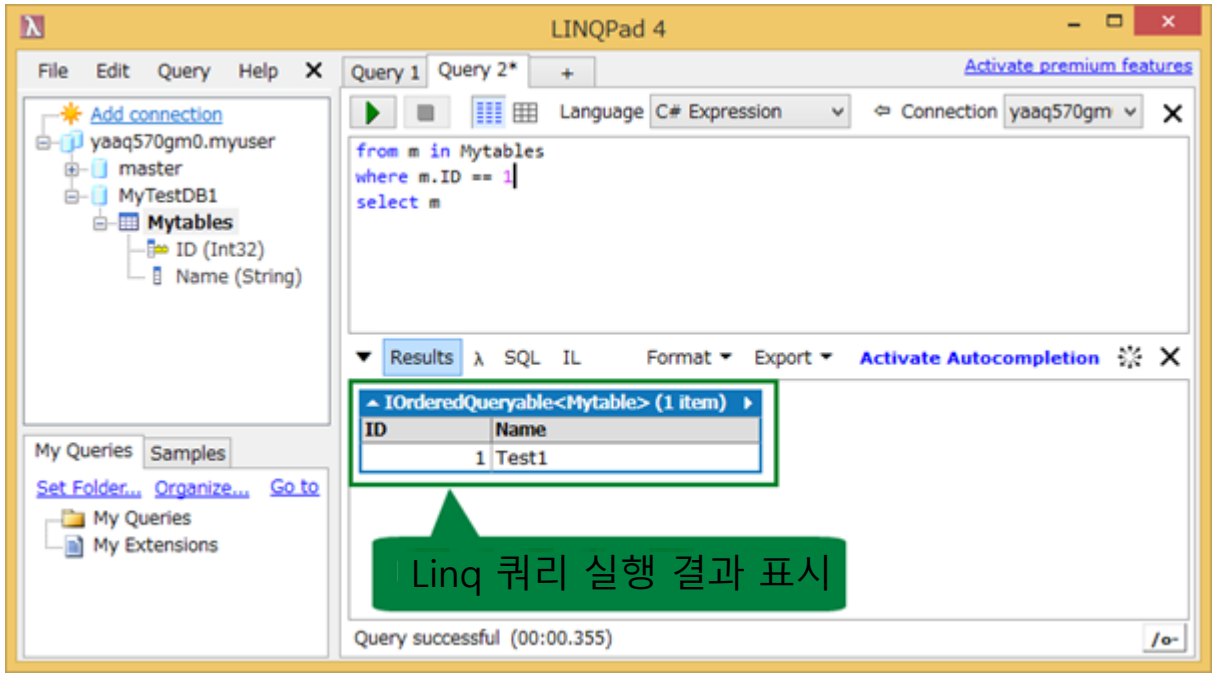
MS SQL 이외에도 MySQL 등의 DB를 선택할 수 있다.

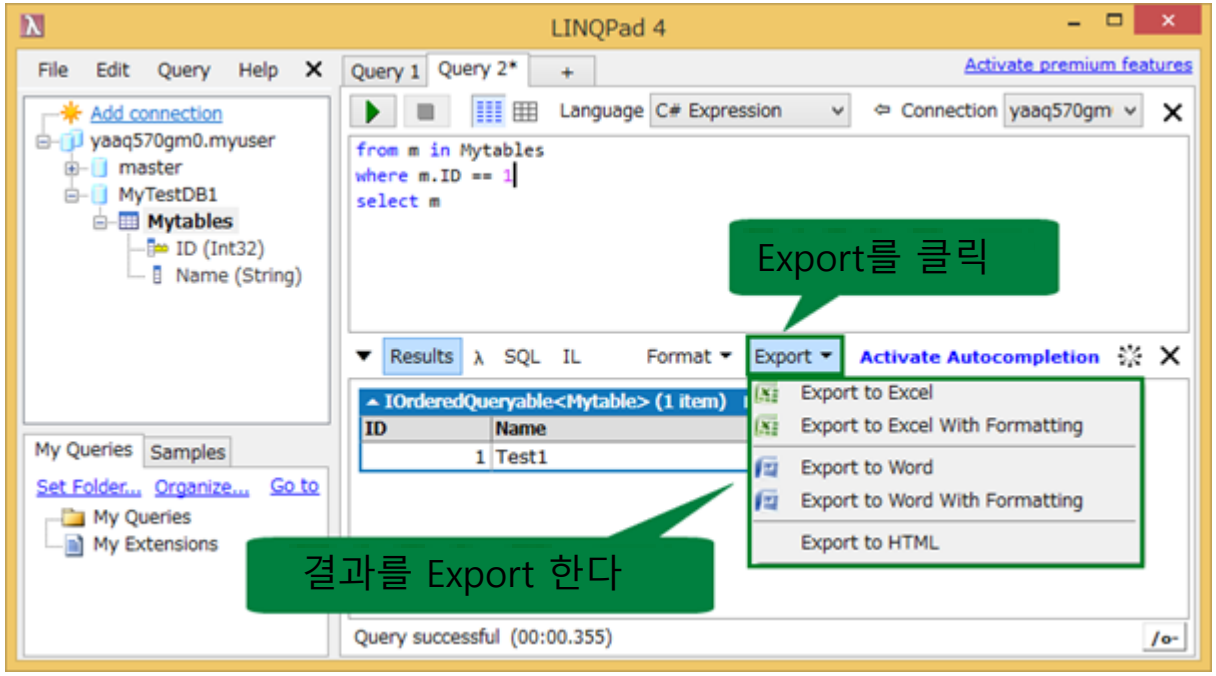
http://troushoo.blog.fc2.com/blog-entry-213.html



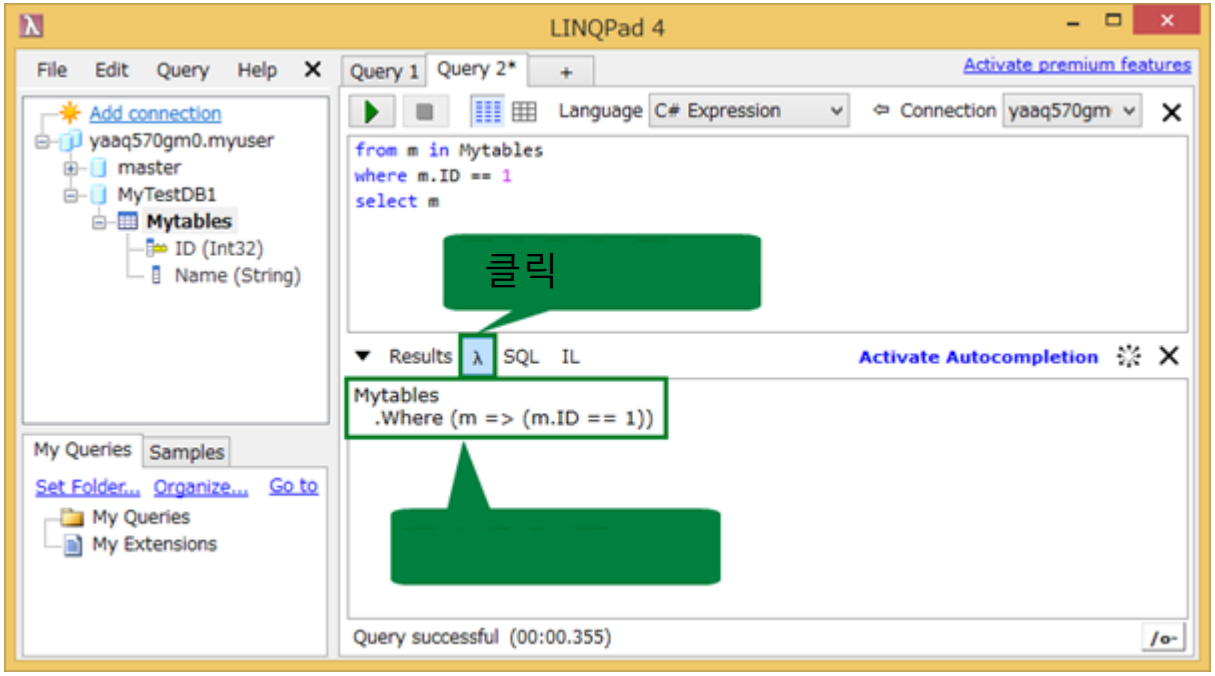


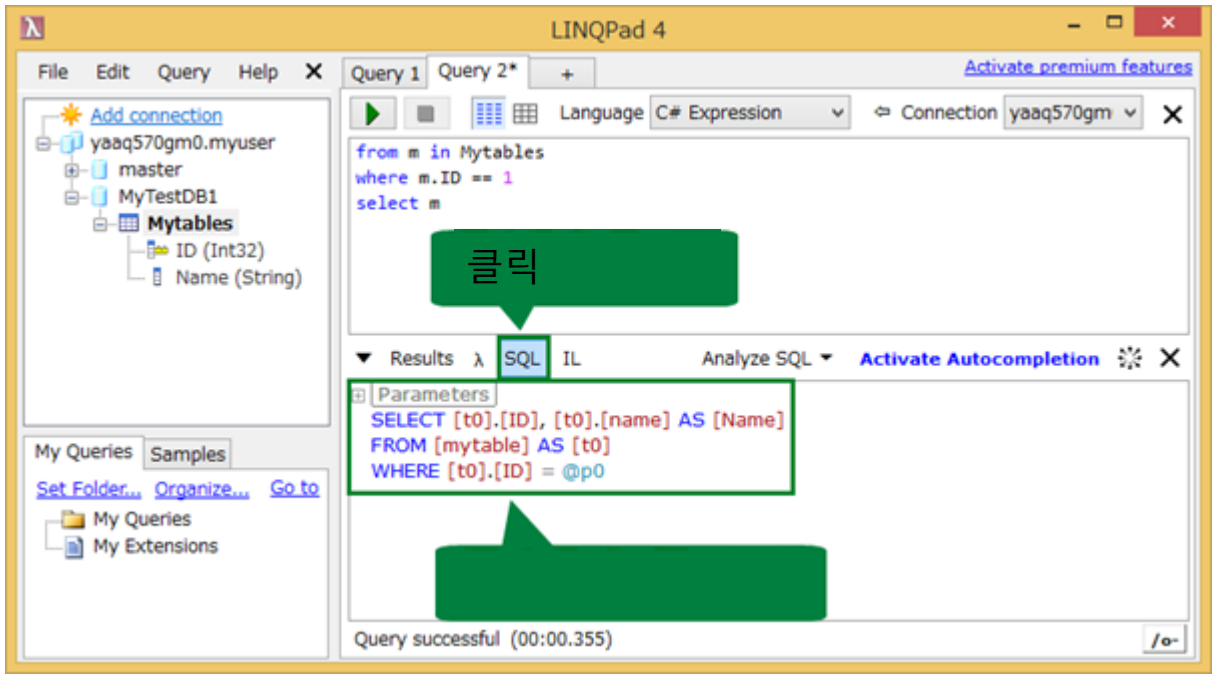
http://troushoo.blog.fc2.com/blog-entry-213.html



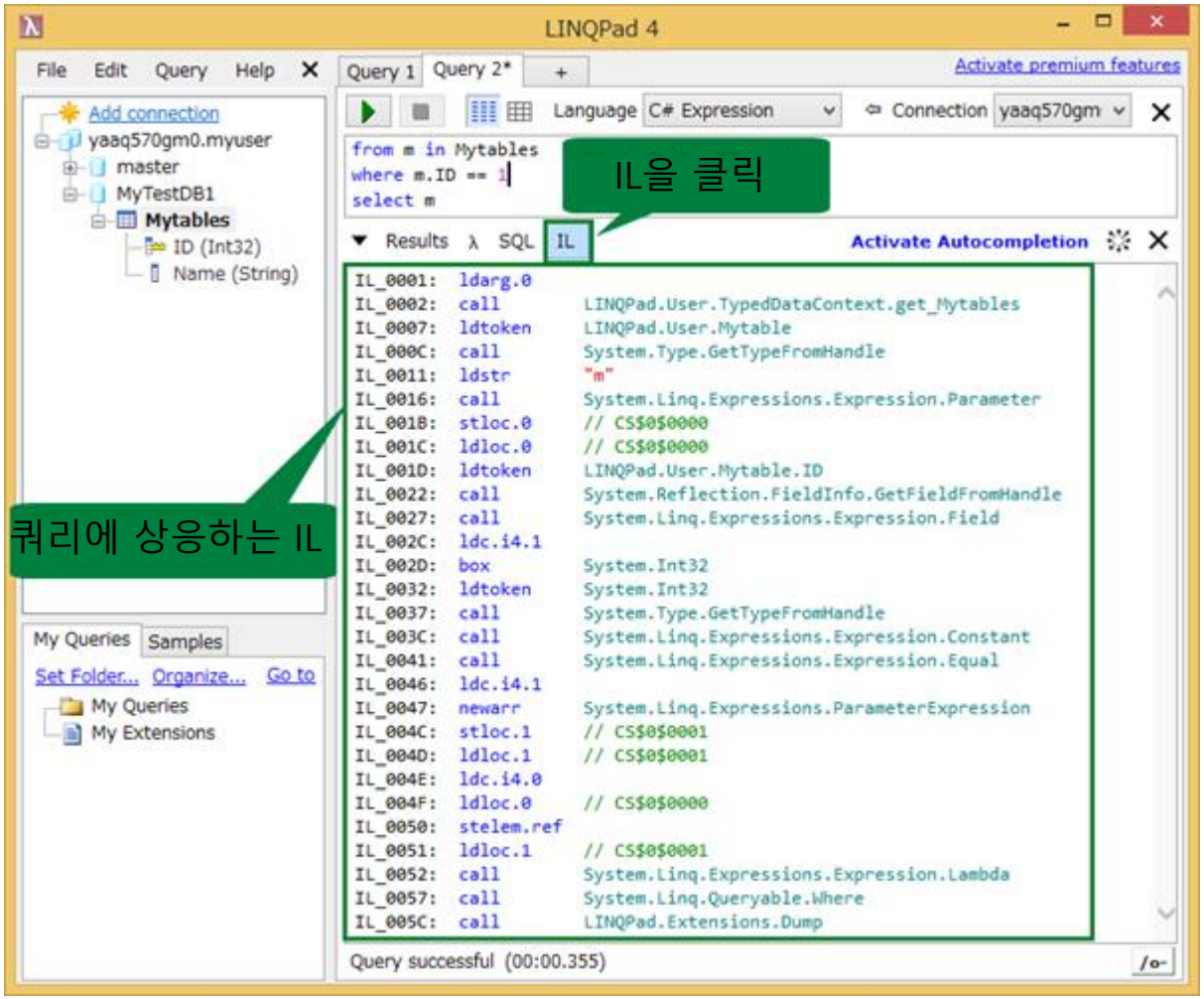


http://troushoo.blog.fc2.com/blog-entry-213.html





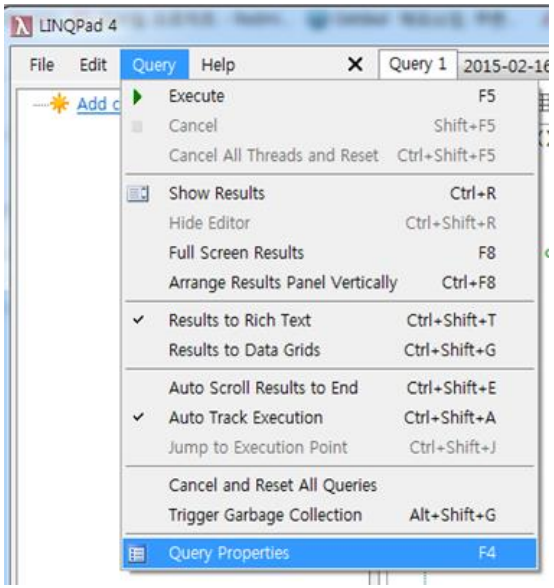
http://troushoo.blog.fc2.com/blog-entry-213.html



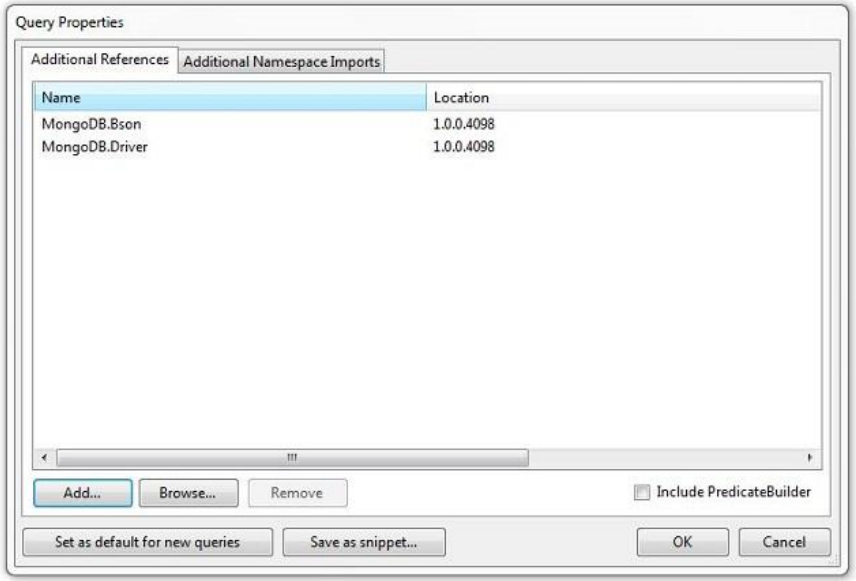
http://troushoo.blog.fc2.com/blog-entry-213.html

LinqPad에 MongoDB 라이브러리 사용하기

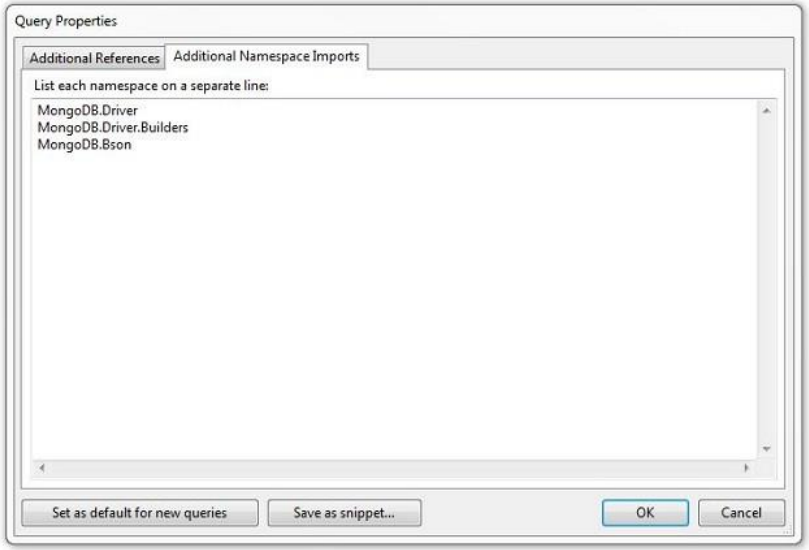
(1)



(2)

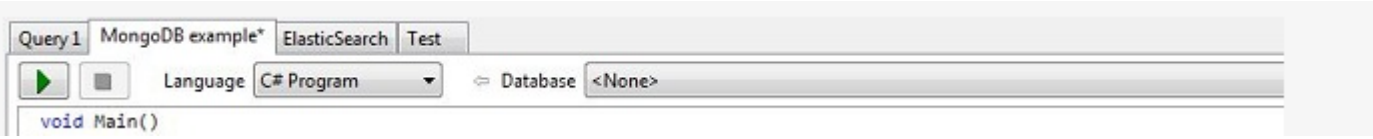


(3)



http://www.rkconsulting.com/blog/mongodb-with-linqpad

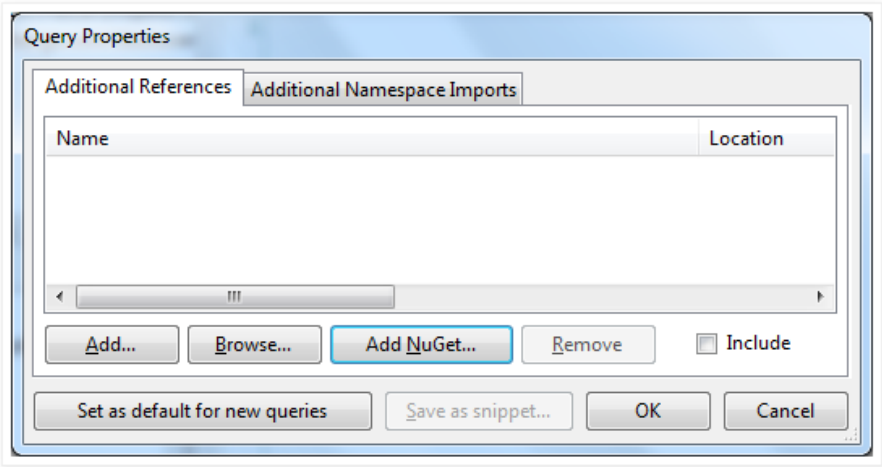
(4)

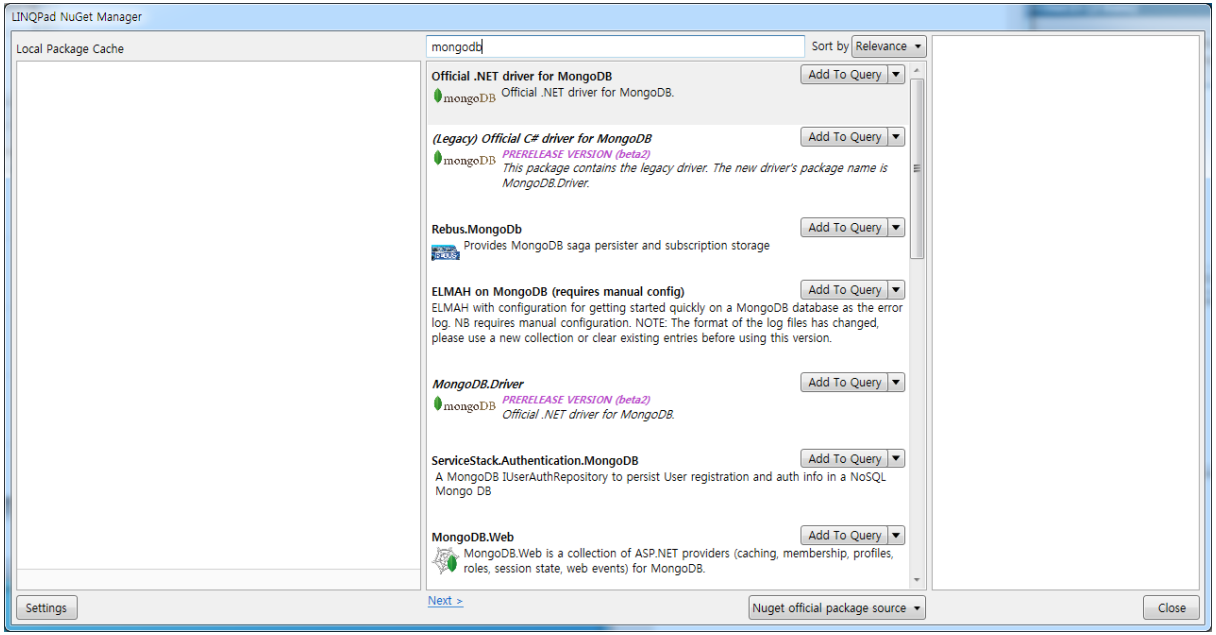



```
void Main()
{
		var server = MongoServer.Create("mongodb://law:pass@pearl.mongohq.com:27071/");
		var db = server.GetDatabase("Universe");
		var collection = db.GetCollection<Planet>("Planet");

		var pluto = new Planet()
		{
			Name = "Pluto",
			Colour = "Purple"
		};

		collection.Insert(pluto);

		var query = Query.And(Query.EQ("Name", "Pluto"));
		var list = collection.Find(query).ToList();

		list.Dump();
}

public class Planet
{
		public ObjectId  Id { get; set; }
		public string Name { get; set; }
		public string Colour { get; set; }
}
```

NuGet integration in LinqPad

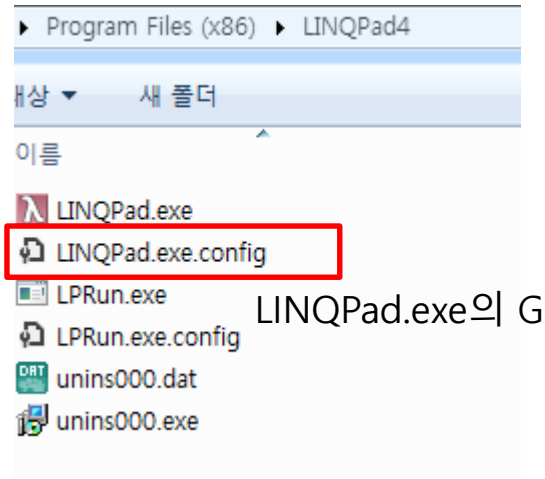
In the Query Properties (F4) we have a new option: **Add NuGet...**





http://pascallaurin42.blogspot.kr/2012/06/nuget-integration-in-linqpad.html

설정 파일



.NET 프로그램의 app.config와 비슷하다.
LINQPad.exe 파일이 있는 곳에 linqpad.config을 만들어 두면 된다.
설정에 기술한 것은 실행하는 코드에 반영된다.

LINQPad.exe의 GUI용 설정파일

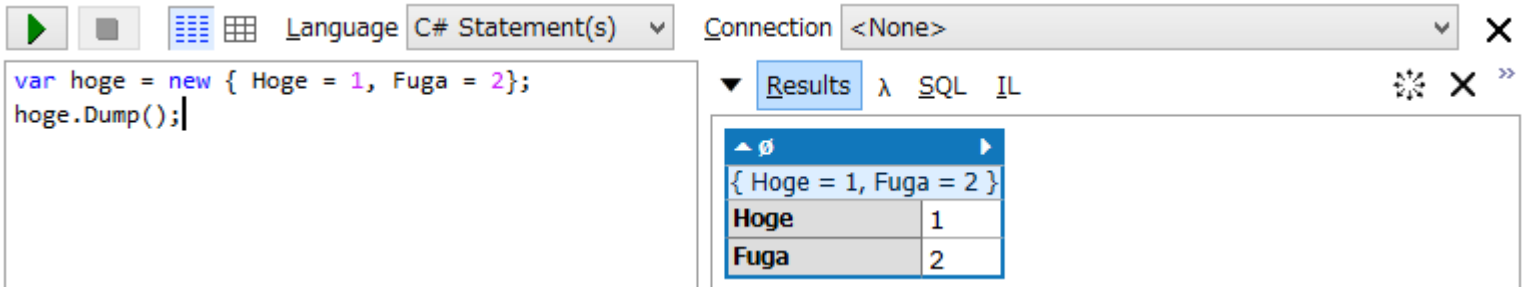
LINQPad.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
  </configSections>

  <appSettings>
    <add key="Application Name" value="MyApplication" />
    <add key="Application Version" value="1.0.0.0" />
  </appSettings>

  <connectionStrings>
    <add name="MyDB" providerName="System.Data.SqlClient" connectionString="Data Source=localhost;Initial Catal
og=test;User ID=User;Password=Pass;Min Pool Size=2"/>
  </connectionStrings>
</configuration>
```

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

Dump 메소드

Dump는 <T>에 대한 확장 메서드로 정의되어 있다.
그래서 기본적으로는 뭐든지 Dump를 실행 할 수 있다.
Dump를 실행하면 현재의 객체를 보기 좋게 [Results]부분에 표시해 준다.



또 Dump는 자신의 객체를 돌려주기 때문에 위의 예는 아래처럼 기술할 수도 있다.

```
var hoge = new { Hoge = 1, Fuga = 2}.Dump();
```

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

Dump를 사용하면 처음 LINQ를 사용할 때 이해하기 어려운 LINQ 관련 메소드의 실행 상황을 확인하기 쉽다

```
string[] Words = new string[] {"   KOOKABURRA", "Frogmouth", "kingfisher   ", "loon", "merganser"};
Words
  .Dump("ORIGINAL:")
  .Select(word => word.Trim())
  .Dump("TRIMMED:")
  .Select(word => word.ToLower())
  .Dump("LOWERCASE:")
  .Where(word => word.StartsWith("k"))
  .Dump("FILTERED to 'K':")
  .OrderBy(word => word)
  .Dump("SORTED:");
```

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

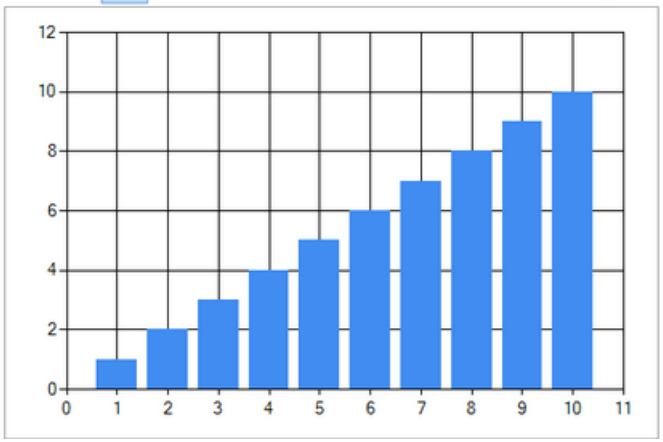
System.Windows.Forms.DataVisualization.Charting 이름 공간을 사용해서, Chart(그래프) Dump도 가능하다.

chartDump

```
//이하 dll의 참조와, namespace의 imports를 설정했어요
//System.Windows.dll
//System.Windows.Forms.dll
//System.Windows.Forms.DataVisualization.dll
//using System.Windows.Forms.DataVisualization.Charting

void Main()
{
    var chart = new Chart();
    chart.ChartAreas.Add(new ChartArea());
    var series = new Series { ChartType = SeriesChartType.Column };
    foreach (var item in Enumerable.Range (1, 10))
    {
        var x = item;
        var y = item;
        var index = series.Points.AddXY(x, y);
    }
    chart.Series.Add(series);
    chart.Dump("Title");
}
```

```
1  void Main()
2  {
3      var chart = new Chart();
4      chart.ChartAreas.Add(new ChartArea());
5      var series = new Series { ChartType = SeriesChartType.Column };
6      foreach (var item in Enumerable.Range (1, 10))
7      {
8          var x = item;
9          var y = item;
10         var index = series.Points.AddXY(x, y);
11     }
12     chart.Series.Add(series);
13     chart.Dump("Title");
14 }
15
16 // Define other methods and classes here
17
```

Results Title λ SQL IL



http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

.Dump() 메소드

```
public static T Dump<T>(this T obj /* , ... */)
{
    // ...
    return obj;
}
```

식 중 임의의 위치에 .Dump()를 삽입하여 출력 할 수 있다.

복잡한 객체는 표로서 출력되지만 계층이 깊은 경우 일부가 접착되거나 링크로 표시됨으로써 숨겨짐을 클릭하여 표시 가능. 또 표의 헤더의 오른쪽?를 클릭해서 표를 그리드 뷰로 표시 할 수 있다.

IEnumerable나 IEnumerable<T>을 .Dump() 하면 목록이 나오는 것은 당연.
아래의 타입도 지원한다:

- Task / Task<T>
- Dump 되면 awaiting... 를 표시하고 실행이 완료된 단계에서 결과값을 대체한다
- Lazy<T> - 표시되는 링크를 클릭할 때까지 실행이 지연
- IObservable<T> - 결과를 받아 볼 때 리스트가 늘어난다. 리스트의 테두리 색으로 완료되었는지 판단할 수 있다
- XDocument / XElement - XML 뷰에서 표시됩니다
- Control (Windows Forms), UIElement (WPF)- UI 요소의 내용이 출력에 표시된다
- 등

또 Dump() 방법에는 여러 오버 로드가 있어서 출력을 세세하게 제어할 수 있다.

http://takeshik.org/blog/2014/12/06/try-linqpad/

Util 클래스

LINQPad 자체에 정의되어 있는 클래스.
LINQPad를 이용하는 데 편리한 기능이 준비되어 있다.

```
public class DummyData
{
    public int Id { get; set; }
    public string Name { get; set; }
    public int Value { get; set; }
}
```

```
var bar = new Util.ProgressBar("Progressbar 표시");
bar.Dump();
foreach (var element in Enumerable.Range(1, 100))
{
   bar.Percent = element;
   System.Threading.Thread.Sleep(50);
};
//Progressbar표시 부분만 실행할 수 있도록";"를 넣고 있다
```

```
Util.Metatext("Metatext").Dump("Metatext 표시를(녹색+이태릭)");
```

```
Util.Highlight( new { a = 1 , b = 2, c = 3} ).Dump("Highlight Dump 메서드에서 출력 결과를 하이라이트 표시(문자열 외에는 효과가 없는듯? 사용법을 틀려서 그런가?)");
```

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

```
var a = new DummyData(){ Name = "1", Id = 1 };
var b = new DummyData(){ Name = "2", Id = 2 };
Util.VerticalRun( Util.Highlight(a), b ).Dump("VerticalRun 복수의 데이터를 수직 방향으로 정리해서 Dump(클래스는 Highlight가 효과 없음)");
Util.VerticalRun( Util.Highlight("abc"), "abc" ).Dump();

Util.WordRun( true, Util.Highlight(a), b ).Dump("WordRun 복수의 데이터를 수직 방향으로 정리해서 Dump. 제1 인수는 표시 영역이 모자란 경우에 접을지를 지정하는 플래그.(클래스의 경우는 옆으로 안 되고 Highlight 안됨)");
Util.WordRun(true, Util.Highlight("abc"), "abc").Dump();
Util.WordRun(false, Enumerable.Range (0, 100)).Dump();

Util.RawHtml( Util.ToHtmlString(
            Enumerable.Range (0, 10).Select (e => new { e, a = e * e, b = e * 2 })
        ) ).Dump( html 형식으로 표시(ToHtmlString와 조합해서 써 보았다)");

"".Dump(문자열 정형);
Util.ToCsvString
        (
            Enumerable.Range (0, 10).Select (e => new { e, a = e * e, b = e * 2 }),
            new string[]{"e", "a"}
        ).Dump(ToCsvString 넘겨진 객체를 csv 단락의 문자열로 반환. 제2 인수로 출력하는 열 선택이 가능");
"".Dump(WriteCsv 파일에 CSV 형식으로 쓴다. ToCsvString에 써넣은 남은 파일 경로가 증가된 느낌의 인터페이스");
//Util.WriteCsv


Util.ToHtmlString(Enumerable.Range (0, 10).Select (e => new { e, a = e * e, b = e * 2 })).Dump();
```

```
"".Dump("LINQPad 자체 정보 얻기");
Util.GetWebProxy().Dump();
Util.GetMyQueries().Dump("GetMyQueries My Queries 폴더에 있는 LINQ 파일 일람을 표시");
//Util.GetSamples().Dump("GetSamples Samples 폴더에 있는 LINQ 파일 일람을 표시");
//Util.GetSamplesFolder().Dump("GetSamplesFolder Smaple 파일 저장소의 폴더 패스를 얻는다");
Util.CurrentQueryPath.Dump("CurrentQueryPath 현재 파일 패스를 반환. 저장하지 않는 경우는 null 반환");

"".Dump("그 외");
var result = Util.Cmd("ping localhost").Dump("Cmd dos 명령어를 실행하는 것이 가능. 실행 결과를 반환 값으로 받는 것도 가능");

Util.CurrentDataContext.Dump("CurrentDataContext  현재의 DataContext를 취득 가능. 오른쪽 위의 Connection 미 선택 시는 null를 반환)");

var sw = Stopwatch.StartNew();
Util.OnDemand("클릭하면 실행한다", () => sw)
.Dump("OnDemand  클릭 시에 제 2 인수에서 지정된 Func를 실행하고 Dump한다. 이 처리 도중 경과 보존 등에 사용할 수 있을 듯");
```

"".Dump("Run 지정한 linq 파일을 실행할 수 있다. 반환 값 형식은 제 2인수 format으로 지정. 실행한 linq 파일 내에서 dump 하고 있는 내용이 반환된다. ");
//var run = Util.Run(@"", QueryResultFormat.Html);
//run.Dump();

"".Dump("ReadLine Console.ReadLine()에 타이틀을 설정 할 수 있다");
//Util.ReadLine("Console.ReadLine() 에 타이틀을 설정할 수 있다").
Dump("ReadLine Console.ReadLine() 에 타이틀을 설정할 수 있다 ");
//Console.ReadLine().Dump("단순한 Console.ReadLine은 타이틀을 설정할 수 없다");

"".Dump("DisplayWebPage Web 페이지 표시 가능");
//Util.DisplayWebPage("http://google.com", "google");

Util.Cache
코드 실행 후의 값을 저장할 수 있다.

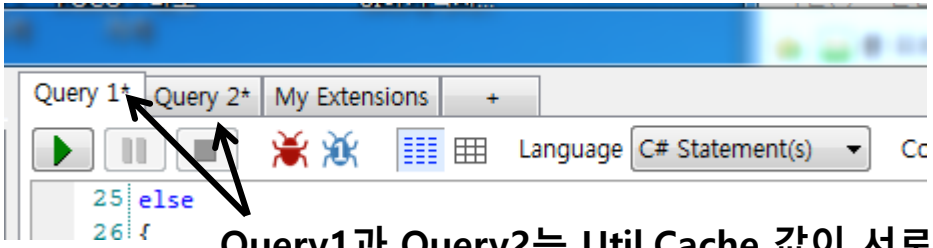
```
var dic = Util.Cache(() => new Dictionary<string, string>());
dic["foo"] = "hoge";
dic.Dump();
```

```
var dic = Util.Cache(() => new Dictionary<string, string>());
dic["bar"] = "fuga";
dic.Dump();
```

위 코드를 각각 실행해보면 두 번째 코드 실행시에 딕셔너리의 내용에 "foo"와 "bar" 2개의 키가 존재하고 있다!
실행이 종료된 후에도 데이터는 저장된다.

지금까지 저장된 데이터를 날리고 처음부터 다시 시작할 때는 [Query]/[Cancel All Threads and Reset]을 선택하면 OK.

Util.Cache에 저장되는 값은 같은 쿼리파일 내에서만 공유된다.



Query1과 Query2는 Util.Cache 값이 서로 다르다.

http://takeshik.org/blog/2014/12/06/try-linqpad/

Util.GetPassword / SetPassword
이 메소드를 사용하면 패스워드(등의 고유 정보)를 코드 내에 직접 쓰지 않아도 된다.

우선 패스워드의 설정.

```
var key = "user";
var password = "pass";
Util.SetPassword(key, password);
```

처럼 Util.SetPassword 를 부르거나 메뉴의 [File]/[Password Manager]에서 패스워드를 관리할 수 있고 그곳에서 설정한다. 설정한 패스워드는 %LOCALAPPDATA%₩LINQPad₩Passwords₩ 에 암호화 되어 저장된다.

이렇게 설정된 패스워드는 Util.GetPassword(key)로 꺼낼 수 있다. 혹은 Util.SetPassword 에서 존재하지 않는 키를 지정하면 비밀 번호를 묻는 대화 상자가 나타나고 거기에 비밀 번호 저장을 할 수 있어서 굳이 미리 비밀 번호를 설정하지 않아다 괜찮다.

DumpContainer 클래스

```
var dc = new DumpContainer().Dump();
while (true)
{
    var now = DateTime.Now;
    dc.Content = new
    {
        Y = now.Year,
        M = now.Month,
        D = now.Day,
        h = now.Hour,
        m = now.Minute,
        s = now.Second,
        ms = now.Millisecond,
    };
    DateTime.Now.ToString("yyyy/MM/dd hh:mm:ss").Dump();
    await Task.Delay(250);
}
```

이 코드를 실행하면 현재 시각의 문자열이 1/4초 마다 흐르는 것과 별개로 현재 시각을 저장한 익명 형의 출력인 표의 내용도(추가해서 출력되는 것은 아니다) 경신되어 간다.

DumpContainer를 사용하면 동적으로 갱신되는 내용을 출력할 수 있다.

http://takeshik.org/blog/2014/12/06/try-linqpad/

Hyperlinq 클래스

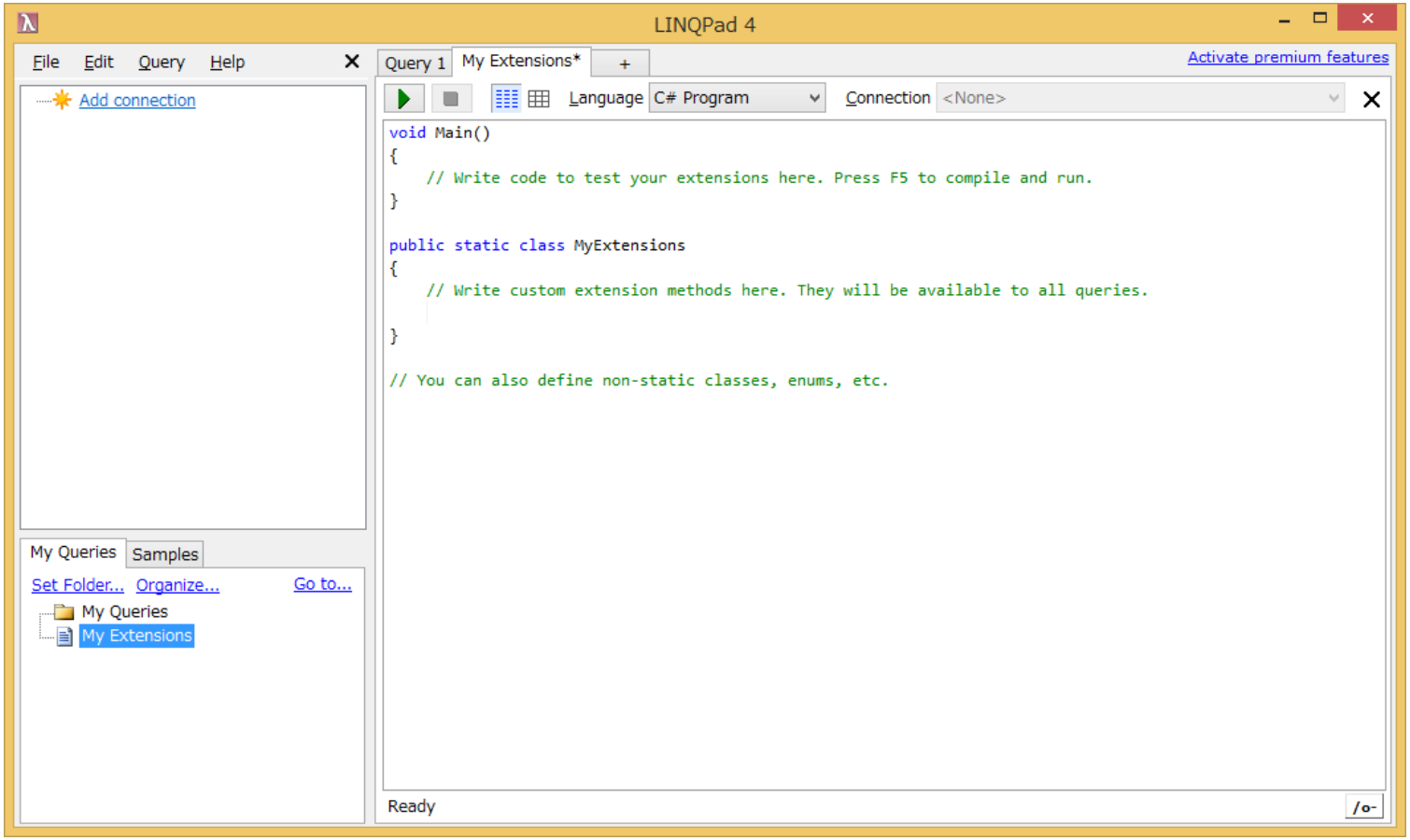
단순히 링크하거나, 처리하거나, 새 linq 파일을 만드는 등 여러 가지 있다

```
void Main()
{
    new Hyperlinq("http://google.com").Dump();
    new Hyperlinq(
            () => Enumerable.Range (0, 10).Select (e => e).Dump(),
            "test",
            true
        ).Dump();

    new Hyperlinq(QueryLanguage.Statements, "var a = 1;a.Dump();", "test").Dump();
}
```

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

My Extensions

자신의 실행 환경에서 항상 참조하고 싶은 처리를 넣어 둘 수 있다.
화면 왼쪽 아래의 [My Queries] 탭 하단에 표시되어 있는 [My Extensions]을 선택.



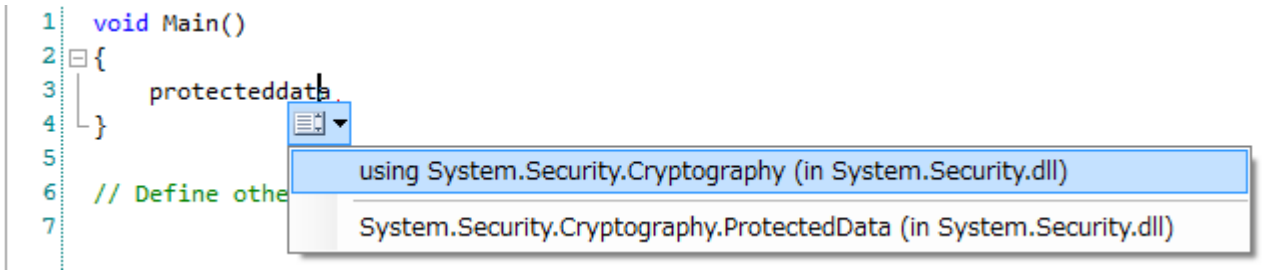
http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

```
public static class MyExtensions
{
    // Write custom extension methods here. They will be available to all queries.

    private readonly static DateTime UnixEpoch = new DateTime(1970, 1, 1, 0, 0, 0, DateTimeKind.Utc);

    public static long ToUnixTime(this DateTime dateTime)
    {
        dateTime = dateTime.ToUniversalTime();
        return (long)dateTime.Subtract(UnixEpoch).TotalSeconds;
    }

    public static DateTime FromUnixTime(this long unixTime)
    {
        return UnixEpoch.AddSeconds(unixTime).ToLocalTime();
    }

    public static IEnumerable<DummyData> GetDummyDatas(int dataCount)
    {
        return Enumerable.Range(0, dataCount).Select(x => new DummyData { Id = x, Name = string.Format("Name{0}", x),
Value = x * 2 });
    }
}

// You can also define non-static classes, enums, etc.
public class DummyData
{
    public int Id { get; set; }
    public string Name { get; set; }
    public int Value { get; set; }
}
```

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

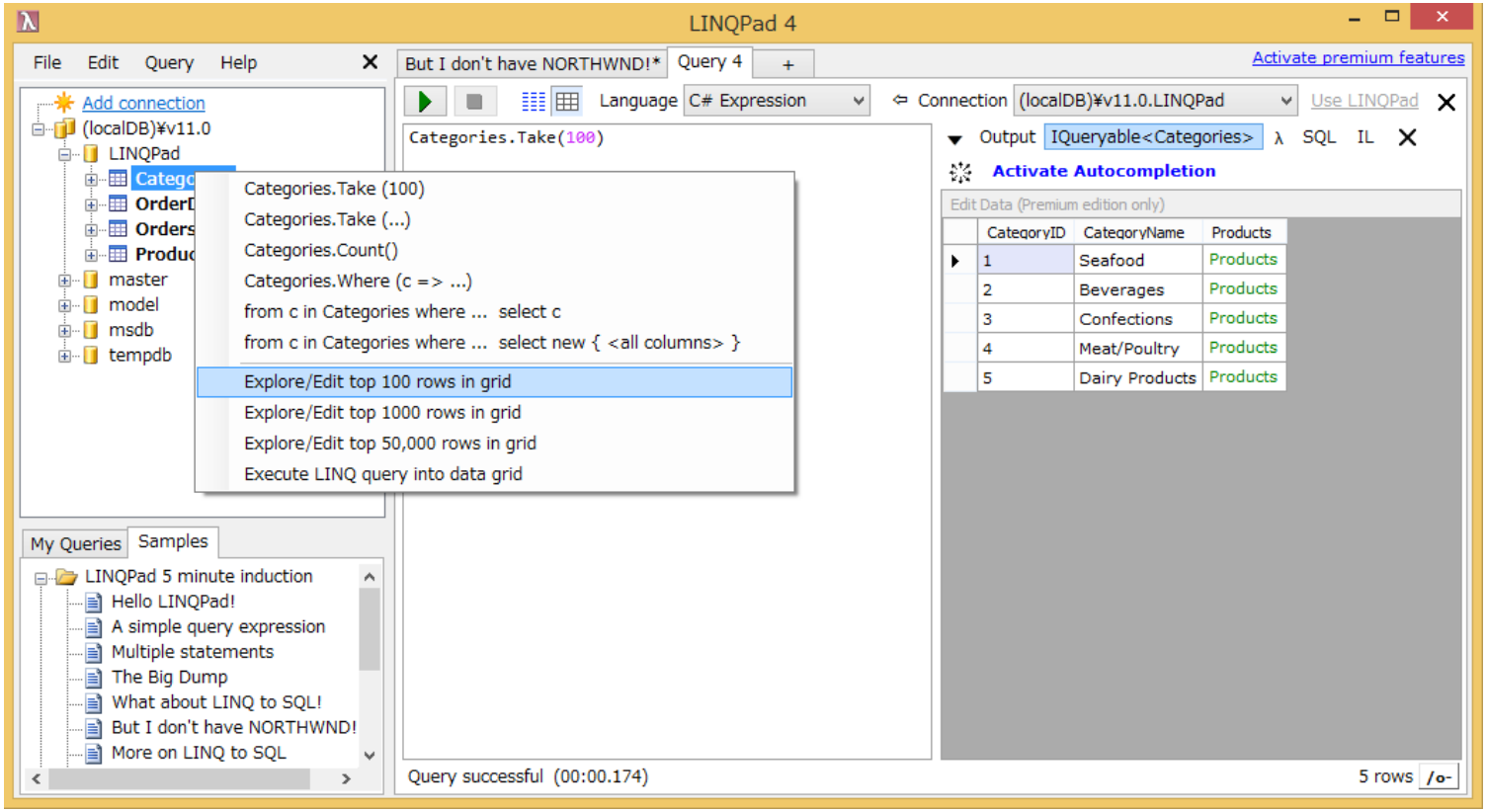
참조하지 않는 형명의 해결(Pro Edition이상)

LINQPad 상에서 아직 참조하지 않는 형명을 입력한 경우 자동으로 해결해 주는 기능.
아직 아무것도 참조하지 않은 상태에서 protecteddata를 입력한 후에 문자열의 뒤쪽에 마우스 커서를 적용하면 다음과 같은 메뉴가 표시된다.



http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

DB에 대한 처리(GUI에서 편집은 Premium Edition만)



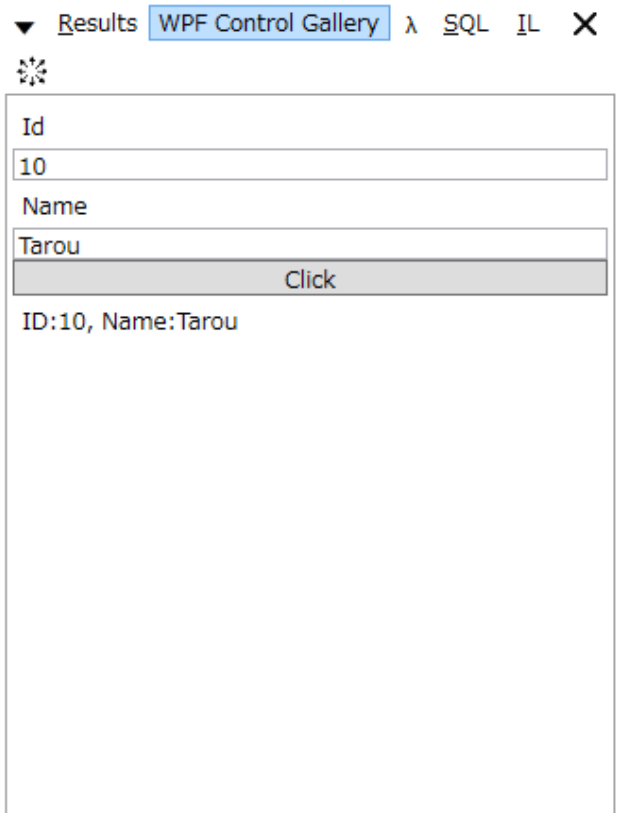
http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

PanelManager 클래스

Dump를 사용하면 WPF와 WindowsForms 컨트롤을 간단하게 표시할 수 있지만 PanelManager를 사용하면 더 편하게 GUI을 사용한 처리를 추가할 수 있다.

참고: http://www.linqpad.net/customvisualizers.aspx

```
void Main()
{
    //GUI部品を準備
    var lblId = new Label { Content = "Id"};
    var txtId = new TextBox();
    var lblName = new Label { Content = "Name"};
    var txtName = new TextBox();
    var btnClick = new Button{ Content = "Click"};
    var lblResult = new Label { Content = "Result"};

    //ボタンをクリックした時のイベントハンドラを追加
    btnClick.Click += (obj, args)=>{
        //テキストボックスの内容を表示
        lblResult.Content =
            string.Format("ID:{0}, Name:{1}", txtId.Text, txtName.Text);
    };

    //PanelManagerに部品を追加
    PanelManager.StackWpfElement (lblId, "WPF Control Gallery");
    PanelManager.StackWpfElement (txtId, "WPF Control Gallery");
    PanelManager.StackWpfElement (lblName, "WPF Control Gallery");
    PanelManager.StackWpfElement (txtName, "WPF Control Gallery");
    PanelManager.StackWpfElement (btnClick , "WPF Control Gallery");
    PanelManager.StackWpfElement (lblResult , "WPF Control Gallery");
}
```



http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

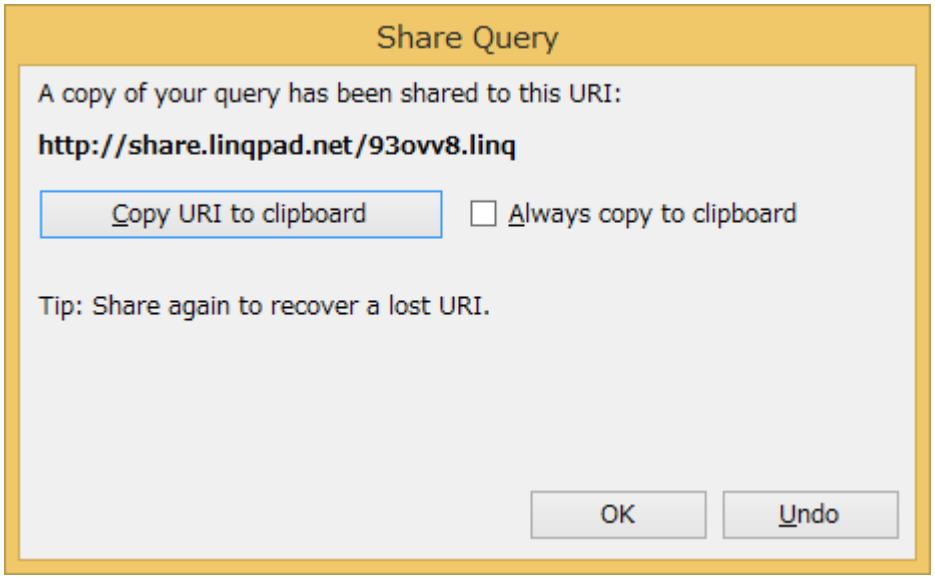
Snippets (Premium Edition)

LINQPad에서도 Snippets를 사용할 수 있다.
기술 형식은 VisualStudio와 같은 형식이므로 이미 작성한 것이 있다면,
LINQPad의 Snippets 폴더로 이동하면 사용할 수 있다.
※기본 값은 My Documents¥ LINQPad Snippets

참고: http://forum.linqpad.net/discussion/287/custom-snippets

Upload to Instant Share

현재 열려 있는 파일의 내용을 올리고 공유하기 위한 기능.
실제로 이 기능을 사용하여 업로드하면, 처리 중 화면 뒤에 다음과 같은 화면이 표시된다



http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7

여기에 기재되어 있는 아래의 URL을 공유하면 다른 사람에게도 파일을 다운로드 하게 할 수 있다.
http://share.linqpad.net/93ovv8.linq

파일 내용에 주의가 필요하다.
LINQ 파일은 다음과 같이 설정 항목을 XML, 실제 소스는 다르게 쓰는 파일로 되어 있다.

```
<Query Kind="Expression">
  <Reference>&lt;RuntimeDirectory&gt;₩System.dll</Reference>
  <Reference>&lt;RuntimeDirectory&gt;₩System.Linq.dll</Reference>
</Query>

DateTime.Now.ToString("yyyy/MM/dd HH:mm:ss")
```

프로젝트의 고유 설정(데이터베이스 접속처, 참조하고 있는 dll 등)이 있는데 그 내용도 업로드 되어 버린다.
URL만 알면 누구든지 다운로드 가능하므로 이용 시는 충분히 주의해야 한다.

명령 라인에서 실행

LINQPad로 작성한 linq를 명령 라인에서 실행할 수도 있다.
LPRUN.exe의 인수에 .linq 파일의 패스를 전달만 하면 된다.
* LPRUN.exe는 LINQPad를 설치한 폴더에 있다

lprun foo.linq
단순히 실행하는 경우는 인수에 전달만

lprun -format=html foo.linq > output.html
실행 결과를 남기경우 -format에서 데이터 형식을 지정 가능하다.
-format={text|html|htmlfrag|csv|csvi}
출력되는 데이터는 .linq 안에서 Dump를 실행하는 데이터이다.

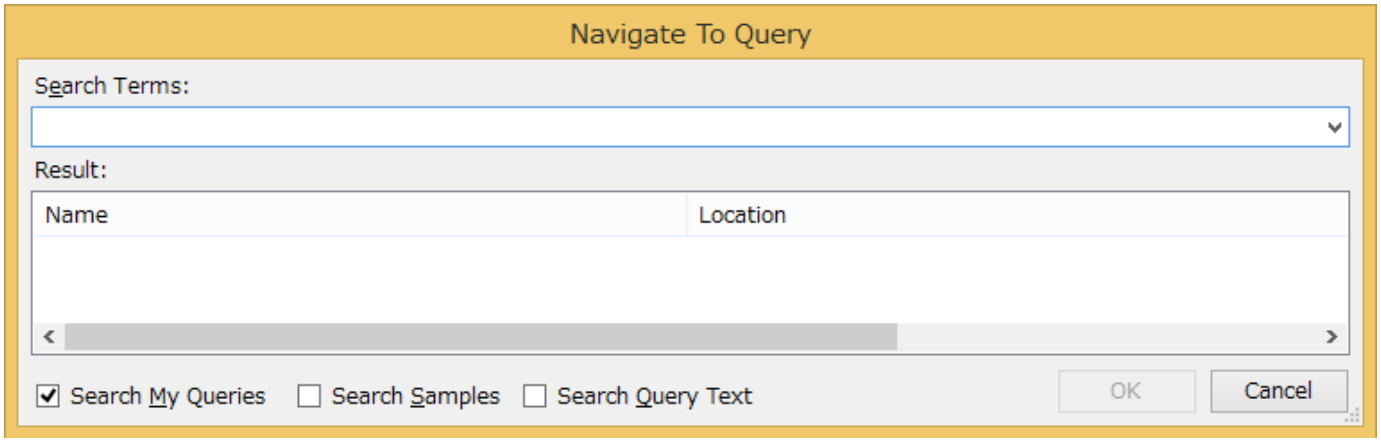
lprun "queries₩foo bar.linq" CustomArg
인수를 넘길 수도 있다.

lprun script1.linq | lprun script2.linq
다른 쿼리 실행 결과를 pipe로 다음 처리에 넘겨줄 수 있다.

자세한 것은 아래를 참조
http://www.linqpad.net/lprun.aspx

작성한 LINQPad의 소스 검색

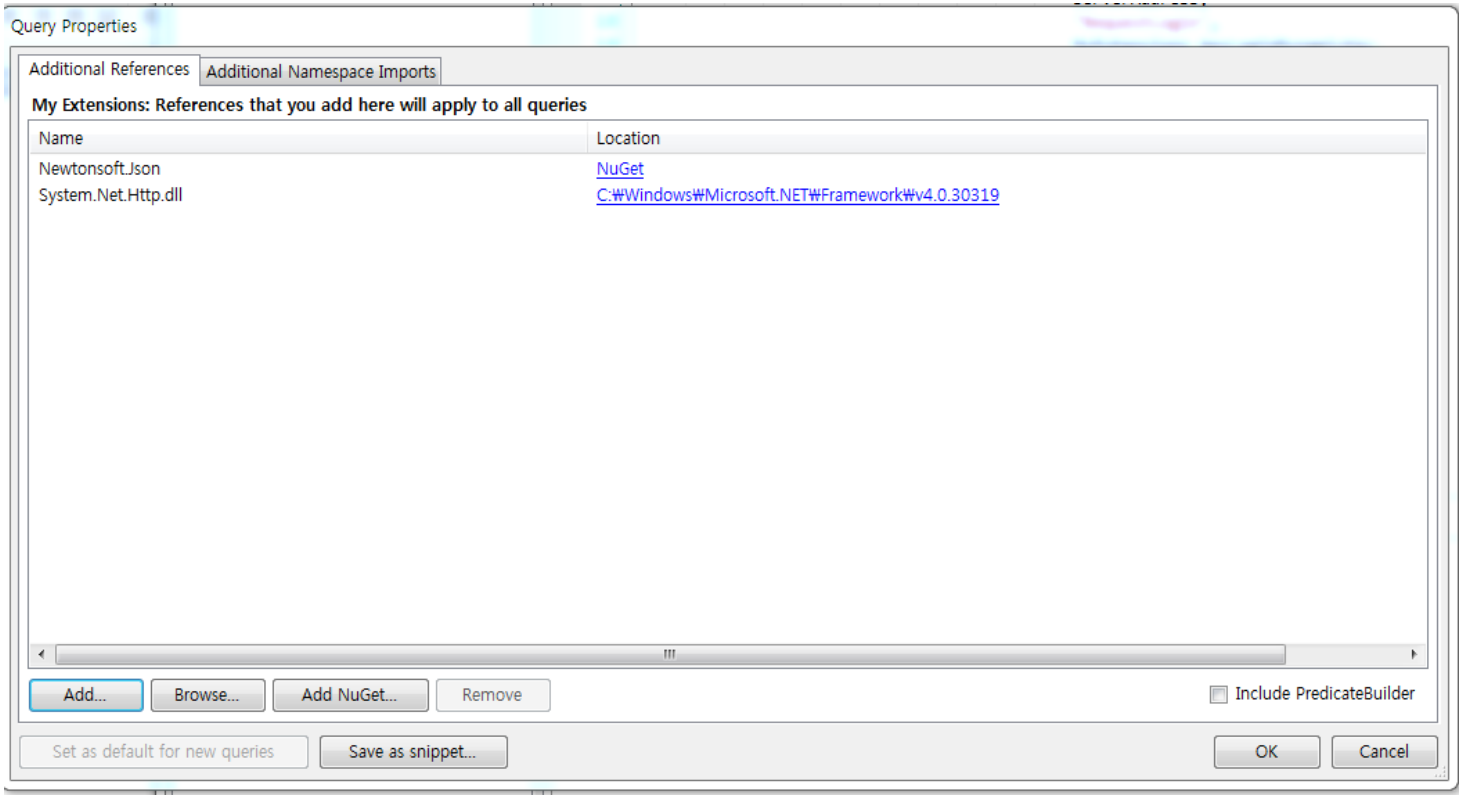
ctrl + ,
입력에서 [Navigate To Query] 화면이 표시된다.
(화면 왼쪽 아래의 영역 My Queries 탭의 [Go to...]를 눌렸을 때와 같은 화면.

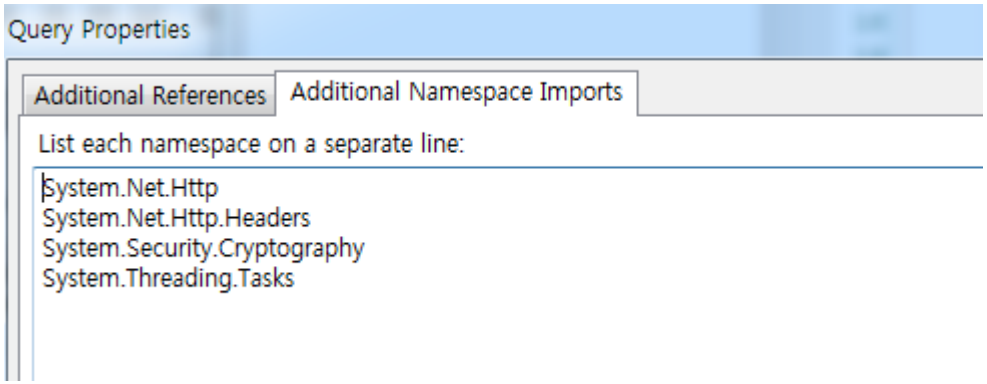


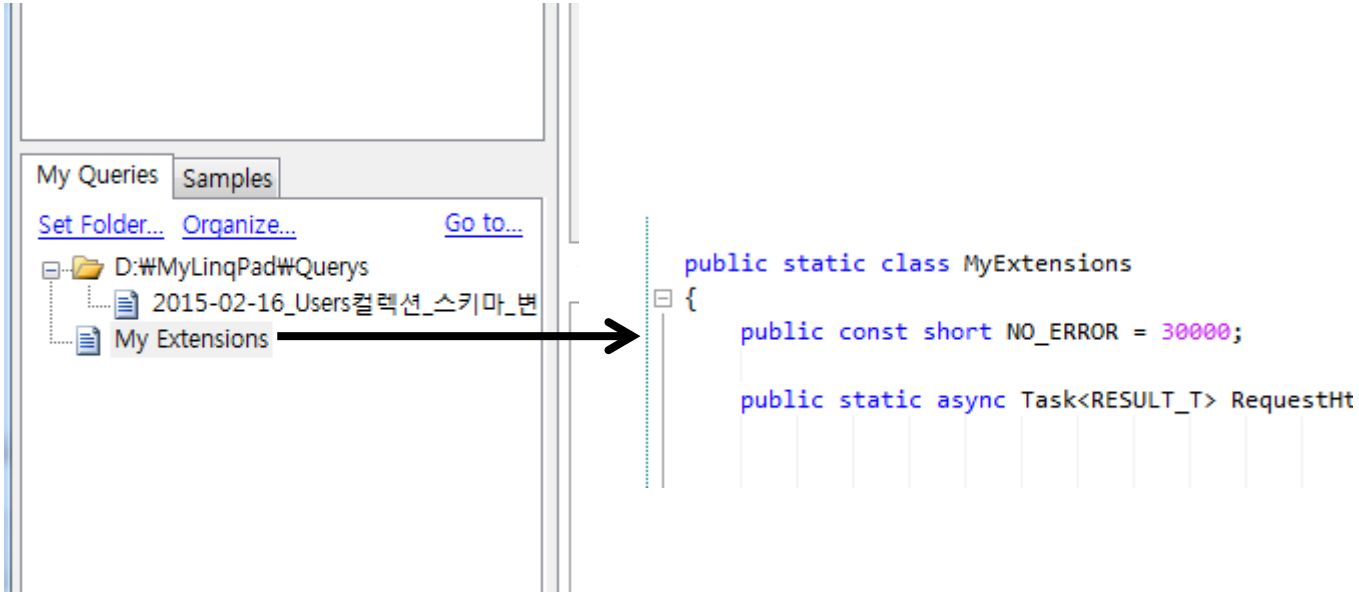
화면 하부에 체크 박스가 있지만 검색하는 범위 변경이 가능.
파일명이 일치하는 것이 검색 대상이지만, [Search Qeury Text]를 선택함으로써 파일 내의 검색도 가능하다.

- Search My Queries: My Queries 폴더에 설정한 폴더에서 파일 이름이 일치하는 것을 검색한다.
- Search Samples: 샘플 파일 중에서 검색한다.
- Search Qeury Text: 실제 파일 내를 검색한다.

http://qiita.com/kazuhisam3/items/293103991a9f0bb8dcb7


Query Properties
Additional References
Additional Namespace Imports
My Extensions: References that you add here will apply to all queries
Name
Location
Newtonsoft.Json
NuGet
System.Net.Http.dll
C:₩Windows₩Microsoft.NET₩Framework₩v4.0.30319
Add...
Browse...
Add NuGet...
Remove
Include PredicateBuilder
Set as default for new queries
Save as snippet...
OK
Cancel


Query Properties
Additional References
Additional Namespace Imports
List each namespace on a separate line:
System.Net.Http
System.Net.Http.Headers
System.Security.Cryptography
System.Threading.Tasks



My Extensions에 서버에 요청할 때 사용할 메소드와 구조체(패킷용)를 정의한다.

```
public static async Task<RESULT_T> RequestHttpAESEncry<REQUEST_T, RESULT_T>(
                                                    string address,
                                                    string reqAPI,
                                                    string loginSeq,
                                                    string userID,
                                                    REQUEST_T reqPacket) where RESULT_T : new()
{
                var resultData = new RESULT_T();

                var api = "http://" + address + "/GameService/" + reqAPI;
                var jsonText = Newtonsoft.Json.JsonConvert.SerializeObject(reqPacket);

                var encryData = AESEncrypt(loginSeq, jsonText);
                var sendData = new REQ_DATA { LSeq = loginSeq, ID = userID, Data = encryData };
                var requestJson = Newtonsoft.Json.JsonConvert.SerializeObject(sendData);

                var content = new ByteArrayContent(Encoding.UTF8.GetBytes(requestJson));
                content.Headers.ContentType = new MediaTypeHeaderValue("application/json");

                var Network = new System.Net.Http.HttpClient();
                var response = await Network.PostAsync(api, content).ConfigureAwait(false);

                if (response.IsSuccessStatusCode == false)
                {
                    Console.WriteLine("[Error] RequestHttpAESEncry Network");
                    return resultData;
                }

                var responeString = await response.Content.ReadAsStringAsync();
                var responseJson = Newtonsoft.Json.JsonConvert.DeserializeObject<RES_DATA>(responeString);

                if (responseJson.Result != NO_ERROR)
                {
                   Console.WriteLine("[Error] RequestHttpAESEncry Response Error: {0}", responseJson.Result);
                   return resultData;
                }

                var decryptData = AESDecrypt(loginSeq, responseJson.Data);
                resultData = Newtonsoft.Json.JsonConvert.DeserializeObject<RESULT_T>(decryptData);
                return resultData;
}
```

My Extensions
클래스 안에
정의한다

My Extensions 클래스 안에 정의한다

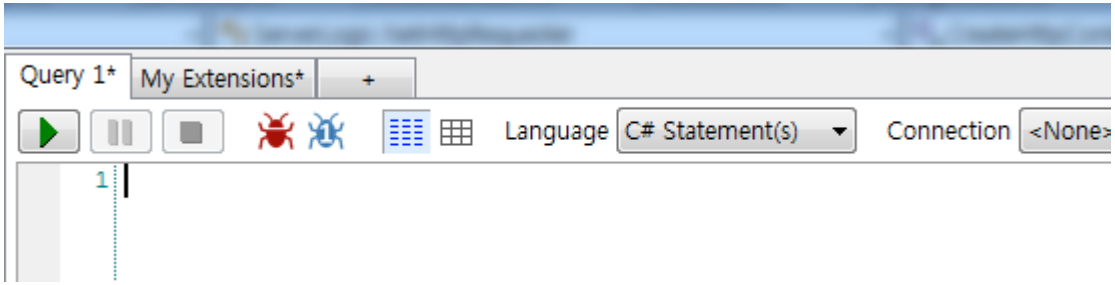
```
static string AesIV = @"!QAZ2WSX#EDC4RFV";  //16
static string AesKey = @"5TGB&7U(IK<";       // 11
public static string AesLoginDynamicKey = @"d_&^t";       // 5

static string AESEncrypt(string dynamincKey, string text)
{
   var comleteAesKey = dynamincKey.Substring(0, 5) + AesKey;

   System.Security.Cryptography.AesCryptoServiceProvider aes = new
System.Security.Cryptography.AesCryptoServiceProvider();
   aes.BlockSize = 128;
   aes.KeySize = 128;
   aes.IV = Encoding.UTF8.GetBytes(AesIV);
   aes.Key = Encoding.UTF8.GetBytes(comleteAesKey);
   aes.Mode = CipherMode.CBC;
   aes.Padding = PaddingMode.PKCS7;

   // 문자열을 바이트형 배열로 변환
   byte[] src = Encoding.Unicode.GetBytes(text);

   // 암호화 한다
   using (ICryptoTransform encrypt = aes.CreateEncryptor())
   {
        byte[] dest = encrypt.TransformFinalBlock(src, 0, src.Length);

       // 바이트형 배열에서 Base64형식의 문자열로 변환
      return Convert.ToBase64String(dest);
   }
}
```

```
static string AESDecrypt(string dynamincKey, string text)
{
   var comleteAesKey = dynamincKey.Substring(0, 5) + AesKey;

   AesCryptoServiceProvider aes = new AesCryptoServiceProvider();
   aes.BlockSize = 128;
   aes.KeySize = 128;
   aes.IV = Encoding.UTF8.GetBytes(AesIV);
   aes.Key = Encoding.UTF8.GetBytes(comleteAesKey);
   aes.Mode = CipherMode.CBC;
   aes.Padding = PaddingMode.PKCS7;

   // Base64 형식의  문자열에서 바이트형 배열로 변환
   byte[] src = System.Convert.FromBase64String(text);

   // 복호화 한다
   using (ICryptoTransform decrypt = aes.CreateDecryptor())
   {
      byte[] dest = decrypt.TransformFinalBlock(src, 0, src.Length);
      return Encoding.Unicode.GetString(dest);
   }
}
```

My Extensions 클래스 밖에 정의한다

```
}

public static class MyExtensions
{...}

// You can also define non-static classes, enums, etc.
public enum MARKET_TYPE : short
{
    ANDROID = 29,
    iOS = 30,
}

#region PACKET_DEFINE
public struct REQ_DATA
{
    public string LSeq;
    public string ID;
    public string Data;
}

public struct RES_DATA
{
    public short Result;
    public string Data;
}

// 로그인 요청
public struct REQ_LOGIN_DATA
```

```
public enum MARKET_TYPE : short
{
   ANDROID = 29,
   iOS = 30,
}

#region PACKET_DEFINE
public struct REQ_DATA
{
   public string LSeq;
   public string ID;
   public string Data;
}

public struct RES_DATA
{
   public short Result;
   public string Data;
}

// 로그인 요청
public struct REQ_LOGIN_DATA
{
   public string ID;
   public string PW;
   public short ClientVer;
   public short DataVer;
   public short MarketType;
}

public struct RES_LOGIN_DATA
{
   public short Result;   // 요청에 대한 결과
   public string AT;          // AuthToken
   public string LSeq;        // LoginSeq;
}
```




```
var serverAddress = "12.10.50.223:21651";
var request = new REQ_LOGIN_DATA()
{
   ID = "jacking",
   PW = "~123qwe",
   ClientVer = 0,
   DataVer = 0,
   MarketType = (short)MARKET_TYPE.ANDROID
};

var result = MyExtensions.RequestHttpAESEncry<REQ_LOGIN_DATA, RES_LOGIN_DATA>(
                                        serverAddress,
                                        "RequestLogin",
                                        MyExtensions.AesLoginDynamicKey,
                                        request.ID, request);

if(result.Result.Result == MyExtensions.NO_ERROR)
{
    Console.WriteLine("성공 : {0}", result.Result.Dump());
}
else
{
    Console.WriteLine("에러 : {0}", result.Result.Result);
}
```