

Entity Framework 快速上手

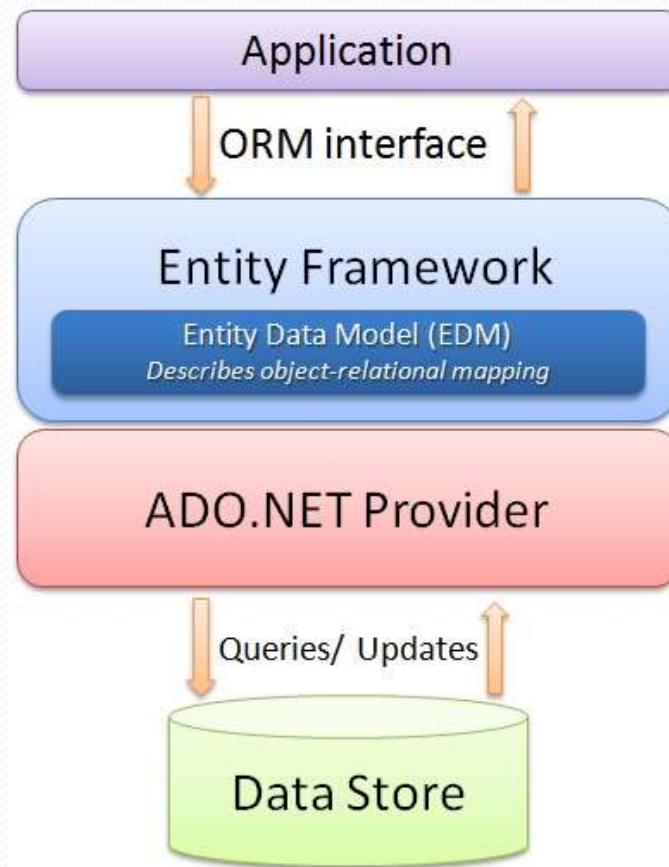
Alan Tsai

Agenda

- Entity Framework
 - 快速介紹ORM
 - EF 整體介紹
 - Modeling
- Linq
 - 概念
 - 基本語法
- Tips 和工具

ORM為什麼出現

- 像操作檔案的方式操作DB
 - DataReader
 - 保持連線
 - 只能往前讀
- 像操作Table方式操作DB
 - DataSet – DataTable, DataColumn, DataRow
- 像操作Object方式操作DB
 - Linq to Sql
 - Entity Framework



Entity Framework 目前狀況

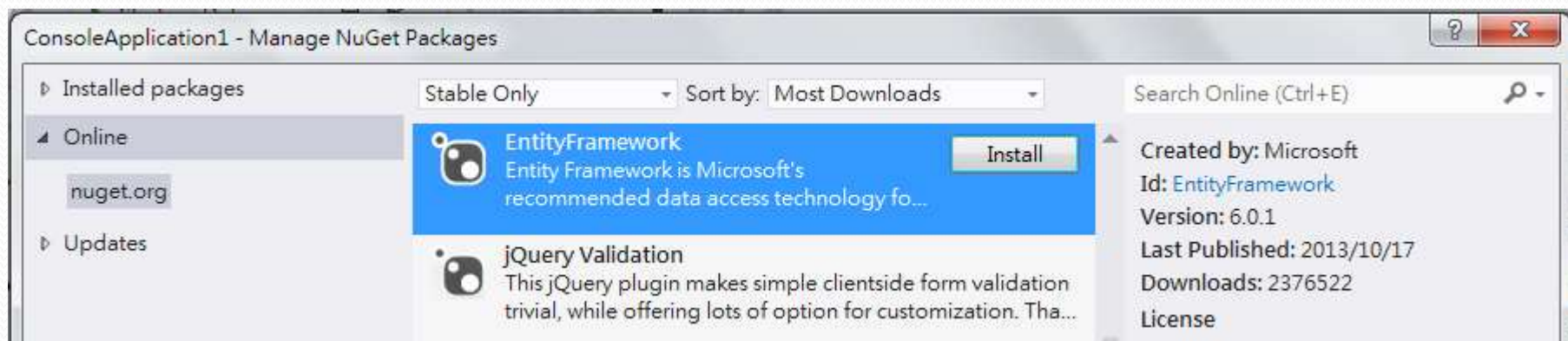
- 最新版本是 EF 6.0.1
- 從EF 5 之後都是Open source
(<http://entityframework.codeplex.com/>)
- 從EF 6 之後慢慢和.Net Framework 切割出來
 - 在EF 5 一定要是.Net Framework 4.5才有Enum 和Spatial 支援
 - 但是EF 6 就不需要了

Entity Framework 6 Highlight

- Async 支援
- 能夠在斷線情況下重新連線
- 在.Net 4.0 支援Enum和Spatial
- Code First支援map SP
- 更多新東西 (<http://msdn.microsoft.com/en-us/data/jj574253>)

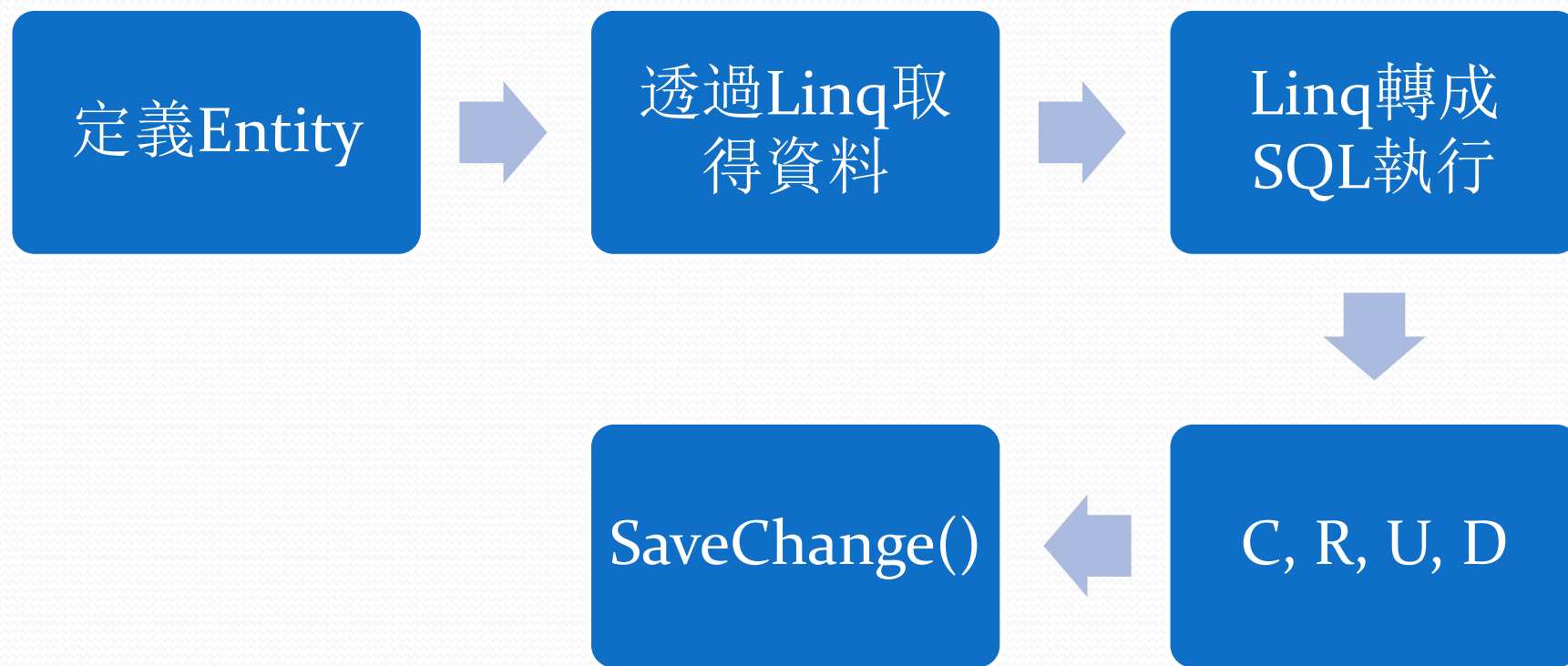
如何取得EF

- Nuget



```
PM> Install-Package EntityFramework
```

EF 流程



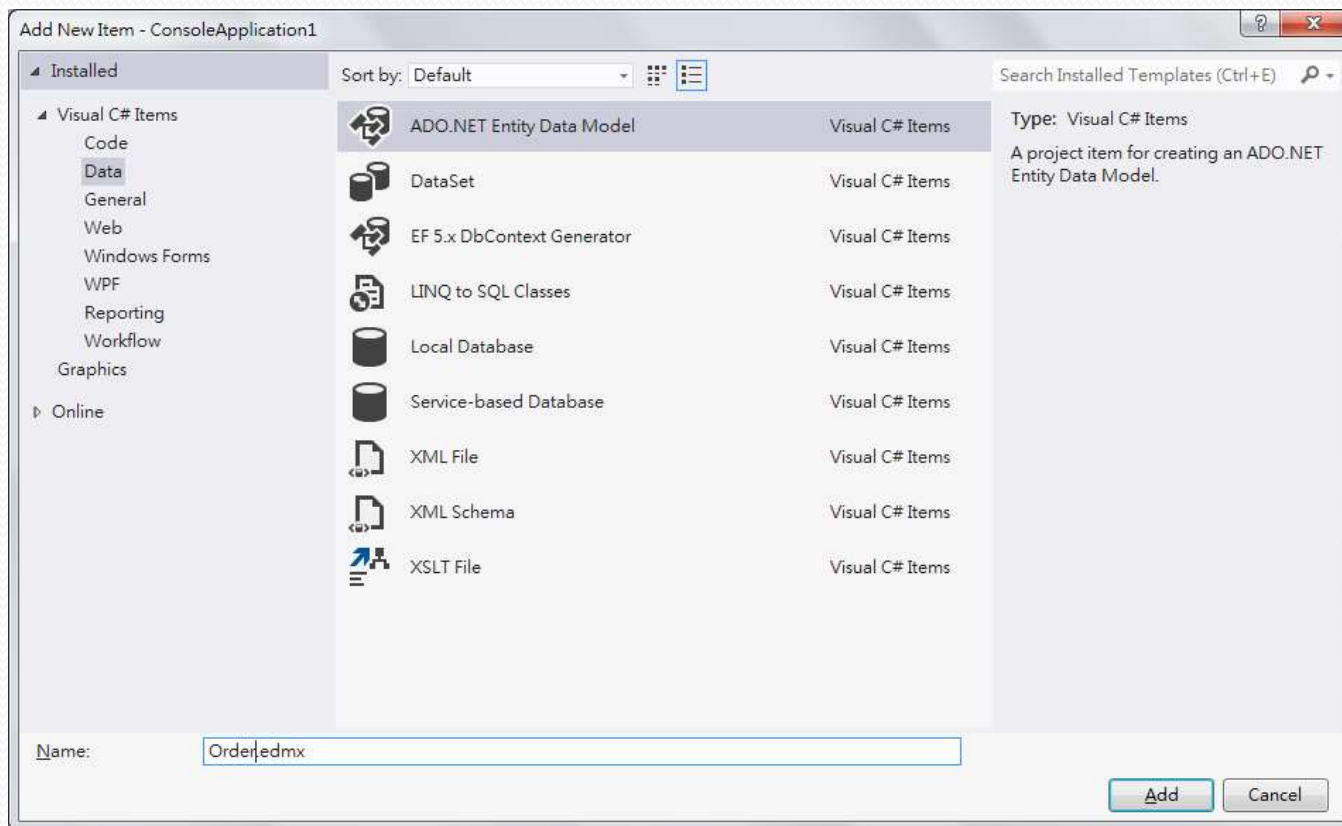
EF Entity Mapping

- 使用拖拉方式（GUI）
 - Database First
 - Model First
- OOP方式
 - Code First

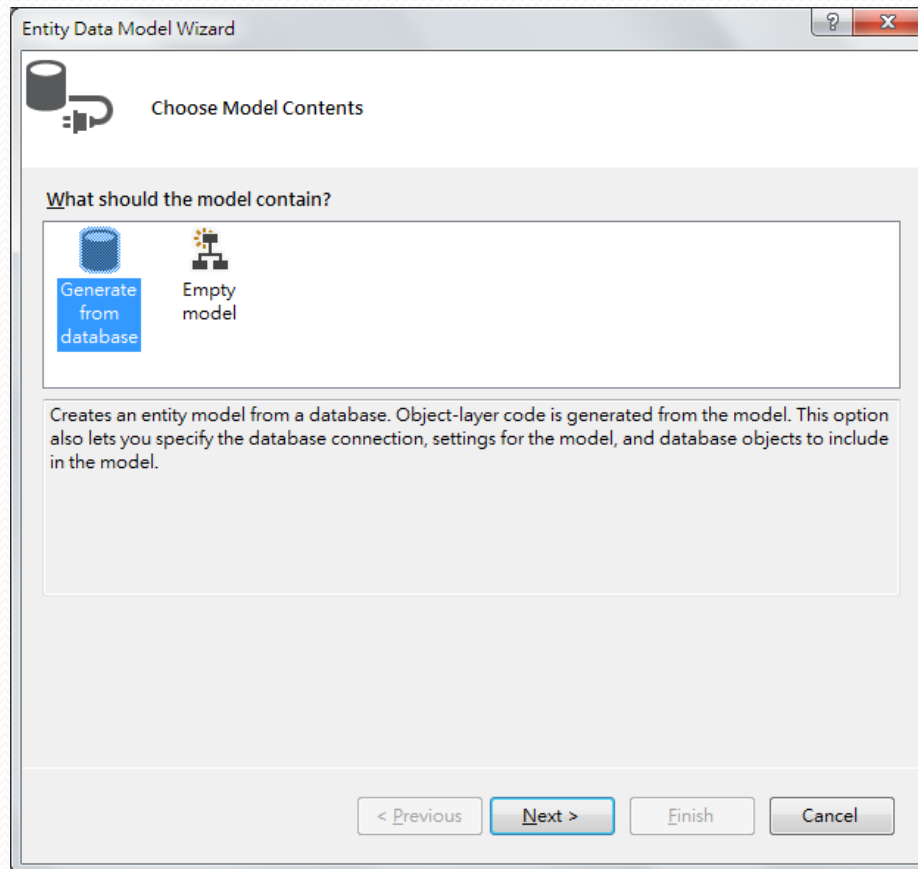
EDMX

- 就是一個XML檔案
- 基本有三個部份：
 - CSDL - Conceptual schema definition language.
 - Entity (Model)
 - SSDL - Store schema definition language
 - 代表DB的結構 – 實際的Table
 - MSL - Mapping specification language
 - Entity 和 DB的對應資料

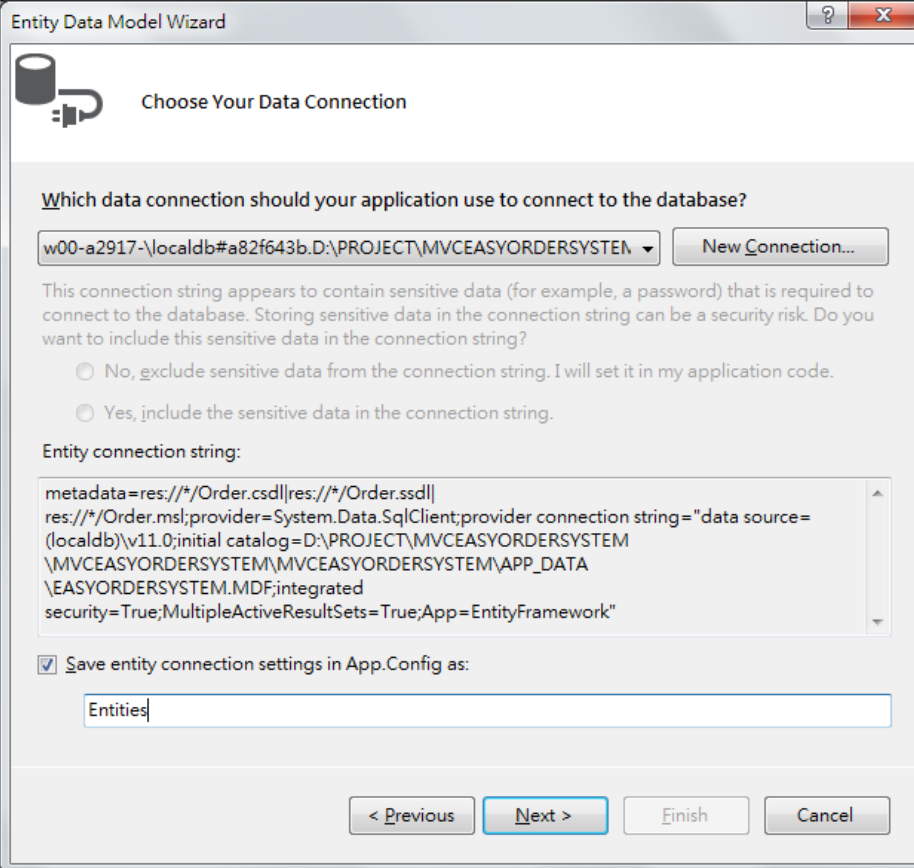
新增EDMX



Database First 還是 Model First



選擇Connection



The image shows a screenshot of the 'Entity Data Model Wizard' dialog box. The title bar reads 'Entity Data Model Wizard'. The main heading is 'Choose Your Data Connection'. Below this, a question asks 'Which data connection should your application use to connect to the database?'. A dropdown menu shows a connection string starting with 'w00-a2917-\\localhost#a82f643b.D:\\PROJECT\\MVCEASYORDERSYSTEM'. To the right of the dropdown is a 'New Connection...' button. Below the dropdown, a warning message states: 'This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?'. There are two radio buttons: 'No, exclude sensitive data from the connection string. I will set it in my application code.' and 'Yes, include the sensitive data in the connection string.'. Below this, the 'Entity connection string:' is displayed in a text box. The connection string is: 'metadata=res://*/Order.csdl|res://*/Order.ssdl|res://*/Order.msl;provider=System.Data.SqlClient;provider connection string="data source=(localhost)\\v11.0;initial catalog=D:\\PROJECT\\MVCEASYORDERSYSTEM\\MVCEASYORDERSYSTEM\\MVCEASYORDERSYSTEM\\APP_DATA\\EASYORDERSYSTEM.MDF;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework"'. At the bottom, there is a checkbox labeled 'Save entity connection settings in App.Config as:' which is checked. Below the checkbox is a text box containing the word 'Entities'. At the very bottom, there are four buttons: '< Previous', 'Next >', 'Finish', and 'Cancel'.

Entity Data Model Wizard

Choose Your Data Connection

Which data connection should your application use to connect to the database?

w00-a2917-\\localhost#a82f643b.D:\\PROJECT\\MVCEASYORDERSYSTEM

New Connection...

This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?

☐ No, exclude sensitive data from the connection string. I will set it in my application code.

☐ Yes, include the sensitive data in the connection string.

Entity connection string:

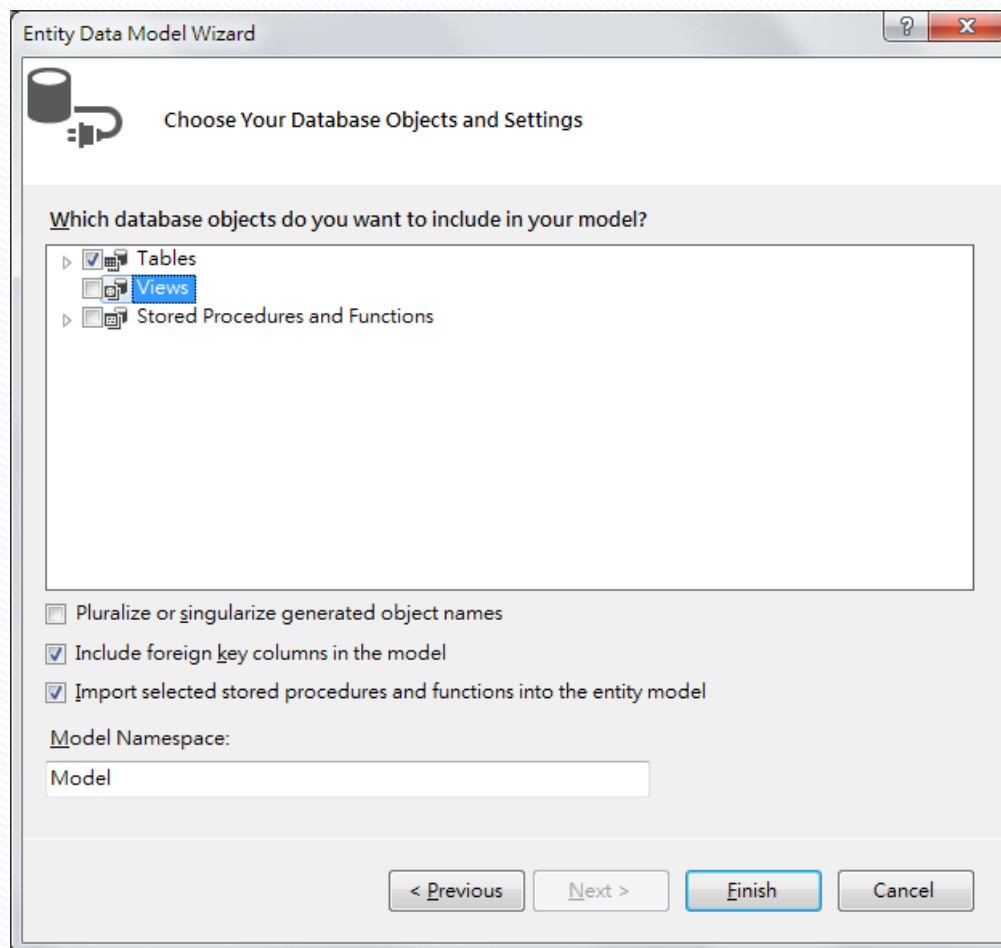
metadata=res://*/Order.csdl|res://*/Order.ssdl|
res://*/Order.msl;provider=System.Data.SqlClient;provider connection string="data source=
(localhost)\\v11.0;initial catalog=D:\\PROJECT\\MVCEASYORDERSYSTEM
\\MVCEASYORDERSYSTEM\\MVCEASYORDERSYSTEM\\APP_DATA
\\EASYORDERSYSTEM.MDF;integrated
security=True;MultipleActiveResultSets=True;App=EntityFramework"

☒ Save entity connection settings in App.Config as:

Entities

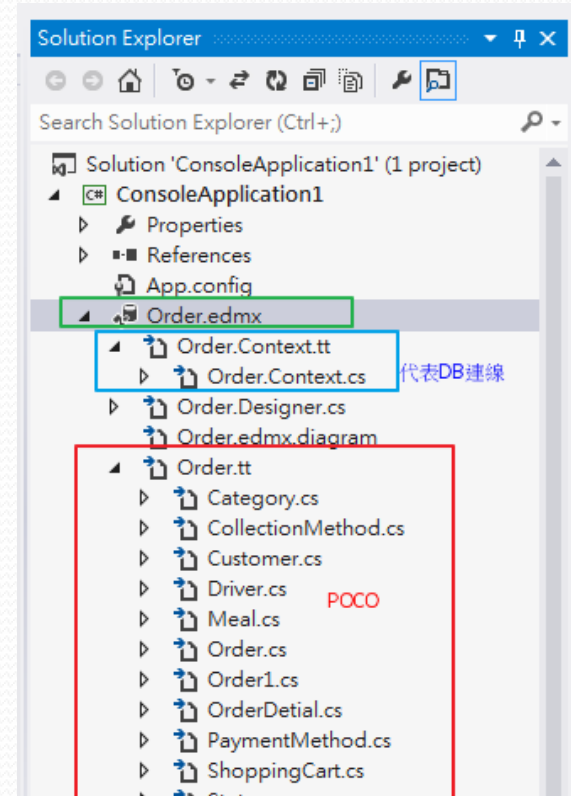
< Previous Next > Finish Cancel

選擇要對應的Model



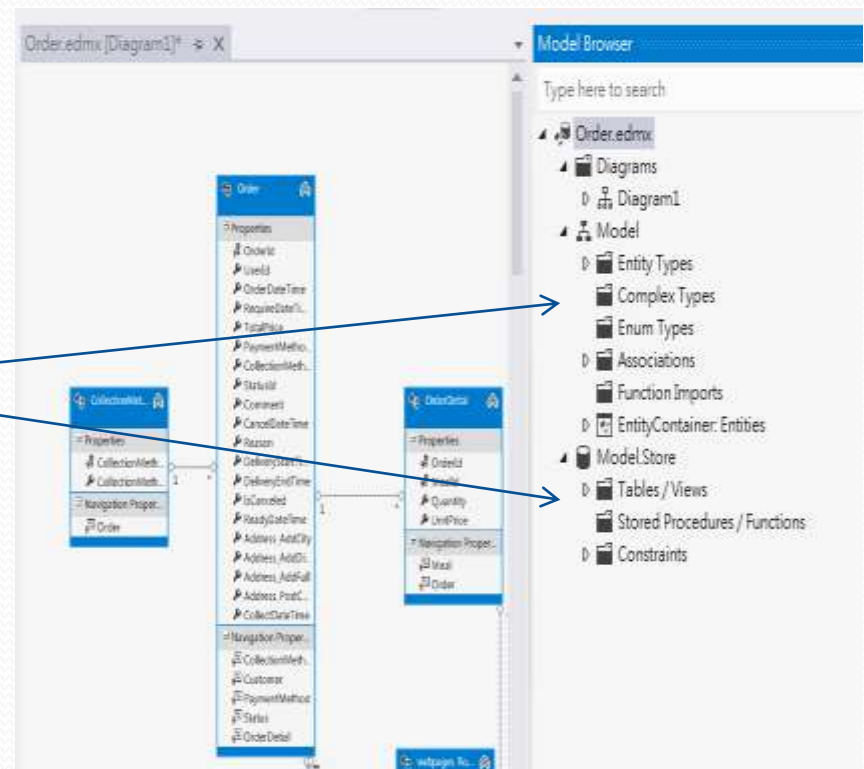
EDMX

- 從EDMX產出對應的
 - DbContext – 連線DB
 - POCO – Plain Old CLR Object
 - 和POJO 一樣概念
- 透過T4 Template



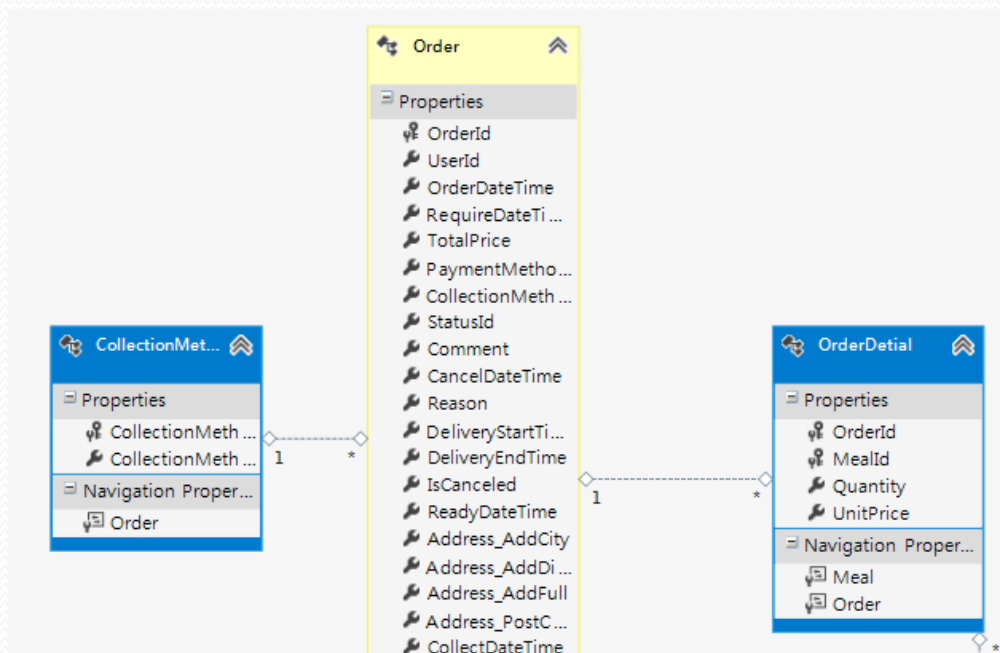
EDMX 和 Model Browser對應

```
Order.edmx X
<?xml version="1.0" encoding="utf-8"?>
<edmx:Edmx Version="3.0" xmlns:edmx="http://schemas.microsoft.com/ado/2009/11/edmx">
  <!-- EF Runtime content -->
  <edmx:Runtime>
    <!-- SSOL content -->
    <edmx:StorageModels>...</edmx:StorageModels>
    <!-- CSDL content -->
    <edmx:ConceptualModels>...</edmx:ConceptualModels>
    <!-- C-S mapping content -->
    <edmx:Mappings>...</edmx:Mappings>
  </edmx:Runtime>
  <!-- EF Designer content (DO NOT EDIT MANUALLY BELOW HERE) -->
  <Designer xmlns="http://schemas.">...</Designer>
</edmx:Edmx>
```



兩個VS 2012 增加的功能

- 可以給Entity不同的顏色來區分：
- 可以把Entity切割到不同圖片



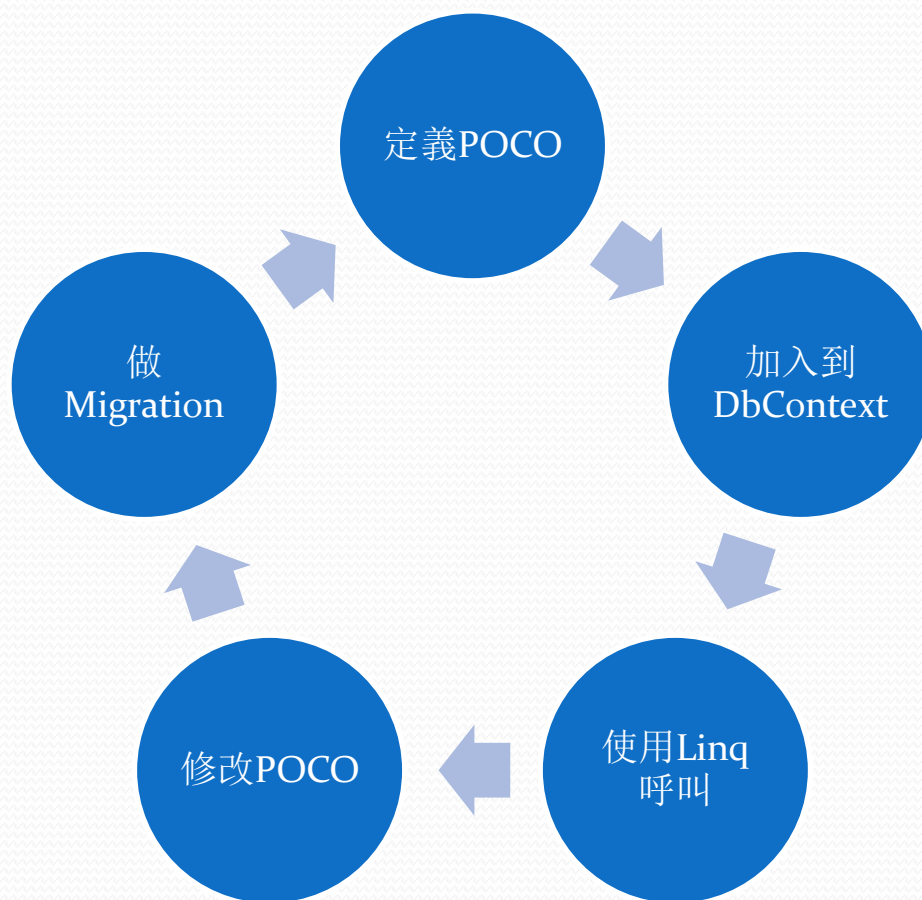
Demo

Database First EDMX

Code First

- 如果你不喜歡使用GUI，那麼可以使用Code First
- 使用POCO來定義Entity
- 和MVC一樣使用Convention over Configuration
 - 可以使用Attribute
 - 可以使用Fluent API
- 定義好的POCO加入DbContext

Code First流程



Connection String

- 設定名字
- Provider
- connectionString
- 在DbContext的建構子就能夠指定要使用哪一個Connection。
- <connectionStrings>
- <add name="BlogContext"
- providerName="System.Data.SqlClient"
- connectionString="Server=(localdb)\v11.0;Database=Bloggging;Integrated Security=True;"/>
- </connectionStrings>

Convention

- Property 叫做 Id 或者 {Class}Id 是 int 會被當做 PK
- Property 是另外一個 Type 預設為 Navigation
- Property 是另外一個 Type 的名稱加上 Id 是 FK

Attribute

- `System.ComponentModel.DataAnnotations`
- `[Key]` – 設定那個為PK
- `[Required]` – 不可為null
- `[MaxLength]` – string 長度
- `[NotMapped]` – 不要對應
- `[ComplexType]`

Lazy Loading

- 我們有Navigation但是不一定每一次都需要對應的內容。
- 因此當我們把一個Navigation 加上virtual的時候，告訴EF這個可以Lazy Load。
- 如果在Context裡面呼叫到Navigation，那麼就會在那個時候Load
- 假設在Context沒有呼叫，然後到了Context外面才呼叫，那麼就不會Load

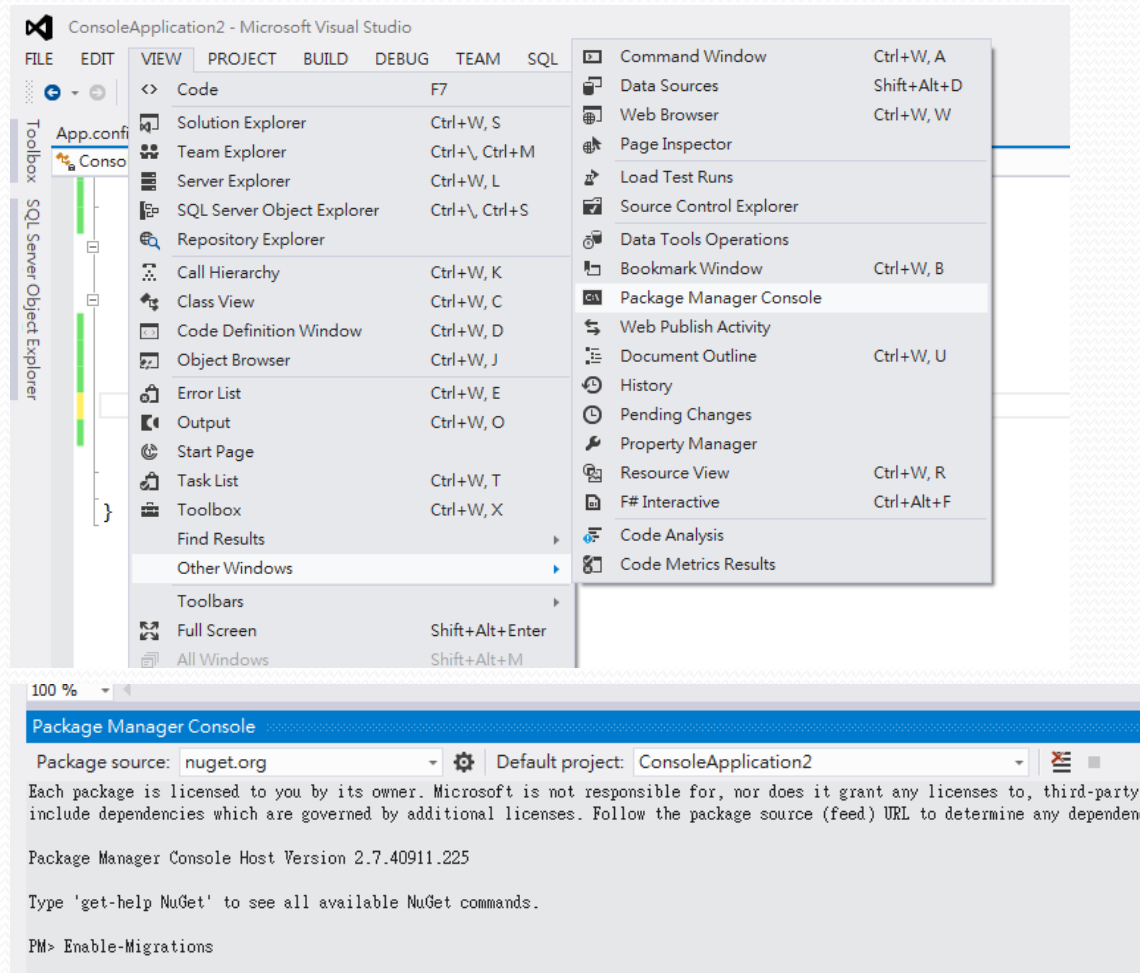
Demo

使用Code First快速建立一個Table

Migration

- 當有異動的時候，就會需要執行Migration
- Migration將會記錄每一次異動的地方，然後更新資料庫
- Migration有分為手動和自動

啟動Migration



Migration 資料夾

- 有一個Configuration.cs
 - 用來設定一些Migration執行時候動作
 - 例如讓DB產生預設值，或者啟動自動migration
- 如果使用手動Migration會有對應檔案產生

手動Migration

- 每一次變動在Package Manage Console 輸入
 - Add-Migration – 顯示異動內容
- 更新到資料庫
 - Update-Database

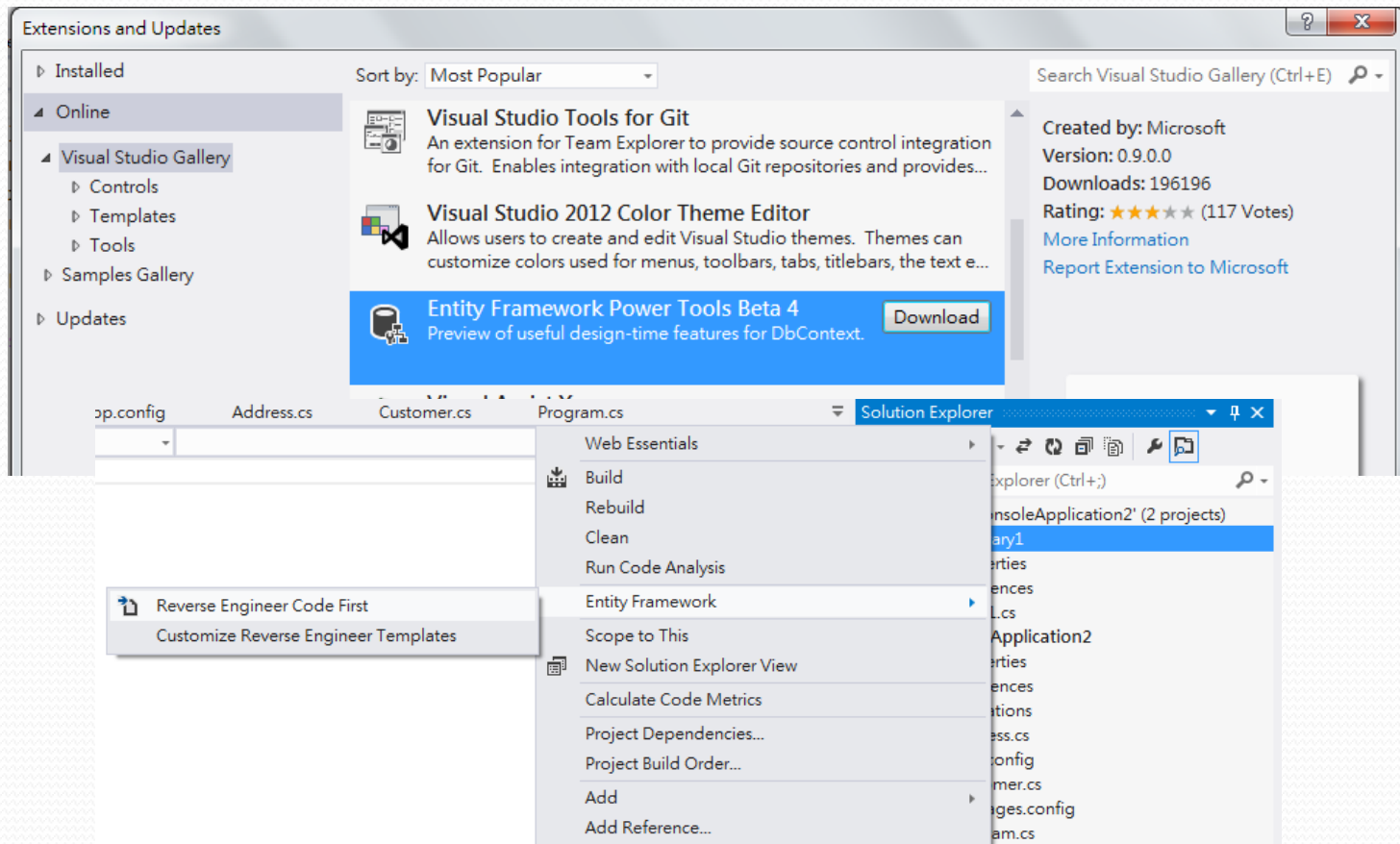
自動Migration

- 修改Configuration.cs 的建構子
 - AutomaticMigrationsEnabled = true;
 - AutomaticMigrationDataLossAllowed = true;
- 在啟動的時候改變啟動模式：
 - CreateDatabaseIfNotExists(預設)
 - DropCreateDatabaseIfModelChange
 - DropCreateDatabaseAlways
 - MigrateDatabaseToLatestVersion

Demo

Migration

Code First 已存在DB



對EF做CRUD

- DbContext就像一個資料庫的物件
- 每一個在DbContext有註冊的POCO就像一個List一樣，所以就和平常List一樣直接做Add、Remove。
- Update可以先用Read找出來，然後做修改儲存，或者是透過DbContext.Entry<T>().SetEntity = modify然後在儲存。

回顧

- 3種方式
 - Database First
 - Model First
 - Code First
- 更多範例：
 - <http://msdn.microsoft.com/en-US/data/ee712907>

LINQ

Language-Integrated Query

Linq 是什麼

- 可以把它想成在程式裡面使用SQL
- 從告訴程式如何做一件事情，變成告訴程式我們要什麼：
- 例如取得一個Dictionary<String, String> Value叫做Alan

```
foreach (var item in names)
{
    if (item.Value == "Alan")
    {
        isFind = true;
    }
}
```

```
isFind = names.Any(x => x.Value == "Alan");
```

Linq 構成

- Query Syntax – 像是 SQL

```
var a = from item in names
        where item.Value == "Alan"
        select item;
```

- Method Syntax

```
var a = names.Where(x => x.Value == "Alan");
```

- Extension Method
- Lambda Expression
- var – strong type

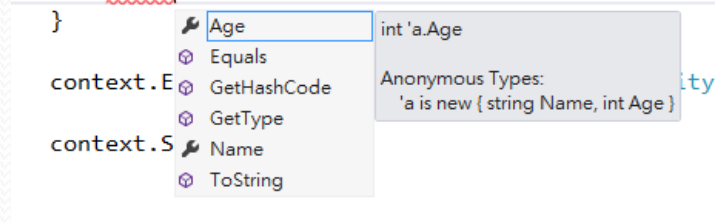
Linq使用範圍

- Linq to Entity
- Linq to Xml
- Linq to Object – IEnumerable<>
- Linq to Excel – 第三方快速讀取Excel資料

Where() Select() Group()

- Where()可以用來取得符合條件的物件
- Select()能夠project我們要的內容。例如在一個Customer的object，我們只需要Name和Age，我們可以：

```
var customers = context.Customer.Select
    (x =>
        new
        {
            Name = x.Name,
            Age = x.Age
        }
    ).ToList();
foreach (var item in customers)
{
    item.
```



Skip() 和 Take() 做分頁

- Skip() 能夠跳過指定的幾個。
- Take() 能夠只取得特定的幾個。

```
names.Skip(10).Take(10);
```

Linq 101 sample

- <http://code.msdn.microsoft.com/101-LINQ-Samples-3fb9811b>
- 幾乎所有常用的都有例子。

Linq 在EF裡面的Eager Load

- EF只有在真的需要資料的時候才會呼叫DB – 稱之為 Lazy Load
- Entity 有Navigation，但是如果做Query的時候沒有包括Navigation的內容，那麼就不會被呼叫。
- 因此需要呼叫Include()來做Eager Load

```
foreach (var item in context.Customer.Include(x => x.Order).ToList())  
{  
    Console.WriteLine(item.Order.Price);  
}
```

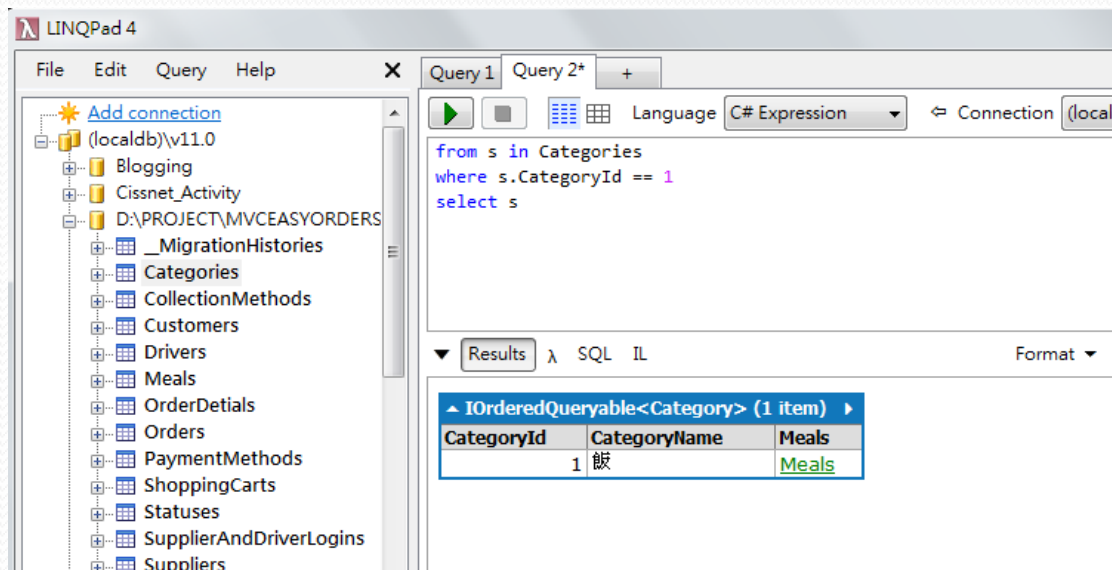
LinqToExcel

```
var excelFile = new ExcelQueryFactory(@"C:\Users\Alan\Documents\Visual St  
  
excelFile.AddMapping<Customer>(x => x.Name, "姓名");  
excelFile.AddMapping<Customer>(x => x.BadgeNo, "工號");  
excelFile.AddMapping<Customer>(x => x.Tel, "電話");  
  
var result = excelFile.Worksheet<Customer>(0);  
  
foreach (var item in result.Where(x => x.Name == "a"))  
{  
    Console.WriteLine("Name: " + item.Name);  
    Console.WriteLine("Badge: " + item.BadgeNo);  
    Console.WriteLine("Tel: " + item.Tel);  
}  
  
Console.ReadLine();
```

	A	B	C	D
1	姓名	工號	電話	
2	a	1	55	
3	b	2	66	
4	c	3	77	
5	d	4	88	
6				
7				

LinqPad

- <http://www.linqpad.net/>
- 可以用來測試Linq語法或者是任何C#語法
- 用來學習Linq的最好工具



Breeze.js

- <http://www.breezejs.com/>
- 在Javascript裡面使用類似Linq語法來過濾資料
- 可以搭配多種目前做火紅的前端Javascript MVC使用

Demo

Mvc Scaffolding 功能

結語

- 對於EF有個基本瞭解
- 對於如何定義Entity的三種方式
 - Database First
 - Model First
 - Code First
- Linq
- 工具
- Mvc 和 EF

Q&A

感謝大家