★ Entity Framwork ★

- way #3 to connect between [DataBase] and [code]
- is an (ORM) → Object Relational mapper

★★ طريقة [#٣] للتوصيل بين DB و الكود
→ [#1] sql connection
→ [#2] sql Data Source
→ [#3] Entity Framwork

— mapping By object and classes.
طرق لتوصيل DB مع ال code
← يتم عليه التوصيل عن طريق ال classes + object

— work with relational DB.
← تعمل مع DB يلي بكون ال tables الهم علاقة ببعض

★★ Benefit / Advantage — الفائدة

- eliminate effort for the developer to write data access code they usually needed to write
← يقلل على ال developer ضرورة كتابة الكود للتوصيل بين DB و code

we have 3 way to work with data

لدينا ٣ طرق للتعامل والعمل مع البيانات

① DataBase First ② Model First ③ Code First

1) DataBase First :-

- we use this way if we already have a DataBase

نستخدم هذه الطريقة عندما يكون هناك DB جاهزة

- the existing designer in Visual Studio automaticly

المصمم الموجود في Visual Studio يعمل بشكل automaticly على

generate a data model that consists of classes and properities {get; set;}

يقوم بشكل automaticly على انشاء Data model تحتوي على classes and properities {get; set;}

وبكونو متوافقين مع DB الموجودة في sql server

correspond to existing Data Base {tables and columns}

متوافقة

① information about database structure "Store schema"

+

② data model "conceptual model"

المعلومات الخاصة بهيكل ال DB

وال Data model

+

mapping between them

والارتباط بينهم

is stored in

يتم تخزينهم في

XML File

مع امتداد .edmx

** LINQ ** Language Integrated Query

- Programming model that you can use to query data from variety of sources

← نموذج يتم استخدامه للاستعلام عن البيانات من مصادر مختلفة

- Features of the { Entity Framework , C# } that enables developers to use language Integrated Query (LINQ)
  - * query data from a relational database

{select, Insert, delete, update}

← عبارة عن ميزة لل { EF , C# } بالي بتسمح لل developer باستخدام هذه اللغة لكتابة جمل ال Query

← لإستخدام ال بيانات من ال Relational DB

- allow the developer to write queries with use operators { where , select , orderBy , Join , GroupBy } ←

← تسمح للمطور/المبرمج من استخدام ال { } مع جمل ال Query

- allow to perform CRUD

{ Create , Read, update , Delete } ← مفهوم جداً مهم

LINQ

← لديها خاصية اظهار الخطأ قبل تنفيذ ال code

↓

- لانو كود تاعة ال Query بنكتب بلغة ال SQL
وعند تنفيذ ال code وبعدوها لـ SQL server

* واذا كان في خطأ بالجملة بتظهر قبل ما يعمل Run عكس
الاشي تاعة زمان

* في عمل ال query باستخدام ~sqlconnection, sqldatasource~

↓

نتم كتابتهم بلغة ال SQL server بحيث يظهر
الخطأ في الكتابة بعد ما نتم تنفيذ الكود
ليش ؟

↓

① عمل ال query بتكون بلغة SQL server
② لما نعمل Run بقوم ال compiler ببعتها لل SQL injen
③ وال SQL injen بقوم بالرد اذا كان
← صحيح برجع فيه ال data
← واذا لا برجع Error
Message

بالكود هيك بصيرو ال tables بال DB

... class student { → table ت

"مفرد"

ما في كود لل get و set

ت column ← Int ID {get; set;} → auto property

يتم معاملة ال column على شكل Handler

}

lambda

* Pointer on the property
* Create small and anonymous function

← يسمح بإنشاء function صغير وبدون اسم

used to

- Perform some action
- code more readable, concise, maintainable

← يخلي الكود قابل للقراءة ، مختصر ، قابل للتعديل

* You don't need to specify the type of value that you input

لسنا بحاجة لتحديد نوع القيمة اللي رح نعطيها

Syntax

input => expresion

Divided into tow part

input => { statements; }

X => X * X ;

* أكثر استخدامها بجمل ال query مع ال where ، order by

Example :-

List <int> num = new list<int> ()
{ 1, 2, 3, 4 }

var S = num.select (X => X * X)

S = { 1, 4, 9, 16 }

Pluralize and Singularize

* يكون اسم ال table في DB

① the first letter → capital letter

يكون اول حرف ← capital letter

② the name is plural → يكون الاسم جمع

Ex: { students , products , categories }

* وكما نعمل class باسم ال table

يتم تسمية ال class بنفس اسم table

ويكن بصيغة المفرد

{ student , product , category }

ليشان نستخدم ال singularize في اسماء ال classes

← ① اسهل بالاستخدام

← ② بدنا الاسم يوصف اسم الجدول وما يحتويه

ومش مقدار الاحتواء

احتواء

☆☆ the Different between function and stored Procedure

☆ function
- always give output (دائما يقوم بارجاع قيمه)
- need only input parameter → تحتاج معامل ادخال
- return only one particular value (تقوم بارجاع قيمة واحدة)
- Can written within select statement

← ممكن كتابتها داخل select ↓

☆ Stored Procedure
- Some times produce no result
- need input and output Parameter ← احياناً ما يرجع قيمه
- Can return 1024 value as a result called independently ← بحاجه لمعامل ادخال + اخراج

يرجع 1024 قيم لا معنى

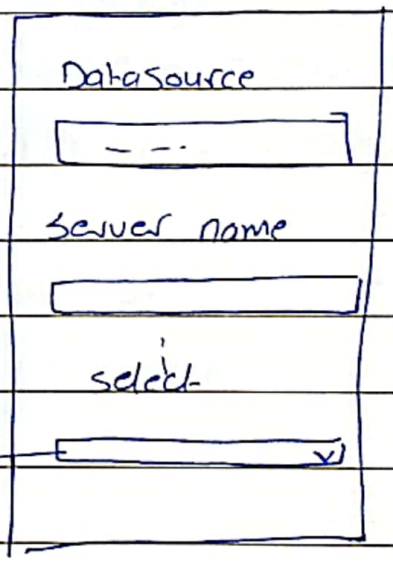
Connecting between DB and code using Entity framework

[1] Create Project → right click → Add → new item

[2] ADO.net Entity Data Model → Add

[3] → EF Designer from database → next

[4] new connection



.\sqlexpress

او

اسم الجهاز و ال sql " connect "

DB منفتا, اسم

[5] next → entity framwork 6.x → next

[6] → check → Tables → dbo → ميز ال tables اللي بدك تستخدمها

[7] check Pluralize --- , check Include FK --- → Finish

[8] Model1.context.cs كانفتح على

classes و ال properties مكان

اذا بدي اعدل اشي على ال Diagram او اضيف
table ما كنت محدده

← بدخل على Model1.edmx

Right click → Update model from DB

→ Refresh → tables → dbo

وبعدين ال tables اللي بدي اياهم

→ finish

Dbset<customer> customers { get; set; }

Data type

typeof

class اسم

وبيحتوي على Property

(Columns) الأسماء ال

customer

بدي اشبك بين ال webform مع ال DB

ميو

Gridview1

create object from class "DB" + "entity" ✱

"DataSource" عشان اعمل access على جداول ال table

اسم ال entity

اسم entity

entity SSS = new entity

DataSource

object

TaskEntity XX = new TaskEntity( );

Gridview1 . DataSource = XX . table_name . ToList();

. DataBind( );

object

select *

return all columns inside table

* linq the entities
  - convert the query statement from (*) to sql server

DB name + Entities ( ); → Data Source

Entity Framework

★★ Select statement { we have tow way to select data from Data Base using entity framework }

← في عندنا طريقتين عشان اعمل select بال entity framework

★ **way 1** الطريقة الاولى

★ TasksEntities obj = new TasksEntities();
↳ DataSource

★ Griduiew1 in webform.aspx

Griduiew1.DataSource = obj.TableName.Select(x => x.colN).ToList();
↳ no object

with where with ID

Grid view1.DataSource = obj.Votings.where(x => x.id == 1).select(a => a.rate).ToList();

with where with name

Gridview1.DataSource = obj.Voting.where(z => z.rate == "good").select(a => a.number).ToList();

Example :-

Votings

Id	rate	number
1	Good	7
2	VG	8
3	Ex	5

بجيب عن طريق الكود

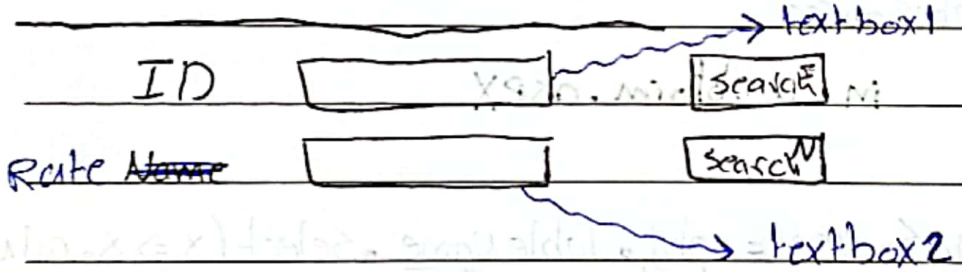
```
Entities XX = new Entities();

--- Page load ( ) {
    Gridview1.DataSource = XX.Votings.toList();
```

بجيب كل القيم يلي موجودة بال table

```
    Gridview1.DataBind();
```

+ هاد بقرأ ال



Search I

```
int x = Convert.ToInt32(textbox1.Text);
var Q = from c in XX.Votings
        where (c.Id == x)
        Select (c);
```

way #2

Search(R) البحث عن طريق الاسم

tablename

```
var y = from c in xx.Voting
        where c.rate.Contains(Textbox2.Text)
        select (c);
```

تعمل على ال like

"select *"

```
Gridview1.DataSource = y.ToList();
Gridview1.DataBind();
```

select الطريقة الثانية لل

table_name

```
var x = from c in xx.Votings
        where c.Id == 7
        select (c.rate , c.number);
```

DataSource "object from"

column name

---

* select statement with orderBy {Desending, Asending}

```
Gridview1.DataSource = xx.Votings.OrderBy(o => o.rate)
                                  orderBy Descending(o => o.rate)
.Select(x => x.Rate);
```

ترتيب تصاعدي "من الأصغر للأكبر"

كترتيب تنازلي

Select Specific aggregation function

استخدام ال Select مع ال aggregation function

هي ال func ~~بترجع~~ بترجعو قيمة وحدة

{Count, Avg, Min, Max}

Sum

~~Gridview1.DataSource~~ = xx.ratings.Select(x => x.rate).Count().ToString();

Lable1.Text

3

Lable2.Text = xx.ratings.select(a => a.number).Sum().ToString();

1+8+5

20

Foreign Key in Entity Framework

① with Join

Customer

CID	CN	CityI	CA
1	xxx	1	33

City

ID	CityN
1	I

```
var x = from cu in xx.customer
    Join ci in xx.city on
        cu.CityI == ci.ID
    where ( cond1 (||/&&) cond2)
    select( cu.CN , ci.CityN);
```

customer من جدول city من جدول

```
Grid1 ---- ();
```

② without Join

```
var x = from cust in customer
        orderby cust.CA descending
        select new { cust.CA, cust.CN , cust.City.CityN } ;
```

city // table من

syntax

→ cust . TableName . Col_name

Insert Statement In Entity Frame work

```
entities XX = new entities();
:
--button-click-- {
```

customers

ID	N	Age

① انشاء object من class تابع ال table_name

```
Customer cu = new Customer();
```

T1
name

② غير نعين بالقيم

```
cu.Col_name
```

T2
Age

```
cu.N = T1.Text;
cu.Ages(T2.Text);
```

Convert.ToInt32

③ عشان نطبق ال insert

```
XX.Customers.Add(cu);
```

④ عشان يحفظ ال insert اللي نعمل

```
XX.SaveChanges();
```

Show Image

★ لإظهار الصورة داخل
ال gridview بدون استخدام
ال (Eval)

① Drag-Drop from tool box
★ Gridview

و داخل ال gridview عن طريق ال Design
↳ ▷ → edit column → Image field

DataImageurl Feild = Col-name
لي بيجي بال DB

عشان يزبط لازم بال Page load
نكون عاملين select
لل table يلي بدنا نجيب
منو القيم
★★ ولازم جوا ال select نكون
مختارين ال image col

و عشان نظهر ال col يلي
بدنا اياهم ونعمل عن طريق
ال Bound field ← Data Field = Col-name
ونصيرها نفس طريقة ال Image feild

$ ~~update~~ update in entity fw

① بدي اعمل button / link مثلاً بعتني على الصفحة يلي رح اعمل فيها ال update

يلي رح اعبي فيها القيم الجديدة

من ← gridview ← Design ← [7] ↓ edit col → Hyberlink ... → Add

select

→ click on it

→ * DataNavigateUrlField = col-name → مهم

يلي رح ابعث قيمتو مع ال (Url)

→ مهم → * DataNavigateUrlFormatString = "UP.aspx ? ids = {0}"

اسم الصفحة يلي بدي ارح عليها

القيمة يلي رح استدعيها بالصفحة الثانية عن طريق Query string

بتتغير حسب ال Row يلي تم اختياره و قيمتو جبيها من Data Navigate Url Field

② انشاء صفحة up.aspx

Name [ ] T1

Age [ ] T2

upload filename

Update

① عشان احط القيم الموجودة معها بالـ DB داخل الـ Textbox

Page load ( ) {

if (!IsPostback) { ← بدونها ما بعمل update

A

T1.Text = (way 1)

XX.Customers.FirstOrDefault (a => a.ID == id)?.Name;

لانو بكل مرة بكبس على update بعمل load بكل الصفحة فبرجع بحط ~~[illegible]~~ و بنفذ نفس الـ Page load

query string من

int id = Convert.ToInt32(Request.QueryString["id"]);

B

way 2

Var xy = XX.Customers.Find(id);

T1.Text = xy.Name → column اسم

T2.Text = Convert.ToString(xy.Age);

③ اطبقه على ال update داخل ال update_click

```
int id = Convert.ToInt32(Request.QueryString["id"]);
Var up = XX.customers.Find(id);
```

← هو لحسبك ال col لما يكون ID == id

```
up.Name = T1.Text;
up.Age = Convert.ToInt32(T2.Text);
```

~~[illegible]~~

مهم (XX.SaveChanges();)

```
string ii = "~/imgPath" + up.Photo;
```

* اذا بدي اعدل image يعني اذا ال fileupload فيو قيمة صورة

```
if(FileUpload1.HasFile){
    ... SaveAs("~/imgs/" + Path ....);
```

على ال if

Delete

* نفس خطوات ال **update**
من ناحية انشاء ال Hyperlink و انشاء صفحة

* نفعل ممكن بصفحة ال delete نظهر المعلومات

② Details View ① lable

Bound field — لازم نعمل

لا لل col بدي اياهم يظهرو

Image field — اذا بدي اظهر صورة

مع تحديد ال Column Name في كل وحدة

منحط فيهم اغنية

زي ال Textbox في update

الفكرة لتبقى اعبي القيم بقى ⟸ Details

Query String

Page load في From QS

var xs from o in customer where o.Id s= id

select (o) — نعمل select

* كل القيم بال موجودة بال customer — القيم الموجودة بال DB

→ Details view1. Datasource = X.ToList();

object from (DB Name entities)

del-Click داخل ال delete على ال

* Query string (id)

→ var del = XX.customers.Single(a => a.Id == id);

لجيب ال Row باي بيحتوي على ال (id)

XX.Customers.Remove(del);

XX.SaveChanges();

★ معلومات بكل المواضيع ★

Entity Framework :-

★★ Object Relational Mapping (ORM) tool that allow developers to interact with database

ORM عبارة عن tool بتسمح لل developer للتفاعل / للوصول الى DB

★★ EF automatically generated necessary sql commands to { create (insert), Read (select), update, Delete } ⇔ CRUD

تقوم EF بإنشاء كل sql المهمة

Database First

★★ used when you already have existing DB

يتم استخدامها لما يكون عنا DB جاهزة اصلا

★★ and you want to avoid creating classes and model properities

" يتم انشاء كلاسات classes (أو table " جدول ") و columns كلاسات

LINQ to entities

★★ Programming feature in .net framework that allow developers to write queries using Entity Framework data models

عبارة عن ميزة في .net بتسمح لل developer بكتابة كل (select, ...) CRUD

★★ it Provides way to work with database in object oriented way using C#

بتوفر طريقة للعمل مع DB باستخدام object oriented بلغة ال C#

و يتم تحويل ال عملية من [sql query C#] لـ [sql query sqlserver]

lambda

★★ can be used with EF to write LINQ queries (update, select, insert, delete)

يتم استخدامها لكتابة عمل ال

★★ work with linq method provided by entity framework to filter, sort, tranform

تعمل مع ال linq يلي بتوفرها ال entity framework

عشان نعمل filter, sort, transform؟

filter → where

sort → order

```
X => X.ColumnName == y
```

(where مع)

```
orderBy(X => X.Age);
```

Pluralize and Singularize ~~concept~~ concept

according

the names of database table and column

the entities classes and properties in data model

will named

حسبا بالاعتماد على اسماء ال table, col الموجودين بال Database

يتم تسمية ال classes / properties يلي موجودين بال Data model

ال table يلي بال DB بكون بصيغة الجمع

يتم تسمية ال class تبعه ال table بس

بصيغة المفرد

Select (2 way)

Way 1

```
entities q = new entities();

var X = from c in q.TN
        where (c.ID == 7)
        select (c);
```

Way 2

```
var a = q.TN
  .where(s => s.ID == 7)
  .select(c => c.colN);
```

tablename

select for one col

```
  .select(c => new { c.CN, c.CN2 });
```

select more than one col

columu name

order By

- orderBy (x => x.cN); → تصاعدي
- orderBy Decending (x => x.cN); → تنازلي