1/5/2023

Abdel-Rahman Gamal Ahmed

no campany

Design pattern

DI



Table of Contents

[1-topic: 2](#_Toc123937504)

[2-business logic : 2](#_Toc123937505)

[3- important notes : 3](#_Toc123937506)

[4- dependence injection : 3](#_Toc123937507)

[4.1-why form take object in contractor ? 4](#_Toc123937508)

[4.2-logic in dependence injection : 5](#_Toc123937509)

[4.3-install library for dependence injection: 7](#_Toc123937510)

[4.4-register object in autofac : 7](#_Toc123937511)

[4.5 – Features of dependence injection : 9](#_Toc123937512)

[4.6- get all type in dll : 10](#_Toc123937513)

[4.7-register type life cycle : 10](#_Toc123937514)

[4.8- important notes for interface container : 13](#_Toc123937515)

[5-the end: 14](#_Toc123937516)

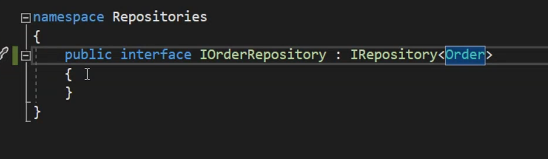
# 1-topic:

احنا اخدنا جزء كبير من ال repository وهنرجع ليه تانى نشوف كام ملحوظه مهمه ومفيدة .

دلوقتى هناخد حاجة اسمها ال dependence injection .

# 2-business logic :

قبل مانروح لل dependence injection . هنعمل بس logic بتاع order على السريع والخطوات هى هى الى عملنها ف ال client بس هنحتاج الخطوه دى واحنا شغلين . وهسبلك اسكرينات بس من غير شرح .







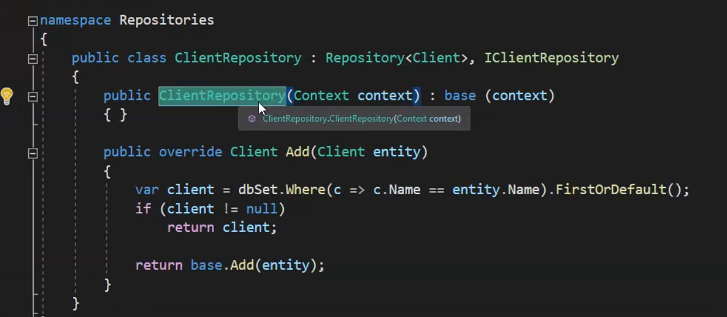
* ملحزظه مهم يغالى قبل منكمل : لو ملاحظ ف الكود بتاع ال ui قلتلك هو tightly couple مع الmode وكمان مع الداتا
* ف موضوع ال tightly couple بالنسبه للdata قصدى بيه ان جواه ال ui ف object من ال context ولحاد دلوقتى هو موجود بس هنحوشه وشخلين ف خطوات حله .
* اما موضع ال tightly coupleبالنسبه للmodel احنا قلنا هناقشه مع pattern تانيه لو ملاحظ ان لما عملتbusiness logic انا مبقتش اعمل object من model . خلى بالك من النقطه دى وهنرجعلها ان شاء الله .

# 3- important notes :

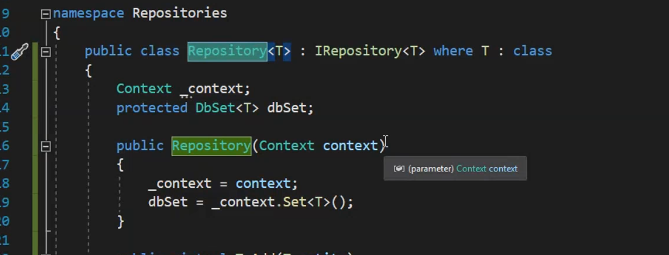
احنا عملنا من شويه repository مخصوص للclient وكمان واخد مخصوص لل order عشان نضيف logic معين . طيب افرض ان ف حد استخدام ال repository القيدم الى مفهوش اى business logic ؟ ازاى هقدر امنع الشخص من حاجة زى دى خلى السوال ف دماغك وهجاوب عليه بعد شويع تابع .

# 4- dependence injection :

* لو رحنا لكلاس clientRepo هنلاقيه بيقول اى حد عاوز يشتغل معايه يبعتلى object من context .

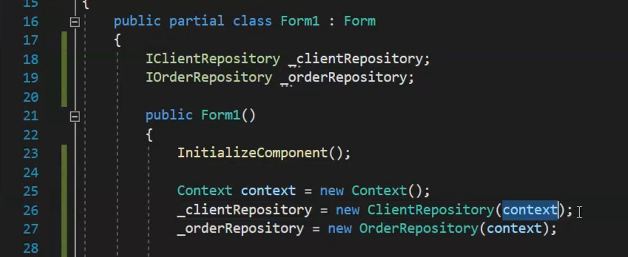


* لو رحنا لكلاRepo هنلاقيه بيقول اى حد عاوز يشتغل معايه يبعتلى object من context .



* ف عشان form عصب عنها عملت object من context عشان تبعته للrepo الى احنا عملنها .

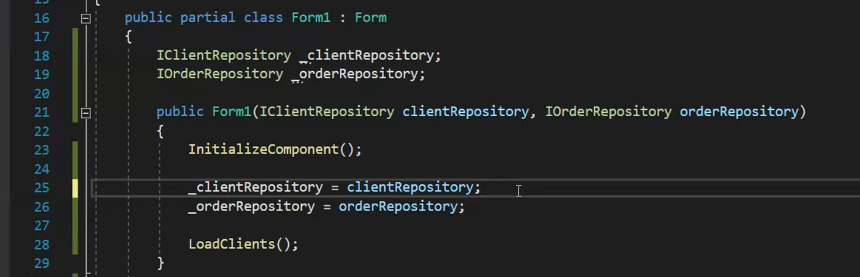
ف كدة ف tightly couple بين ال ui وال context class



* عشان نحل مشكله ال tightly couple مفروض الكلاس سئل نفسه هو انا عاوز اتعامل مع context ولا انا عاوز اعمل create ل context ؟ ف فرق كبير بنهم على فكرة

1. لو انا عاوز اتعامل مع context سواء على مستوى function واحدة او حتى على مستوى الكلاس فالمفوض ان دة علاقه aggregation يعنى الكلاس بيحتوى على context ف لو بتكلم عن كلاس repo ف هنقول ان كلاس repo يحتوى على context
2. اما لو انا هعمل create للcontext جوايه يبقا دى علاقه composition يعنى علاقه consist of يتكون من .

* طيب لو فرضنا ان ال repo class هو الى بيعمل create لل context جواه ف المشكله هنا ان ال repo هو الى هيكون مسئول عن ال life cycle للcontext يعنى طول م object from repo هيكون ال context object عايش .
* طيب انا دلوقتى as repo اناى مالى ومال ال create انا كل الى محتاجه اتعامل مع context ف عشان كدة خليناrepo ياخد ال context على انه input parameter ف الكونستراكتور . نفس الكلام باضبط هتطبقه على form
* ف كلاس form هيسئل نفسه هل انا مسئول عن الcreate بتاع ال context او متى مسئول عن creation للrepo ؟ الاجابه لا ايوة انا محتاجهم جوايه بس مش انا الى هعمله create يعنى ال form مش محتاج object من ال context الى محتاج يستخدام ال context هو ال repository . وحنا سئلنا نفسنا السوال ده نطبق مبداء ال single responsibility .
* ف كدة ال form هتقول انا محتاج ال context وال repo يتبعتوا ليا ف ال contractor انا مش هعمل create لاى حاجه منهم.
* ودة شكل كلاس ال form دلوقتى يحب .



## 4.1-why form take object in contractor ?

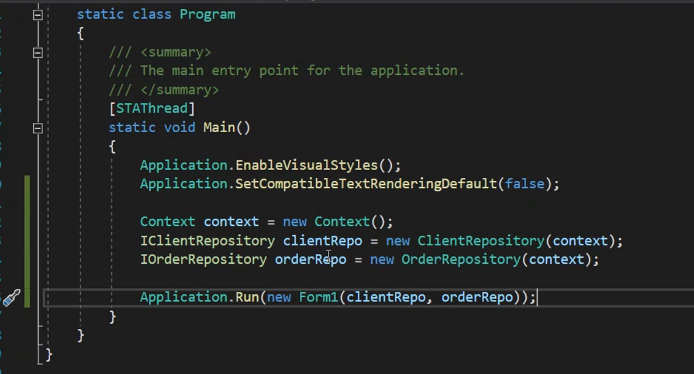
ليه خلينا الform تاخد object ف ال contractor ؟

1. عشان مش مسئوليه ال form تعمل create للobject وكدة حققنا ال single responsibility .
2. طيب ايه الى هيحصل لو خلينا موضوع ال create دة جوة ال form .
3. اول حاجة form هيشوف كل شويه ال repo الى جواه دى محتاجه انهى object ويفضل يجهزهملها .
4. افرض ان ال repo الى جوة form بدال مكان بياخدobject واحد من ال context ف ال contractor بقى ياخود اكتر من parameter ؟ ف معنى كدة ان الكود بتاعك هيضرب عشان عدد الparameter للكونستراكتور اتغير ف هتفتح الكود ودعدل فيه ودة مشكله

## 4.2-logic in dependence injection :

طيب دلوقتى مين الى هيبعت لل form ال object الى هيا عوزها ف الcontractor ؟كدة مفروض الى بيعمل object من form الى

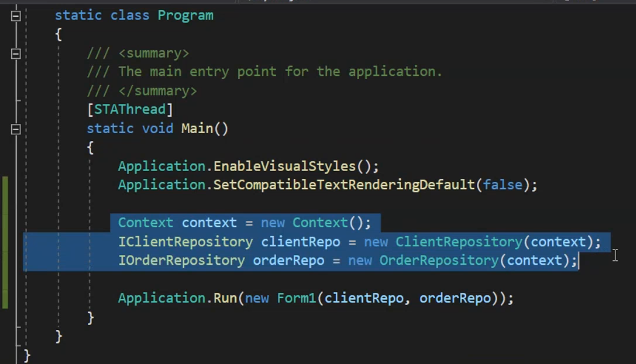
هو class program .

* ف تعاله تكتب شويه ال object الى محتاجه كلاسform وتبعتهوله .
* طيب انت كدة حليت المشكله ؟ طبعا لا انت كدة مرجد رحلتها طيب الحل ايه؟ الحل ف ال dependence injection ف تعاله نشوف ازاى هنتعامل معاه ؟
* يعنى ايه طيب ف الاول كلمه dependence injection : dependence دى هى ال object الى بيعتمد عليهم كلاس ميعن عشان يشتغل . فحالتنا هنا كدة الكلاس form بيعتمد او dependence على الclientRepo,orderRepo . ومعنى inject يعنى حقن او ارسال



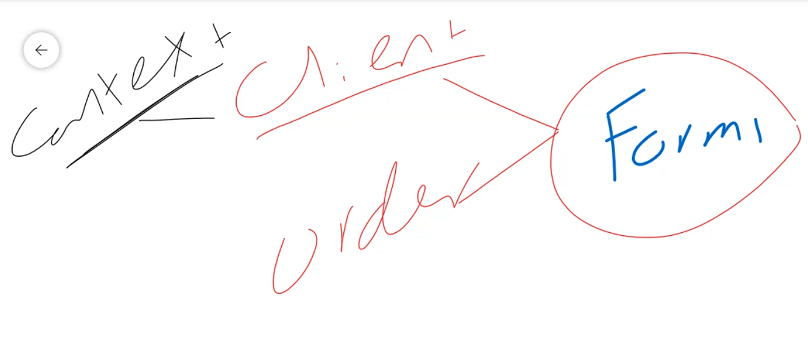
* ف كدة معنى dependence injection يعنى ارساله الobject الى محتاجها كلاس معين بدون تدخلى منى .
* ف تلت انواه من ال dependence injection :
* Contractor injection ودى معنها هيعمل inject لل dependence وهو بينادم ال Contractor بتاع ال object . ودة الى هنستخدامه .
* Setter injection : ودة معناه ان هعمل property وف ال set بتاعتها هعمل inject for dependence
* Interface injection
* معظم شغلنا هيكون على ال contractor injection ف تعاله نشوف الخطوات الى هنعملها عشان

نعمل dependence injection in contactor .

* ف الخلاصه ان معنى ال dependence injection : ان هيكون ف layer جديدة تكون هى المسئوله انها تعمل inject لل dependence الى عوزها object عشان يشتغل . وف حالتنا احنا هنعمل inject ف ال contractor واحنا بنعمل object من ال form .
* ف طبعا احنا مش هنكتب الى مظلل دة الى هنعمله كالاتى :

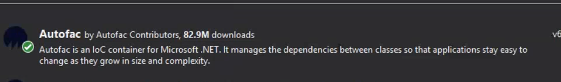
1. هنعتمد على layer جديدة هقلها عاوز اعمل object من form1 وال layer دى هتروح تشوف هو ال form1 محتاجه ايه عشان تشتغل ويبعتلها وتروح تعمل كل ال object الى محتجها ال form وتبعتها للform .
2. طيب ازاى ال layer دى هتعرف ان لما اطلب object من IClientRepository هورح اعمل تبعتلى object من ClientRepository . ف دى مسؤليتك انت انك تحدد ال data type الى هتعملها registration .
3. تانى الlayer الى هنعملها مسئليتها لما اقلها عاوز اعمل object من form1 هى هتورح تشوف ال form محتاجه ايه

ف هتلاقى ان ال fom1 ومحتاجه client وorder ف لما يجى يعمل ال client هيلاقيه محتاج object من context فهيعمل ال object من context و ف الاخر يرجعل object from form1 .



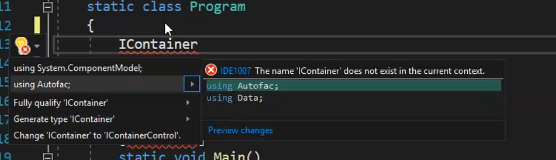
## 4.3-install library for dependence injection:

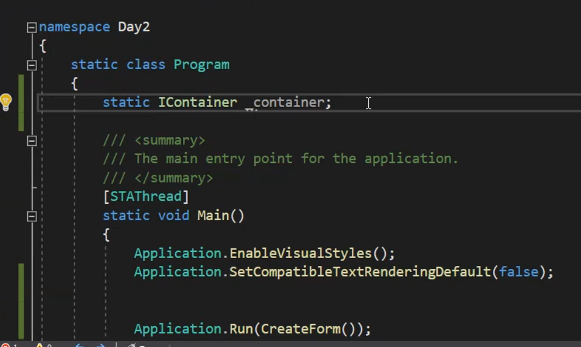
ف اكتر من library الجاهزة عشان نشتغل بحوار ال DI ومن اشهرهم autofac . وطبعا بنحمل الlibrary دى ف ال ui layer .



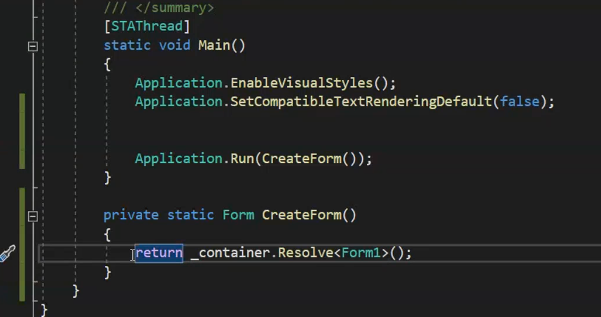
## 4.4-register object in autofac :

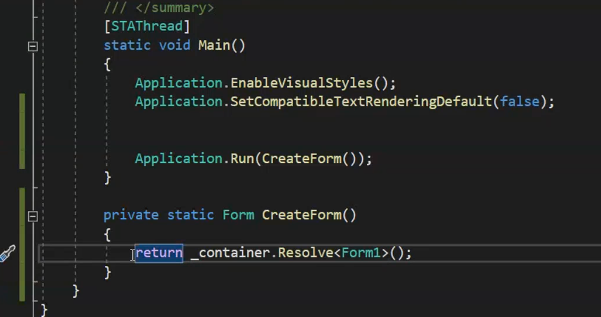
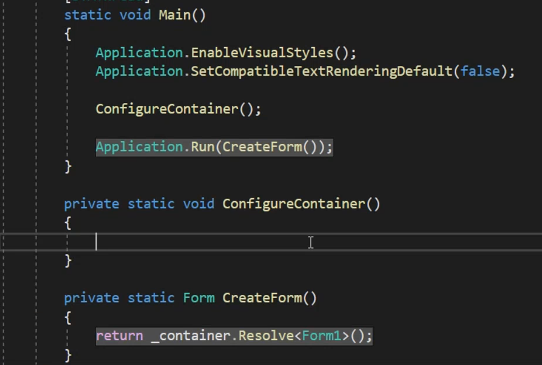
* يعنى هنا هحدد وقول لما يطلب منك الdata type الفلانى اعمل object من كذا
* عشان نعمل register هنتعامل مع حاجتين كلاس و interface . طبعا الكود دة هيكون ف class program .
* اول حاجه هحتاجها هى interface Icontainer و خلى بالك من ل use الى هتستخدمعا هى ال autofac.

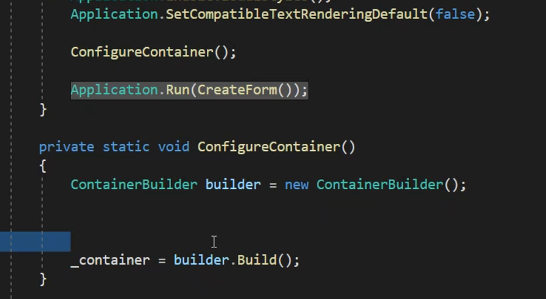
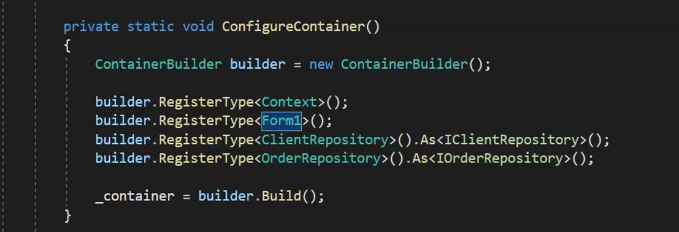


1. ودة شكل ال reference الى هعمله من container ودة static عشان الكلاس الى هو فيه static ودة هيكون مسئول عن عمله ال resolving الى هتحصل يعنى هيكون container دة هو الى هطلب منه يعملىobject من form1 او اى كلاس تانى انا عامل ليه registration .

* يعنى مفروض بعد مخلص registration المروض هستخدام ال container عشان اعمل الشكل دة . container.resolve<form1>(); والسطر دة مسئول عن انه يعمل form ويرجعهالى



1. ناقص بس اعمل register لشويه object الى هشتغل بيهم عشان السطر دة container.resolve<form1>(); يشتغل فهنعمل الconfiguration ف داله اسمها configercontainer .
2. ف عشان نحدد ايه ال object الى هعملها register وكمان احدد ال lifetime بتاعهم هستعين بكلاس جاهز اسمه containerBulider وطبعا الاسم مفروض يفكرك ب ال builder design pattern الى اخدناه واحنا بنستخدام ال builder لما نكون عوزين عاوزين نعمل complex object هيحتوى جواه على مجموعه من ال object الاخره مجموعه من ال option الى انت مش عارف ال user هيختار انهى فيهم .

* containerBulider دى عباره عن implementation للdesign pattern الى اسمه builder والى يهكون مسئول عن building للcontainer بتاعى حسب ال option الى انا عملها
* ف كدة ممكن تتخيل ان ال containerBulider من جواه هيكون ف اكتر من option كل option فيهم مسئول عن تكوين object ف لو هنفرض ان ال containerBulider جواه 10 option وانا لما اعمل form هختار 2 منهم بس .
* ف كدة الى ناقص ف الاسكرين الى فاتت ان اضيف ال option الى هعملها build .
* هنستخدام function اسمها registertype .
* واول registertypeهعملها للcontext وال context دة concrete class ف عشان كدة مختلف عن السطر الى بعده .
* تانى registertype انا عامل فيها interface ف مفروض اقله لما يطلب ال interface الفلانى يعمل class من انهى object .

## 4.5 – Features of dependence injection :

1. كدة مفيش ولا واحد فينا مسئول ان يعمل create للobject
2. لو بعد شويه انت as form1 قرر ان اخلى contractor بتعها ياخد parameter اكتر ف ال contractor . ف كدة التعديل مش هياثر على اى حد ومش هفتح اى كلاس وهعدل فيه .
3. ولو انت مش حاسس باهميت الموضع ف المضوع هيبان اكتر ف ال mvc **مش** انت الى هتكون مسئول عن عمل create للcontroller ف عصب عنك هتروح تستخدام الDI .

## 4.6- get all type in dll :

لو انا عاوز اجيب كل ال type الى جوة dll معينه بدال مكتبهم بايدى ودة هياخد وقت .

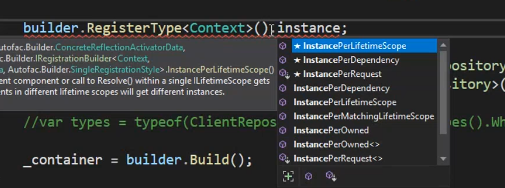
لو فاكر الحجات الى كل الكلاس ورثها من كلاس object كان فيه typeof .وكانت بترجع object من كلاس type وبيكون شايل شويه معولمات عن data type بتاع الكلاس.

* فال object الى رجع من typeof لو عمل.assembly كدة هيبقا معايه كل المعولمات عن ال dll
* ف من ضمن المعولمات الى جوة ال assembly عندى داله هتجبلى كل ال type الى جوة الdll اسمها getType() . ف انت تقدر تعمل forech على الى راجع من gettype وتعمله register ف ال builder
* طيب افرض ان انت مش عاوز تجيب كل ال type الى جوة ال dll عشان ف حجات ملهاش علاقه ب ال repository ؟ بسيطه عشان الى راجع من ال gettype هوIEnearable ف انا اقدر اعمل where . وسيبك من الارور متقلقش منه .



## 4.7-register type life cycle :

ال function الى جوة ال builder الى اسمها register type فيها تلت دوال مهمين خاااالص . كلهم بيبداءه ب instance



* ف تعاله نشوف ايه الموضع دة يبوا عموا :
* احنا قلنا ان ال form هتكون عاوزة clientRepo وال clientRepo هيكون عاوزه context ف ال DIهتعمل context للclientRepo
* طيب ف نافس الوقت ال form هتكون عاوزة orderRepo وال orderRepo هيكون عاوزه context ف ال DIهتعمل context لل orderRepo وخلى بالك ال context هتكون مختلفه عن ال context الى اتعملت لل clientRepo
* طيب لو انت مش عاوز الحاله الى فاتت وهى ان هيتعمل اكتر من context كل محد يطلب نعمل ايه ؟ عندك مجموعه من الاختيارت عشان تغير ال default الى قلنا عليه :

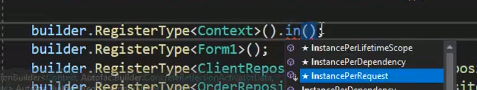
1. لو كتبنا singleInInstance بعد داله registertype كدة انت بقله مفيش غير نسخه واحدة بس الى هيحصلها create من ال context واى حد بعد كدة يشتغل مع النسخه دى وده مثال طبعا لل singleton pattern .

وطبعا ف حاله الcontext مهم ان نخلى عمليه ال creation ليه singleInstance عشان ميبقاش ال

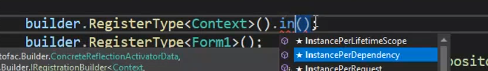
clientRepo , و ال orderRepo كل واحد فيهم معاه object مختلف عن التانى



1. وفى INstanceBerRequest ودى معنها كل مطلب منك object روح امعلمى واحد جديد .

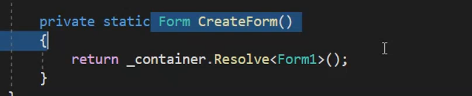


3-وفى تالت نوع الى هو instanceBerDebendance وده معناه ان هعمل object واحد على مستوى الdepandence الواحد يعنى لو فرضا كان عندى dependence واحده بياخد two object منcontext جواه ف هعمل object واحد من context وابعته مرتين لل dependence اما لو فى dependence تانيه عاوزة context هعملها واحد جديد .

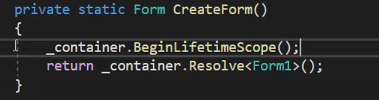


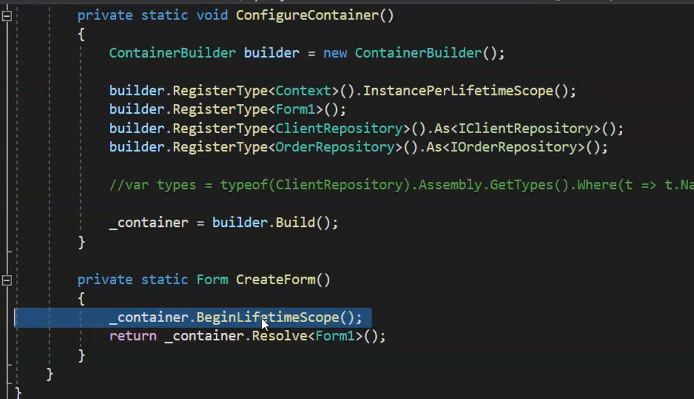
1. رابع واخر واهم نوع هوinstancePerLifeTime



* طيب ايه معناه ؟ دى باضبط زى ال transaction ف الداتا بيز تعاله ناخد مثال على الماشى افرض ان كان عندى اكتر من form ف البرنامج ف اكيد السطر بتاع Resolve هيتنفز اكتر من مرة

ف لو جيت وكتبت \_beginLifeTimeScope قبل سطر ال resolve ايه الى هيحصل ؟ اول حاجة beginLifeTimeScope دة بترجعل حاجة من نوع ILifeTimeScope اكنها بترجعلك reference بيشاور على ال lifeTimeScope الى اتعمل عشان لو حابب تمسحه او حاجة



ف الاجابه عن سوال بتاع معنى ان كتب بعد ال registertype كلمه LifeTimeScope واكتب beginLifeTimeScope قبل resolve ؟

* اول مرة تادم فيها ال resolve : وطبعا قبل الresove ف كلمه beginLifeTimeScope ف كلمه beginLifeTimeScope دى هتخليه يبداء transaction جديد ,ويعمل فيها كل ال object الى ال form محتجها طيب ال form محتاجه معانا

1. الclientRepo و الclientRepo محتاج context ف هتعلم object من الclientRepo وcontext
2. ,وتانى حاجة ال form محتجاها هى ال orderRepo ف هتلاقى ال orderRepo محتاجه context ف **مش** هتعملobject من ال context وهتبعت الobject القديم بتاع الى اتعمل للClientRepo .



* طيب ف تانى مره انادم فيها ال resolve ؟ دى تعتبر lifetime مختلفه عن اول مرة ف كدة هيتعمل context تانى غير الى ف ال lifecycle الى فاتت .
* F1 هنا دة اختصار لform1
* C اختصار ل client Repositary
* 0 اختصار ل order Repositary
* وكل مربع من الى ف الاسكرين دول

بيعبر عن lifetime مختلفه

وكل مربع فيه context واحدة بس

حتى لو ف اكتر من حد ف الlifetime

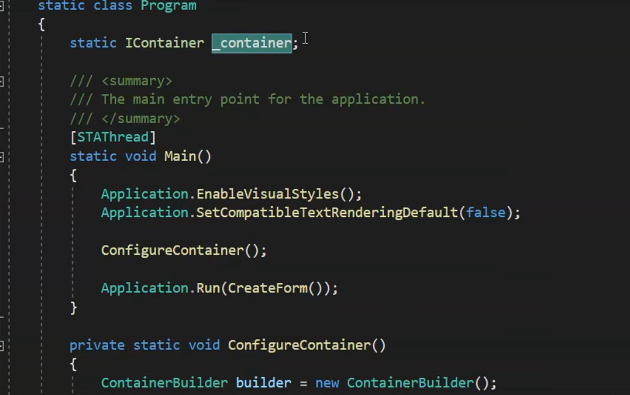
كان محتاج ال context .

## 4.8- important notes for interface container :

هقلك على حاله اوعى تقع فيها :

هتلاقى حد وانت شغال عاملك ال registertype من نوعsingleInstance يعنى هتكون نسحه واحدة على مستوى ال app كله.

وبعد شويه تلاقيه عمل container جديد غير الى كان عامله ومعنى كدة ان انت رميت ال context القيدم الى كان موجود وعملت واحد جديد مع انك محدد ان ال context هو singleton ف انت كدة بتعك يرفيق .



* العبرة من الى فات مينفعش يكون عند غير container واحد بس على مستوى ال app كله .

# 5-the end:

* احنا كدة خلصنا ال DI وشفنا تطبقين على pattern الى هم ال builder وال singleton .
* فالى جاى بقا هنشوف حاله مهمه وحلوه الصراحه وسريعه متقلقش ليها علاقه ب ال repository .