12/29/2022

Abdel-Rahman Gamal Ahmed

no campany

Design pattern

Creation Design pattern



Table of Contents

[1-topic : 2](#_Toc123251782)

[2-builder : 2](#_Toc123251783)

[2.1 –problem: 2](#_Toc123251784)

[2.2 –builder step : 4](#_Toc123251785)

[2.2.1:when we use builder : 5](#_Toc123251786)

[2.3-importain notes : 5](#_Toc123251787)

[2.4-stcture for builder : 5](#_Toc123251788)

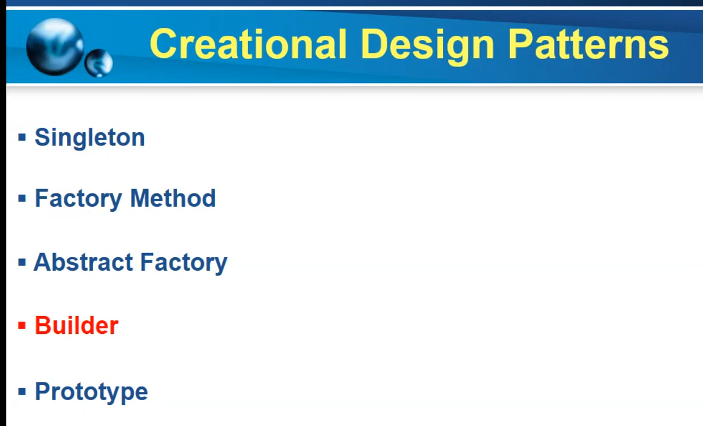
[2.5-client code or code in main: 8](#_Toc123251789)

[2.6 –make mix product? 8](#_Toc123251790)

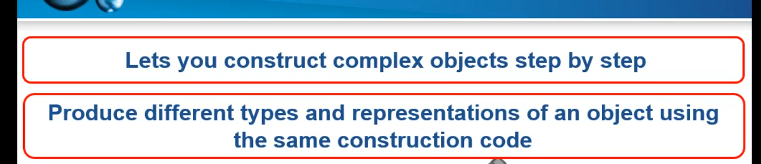
[2.7-applicability : 8](#_Toc123251791)

[2.8 –cons and pros: 9](#_Toc123251792)

# 1-topic :

مكمللينى يغالى ف ال creation DP . هناخد النهاردة اخر واحد الى هو ال builder . وال prototype احنا اخدناه قبل كدة وهقلك امتى بس ف الاخر .

# 2-builder :



ال builder هيسعدنا ان احنا نبنى complex type خطوة خطوه . بمعنى ان احنا ف ال app المختلفه بتاعتنا يكون عندنا object حجمها كبير . وحجمها الكبير دة مش بسبب ال attribute الى فيها دة بسبب علاقتها مع مجموعه من ال class التانيه . يعنى هتلاقى aggregation وcompositions جوة الكلاس دة يعنى . ف الخلاصه الكلاس دة هيكون محتاج يعمل create لمجموعه من ال object جواه .

* والموضع مش مجرد ان محتاج ابنى object كتير جوة class معينى . لو المضوع ان عاوز اعمل create لمجموعه object جوة كلاس ف احنا خدنا امبارح factory method . الفكرة هنا ان كل مرة هبنى ال object بتاعى دة ممكن ابنيه بشكل مختلف .

بمعنى ان كل مرة مش بكون عارف انهى object هحتاج ابنيها وانهى لا .

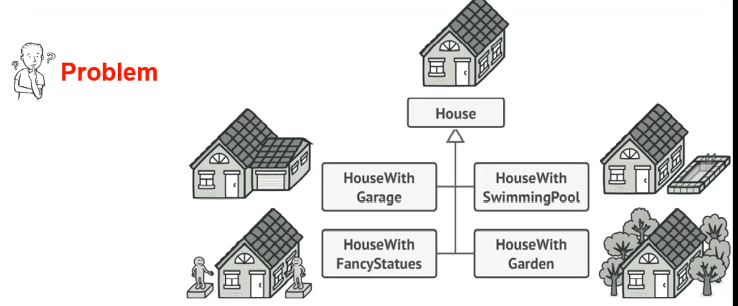
* يعنى ممكن تتخيل ان معايه مجموعه من ال option جوة كلاس معينى هحتاج احدد انهى option هحتاج اشتغل عليها وانهى الى هسبها . وال option الى هخترها ممكن تختلف كل مرة عن الى قبلها .
* ف الى هيسعدنى عشان اعمل الكلام دة هو ال builder .

## 2.1 –problem:

هتتخيل معايه يصحبى ان عاوز اعمل SW يمثل ان ببمنى بيت . ف لو الموضع بس كدة ف كنت هعمل class اسمه house واحط فيه مجموعه من ال object . ف جوة الكلاس الى اسمه house يعنى هعمل شويه object بتمثل الشباك والباب والحطان وكدة يعنى .

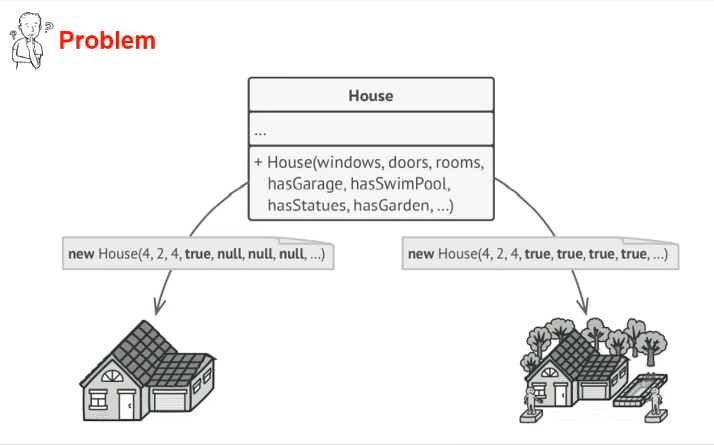


* بس المشكله هنا ان الراجل حب الموضع وزود ف الطلبات بتاعته , قلك عاوز اقدر ابنمى بيت بحمام سباحه وبيت بحديقه خلفيه وبيت بجراش وبيت بمخزن الخ .
* ف انا الحل الى عملته ان زوت ف ال hierarchy الى موجودة .يعنى عملت كلاس تانى وخليته يورث من ال house .
* لو الموضع واقف عند الكلام الى ف الاسكرين دة اشطا عادى اما المشكله ان ممكن حد يقلك ان عاوز بيت فيه جراش وفيه حمام سباحه دمج بين اتنين يعنى . وحد تانى يجبلك اضافه جديدة وهكذاا ف الخلاص ان ممكن الطلبات تفضل تزيد بشكل كبير

يعنى لو عندك 6 class ممكين يقلك عاوز class فيه مزتين من كلاس موجود ف انت كدة ممكن تعمل inherit بعدد ال permutation بين ال 6 class الى عندك ف كدة مستحيل تعمله ب ال inheritance . عشان باشكل دة هيطلع hierarchy كبيرة جدااا بعدد ال permutation بين الكلاس الى موجودة .

* ف واحد تانى جه وقلك : خلاص انا مش هعمل inheritance عشان الدنيا هتعك معانا . كل الى هعمله ان اعمل كلاس واحد اسمه house وانت بتعمل object من ال house هتبعتلى شويه ال option الى انت عوزها ف ال constrictor .
* ف الشكل دة كل مرة هعمل object من house هبعت شويه object كتير ليه وكدة بيخلى ال readable بتاعه الكود متكونش كويسه . يعنى الاسكرين الى قدامك دى اعتقد لما تشوفها صحب

تفتكر هو المتغير التانى دة بيمثل ايه وهترجع للdefinition .

* وكمان ممكن ال option تزيد او تقل .
* ف عشان كل الاسباب الى فاتت دى هنحتاج نستخدام ال builder :

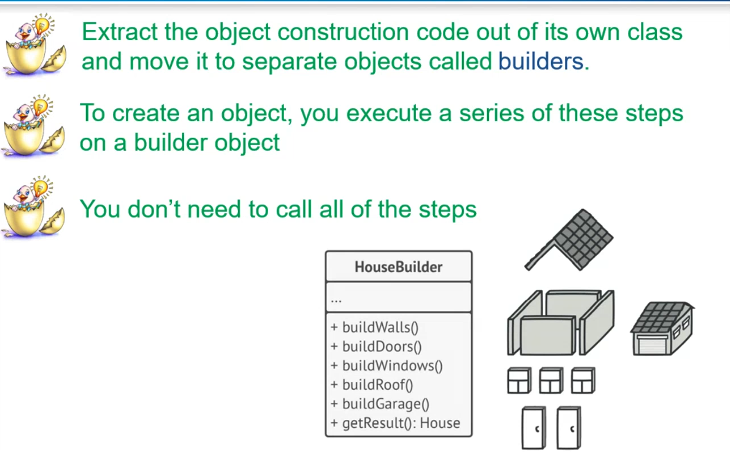
## 2.2 –builder step :

1. هنخلى ال creation بتاع ال object بدال مكانت ف الكلاس نفسه هنخليها ف class مختلفه وهنسميها builder .

وهيبقا كل builder مسئول عن بناء option معينى . يعنى حمام السباحه دة option لوحده ف هعمله builder يكون مسئول عن ال boll

1. على حسب الoption الى عوزها هستعين بالbuilder المختلفه . بمعنى كل option هيكون ف builder انت بقا هتستعين بالbuilder الى محتاجها ولو ف option جديد طلع هتعمل ال builder بتاعه وتستعين بيه وانت بتبنى الobject بتاعك بعد كدة .

* ف كده لو هنتكلم عن كلاس الhouse ف انا عند الباب دة object و ال حديقه object يعنى كل حاجة ف ال بيت هى عبارة عن object .ف كدة الكلاس house ف مجموعه من ال object جواه ف انا عشان اتحكم ف الموضوع خليت كل option من الى فاته ف builder وهنشوف ازاى بعد شويه ؟
* ف كدة زى ما الاسكرين موضحه : هيكون معايه كلاس اسمهhousebuilder وجوة الكلاس دة هيكون معايه method عشان تبنى عناصر البيت زى فنكشن عشان تبنى ال حيطه وواحدة تانى عشان تبنى الشبابيك وهكذا . وانا الى هختار هعمل call لانهى function فيهم . وف الاخر ف function بترجعل house بالoption الى انا حدتها .



* لحاد دقوتى دة مش الحل النهائى . عشان كل ال option حتطها جوة ال housebuilder ف ال client لما يعوز يبنى بيت هيستعين ب ال housebuilder عشان يبنى ال house الى عاوزة . وهنشوف ازاى بعد شويه .
* وملحوظه كدة احنا ممكن ندمج اكتر من pattern مع بعض . عادى وهنشوف برضو مثال على دة
* ف احنا مشين الموضوع واحدة واحده : واول خطوة زى مشفنا ان حطينا كل ال option ف housebuilder . والى عاوز يبنى بيت يعملobject من housebuilder ويقله ال option الى هو عوزها .
* تعاله نحسن التصميم شويه .
* ممكن حد يقلك ان عاوز ابنى البيت من الخشب وواحد تانى يقلك ان عاوز ابنى البيت من الحجارة .ف كدة مينفعش يكون عندى house builder واحد .ف انا كدة محتاج انواع مختلفه من ال builder house كلهم ف الاخر ليهم نفس الinterface . ف بختلاف ال material الى هشتغل بيها هبقا محتاج اكتر من نوع من انواع ال builder .
* ف بالشكل دة انا as client هقول للbuilder انا عاوز بيت بيتكون من كذا وكذا . ف تعاله ناكد على معلومه امتى هتستخدام ال builder .

### 2.2.1:when we use builder :

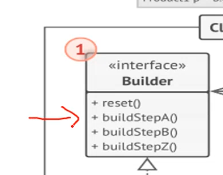
* لما يكون عندى كلاس بيتكون من مجموعه من ال object وال option دى بتختلف من creation للتانيه . يعنى مرة هعمل object بoption 1,2,3 ومرة عاوز اعمل object ب option 4,5,6 .
* الحل بيبقا ايه : هاخد ال creation بتاع ال option وحطه ف كلاس هنسميه builder .
* الbuilder الواحد هيكون مسئول عن بناء ال option المختلفه دى .
* لو انا عندى material معينه عاوز ابنى بيها ال option دى ف كدة لازم يبقا عندى اكتر من نوع من builder وكل builder يكون مسئول عن بنه البيت بتاعى دة .

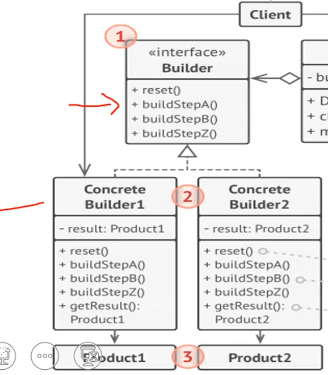
## 2.3-importain notes :

اخلينا متفقين ان ال pattern دة حتى الان احنا بنستخدامه واحنا عرفين كل ال option المتحاه .

المشكله ان احنا مش عرفين ال object الى طالع هو طالع من انهى option من دول.

## 2.4-stcture for builder :

1. هحط كل ال option الى ممكنه ف interface . وكل builder مسئول عن material معينه هيعمل implement للinterface ده .



1. جوه ال concrete builder هبداء اعمل implement ال function .
2. Product1,product 2 دة ال house الى مفروض يطلع لو هنتكلم عن المثال

الى فات

* لاحظ ان ال concrete builder ة هيكون جوها reference من الproduct

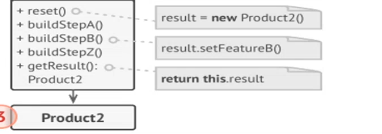
بتاعى الى هو ال house ف المثال الى فات عشان بعد ميعمل build هيكون عندى

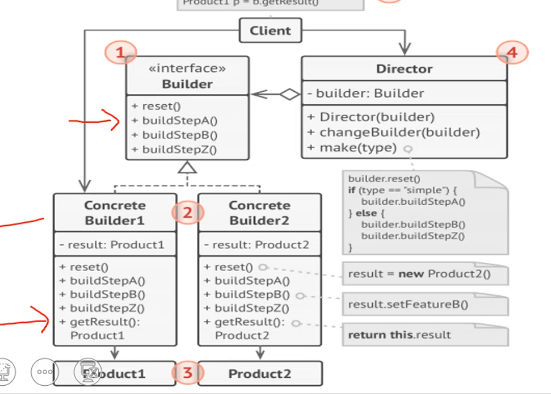
داله اسمها get result عشان ترجعلى ال final product بعد معمل مجموعه ال build

* يعنى ال client هيقول للbuilder ابنى الباب والحطان مثلا وبعد متخلص رجعلى

Object بال final product .

1. دة مثال للكود الى هيكون ف ال concrete class .

* reset بتعمل creation للproduct .
* buildstepB دى بضيف future جديدة للproduct .
* get result دى بترجع ال final product .

1. الdirector دة زى المقاول باضبط . لو مكنش ف director ف انا as client هبقا مطالب ان استعين ب ال builder واقله على الخطوات الى يمشى عليها . يعنى ف الmain هعمل object من concrete builder واروح اقول لل object دة اعمل stepB, و stepA وهكذا . ف كدة كتبت الخطوات بتاعه بناء ال object جوه كود ال client يعنى جوة ال main . ف لو الclient محتاج يعمل اكتر من بيت ف كدة كل مرة ال client هيفضل يعيد الخطوات بتاعه بناء البيت . ف بدال مفضل اعيد الخطوات هستعين بال director .

* ف الdirector بياخد object جوة الconstrictor

من ال builder الى انت عاوز تبنى بيه البيت .

* يعنى هعمل object من director وابعتله object من

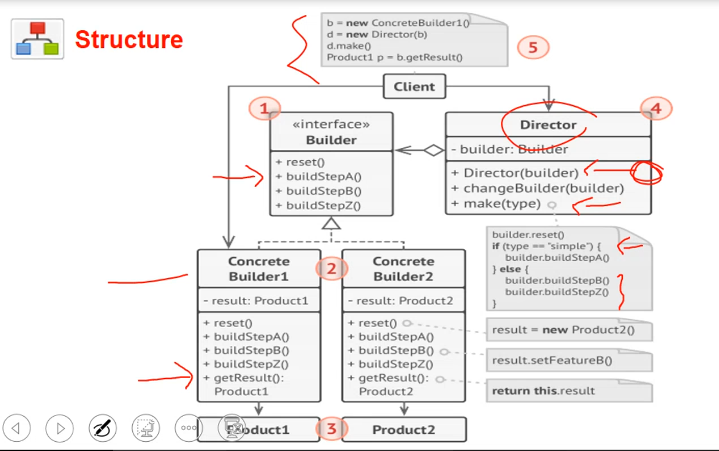
Concrete builder

* وبعد كدة بنادم على داله make من
* الdirector object
* ودة شكل داله make . كل الحكايه انها حسب ال type الى هتاخده هتعمل product فيه option معينه . يعنى لو كان ال type=simple . ف هى هتخلى ال producr فيه ال option رقم A بس وهكذا .
* 
* ف بالشكل دة ال director هو المقاول المسئول عن بناء ال product . ف كدة انا عندى اختيارين

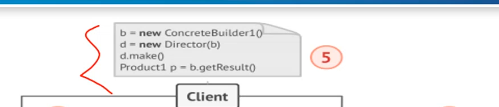
1. اما استخدام ال director واحد وابعتله ال type وهو حسب الtype الى اتبعتله هيعمل ال product .
2. اما اعمل interface اسمه directory واعمل منه concrete directory وكل concrete directory يكون مسئول عن بناء بيت معين او جواه خطوات بناء بيت معين .

ودة الافضل لما يكون عندى اكتر من نوع .

* لاحظ ان كلاس directory بيحتوى عل builder مش بيعمل implement for directory
* ال directory مش حاجه اساسيه ف ال builder pattern دة حاجة اضافيه مش لازم تعملها اصلا .

- ف دة شكل ال structure النهائى :

## 2.5-client code or code in main:

تعاله نشوف الكود الى ف الmain او ازاى الليله دى هتمشى واحدة واحدة :

1. هعمل object for concrete builder .
2. هعمل object from directory , و هبعت ال object بتاع ال builder للdirector constrictor .
3. هنادم داله make الى فهيا ال محدد الخطوات او ال option بتاع ال product الى عاوزة .
4. بعد مداله ال make تخلص كده ال builder عمل ال product ف انا عاوز ال product الى خلص دة ف هنادم داله ال get result عشان ترجعلى ال object بتاعى .
5. ولاحظ هنا ان ال directory كل الى احتاجه ان ابعتله object من ال builder بس الى عاوز ابنى البيت بيه .

## 2.6 –make mix product?

افرض مثلا ان احنا عوزين نبنى product يكون فيه خصائص من two builder .

لو عاوز ابنى product يكون mix بين 2 builder يبقا لازم يكون عندى concrete builder يكون مسئول عن عمل ال mix builder طيب ليه ؟ عشان انا ف الاخر بتعامل مع ال concrete builder وكل concrete builder بيرجع product واحد ومينفعش ادى product لل concrete builder وخليه يضفلى عليه حجات .

ف لو عاوز حاجة mix هتعمل concrete builder وتحط فيه الoption الى انت عاوز عشان تخليه mix .

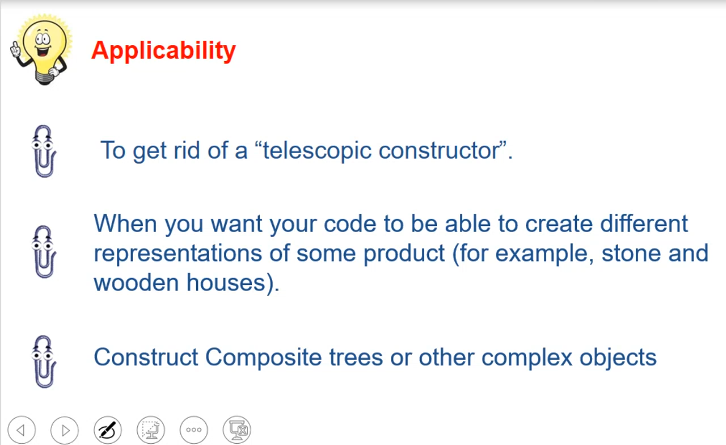
## 2.7-applicability :

امتى هنستخدام ال builder pattern :

1. لما الاقى عندى contractor حجمه كبير جداا . ايه الى هيخلى ال contractor كبير ان باخد parameter كتير يه الى هيخلينى اخد برامترز كتير ؟ متقليش ان فيهfield كتير وعاوز اعملها initlazation . احنا هنا مش بنتكلم ان عندنا fields اساسيه ف الكلاس بتاعى .

* احنا بنتكلم عن عندنا مجموعه من ال option المختلفه . وممكن استخدام منها شويه وشويه لا ف كل مرة بعمل فيها ال object .

1. زى مقلنا لو عاوز اكون قادر ابنى ال product هو هو لاكن بoption مختلفه كل مرة عن الى قبلها ز
2. طبعا تقدر بالشكل دة استعين ب ال builder عشان اقدر ابنى complex type .

* ودول نفس الخطوت الى قلتهم فوق بس مختصرين :

## 2.8 –cons and pros:

مميزات وعيوب الموضع .

1. الميزة ان ببنى الproduct بتاعى خطوه خطوة وانا الى بحدد الخطوات مش لازم امشى عليهم كلهم كل مرة .
2. اقدر اكتب الكود مرة واحدة بس زى مشفنا بمساعدة ال director كتبت الكود مرة واحدة بس وهو كان المسئول عن بناء الproduct.
3. Single responsibility : كل واحد مشغول ف حاجة واحدة بس . ولو حابب اضيف اىbuilder جديد مش هحتاج اعدل ف اى كود موجود .
4. والعيب كلعادة ان الكود بقى معقد شويه صغيرين .

## 2.9- :