12/12/2022

Abdo Gamal Ahmed

no campany

Design pattern

Intro



Table of Contents

[1 –references : 2](#_Toc121836386)

[2 –Goal of course : 2](#_Toc121836387)

[3 – what is Design pattern “DP” ? 2](#_Toc121836388)

[4- What does the pattern consist of ? 3](#_Toc121836389)

[5 – who invented DP ? 4](#_Toc121836390)

[6 – Why should learn pattern ? 4](#_Toc121836391)

[7 – classification DP : 4](#_Toc121836392)

[8- Future of good Design: 4](#_Toc121836393)

# 1 –references :

* تانى كتاب الى اسمه head first دة من احسن واسهل الكتب الى هتلاقيها بتشرح ال design patterns او اى محتوى عموما .
* اما اول كتاب دة حلو لو انت عندك خلفيه شويه .

# 2 –Goal of course :

1. نفهم يعنى ايه DP .
2. بنستخدام ال DP فى ايه .
3. ليه ال DP اتوجدت .
4. هو ليه ال DP مفيد بالنسبالنا .
5. ازاى نطبقه وايه المشكال الى بيحلها ال pattern المختلفه .
6. ايه المشاكل الى بيحلها ال pattern المختلفه .
7. اخر حاجة ال implementation بتاع ال pattern . والنقطه دى مش هتهمنا للدرجه , عشان اهم حاجة ف ال course انك تفهم المشاكل الى بيحلها pattern معين .
8. ف المهم بانسبالنا اكتشاف ال pattern الى ممكن يحل المشكله الى وقعنا فيها . ف مش هتعرف ال pattern الى محتاجه الا لو كنت فاهم كويس ال pattern .

# 3 – what is Design pattern “DP” ?

* ال DP هى مجموعه من الحلول لمجموعه من المشاكل الى دايما بتقبلنا واحنا بنعمل design للSW بتاعنا .

بمعنى ان ف مشاكل بنقبلها على المستوى الشخصى لنيا as developer واحنا شغلين ف design بتاعنا ف الى بنعمله ان بنبداء نبحث عن مجموعه من الحلول ونجربها لحاد منوصل للحل المظبوط ف بنبداء نعتمد الحل دة ولما تقبلنا نفس المشكله بنستخدام نفس الحل جربناه قبل كدة .

* ف فى مجموعه من المشاكل الى لحظنا انها بتتكر مع معظم الcommunity بتاعه ال developer .ف بدءنا ندرو على حلول للمشكال دى لحاد ملقينا انسب حل للمشكله الى بتقبلها . ف بداءنا نقول لما تقابلك المشكله الكذا استخدام الحل الكذا .
* نقدر نقول على ال Design pattern انها template او انها blue prints او انها road maps بتقلك لما تقابل المشكله الفلانيه المفروض تمشى على الطريق دة عشان تحل المشكله .
* ال DP as a template مش بيقدرملك الحل السحرى لحل المشكله .او مش بيدخل ف تفاصيل الحل هو يدوب بيوضحلك الطريق الى هتمشى عليه .حتى لو بنتكلم ان ال DP دة بيكون ليه implementation بس مش هو دة ال implementation الوحيد للDP يعنى ممكن ال DP الواحد يتعمله اكتر من implementation لاكن ف النهايه هى road map واحدة بتمشى عليها عشان توصل للحل .
* وخلى بالك ف فرق كبير بين ال algorithm وال Dp .يعنى ممكن تقلى ال algorithm بيديك حل للمكشله صح ؟لا مش صح ال DP دة اعم واشمل من ال algorithm .عشان ال algorithm ف الاول وف الاخر بيديك مجموعه من الخطوات الى مفروض تمشى عليها عشان تحل مشكله معينه .ولو فيه اكتر من واحد شافوا خطوات ل الجروزم ف الاخر كلهم هتلقيهم كتبه نفس الكود تقريبا .

امال ف ال DP احنا بنتكلم عن General solution بنقول فيه المفروض يكون عندك كذا وكذا والمفروض توفر مذا وكذا .

يعنى مثلا ف الاخر الكذا دة اعمله class ولا interface بارحتك بس كل واحد هيكون ليه مميزات وعيوب . لكن ف الاخر ال DP مش هيجبر يكون عندك الحاجة الفلانيه ف الكود .

* الالجروزم دة باضبط زى الوصفه بتاعه اى اكله هيكون فيه المقادير والخطوات الى هتمشى عليها عشان تطلع طبخه معينه .
* اما ال DP دة حاجة اعم واشمل الى مثلا هتقلك ازاى توفر ال environment المناسبه عشان توفر الطبخه الفلانيه .ف ال Dp ف الاخر هيديك مثال على ال implementation لاكن انت مش مجبر تعمل implement للpattern بشكل دة.

وعشان كدة كمان هتلاقى ف two pattern كل واحد فيهم بيحل مشكله غير التانى لكن لما تيجى تبص على ال implementation بتاعهم جايز جدا تلاقى ال implementation ده واحد ؟ّ! يعنى مثلا انا بحل المشكله x وانت بتحل المشكله y كل واحد فينا هعمل implement للحل للمشكله لوحدة لكن ف الاخر ممكن تلاقينا مطلعين نفس الكود . عشان زى مقلت قبل كدة ال pattern مش بيقلك انت هتكتب كود عامل ازاى لاكن بيقلك الطريق الى هتمشى فيه عامل ازاى .

# 4- What does the pattern consist of ?

* اى pattern ف الدينا لما تيجى نتكلم عليه او تقراه من كتاب هتلاقيه دايما مكون من اربع حجات .

1. : Internالمشكله الى بيحلها ال. Pattern يعنى هيقلك ف جمله واحدة المشكله والحل بتاعها .والجمله دى لو طلعنا بيها وفهمنها ف كل pattern كدة زى الفل مش عوزين حاجة تانى, طيب ليه عشان كونك فهمت الpattern بيحل انهى مشكله دة هيخلينا ان شاء الله لما نقع ف المشكله المشابه لدى هنعرف حل المشكله ف اروح ابحث عن ال implementation بتاعها .
2. : Motivation تانى حاجة بنتكلم باتفصيل شويه عن المشكله وواسبابها والحلول المتاحه وليه اخترنا الحل الفولانى وساب باقى الحلول ليه اكيد هيكون فيه اسباب ؟
3. Structure for pattern: يعنى هيقلك لما تشوف المشكله دى المفوض تعمل structure بتاع ال class بالشكل معين عشان تحل المشكله.
4. Code example: :هيقلك implementation بتاع المشكله زى مقلنا مش ملزم تلتزم ب ال implementation دة .بس المهم تلتزم بحل المشكله as concept .

* وعاوز تتخل لما بنتكلم عن ال pattern زى بالضبط لما كنا بنتكلم عن ال Data structure ف الجزء بتاع ان هى مش مرتبطه بلغه معينه عشان كدة هتلاقى ال DP مش بتتكلم معاك ف تفاصيل كتير .

ف مثلا ممكن تكون ف programing language مفهاش interface ف هتعمل ايه ؟ ف عشان كدة ال DP مش مرتبطه بلغه معينه .

# 5 – who invented DP ?

* مفيش حاجة امسها مين اخرت ال DP عشان ال DP تم اكتشفها مش اخترعها . زى مقلنا مش نشويه ان ال Dp هى عباره عن حلول لمجموعه من المشاكل الى ظهرت معانا . ف احنا مخترعناش ال pattern احنا اكتشفنا ان دى انسب طريقه بحل مشكله يعد التجربه وكدة يعنى .ولما اكتشفنا الطريقه الى هنمشى بيها لحل المشكله دى بدانا نديلها اسم .
* اول ناس اتكلمت عن ال DP كانوا اربعه جمعه شويه مشاكل عامه وقالوا يجمعاه احنا لقينا الناس لما بتيجى تواجه المشاكل دى لقينا ان الحلول دى هى الانسب . والكتاب بتاعهم اتسمه the gang of four
* ومن اشهر الناس الى اتكلمت عن الموضع هم uncle Bob و reboot see marted انا شخصينا شوفت كتاب clean كود ل uncle bob وكان عظيم الصراحه . "تانى اسم شكلى كاتبه غلط "

# 6 – Why should learn pattern ?

* عشان ديما هتقابل مشاكل كتير ودايما هتكون الحلول قدامك ليه بنقول هتقابل مشاكل كتير عشان ف ناس جربت قبلك ووقعت ف مشاكل انت هتقع فيها بعدين
* الناس دى من كتر موقعه ف مشاكل معينه حطه حلول للمشاكل دى .
* والحاجة المهم جدا ان تعلم ال DP بيسهل عمليه ال communication بينا كتيم . يعنى لما نقع ف مشكله معينه واحنا بنعمل design للSW ف بدال محد يشرحلك خوات حل مشكله يعنى بدال ميقلك اعمل 2 class زيادة واعمل implement للinterface الخ ف يقلك استخدام ال DP الفلانى ف انت تفهم الحل من غير شرح كتير ملهوش لزمه ف دة هيسهل ال communication بين التيم يخويا .

# 7 – classification DP :

لما شفنا المشاكل بتعتنا لقينا ان احنا بنواجه 3 مشاكل اساسيه . بيتفرع منهم شويه مشاكل كتير .

1. creation pattern :لقينا ان احنا بنواجه مشاكل واحنا بنعمل create لobject ف عشان كدة طلع مجموعه من الحلول مجموعه من ال pattern صنفنها تحت category اسمه creation pattern .ف ال creation pattern من اسمها هى عباره عن مجموعه من ال pattern الى هتلحنل المشاكل الى هتقبلنا واحنا بنعمل create لobject

وايوه ف مشاكل هتقبلها وانت بتعمل create ل object وفايه الpattern هنا ان هيسعدنا لحلها من غير متفتح الكود وتبداء تغير فيه حاجه .

1. : Structure pattern ودى هتساعدنا ف حل مجموعه مشاكل هتقبلنا واحنا بنبنى ال class class وعلاقه ال class ببعض.
2. Behavioral pattern: دى مجموعه ال pattern الى هتسعدنا ف حل مشاكل ال communication بين ال object. طبعى جدا ف اى app ان احنا بنبنى class وبطلع منها object عشان ال object دى ف الاخر تكلم بعض .

# 8- Future of good Design:

* قبل منبداء ف ال pattern هنتكلم على مجموعه من ال concept وال principle لازم تكون موجودة ف ال app بتاعى . يعنى هنشوف مجموعه من ال concept وال principle الى ال pattern جايه عشان تاكد عليها .
* يعنى كل منخلص pattern هنقول الpattern دة قدر يطبق ال principle كذا وكذا .
* خلى بالك من ال principle الى جايه عشان هنتكلم عليهم ف كل ال pattern وكمان مفيش اى SW هيطلع غير لما نتاكد ان هو ماشى على مجموعه من ال principle الى جايه مش لازم يكون ماشى على الكل بس على الاقلى يكون متبع مجموعه من الى هنقولهم .
* ايه الحجات الى بحكم بيها ان ال SW بتاعى الى طالع دة كويس ولا لا وهنا **مش** بتكلم من وجه نظر ال client . احنا هنا بنتكلم عن ال good design من وجه نظر ال Developer

يعنى هل انت بنيت ال SW صح ؟ لو بعد فترة من لوقت ال requirement اتغيرت ف بناءا عليه عاوز تغير حاجة فى الكود عندك ف هل هتقدر تعمل دة بسهوله ولا هتهد حاجة موجودة عندك ؟ولا هتشيل كود وتحط كود تانى .

* **ف الحجات الى اقدر احكم بيها على ال good design هو كويس ولا لا هى :**

1. Code reuse: :يعنى الكود الى كتبته دة لو حبيت بعد شويه هل تقدر تستخدامه ف app تانى من غير مشاكل ولا هتحتاج تهد وتبنى الدنيا من الاول ولا هتغير حجات بسيطه ولا هترمى الكود اصلا وتبنى من الاول خاالص ؟ وهيكون ف معاير نقدر نحكم بناء عليها هل ف code reuse متحقق ف الكود الى كتبته ولا لا ؟
2. Extensibility :الحاجة الوحيدة الثابته عندنا ان ف change على مستوى ال requirement .فدايما هيحصل تغير ف بالتالى انت مطالب دايما تغير ف الكود بتاعك ف ازاى اخلى ال app بتاعى قادر ان اغير فيه ازود فيه future و requirement من غير ماثر ف الكود الى كان مكتوب قبل كدة . ودة برضو ف مجموعه من المعاير اقدر احكم من خلالها على هل الكود الى انت كاتبه دة لو حولت بعد شويه ان اغير او ازود future بنسبه قد ايه انت هتغير ف الكود بتاعك دة .

* ف من اهم الحجات الى بنتكلم عليها واحنا بنتكلم عن ال good design من وجه نظر المطور هى ال code reuse , extensibility والنسبه الى هتغير بيها الكود بتاعك لما ال requirements تتغير .

# 9 - Design principle :