



binary

team

عدد الصفحات : 4

م. روان قرعوني

المحاضرة : 1

عملي هندسة البرمجيات 3

➤ سنتناول في هذا المقرر المواضيع التالية :

- Design Pattern (DP) : 7 marks(home Work) — 3 or 4 lectures
- Meta Model (MM) : 7 marks
- Petri Net (PN) : 6 marks
- Aspect : بدون علامات فقط من أجل المساعدة وتوضيح النظري




➤ إن ال Design هي أفكار وحلول لمشاكل معروفة مسبقاً ومكررة بحيث عندما تتكرر المشكلة أمامنا فبمجرد رؤية ال Class Diagram سنعرف المشكلة المشابهة ومن ثم نقرر ال Design الأنسب .

➤ وفي كل Design سنذكر :

- (1) المشكلة .
- (2) اسم ال Design .
- (3) نوعه : هناك 3 أنواع : Behavioral — Structural — Creational
- (4) كود
- (5) مخطط الصفوف .

➤ وبما أن ال Class Diagram هو شيء ديناميكي فإنه من الممكن أن نغير في الصفوف بحسب ما يناسبنا .

➤ إن أي Class Diagram فيه العلاقات التالية :

- Association 
- Composition 
- Aggregation 

➤ وفي الأنواع الثلاثة في كل Class نحن نعرف غرض من الآخر لكن الفرق مفاهيمي أما في ال Code فهي نفسها .

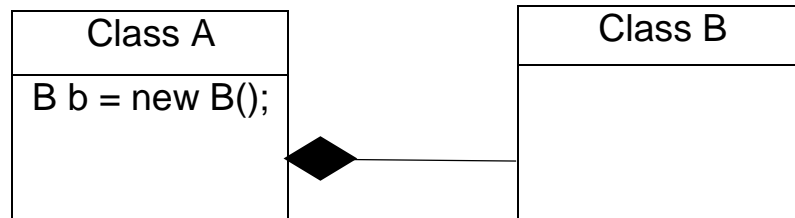
✓ مثال توضيحي :

✓ Association:



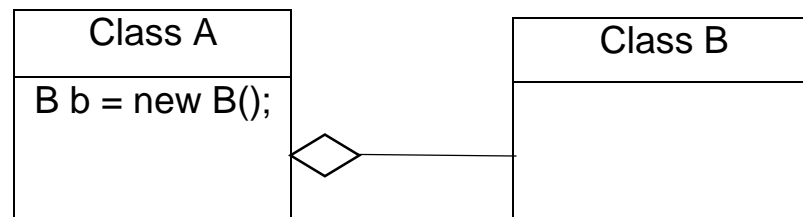
لنفرض لدينا صف طالب وصف سيارة ؛ لا يوجد أي ربط مفاهيمي أو علاقة بين الصفين لكننا قد نحتاج إلى أن نعرّف في صف الطالب غرض من سيارة .

✓ Composition:



لنفرض لدينا صف سيارة A و صف دولاب B ؛ عندما أدمر غرض السيارة يتدمر غرض الدولاب .

✓ Aggregation :



لنفرض لدينا صف شركة A و صف موظف B ؛ عند تدمير غرض الشركة فإن غرض الموظف يبقى على قيد الحياة و لا يتأثر .

✚ سنبدأ اليوم بأول أنواع ال Design :

ملاحظة :

لن نتناول جميع أنواع ال Design ومن الممكن أن يطلب في الوظيفة Design لم يتطرق له الدكتور في المحاضرة ؛ لكن ما يهمه هو أن يكون ال diagram صحيح بغض النظر عن الاسم .

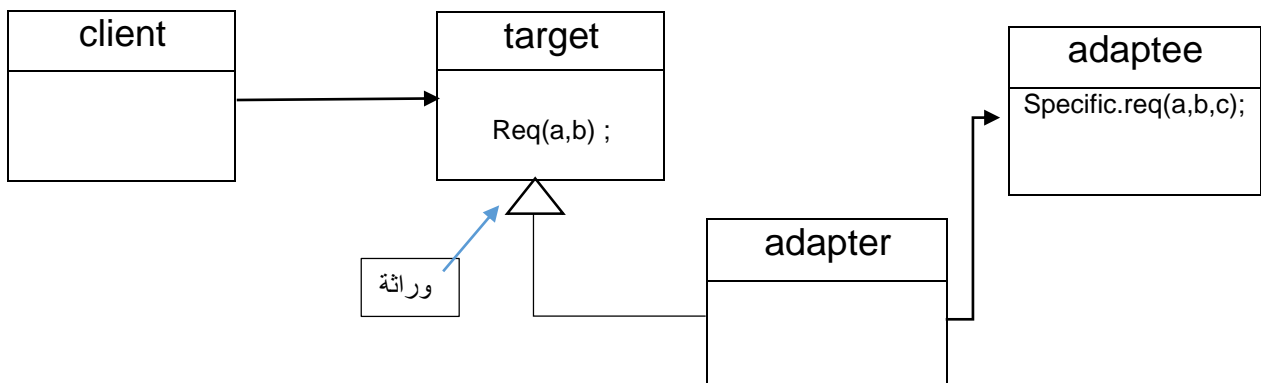
➤ Adapter:

- نوعه : structural .
- المشكلة : لدينا غرضين غير متوافقين ونستخدمه لتحقيق التوافق بينهما.

مثال: ليكن لدينا مكتبة فيها تابع له (3 باراميترات) و التابع المتوفر في الصف الذي أنشأته له (2 باراميترات)؛ ومن الممكن أن يكون الاختلاف في نوع الباراميترات وليس في عددها (لأننا نتحدث مفاهيميا وليس حرفيا) .
الحل في هذه الحالة هو أن نضع وسيط بحيث يوافق بين المكتبة و ال Code لدينا .
و يكون ذلك بعدة طرق ؛ لكن الحل الأمثل هو ما وضعه ال Adapter .

مثال:

- قمنا بإنشاء (2 Classes) : Client , Target ؛
- والكلاس الوسيط : Adapter ؛
- والمكتبة : Adaptee .



```
class Client :
Target tar = new Target();
Void F()
{
tar.req(a,b);
}
```

```
class Adapter extends Target :  
    Adaptee A = new Adaptee();  
    int Req (a,b)  
    {  
        // نكتب كود مناسب لتحديد قيمة المتحول c  
        A.specificReq(a,b,c);  
        Return y;  
    }
```

✚ إن المعرفة العميقة والفهم لل Design Pattern يسهّل علينا نمذجة المشاكل التي تعترضنا و إيجاد ال Design Pattern المناسب لكل مسألة .

انتهت المحاضرة

Written by :

Shorouq Abu Hasan

Word press and preparation :

Afaf AlAwam

Reviewed by :

Walaa Jaweesh