12/14/2022

Abdo Gamal Ahmed

no campany

DP

Part2



Table of Contents

[1 – In previous part: 2](#_Toc122211565)

[2 – encapsulation on method lever : 2](#_Toc122211566)

[3-program to interface not an implementation . 4](#_Toc122211567)

[4 – What is step to make class depend on interface not implementation. 5](#_Toc122211568)

[4.1 - over engineering : 6](#_Toc122211569)

[5 – program to interface not an implementation ex2. 6](#_Toc122211570)

[5.1 – tightly coupled 7](#_Toc122211571)

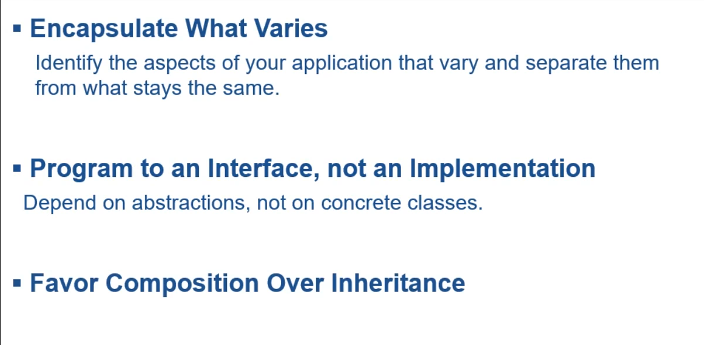
[6.1 - favor composition over inheritance : 9](#_Toc122211572)

[6.1- - favor composition over inheritance in English : 12](#_Toc122211573)

[7 - solid principle : 12](#_Toc122211574)

# 1 – In previous part:

ف الpart رقم 1 اتكلمنا حن حجات كتير من اهمهم Design principle . ف هنخدها بلتفصيل دلوقتى . ودول ال principle 3 .



# 2 – encapsulation on method lever :

ال function الى اسمها getorderTotal دى مسئوله انها تحسب ال total بتاع ال order ف كل الى بتعمله انها بتمشى على المنتحات الى هيشتريها الشخص ويضربها ف السعر . ودة مفهاش مشكله

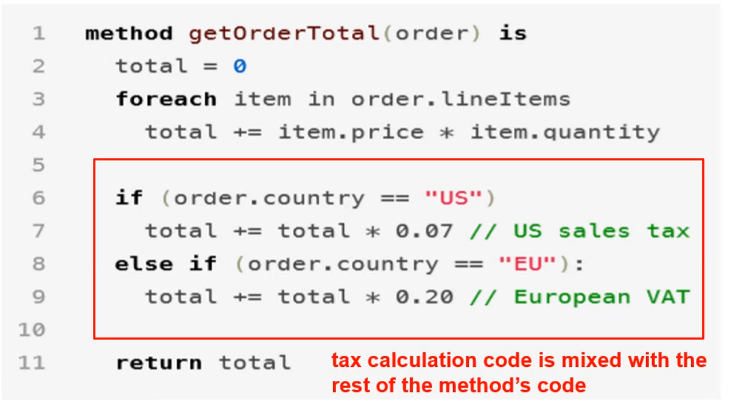
**المشكله بقا :**

1. ف الجزء التانى بتاع حساب ال tax الضرايب الى علىorder ف المضوع دة بيحسبه على حسب المكان الى ريحله ال order

المشكله بقا : كل متتغير طريقه حساب ال tax انت هتفتح ال func وتغير فيها وكمان كل مضيف متزيد دوله جديدة هتحتاج تفتح ال function وتغير فيها .

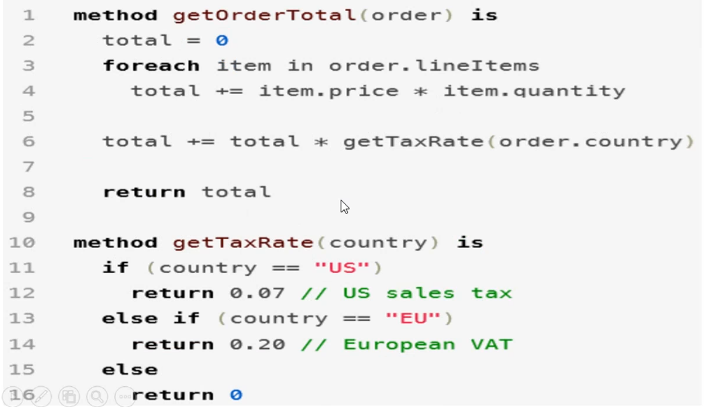
1. وكمان ال code الى بيحسب الضرايب موجود ف نفس المكان الى بيحسب الtotal على ال order . ف كدة ف mixing بين والاتين على فكرة business مختلف

* يعنى حساب ال total عن طريق ان هيلف على العناصر ويحسبهم دة business .
* وحساب الtax دة ملهوش علاقه بالجزء الاول ودة معتمد على البلد الى هروحه ال order .

1. ف اول مشوف حاجة زى كدة ان كل ما الcountry اتغيرت هفتح الfunction دى واغير فيها .

**ف الحل :**

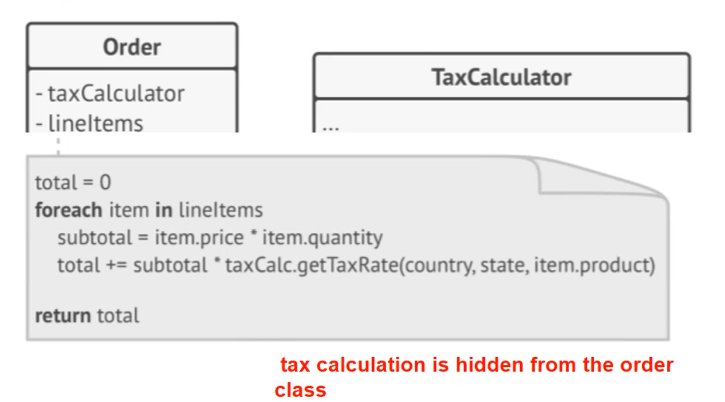
1. ف الحل ان افصل الجزء الى هيتغير ل function لوحدة .
2. دة الشكل بعد مفصل الجزء الى هيتغير لوحدة . ال function الى اسمها getTaxRate دة هتاخد ال counter وعلى اساس الcounterty هتحسب ال tax وهنادى ال func دى من جوة ال getOrderTotal .



1. كدة تمام حلينا المشكله .؟ لا للاسف لسه لو خت بالك هتلاقى ان ال function الى اسمها getTaxRate دى موجودة جوة order class لسه ف كدة كل ما ال country تتغير هفتح ال class اسمه order وغير ف الداله getTaxRate . ف عشان كدة حد جه وقال ال function الى اسمها getTaxRate دى ملهاش علاقه ب ال order class عشان حساب ال tax معتمد على ال counter مش على الorder والكلام دة as a logic . ف مش منطقى احط getTaxRate جوة كلاس order وكل متتغير حاجة ف الضرايب اغير ف class order .
2. لاحظ التدريج الى حصل ف الاول كنا بنتكلم عن جزء ف function كان بيتغير بستمرار ف فصلناه ف func لوحدة وبعد كدة او دولقتى بقنا بنتكلم عن جزء ف class كل ميتغير هفتح الكلاس وغير فيه وكمان ان ال class order ملهوش علاقه بالجزء الى

بيتغير دة as logic . ف عشان كدة هنقرر نفصل الجزء الى بيتغير ف class لوحدة .



1. ف هنحط الداله بتاع getRaxRate ف كلاس وهنسميه TaxCalculator ,وهناخد object من ال TaxCalculator class جوة ال order عشان نحسب ال Tax .

**الهدف من الى حصل :**

ان اخلى كل واحد مسئول عن الحاجة بتاعته . لما اتكلمنا زمان عن ال class والكلاس دة بيشيل

member variable و member function . وقلنا ان المفورض ال member variable دى بتعبر عن الخطائص والصفات بتاعه الحاجة الى بتوصفها .يعنى انا لو عامل class اسمه employee طبيعى جدا الاقى جوه variable اسمها name ,address

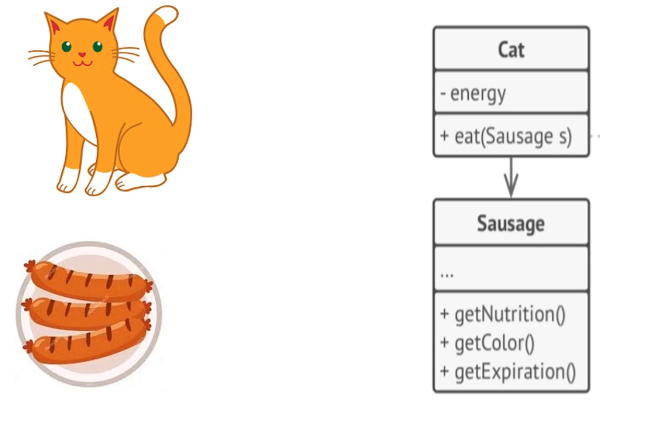
وقلنا ان ال member function دى بتوصف ال behavior بتاع ال object فطبيقى الاقى function اسمها getabsentDay .

ومثلا من الطبيعى تلاقى داله لحساب ال salary جوة ال employee بس مش دايما طبيعى ؟؟

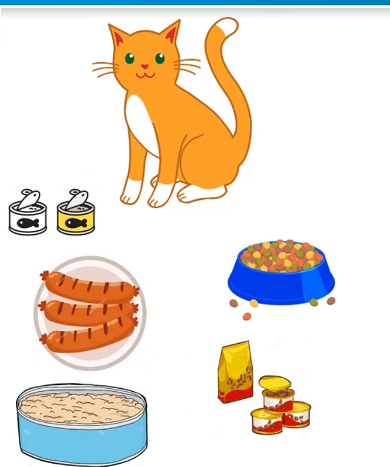
* لو الsalary الراتب الشهرى بتاع الموظف والراتب السنوى وكدة ف طبيعى جداا احط حساب ال salary جو الموظف
* اما لو كان مع المرتب بحسب هحط موضوع الضرايب وحساب الضرايب مش معتمد على ال business بتاع ال employee ف كدة هنفصل موضوع ال salary وال tax عن الموظف او ممكن نفصل ال tax بس عن الموظف حسب الlogic وهنشوف مثال تانى على الجزء دة لما نشرح ال solid principle .

# 3-program to interface not an implementation .

وهنا لما شرحنا الجزء دة قلنا اعتمد على interface not concrete class . ف تعاله ناخد مثال ونوضح ايه السبب .

لو قلتلك اعمل برنامج عشان تاكل القطه دى السجوق. ف انت عملت 2 class واحد بيمثل القطه والتانى بيمثل السجق وعملت object من ال class سجوق جوة كلاس القطه وبعت ال object دة لداله اسمها eat ف هل كدة ال design صح ؟

ايوة صح بنسبه 100% بالنسبه للrequirment الى موجودة عندنا

 بس لو غيرنا requirement بسيط هنحتاح نغير ف الكود ؟

* لو جيت وقلتلك عاوز اخلى القطه دى تاكل مجموعه من الاكلات المختلفه ؟

ارجع وبص على ال design الى كنت عامله هتلاقى نفسك عاوز تغير ف

Design الى عملته طيب ليه? السبب الاساسى الى هيخليك تغير

ف ال design هو cat class عشان فيه داله eat معتمدة على

Concrete class. طيب الحل دلوقتى ان نخلى الeat تعتمد على

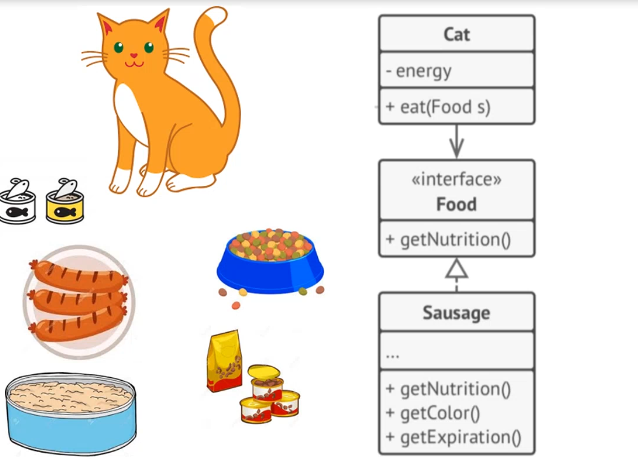
Interface .

وقصدى ب interface من ناجيه الوجك layer

وقصدى ب interface من ناجيه implementation اما interface او

Abstract class .

وده المنظر لما نغير ال design ونخليه معتمد على interface مش concrete class .

**الى عملته عشان احل المشكله او احسن ال design :**

1. عملت abstract class اسمه food
2. خليت الداله eat تاخد object من class food

* ف كدة انا معتمد على layer بينى وبين

الclass concrete الى هى

Class abstraction

1. فاقدر بعد كدة اخلى القطه تاكل اى حاجة ف الدنيا عادى

كل الى هتعمله Class وتخليه يورث من food

وتبعت object من ال class دة لeat .

1. ف خلى لسوال دة ف دماغك دايما :
2. اعتمد على abstraction ومتعتمدش على concrete class .

وبقصد بكلمه يعتمد على يعنى بيستخدام جواه .

1. الميزة لما اخلى ال class بتاعى يعتمد على abstraction ان

اخلى ال design بتاعى fixable يعنى اى حد ف الدنيا يقدر

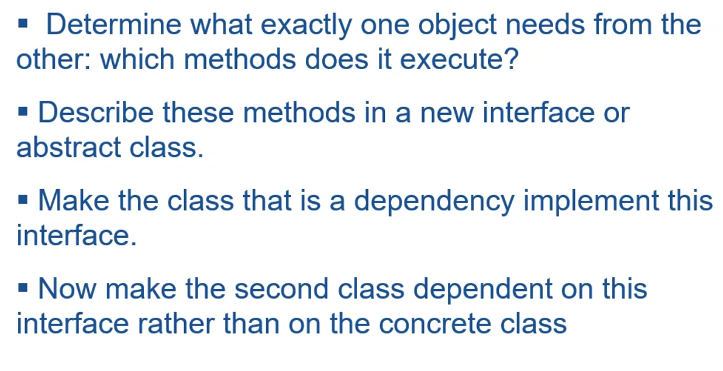
يضيف future جديدة من غير مرجع اغير او اضيف حاجة ف class كات الى عملته .

1. وللصراحه الموضع بقى flexible بس يقى فيه سكه من التعقيد بس مش مشكله . كل الشرح الى جاى هيكون فيه شويه تعقيد بس عشان اقدر اخلى الكود بتاعى flexible . ف معظم او كل الحلول الى هنشفها بعد كدة ان اضيف class او interface .

# 4 – What is step to make class depend on interface not implementation.

ازاى نعتمد على abstraction مش implementation . هنمشى على الخطوات الى جايه .

1. هحدد ايه ال function الى محتاها من ال class التانى . هناخد مثال القط السجوق هحدد ال cat class هو عاوز function ايه من ال object السجوق .
2. بعد محددت شويه الحجات الى عوزها هحطهم ف interface او abstract class .
3. هخلى class cat معتمد على ال interface الجديد دة . ولاحظ هنا cat كمثال مش اكتر .
4. بعدين خلى باقى الناس الى هتشتغل معاك يعمله inheritance او implementation .يعنى خلى السجوق class يعمل implementation و لو عندك اكتر من اكله خليها كلهم يعمله implementation ل interface

ودى الخطوات بالنجليزى لو انت جامد سحليقه ومش بتفهم عربى . وملحظه كلامه سجوق كتبها غلط تقريبا كتبها سجق ف لو عدلها لو انا نسيت اعدلها .

## 4.1 - over engineering :

دة معناه انك تدى الموضع اكبر من حجمه .

بعد شويه هنشوف حاجة اسمها sold principle ودة بيتكون من 5 principle ف هتلاقى واحد وهو شغال بيحاول يطبق ال 5 كلهم مع ان احنا زى ماهنشوف ان اى principle هضيفه هيعقد الكود شويه ف فى حلات المشكله الى بتحلها هى مشكله صغيره بس انت بتعملها

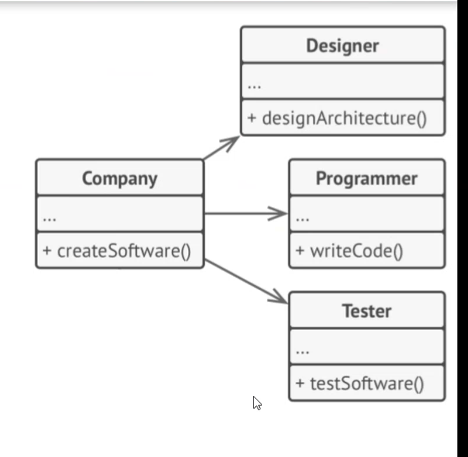
Engineering over يعنى بتدى الموضع اكبر من حجمه.

الخلاصه : متحولش دايما تطبق كل ال 5 prince بختصار متعملش الحاجة ام جنيه وهى هتكلفك 100 جنيه .

# 5 – program to interface not an implementation ex2.

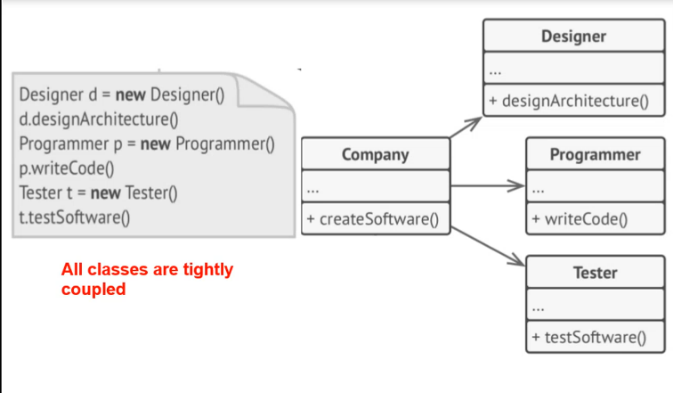
عاوز اعمل app يحاكى ان عندى شركات SW .

ف هعمل company class وحط فيه method اسمها create Software

وعندى مجموعه من الموظفين لكل واحد هعمل class .

وكل واحد من الموظفين جواه فنكشن بتخليه يعمل الشغل بتاعه .

و

طيب حسب التصميم الى انا عامله هاجى جوة class company واخد object من كل الموظفين الى عندى جوة فنكشن createSoftWare ومن ال object ده هنادم فنكشن الى محتاجها .

طيب التصميم دة لسه فيه مشكله ? ايوة والمشكله ان ال

Class معتمدين على بعض مش على interface.

وهنا ف مشكله تانى اسمها tightly coupled . تعاله نشرحها ونرجع بعدها نحل المشكله

## 5.1 – tightly coupled

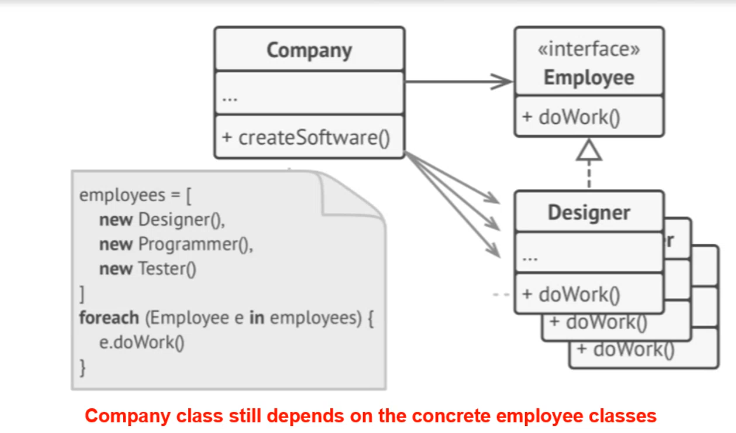
معنها ان ال class ماسكه ف بعض قوى .وده عيب كبير خاالص ف احنا دايما بنفضل نعمل حاجة اسمها lose coupled ليه ? معنى tightly coupled ان ال class مرتبطين ببعض قوى ف اى تغير ف اى واحد فيهم بنسبه مليون الميه هياثر على التانى

وانا as class company بقيت مرتبط بشكل كبير بباقى الclass الى معايه .

ف لو انا جيت بعد شويه وقلتلك ان هنضيف عضو جديد للموظفين يعنى هضيف GameProgramer class ف كدة تقدر تعمل extent للmethod الى جوة GameProgramer من غير مفتح company class ؟؟

اكيد لا لازم افتح ال company class واحد object من ال class الجديد.

* ف كدة قدرنا نعرف ان كل ما انواع الموظفين التغيرت هنعدل ف company class .
* ف كدة هنرجع لل principle الى بيقول encapsulate what vans و program interface not implementation

**تعاله ترجع نحل المشكله** : احنا هنعمل interface ونخلى الموظفين والشركه تعتمد عليه . زى معملنا ف لمثال الاول بتاعى

program interface not implementation

طيب كدة حلينا المشكله ؟ لا محلنهاش للاسف عشان

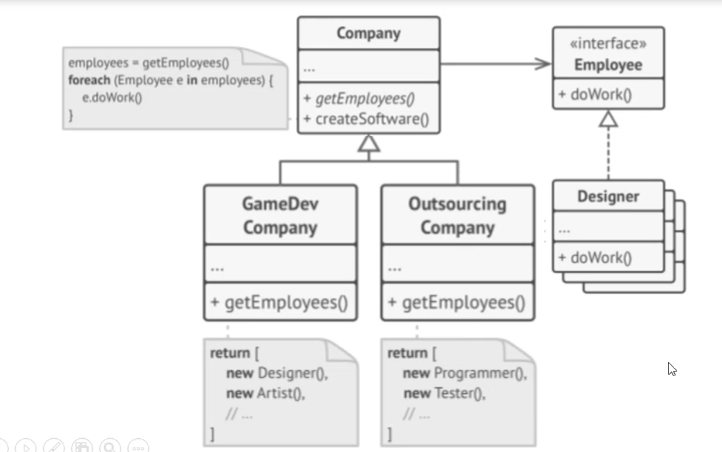
كدة برضو هفتح ال company كل معمل class جديد تبع

الموظفين عشان استخدام ال function الى فيه .

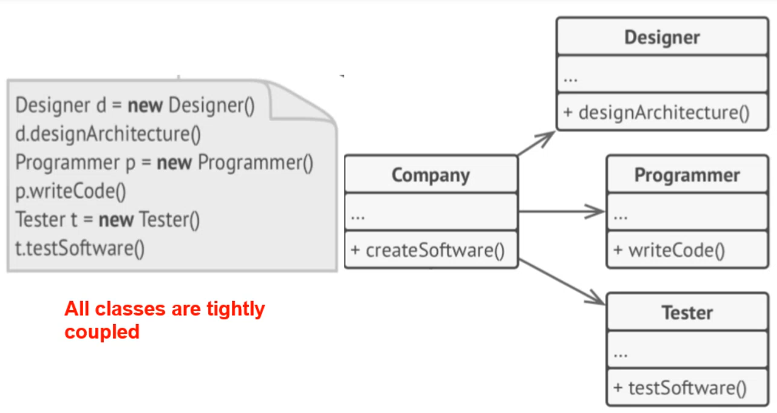
طيب لو جيت بعد شويه . وغيرت الrequirement وقلت ان ممكن يكون عندى انواع للشركات مثلا شركه مختصه ف الDatabase وشركه ف ال Gaming الخ ف انواع الموظفين الى ف كل شركه هتحتلف حسب التخصص بتاع الشركه :

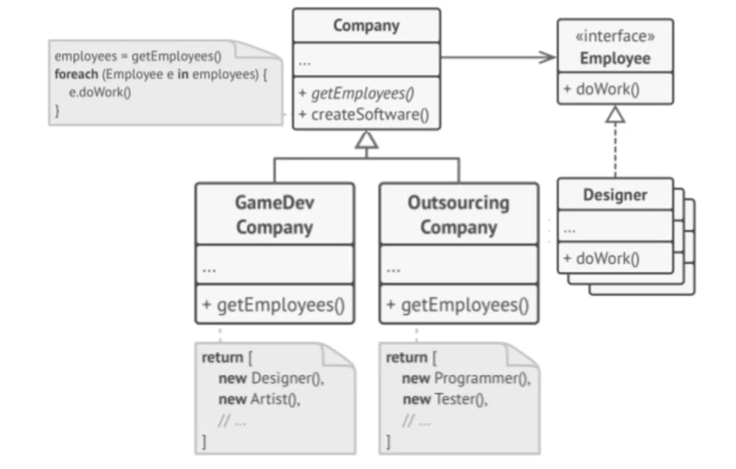
ودة هيكون شكل التصميم الجديد وكل الى عملته خليت companyتكون abstraction class وفيها 2 method الاوله هى abstract واسمها get Employeeودى الى هيعمل implement ليها الى هيورث من companyوالداله التاينه هى createsoftware ودى يغالى كل الى بتعمله انها بتنادى داله get Employee الى هيكون معمها implement جوة child class بتوع ال company وكمان بتمشى لوب على كل ال employee وتنادى داله do work . طيب كدة كسبت ايه اكتر من التصميم الى فات :انت كدة مش هتفتح الabstract class الى اسمه employeeخلاص هيكون ثابت

بس للاسف انت هتفتح الكلاس الى ورث من ال company وتعدل فيه لو حبيت تضيف موظف جديد ف شكره معينه عشان هم معتمدين على concreate class مش interface . طيب والحل ؟؟ هو للاسف لحاد دلوقتى احنا متعلمناش الحل قدام هناخد طريقه لحل المشكله دى. بس لاحظ ان ف تحسن عن التصميم الى فات وشرحتلك ازاى .



ممكن حد يقلى عشان اتفاده المشكله ان ممكن اغير ف الclass الى ورثت من company ان احول ال class دى ل abstract class وخلى ناس تانى تورث منها وهكذاا . هنا بقا ف نقظه انت ف شغلك الافضل تعتمد على ال abstraction مش على ال concrete class ولاكن دة هيكن لانهى level ؟ بالعقل كدة ان ف الاخر لازم اعمل concrete class عشان اعمل object من class يعنى ماشى هفضل اعمل abstract او interface وخلى class تورث منهم بس هفضل اعمل دة لحاد امتى يسطا ف مرحله لازم يبقا ف concrete class اوابقا معتمد على concrete class الفيصل او الفرق فين هو انهى level انت معتمد فيه على concrete class :

* ف الdesign الاول الى عملناه دة تصمصم وحش خالص ليه عشان انت من اول level معتمد على concrete class ف لو ف اقل تعديل حصل هتشقلب ال design كله .
* اما تانى تصميم دة : ايوة انا معتمد على concrete class بس امتى ؟ ف ال level رقم اتنين او تلاته ف لو قلتلك ان ف شركه جدية هتفتح او قصدى نوع جديد من انواع الشركات فتحت هل هحتاج اغير حاجة ف الكود الى عامله دة ؟ الاجابه لا كل الى هعمله هخلى ال شركه الجديده تعمل inherit للcompany .
* المشكله كلها ف مثلا لو class gamerDev عاوز يزود عنده employee ف غصب عنى هحتاج اعدل ف ال class والمشكله دة هنشفلها حل بعدين متقلقش .



# 6.1 - favor composition over inheritance :

اخر نقطه ف ال inheritance . هنفضل ال composition عن ان نعمل inheritance : اكيد ف مشاكل ف ال inheritance عشان كدة اوقات بنفضل ال composition .ف تعاله نشوف المشاكل

1. ال child class مينفعش يقلل من ال interface بتاع ال parent class . يعنى مينفعش تقلل من الحجات الى هتورثها من ال parent class
2. وانت بتعمل override لازم تتاكد ان ال behavior الجديد يكونcompatible مع ال behavior القديم .يعنى لما تعمل override انت مش بتغير ملامح ال function بمعنى ان روح الfunction هتفضل هى هى انت مجرد بتغير ال implementation يعنى لما اعمل override ل function دة معناه ان هعمل نفس ال function زمان بس بطريقه مختلفه لاكن انت لسه بتعمل نفس ال action . لو هتعمل override ل function اسمها eat يبقا انت هتعمل نفس ال action الى اسمه eat بس بطريقه مختلفه .
3. لو انا parent class وحاتط فيه protect member وجه ال children class ورث منى ف كدة ال child class هيورث الحجات ال protect . طيب انت مفروض عامل ال protect member ليه ؟ عشان محدش يقدر يغير ف ال member ده وعشان كده عملين ال set ,get function ف كدة الناس الى معاه object من ال class بتاعك عشان يوصله لل protect ده هيوصله من خلال

geter ,seter طيب جميل . اما الناس الى وارثه ال protect member دول مفيش اى اجبار يخليه يستخدام ال set function

عشان يغير ف قيمه المتغير . ودى جريمه زى متفقنا زمان ف دة تالت عيب ان ممكن اعدل ف قيمه prodect member من غير

مرجع لل set func .

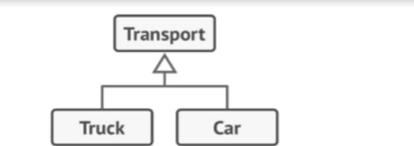
1. الsubclasses مرتبط ارطبات وثيق ب ال upper class ودة الى اسمه tightly couple .

وشرحته قبل كدة ودة مثال سريع تانى : لو ال superclass زود function او غير ف function وبدال مهى public خلاها private كدة ال subclass للاسف هتتاثر .

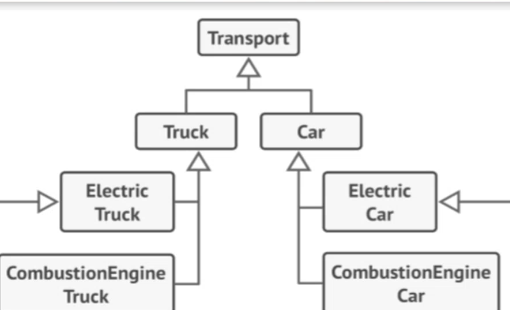
1. من المشاكل الجباره الى هنشوفها : تسبيب ال parallel inheritance hierarchies وتعاله ناخد مثال عشان نوضح المشكله :

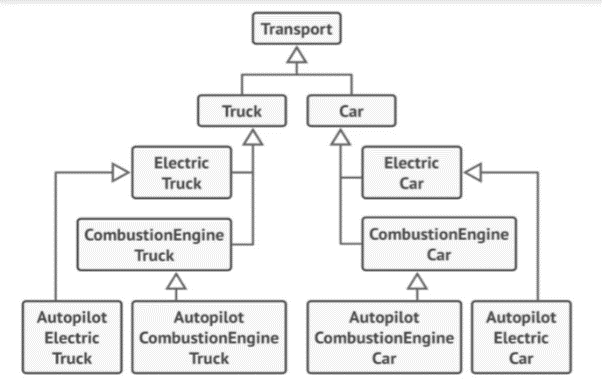
هعمل design system for transportation .

* ف الاول عملت 3 class الى هم transport وcar ,track وال car ,track ورثين من transport



* بعد شويه جه قلك هنزود حاجة عندى نوعين من انواع العربيات غير ال car ,track الى هم عربه بتمشى بالبنزين ولا بالكهربه . ف لو عاوز اكمل ال design ف كدة هعوز اعمل نسختين من كلاس car ونسختين من كلاس track نسخه تمشى بالبنزين ونسخه بالكهربه .



* جه واحد تانى وقلك استنى ده احنا عندنا diminution تالت الى هو عربيه autopilot ف هضيف النوع دة على التصميم بتاعك . ف دة هيكون الشكل .
* مفروض تكون عرفت المشكله لو

استمريت بالشكل دة مش هتخلص ودة

شرح ال parallel inheritance .

و دة مثال تانى سريعا على ال parallel inheritance : عوزين نعمل represent للطلبه ف الكليات

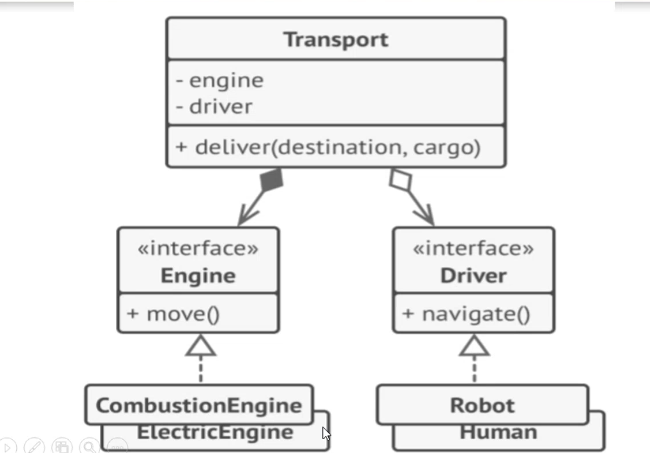
* ف هتعمل student class .
* وجيت بعد كدة وقلتلك استنى عوزين نقسم الطلبه دول لزكر و انثى ف لو انت عاوز تطبق ال inheritance ف هتورح تعمل لكل نوع جديد "زكر وانثى " class وتخليه يعم inherit من ال student class
* جيت بعد كدة قلتلك استنى عوزين كمان تقسم student حسب الكليات وعندنا 3 كليات مثلا .

فكدة هتعمل 3 class جداد وتخليهم يعمله inherit من mal class وتعمل 3class وتخليهم يعمله inherit من female class .

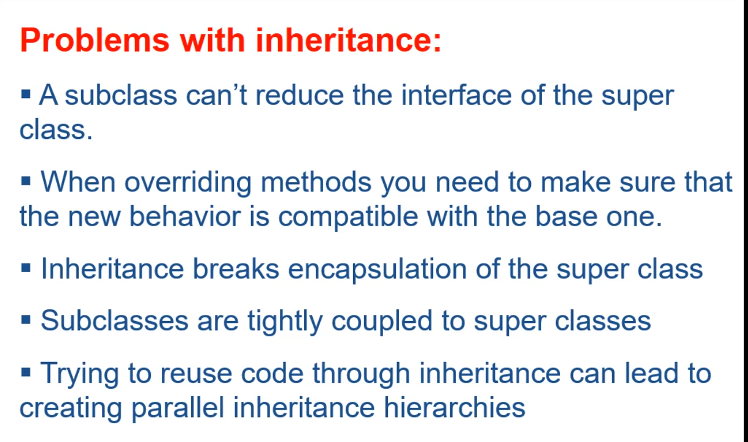
* جبت بعد كدة وقلتك انا عاوز افرق بين الطلبه الى ف اسيوط ولا بره اسيوط . ف كدة هضيف 2class لكل لل 6 class الى ظهره عندى .
* ف مش هتخلص **. طيب الحل ؟؟بدال م انا معتمد على inheritance هنعتمد على composition**

**الحل دة هطبقه على مثال ال transport**

* وملحوظه  **لو قلتلى ان مثلا كل منوع جديدى يضاف لل requirement هفتح الكلاسstudent واضيفه فيه اقلك كلمه وحشه والله احنا من الصبح بنقول ان عوزين منعدلش ف اى حاجة موجودة ف ركذ كدة** معايه .
* الغلط الى عمناه المرة الى فاتت ان خليناcar, track class يورث من transport class . ف هنحل الموضع دلوقتى ب ال composition class بدال ال inherit .
* كل الى هعملته ف التصميم دة ان عملت interface وخليت كل انواع العربات تورث منه وكمان بدال ال inherit خليت جوة ال transport class يكون فيه composition class من ال من انواع المركبات الى عندى بس ال reference بتاع انواع العربات من نوع ال interface .
* لو مستغرب النص اليمين من الاسكرين الى جايه هقلك اول حاجة مش مهمه تانى حاجة هو هنا عامل حاجة اضافيه عن الفرضيات الى كنا بنحلها ف ال transport الى هى عامل فرض لسائق المركبه بواسطه الانسان ولا robot .

.

## 6.1- - favor composition over inheritance in English :

دول يسيدى نفسه المشكال الى ف ال inherit بس بالانجليزى .

# 7 - solid principle :

دة هنشرحه المرة الجايه . ومن اهم ال principle الى مفروض تتاكدان على الاقل مجموعه منهم متوافره ف ال app بتاعك .