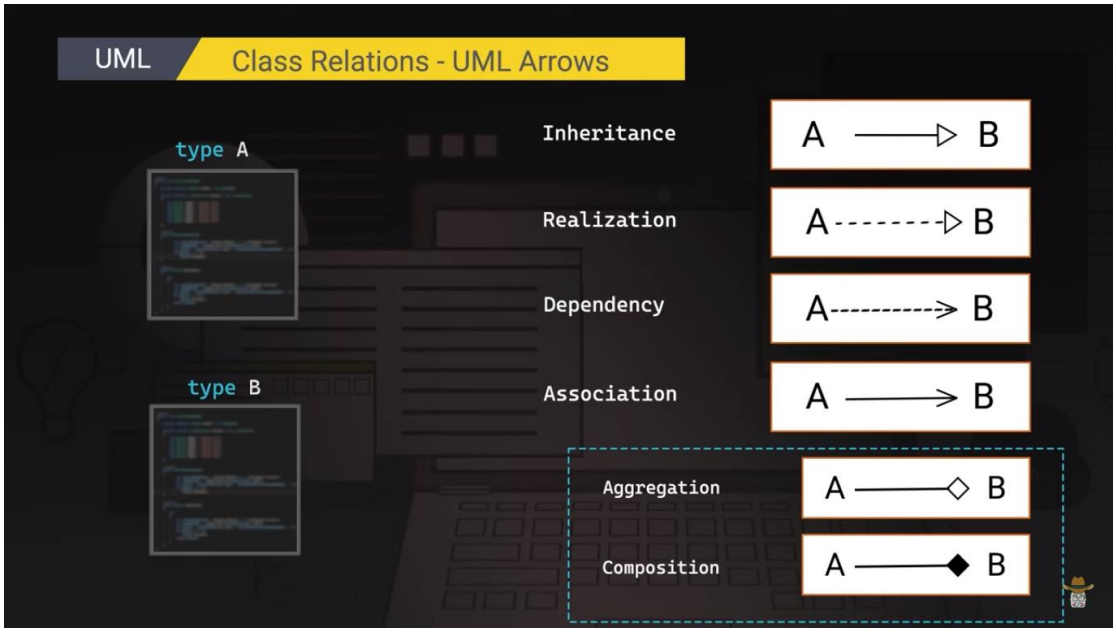


إذن نبدأ في مرحلة التصميم باكتشاف العلاقات مابين ال classes، فما هي هذه العلاقات ؟

إن العلاقات المستخدمة في ال class Diagram على نوعين :

1) علاقات تربط بين الصفوف مباشرة .

2) علاقات تربط بين ال objects من هذه الصفوف .

و سنتحدث عن هذه العلاقات بالتفصيل :

سنبدأ أولاً بالعلاقات التي **تربط بين الصفوف** و هي على نوعين :

1- الوراثة (generalization) .

2- التحقيق (realization) .

الوراثة

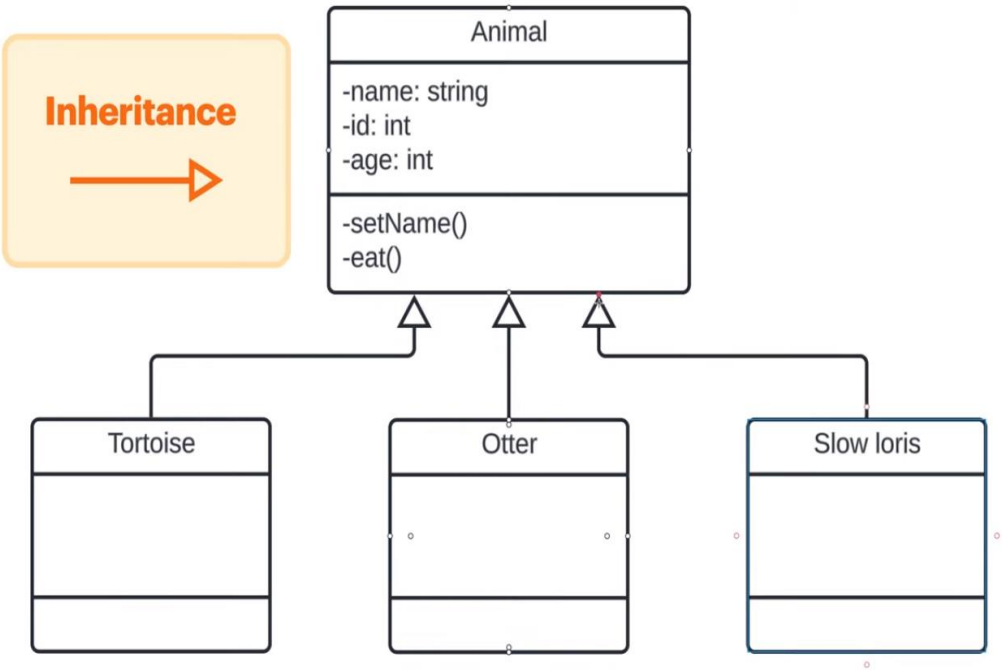
Animal
-name: string -id: int -age: int
-setName() -eat()

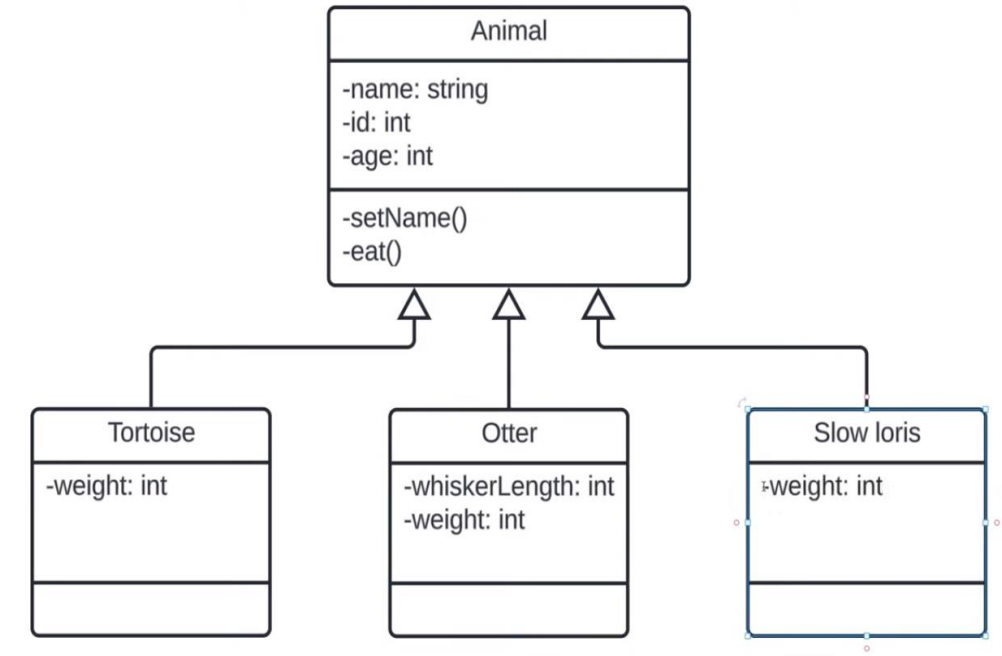
Animal
-name: string -id: int -age: int
-setName() -eat()

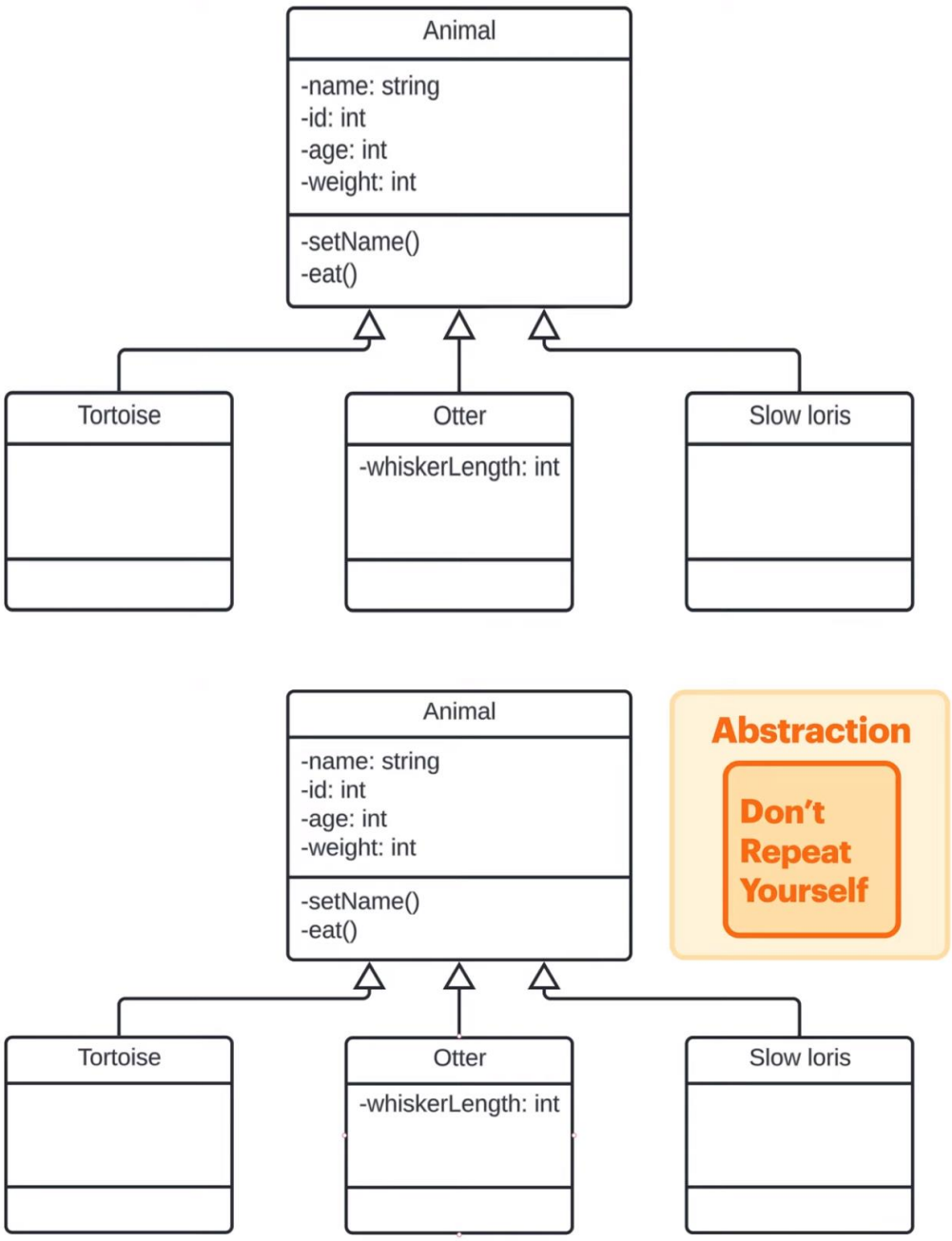
Tortoise
-name: string -id: int -age: int

Otter
-name: string -id: int -age: int

Slow loris
-name: string -id: int -age: int


Inheritance
Animal
-name: string
-id: int
-age: int
-setName()
-eat()
Tortoise
Otter
Slow loris


Animal
-name: string
-id: int
-age: int
-setName()
-eat()
Tortoise
-weight: int
Otter
-whiskerLength: int
-weight: int
Slow loris
-weight: int

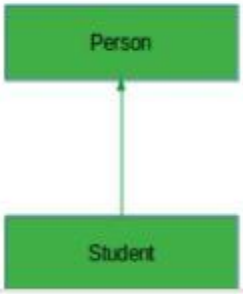

Animal
-name: string
-id: int
-age: int
-weight: int
-setName()
-eat()
Tortoise
Otter
-whiskerLength: int
Slow loris
Animal
-name: string
-id: int
-age: int
-weight: int
-setName()
-eat()
Abstraction
Don't Repeat Yourself
Tortoise
Otter
-whiskerLength: int
Slow loris

1. Inheritance (or Generalization): التعميم

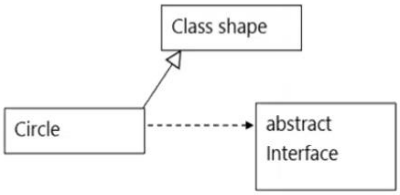
A generalization is a taxonomic relationship between a more general classifier and a more specific classifier. Each instance of the specific classifier is also an indirect instance of the general classifier. Thus, the specific classifier inherits the features of the more general classifier.

- Represents an "is-a" relationship.
- An abstract class name is shown in italics.
- SubClass1 and SubClass2 are specializations of SuperClass.

يساعد التعميم على توصيل صنف فرعي (subclass) بصنفه الأعلى(superclass) ، يستمد الصنف الفرعي متغيراته (يرث) من الصنف الأعلى، لكن لا يمكن استخدام علاقة التعميم هذه لنمذجة تنفيذ الواجهة (interface implementation)، ومن الجدير بالذكر أيضاً أن مخطط الصنف يسمح بإستمداد المتغيرات (التوريث) من عدة أصناف أعلى. في هذا المثال، يتم تعميم الطالب في الفصل الدراسي من صنف الشخص.



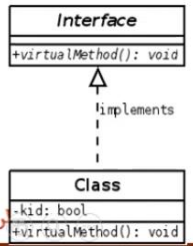
ثانياً : التحقيق (realization) :



لدينا بعض لغات البرمجة غرضية التوجه تمنع ال Multi Inheritance و لذلك إذا أردنا أن نجعل class ما يرى أكثر من صف ماذا علينا أن نفعل ؟؟

نقوم ببساطة بتعريف Interface و تكون abstract أي نعرف لها سلوك معين لكن لا نستطيع أن ننشئ غرضاً من تلك ال Interface أي لانستطيع أن نقوم بعملية تحقيق لهذه ال Interface ، و نجعل ال class الذي نريده أن يرث من أكثر من صف أن يرثَ من هذه ال Interface .

و بالتالي class circle قام بالوراثة من shape و بتحقيق ل interface معينة ، و كما نلاحظ تمثل علاقة التحقيق بسهم منقط من الاين إلى ال Interface.



هنالك علاقة مختلفة في UMLللواجهات ((interfaces، فالوراثة من واجهة تسمى "implementation" "التي هي علاقة "تطبيق" ((realizationفي مخططات UML. تمثيلها الشكلي مشابه للوراثة، إلا أنَّ الخط مقطّع ((dashed، ويجب تحديد أنَّ الواجهة هي «مجرَّدة» (– (abstractأي أنَّ اسمها مكتوبٌ بخطٍ مائل؛ كما هو مبيّن في هذا الرسم.

تدعم بعض لغات البرمجة مبدأ الوراثة المتعددة، أي إمكانية وراثة فئة ما لأكثر من فئة أخرى. في C # لا يمكن للفئة أن ترث إلا من فئة واحدة..
وبالتالي للوراثة من أكثر من فئة فنستخدم مفهوم الواجهات.
يشابه مفهوم الواجهات بشكل كبير مفهوم الفئات المجردة، والتي لا يمكن استساخ كائنات منها، بالإضافة لعدم إمكانية كتابة أي كود عملي داخل الواجهات وإنما فقط الوظائف والخصائص التي ستستعمل هذه الواجه

يمكن لفئة واحدة أن ترث من أكثر من واجهة، وهو الأمر غير الممكن بالنسبة للوراثة من الفئات، وتسمى عملية الوراثة من واجهة بـ Implementation.

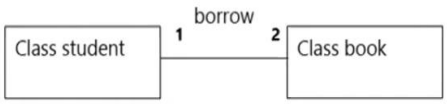
```
public interface ICar
{
  int carMaxSpeed { get; set; }
  void AddItem(string Item);
}

public class BMW : ICar
{
}
public class BMW2 : ICar, ITruck
{
}
```

المهندس : محمد الرسلان

علاقة التشارك أو الارتباط أو الاقتران Association

علاقة ال association لها 3 عناصر :

1) اسم العلاقة : يكتب فوق الخط الواصل ما بين ال classes .
2) لها تعددية : نلاحظ في المثال السابق وجود (n و 1) على طرفي الخط الواصل و هذا يعني أن ال customer يطلب n طلب أما الطلب يعود ل customer وحيد .
3) لها أدوار .



لاحظ اسم العلاقة (borrow) (استعارة) .

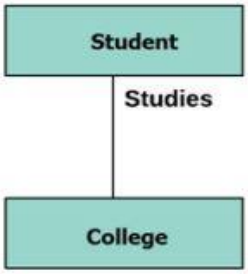
و كذلك التعددية هي (1,2) على طرفي العلاقة و تعني أن الطالب يحق له استعارة كتابين أما الكتاب يكون في الوقت الحالي مُستعار من قبل طالب واحد .

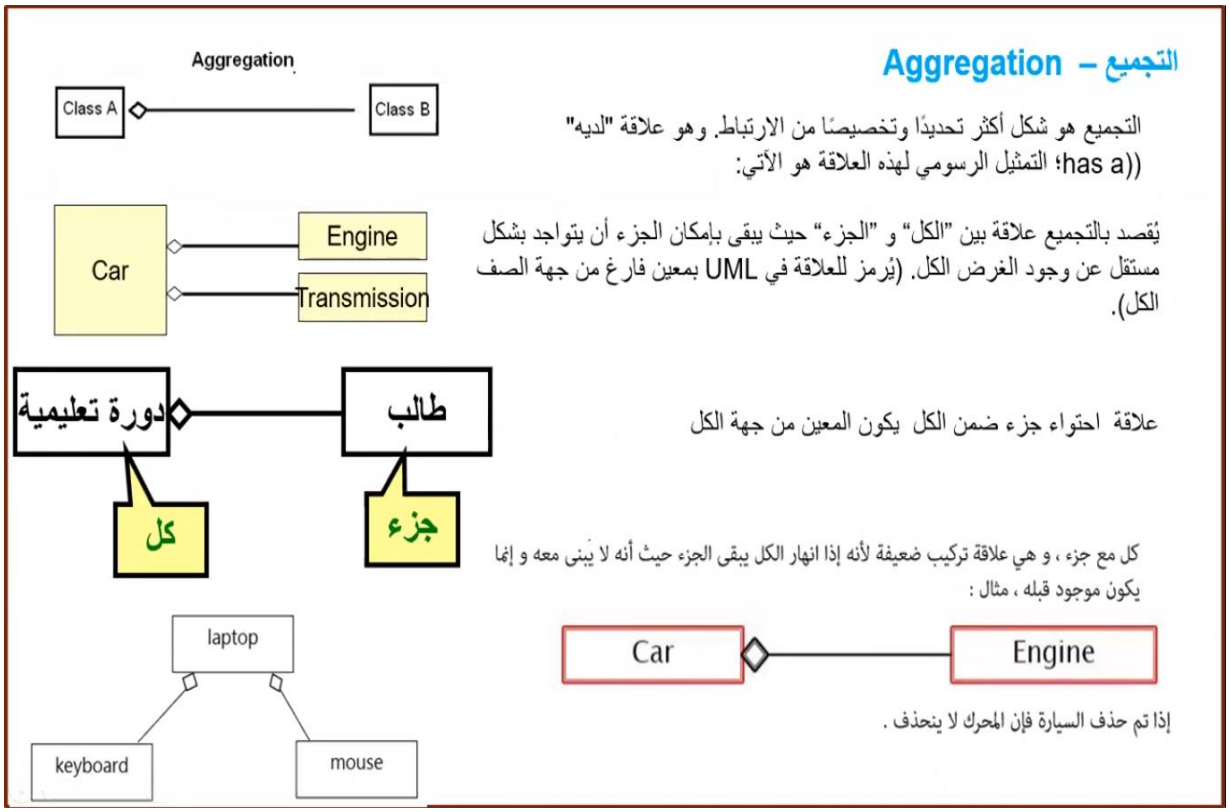
المهندس : محمد الرسلان

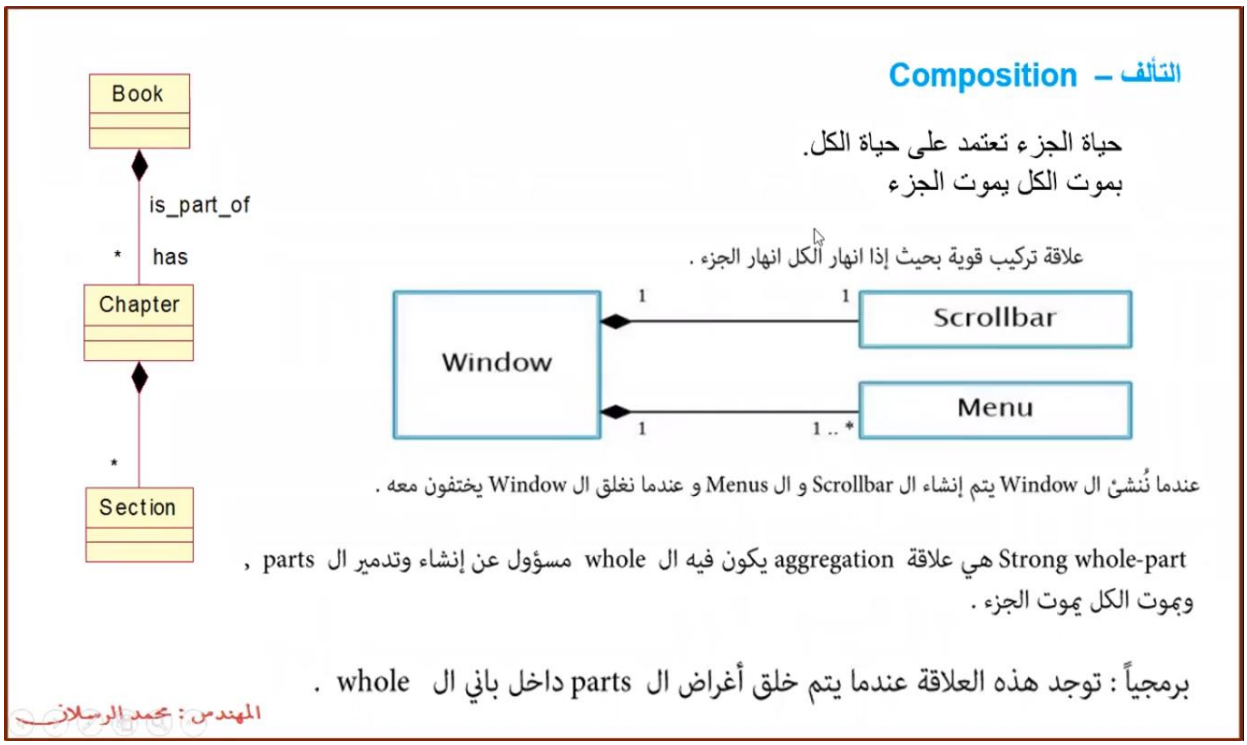
2. Association: الارتباط

Associations are relationships between classes in a UML Class Diagram. They are represented by a solid line between classes. Associations are typically named using a verb or verb phrase which reflects the real world problem domain.

يمثل هذا النوع من العلاقات علاقات ثابتة بين الفئتين "أ" و "ب". يعني مثلا موظف يعمل في منظمة .في هذا المثال تظهر العلاقة بين الطالب والكلية وهي الدراسات .





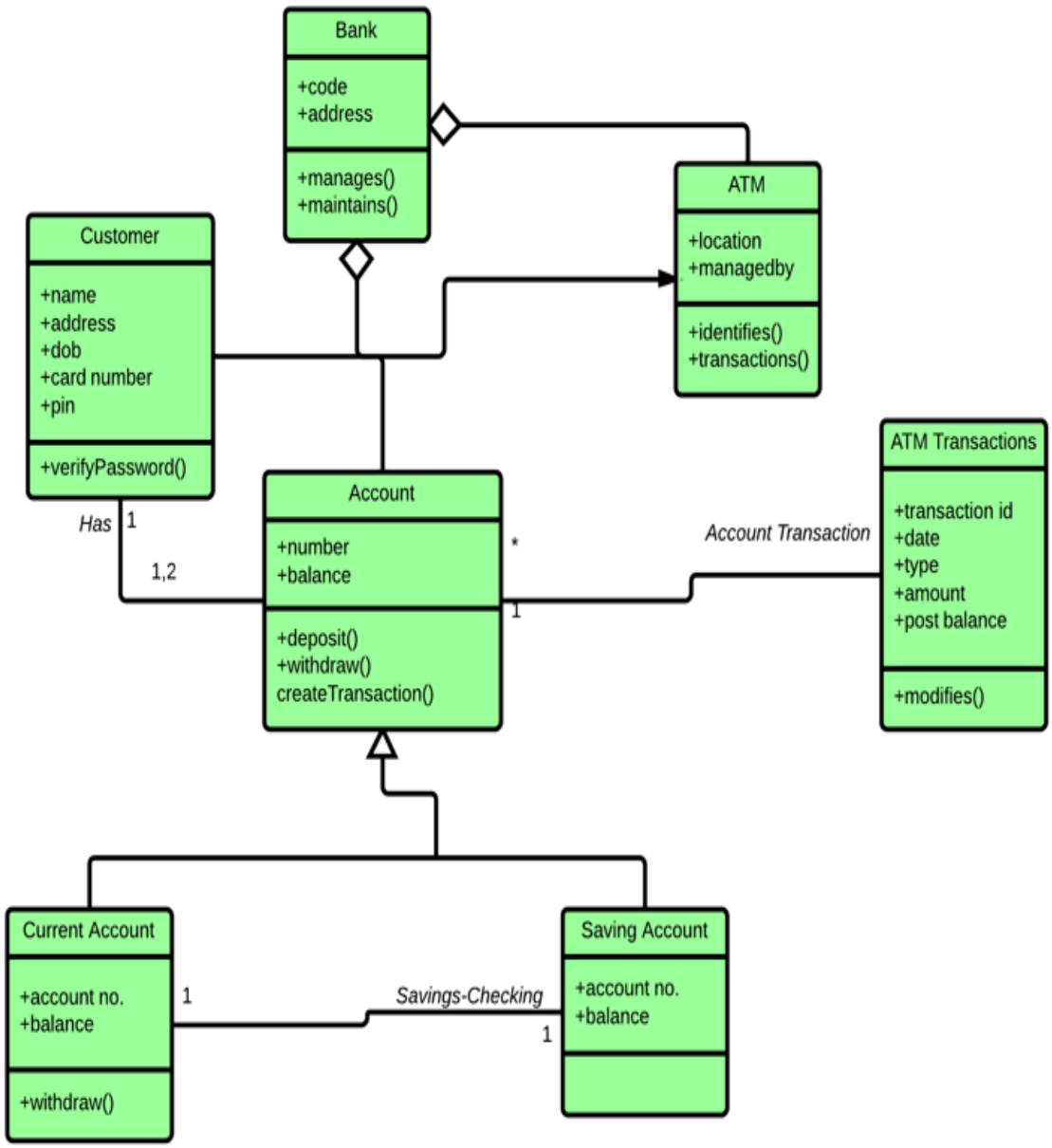


مقارنة بين التجميع و التكوين

التجميع	التكوين
يشير التجميع إلى علاقة يمكن أن يكون الصنف الطفل فيها موجودًا بشكل منفصل عن الصنف الوالد. مثال: السيارات (الأصل) والسيارة (الطفل)، لذا إذا قمت بحذف السيارات، فإن السيارة الفرعية لا تزال موجودة.	علاقة تركيب ثابت حيث لن يكون الطفل موجودًا بشكل مستقل عن الوالد. مثال: المنزل (الوالد) والغرفة (الطفل)، لذلك نحن نعلم أنه لن تنفصل الغرف عن المنزل أبدًا.

نظام الصراف الآلي نظام بسيط للغاية حيث يحتاج العملاء فيه إلى الضغط على بعض الأزرار للحصول على أموالهم لكن بنفس الوقت هناك طبقات أمان متعددة يحتاج أي نظام ATM لتمريرها مما يساعد على منع الاحتيال وتقديم المال أو التفاصيل لعملاء الخدمات المصرفية عندما يحتاجون إليها.

فيما يلي مثال على مخطط UML Class:


Bank
+code
+address
+manages()
+maintains()
ATM
+location
+managedby
+identifies()
+transactions()
Customer
+name
+address
+dob
+card number
+pin
+verifyPassword()
Has
1
1,2
Account
+number
+balance
+deposit()
+withdraw()
createTransaction()
*
1
Account Transaction
ATM Transactions
+transaction id
+date
+type
+amount
+post balance
+modifies()
Current Account
+account no.
+balance
+withdraw()
1
Savings-Checking
1
Saving Account
+account no.
+balance