* مادة first :

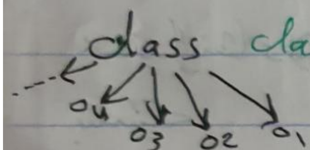
object + class

Class : مجرد تعريف + يحتوي على Attributes و functions

object : محجوز مكان في الذاكرة .

* شرح object :

* object هو نسخة من class : يتم اشتقاقها من نفس class



* مثال: أجزاء البيت، أساس الجدول هو class (مجرد فاضي).

* لما بدي أعمل object بأخذ نسخة من class .

ال class ينشأ مرة واحدة أما ال objects يمكن إنشاء عدد لا نهائي منه .

نسخة من ال class يتم اشتقاقه من نفس ال class موصول إلى المتغيرات و ال functions الموجودة داخل ال class لإعادة استخدامها .

* عندي class اسمه car ⇐ .class يعني أنشئ object من هذا ال class بأخذ نسخة من الجدول وبعبيه .

Class

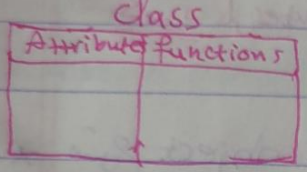
functions	Attributes

ما يربط بين object و class .

عند إنشاء objects بأخذ نسخة من الجدول وبعبي .

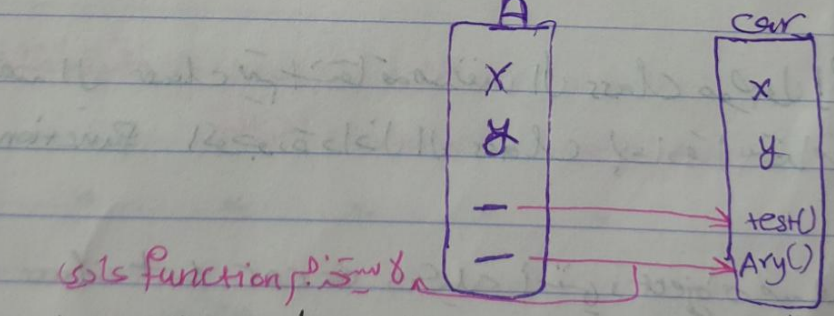
Object:

- هو عبارة عن نسخة عن class.
- عند إنشاء object يقرأ وصل لـ Attributes وال functions الموجودة داخل class.
- الـ class عبارة عن جدول فارغ يوجد فيه Attributes و functions دون قيم.



* عند إنشاء objects يتم أخذ نسخة من هذا الـ class الفارغ أو الجدول ثم أقوم بتعبئته
* الكلاس ينشأ مرة واحدة
* الأوبجكت أكثر من مرة يحكم إنشاؤه.

عندي Car ، object من A



Constructor: ميثود خاصة بيستدعى تلقائياً وقت إنشاء ال object

هو function له نفس اسم ال class، يتم استدعاؤه تلقائياً عند إنشاء ال object.

* يستخدم لتهيئة المتغيرات وإعطاء قيم أولية واستدعاء function التي نريد تنفيذها مع بدء البرنامج
* يمكن بناؤه مع Attribute و functions وبدونهم.

* لا يشترط على أنه boolean , char , string , int.
* لا يقبل نوع ولا يحدد قيمة، لأنه مهمته فقط العمل عند إنشاء ال object لإعطائه قيم أولية

* ممكن أنه يكون أكثر من Constructor لكن بشرط:
- أنه يختلف كل Constructor عن الثاني في عدد البارامتير أو في ترتيبهم.

* عندما لا يكون داخل الكود أي Constructor وقت بإنشاء object فإن البرنامج تقوم بعمل Constructor افتراضي.

* * * * * * * * * *

* this:

to call a constructor by another constructor.

* تأتي أول جملة داخل constructor.
* لا يجوز مناداة constructor داخل method حتى لو استخدمنا this.
* لا تأتي this إلا داخل constructor.
* لا يمكن أن تكون المناداة Circular.

Array:

نوعين من Array :

* Array of object

class اسم ال + اسم الobject = new + اسم الclass + [الحجم]

Array of method

public static اسم class [ ] اسم method ()

{

}

* second يلي جاي :

Static attributes + Static methods

* Static attributes :-

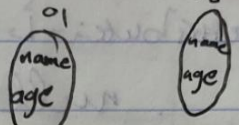
عندي class ، هذا ال class موجود فيه Attribute عادي.

Attribute يعني : int, char, String, float, boolean

هذا Attribute لجميع objects ، 2 objects : O_1 and O_2 :-

age

name



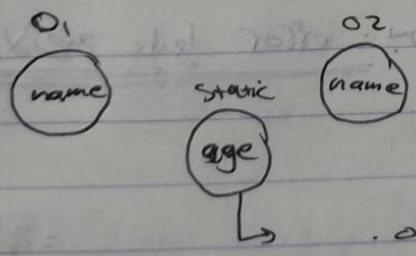
* في هذه الحالة عندي 2 objects وعندي 4 Attributes.

في هذه الحالة ال Attributes مش Static.

- عشان أعرف Attributes على أنها Static :

نفس class في name ، static age

غالباً object هي age ، فإنه يتم تعريفها مرة واحدة فقط و نقوم باستدعائها لكل objects ، مشتركة لكل ال objects.



مشتركة مع جميع objects.

Static attributes

- لا تكون موجودة داخل ال object.
- تعرف مرة واحدة فقط.

*ملاحظات :

- هذا ال Attribute لا يتبع لأي object. ويشمل new object ومش
  رح يكون ال Attribute يلي مستوى static موجود داخل ال object new

عند عمل run ، يأخذ static attribute قيمة بدائية (default)
يا Zero يا null.

ممنوع داخل ال object تكون موجودة ، لازم تكون خارجها لأنها مشتركة.
يتم إنشاء static attributes عند إنشاء أول object

يتم عمل ال acsses (Attributes , methods) static عن طريق اسم
class

* إذا في الامتحان أعطاني إياه مناديه عليها من اسم ال object بغيره وبحط اسم class.
لكن لا اضع عليها error : بس بغير اسم ال object الى اسم ال class .

*ملاحظة :
لازم داخل static method تكون Attributes من نوع static .
لكن ال method يلي مش static عادي يكون (وما يكون) داخلها static attributes .

Error : private with static
* الاستدعاء يتم بواسطة اسم ال class .

* تلخيص : Static attributes

- يتم إنشاؤها مرة واحدة عند أول object .
- تكون مشتركة لجميع الـ objects .
- يتم مناداتها باسم الـ class وليس اسم الـ object .
- Static method لازم يكون جواها static attribute .
- إذا كان عندي static method وفي جواها method بدي أناديها لازم الـ method اللي جوا تكون من نوع static .

* الـ static يتم تنفيذها قبل main *

- يمكن أن تتواجد static attribute داخل method عادي (يعني مش static).

* إذا تشابه attribute مع local variable نستخدم this كلمة attribute الـ *

attribute + method = member

* Inheritance: *الوراثة:

* يمكن لـ class أن يرث صفات من class آخر.
الأب ، الابن

* يمكن الابن الوصول إلى كافة الـ attributes والـ methods الموجودة عند الأب ما عدا الـ attributes التي من نوع private.

وظيفة الـ Inheritance هي اختصار الكود.

Public class a extends A
الابن ↓ الأب ↓

الأب: super class.
الابن: Sub class.

عادةً: لما أعمل object من الابن وأنادي على method، يتم الإشي كالتالي: (هاظ بنادي على هاظ)
- بروح بدور على الـ method داخل الابن ← موجودة بستخدمها.
- غير موجودة بروح بدور عليها عند الأب ← موجودة بستخدمها.
- غير موجودة هيك بعطي Error.

لكن إذا كانت الـ method موجودة عند الابن والأب يبدأ التنفيذ من الابن.

✗ Static method can be Inheritance ✗

* Override + overload:

{over riding}

- إمكانية تسمية أكثر من method بنفس الاسم بحيث نستطيع أن نكتب 2 methods عند الأب وعند الابن بنفس الاسم ونفس ال parameter ونفس النوع لكن الكود مابين ال paracies يكون مختلف.

* static method يمكن أن تكون Inheritance لكن مستحيل *

* overriding أن تكون *

يستطيع الأب الوصول إلى هذه ال methods الموجودة عند الابن بحيث تكون بنفس الاسم.

{over loading}

- هو method موجود عند الأب والابن، يكون نفس الاسم لكن يختلف في عدد ال parameters أو نوعها أو ترتيبها.

* * * * * * * * * * * *

Super:

هي كلمة محجوزة في جافا تشير إلى ال class الأب ومتغيرات ودوال ال class الأب.

* استخدامات كلمة super:

- استدعاء متغيرات ال class الأب.
- استدعاء ال method ال class الأب بطريقة فورية.
- استدعاء دوال البناء (constructors) في ال class الأب.

* لا تأتي super أو this في نفس ال constructor.

- عند إنشاء object من الابن يجب أن يمر في ال constructor الخاص بالأب وال constructor الخاص بالابن.

* Equals

*تقارن address مع address .

- تعيد True إذا كان ال 2 addresses يشيرن لنفس ال object.

- لاتقوم بمقارنة محتويات ال object.

Student $S_1 = S_2$;

النوع	نفس ال class	نفس ال Package	الأبناء	package مختلفة
public	✓	✓	✓	✓
protect	✓	✓	✓	X
defult	✓	✓	X	X
private	✓	X	X	X

الأبناء من باكيج ثانية

- حتى أعمل object من package ثانية لازم أعمل import

* Polymorphism :

- هو pointer من نوع الأب يؤشّر على object من نوع الابن .
- يمكن وصول الأب إلى ال attributes وال functions الموجودة عند الابن ، بشرط أن تكون ال functions عليها overriding .
- يعني يكون اسم ال functions عند الابن نفس اسم ال functions عند الأب .

علّل :

* عند إنشاء object لا يستطيع الوصول إلى ال attributes أو ال ~~functions~~ methods الموجودة عند الأب ؟؟

- لأنّه تكون هذه ال attributes وال methods موجودة عند الابن وغير موجودة عند الأب ، لذلك لا يستطيع الوصول إليها .

* كيف تعمل ال polymorphism ؟

- نجعل object من الأب ونخلّيه تأشر على ال class الابن .

```
public class a extends A
{
}
A ab= new a();
```

طريقة أسهل (من نوع upcasting).

* Casting : *التحويل

up casting + down casting :

Up Casting : polymorphism

- الابن ينادي على أبوه .

الأب ____ = new الابن ();

Nidal A = new Malak();

* هي التحويل إلى supertype .
* هي دائماً allowed .
* تستطيع عل Upcasting إذا كان هناك أي relationship بين 2 classes .
* من الصغير للكبير .

to call super class's method :

⇒ super. method () / performing upcast.

Down Casting :

- الأب ينادي على ابنه

الابن ____ = (الابن) new الأب ;

Malak A = (Malak) new Nidal;

أي أن object من الأب

* هي التحويل إلى subtype .
* تخضع check قبل التحويل
* فيها Cast Exeption .

to call Sub class's method :

⇒ do down casting.

* Instance of:

- بتيجي أفحص إذا هذا الـobject ابن هذا الـ class.

if (object instance of class)

* * * * * * * * *

* Abstract: التجريد

- شكلعزي أي class فيه constructor , attribute.
- جمنوع أنشيء أي object منه.
- يكون super class.

* Abstract method:

- لايكون لها جسم (No body).
- موجودة داخل abstract.

public abstract Maluk ();

- يترك تنفيذها أو كتابتها في الأبناء اللي من الـ abstract class.

لازم الـ abstract method تكون في الـ abstract class.

- إذا كان عندي abstract method لازم يكون عندي أبناء.

- هي كلمة محجوزة في Java تعني أنه الـ class أو الـ method اللي تم تعريفها على أنها abstract تكون محجوزة.
- يستخدم فقط للتصريح عن الدوال بدون بناء كود داخل هذه الدالة.
- أي class يرث من الـ abstract class يجب عليه إلزاماً تنفيذ كافة الدوال التي تم الإعلان عنها في الـ class المجرد.
- الفائدة منه هي إيجاد نمط موحد.
- يجبر المبرمجان الوارثة على تنفيذ دوال بشكل إجباري.

* Interface:

- هي طريقة أو أسلوب لربط ال objects مع بعضها البعض.

- كيفية إنشاء ال Interface :

1. بعمل class عادي بس بغيّر كلمة class ل Interface

```
Interface A
{
}
```

- الفرق بين ال class العادي وال Interface class :

① كل ال functions اللي داخل ال Interface ما إلها body.

- دايماً ال function بكون public حتى لو ما حطيت كلمة public.

```
public void Malak ( );
```

② المتغيرات اللي فيها دايماً final حتى لو ما حطيت كلمة final.

* طب سؤال قوي ع قولة حلا... مين بعمل الميثود هاي ؟

class ثاني بكون على الشكل :

```
public class Malak implement Hala
{ }
```

بس بشرط، لازم هاد ال class يحوي جميع ال method يلي جوا ال Interface

يعني ؟؟

معمولهم overriding

- ما بقدر أعمل منه object بس بقدر أعمل polymorphism.

* الوراثة المتعددة :

```
public class Malak implement Hala, Suzan
  {
  }
```

بتوخذ ال method اللي عن Hala ، Suzan .

- يمكننا تعريف ال method على أنها final .
- عندما تكون ال method = final .

- ممنوع أعملوا override

final class ⇒ Super ممنوع يكون ⇐ ممنوع أعمل له وراثه

Attribute final ⇒ ممنوع تغير قيمتها