♥ Java ♥

المحاضرة التاسعة عشر — الأحد 10/12/2017

♡ الوراثة ♡

الوراثة: هي الطريقة الرئيسية لتعديل الصفوف. تستخدم الكلمة المحجوزة extend extends للتعبير عن وراثة صف لصف آخر.

نسمي الصف A الذي يرث الصف B بالصف الوارث أو الابن أو المشتق ويدعى الصف B الموروث أو الأب أو القاعدة.

الشكل العام: class A extends B {

هنا نقوم بتعريف الصف A كوارث للصف B

«إن الصف B موجود مسبقاً في نفس الملف»

فهنا نعرف كل ما هو جديد بالنسبة لـ B «كل ما هو زيادة أو تعديل على B»

}

ماذا يعني أن الصف A يرث الصف B؟؟

1. هذا يعني أن A هو B (B ليس A) الوارث هو مورث ولكن المورث ليس وارث.

كل ما ينطبق على B يمكن تطبيقه على A ولكن العكس غير صحيح فليس كل شيء معرف على A يمكن استخدامه على B.

2. تنتقل إلى الصف الوارث A جميع مكونات الصف B العامة (public) والمحمية (protected) «تلقائياً ولا يظهر ذلك في الكود»

أما مكونات الصف B الخاصة فإنها لا تنتقل بعملية الوراثة.

ملاحظة: عند كتابة A نحن نقوم بتعريف صف أي لا نفقد شيء من ميزات الصف ولكن هناك علاقة تربطه مع B كأب له أي موجود داخل A مكونات من B قد ورثها يستطيع استخدامها.

3. كل غرض من الصف A يحوي ضمنياً غرض من الصف B:

حتى يمكننا انشاء غرض من A يتوجب ضمنياً في الانشاء انشاء غرض من B الصف الأب (كل object من الصف A يجب أن يكون داخله object من الصف B ضمنياً)

أي:

باي الابن لازم يستنى باي الأب

Scanned by CamScanner

مثال: ليكن لدينا الصف B التالي:

```
class B{
 public int x;
  protected float y;
   private char c;
 public B(){x=0; y=0; c='B';}
  public B(int x, float y){
     this.x = x;
     this.y = y;
       c='B';}
   public void print(){
   System.out.print(x+y+c);
                        }
    public char get_c(){
          return c;}
            }
```

ولنعرف الصف A كأبن للصف B:

```
class A extends B{
   int 3;
    char c;
   A(){3=0; c='A';}
    A(int 3){this.3=3; c='A';}
    void print(){
    System.out.println(x+y+3+c);
    System.out.print("My father is" +get_c());
                }
             }
```

Scanned by CamScanner

ملاحظات: * عند تصميم الصف لا نعلم إن كان سيكون مورث (أب) أم لا لذلك نضع محدد وصول لكل مكون

* إن باني الابن يجب أن يستدعي باني الأب ويجب أن يكون استدعاء باني الأب هو أول تعليمة في باني الابن

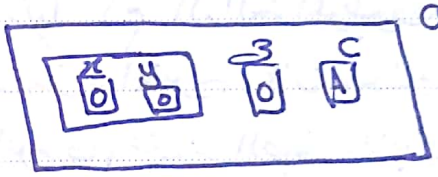
* في حال لم نذكر ضمن باني الابن أي باني قمنا باستدعائه للصف الأب فإن اللغة افتراضياً تقوم باستدعاء الباني الخالي (الافتراضي إن لم يكن هناك أي باني) الموجود في الأب

* في حال لم نذكر باني الأب المستدعى ولم يكن هناك باني خالي في الأب فإن هذا سيولد خطأ

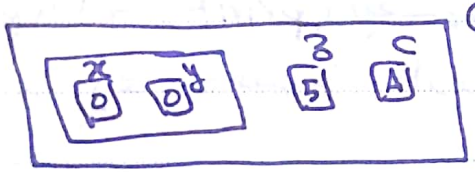
بالعودة للمثال: داخل الـ main :

```
A a1 = new A();
A a2 = new A(5);
```

a1



a2



ملاحظة: إذا كان الباني الخالي غير موجود في الصف B فعند كتابة الباني في الصف الابن دون استدعاء الصف الأب فإنه يعطينا الـ compiler خطأ. وبالتالي يجب إلزاماً استدعاء باني الأب داخل باني الابن كأول تعليمة.

* إذا لم يتوفر في الصف الأب باني دون وسطاء (خالي أو افتراضي) عندئذٍ يلزم (يتوجب) علينا استدعاء باني الأب صراحة كأول تعليمة في باني الصف الابن أي لا يمكننا الاعتماد على الاستدعاء الافتراضي الذي تقوم به اللغة لأنه لا يوجد شيء للاستدعاء ((استدعاء شيء غير موجود)).

كيفية استدعاء باني الأب داخل باني الابن:

يتم استدعاء باني الأب داخل باني الابن عن طريق استخدام الكلمة super إن super اسم للأب ضمن الابن وهي تفيد في استدعاء متغيرات ودوال الصف الأب

بالعودة للمثال: استدعاء الباني الخالي للأب ضمن بواني الابن:

```
A(int z){ super(0,0);
          this.z = z;
          c = 'A'; }
```

```
A(){ super(0,0);
     z = 0;
     c = 'A'; }
```

Scanned by CamScanner

استدعاء باني الأب الفيرضالي ضمن باني الابن

```
A(int x, float y, int z){ super(x,y);
                          this.z=z;
                          c='A';}
```

هل يمكننا انشاء غرض من الصف B؟:

طبعاً يمكننا ذلك لأن B هو صف بحد ذاته وبالتالي يمكننا مثلاً إنشاء الغرض:

```
B b = new B(11,15.5);
```

* لو قمنا بكتابة التعليمة التالية ضمن ال main :

```
a1.print();
```

فأي دالة سيتدعى؟ الدالة التي ورثها أم الدالة الموجودة ضمن الصف الابن؟

سيقوم باستدعاء الدالة الموجودة ضمن الصف الابن لأنه عندما قمنا بكتابة نفس اسم الدالة (print) بنفس قائمة الوسطاء ضمن الصف الابن فإن هذه الحالة تسمى إعادة تعريف للدوال «دالة موجودة في الاب وأعيدت كتابتها في الابن»

إعادة تعريف الدوال: هي إمكانية كتابة دالة موجودة في الصف الأب ضمن الصف الابن بحيث يكون لها نفس الاسم ونفس قائمة الوسطاء ولكن تقوم بعمل مشابه للدالة الأب أو مختلف كلياً.

* عند إعادة تعريف دالة موجودة في الأب ضمن الابن واستدعائها عن طريق غرض من الابن فإنه سيتم استدعاء دالة الابن وليس دالة الأب
* عند استدعاء الدالة get_c ضمن الابن فيتم استدعاء دالة الأب لأنها دالة ليست موجودة في الابن موجودة فقط في الاب. (هي عامة وموروثة)

وبالتالي نتيجة الطباعة للتعليمة a1.print(); هي:

```
OA
My Father is B
```

المكونات الخاصة في الصف الأب لا يمكن استدعائها أو استخدامها ضمن الابن أبداً لأن المكونات الخاصة لا يمكن الوصول لها إلا ضمن الصف نفسه وهي مكونات لا تورث.

انتهت المحاضرة

ALAUID.net
Scanned by CamScanner