بعض الكلمات المفتاحية في لغة الجافا

محددات الوصول : Access Modifiers

1) **public** : محدد الوصول العام.
- إذا ظهر في بداية التصريح عن حقل أو طريقة فإنه يجعلها عامين، أي يمكن الوصول إليهما من أي صف آخر.
- وإن كتابة الكلمة المفتاحية public قبل التصريح عن صف ما أو طريقة يدل على أن الصف (أو الطريقة) قد أصبح عاماً بحد ذاته، ولا يعني ذلك أن كل مكوناته عامة.

2) **private** : محدد الوصول الخاص.
- إذا ظهر في بداية عبارة التصريح عن حقل أو طريقة فإنه يجعلها خاصين، ولا يمكن الوصول إليهما من خارج الصف الذي تم التصريح عنهما فيه.
- إن كتابة دالة خاصة private في صف ما، يكون الهدف منه عادة استدعاء هذه الدالة فقط ضمن دالة أخرى موجودة في نفس الصف.
- وكذلك فإن كتابة صف داخلي خاص ضمن صف ما، يكون الهدف منه عادة استخدام هذا الصف فقط ضمن صف داخلي آخر موجود في نفس الصف.
- إن تحديد كل البواني في صف ما من النوع الخاص private يمنعنا من القدرة على اشتقاق كائنات من نوع هذا الصف، وفي هذه الحالة لن تصبح لهذا الصف فائدة، لذا غالباً لا يُحدَّد الوصول إلى البواني إلا على أنها عامة public.

3) **protected** : محدد الوصول المحمي.
- إذا ظهر في بداية التصريح عن حقل أو طريقة فإنه يجعلها محميين، حيث يمكن الوصول إلى العنصر المحمي فقط من الصفوف الأبناء (عند الوراثة) والصفوف المنتمية إلى المكتبة نفسها.

4) **المحدد الافتراضي** : لا توجد له كلمة محجوزة، بل يكفي ألا نضع أياً من المحددات الثلاثة السابقة.
- وهو يتعامل مع ما يسمى بالحزمة package، إذ تجعل الحقول والطرق والصفوف عامة بالنسبة للصفوف المنتمية إلى نفس الحزمة، ولكنها ستكون خاصة بالنسبة للصفوف المنتمية إلى حزم أخرى.

Scanned by CamScanner

مثال: ليكن لدينا الصف التالي:

```
public class Complex {
    public double x;
    private double y;
    public Complex() {}
}
```

وليكن لدينا البرنامج التالي:

```
class MainComplex {
    public static void main (String args[])
    { Complex c;
      c = new Complex();
      c.x = 5;
      c.y = 5;
      :
    }}
```

- يمكن كتابة السطر الأول في الدالة الرئيسية والتفرع عن كائن من النوع Complex دون أي مرجع لأن الصف Complex عام.
- كما يمكن إنشاء هذا الكائن عن طريق الباني الخالي لأنه عام أيضاً.
- ويمكن كذلك إسناد قيمة للمعطى العضو c.x بشكل مباشر كما في السطر الثالث.
- ولكن من غير الممكن فعل ذلك مع العضو y لأنه خاص في الصف ولا يمكن الوصول إليه بشكل مباشر بهذا الشكل.

* نتابع في كثير من الأحيان إلى جعل بعض المعطيات الأعضاء خاصة بالصف حتى لا يتسنى لأي آلية كانت بالوصول إليها وتغييرها، ولكن ذلك سيمنعنا من التعديل عليها فيما بعد إذا لم نوجد وسيلة بديلة لذلك.

إن الوسيلة البديلة التي تتيح لنا الوصول إلى المعطيات الخاصة والتعديل عليها بشكل غير مباشر هي إضافة دوال إلى الصف تقوم بذلك بحيث أستطيع الوصول إلى الدالة

Scanned by CamScanner

/ /

بشروط معينة ، وبما أن هذه الدالة داخل الصف نفسه فإنها ستتمكن من الوصول إلى المعطيات الخاصة فيه .

مثلاً : في المثال السابق لا نستطيع الوصول إلى y أو التعديل عليها ، ولكن لو أضفنا الدالتين التاليتين في الصف Complex :

```
public void set_y (double m) { y=m; }
public double get_y () { return y; }
```

عندئذ سيصبح بإمكاننا في أي صف آخر الوصول وتغيير قيمة y بشكل غير مباشر عن طريق هاتين الدالتين العامتين .

وبالتالي حسب المثال يمكننا إعطاء قيمة لـ y في الدالة الرئيسية كما يلي :

```
c.set_y (5);
```

كما يمكننا الحصول على قيمة y متى أردنا كما يلي :

```
double m = c.get_y();
```

وستصبح بشكل غير مباشر قيمة m هي نفسها قيمة y التابعة للكائن c .

الكلمة المفتاحية Static : (ساكن)

رأينا سابقاً أن الوصول إلى حقل في الصف أو استدعاء طريقة يتابع إلى إنشاء كائن من الصف ثم الوصول أو الاستدعاء من خلاله . ولكن في كثير من الأحيان نريد استدعاء طريقة أو الوصول إلى حقل ما دون الحاجة إلى إنشاء كائنات واستهلاك ذاكرة بشكل غير مسوغ .

توفر لغة الجافا الكلمة المفتاحية «static» لتعريف حقول أو طرق على مستوى الصف حيث يصبح هذا الحقل مشتركاً بين كل كائنات هذا الصف ، ويصبح بالإمكان الوصول إليه من خلال اسم الصف دون الحاجة لإنشاء كائن .

مثلاً : إذا كان لدينا الصف التالي :

```
class Titi {
    Static int x;
}
```

/ /

وكتبنا البرنامج:

```
class UseTiti{
  static void main (String args[])
  { Titi t1, t2;
    t1 = new Titi();
    t1.x = 5;
    t2 = new Titi();
    System.out.print(t2.x);
  }
}
```

هنا سيطبع القيمة 5، لأنه من السطر الثالث في الدالة main أصبحت قيمة الحقل x لكل كائنات الصف تساوي القيمة 5.

كما أنه كان من الممكن إعطاء القيمة 5 للحقل x دون إنشاء الكائن t1 وذلك بكتابة:

```
Titi.x = 5;
```

أي باستدعاء x عن طريق اسم الصف مباشرة.

سؤال: لماذا نضع الكلمة المفتاحية static دوماً عند التصريح عن الدالة main؟

إن أمر التنفيذ run هو فعلياً استدعاء للدالة main.

وبما أن الدالة main تُعرَّف داخل صف، فاستدعاؤها في الحالة العادية يتم من خلال إنشاء كائن من هذا الصف.

وبما أنه لا يوجد مكان لإنشاء هذا الكائن كون الدالة main هي نقطة بداية التنفيذ لذلك كان من الضروري أن تكون دالة ستاتيكية static حتى يكون بالإمكان استدعاؤها من خلال اسم الصف مباشرة دون إنشاء كائن.

ALAUIB.net

Scanned by CamScanner

/ /

الكلمة المفتاحية final: (نهائي)

تستخدم هذه الكلمة المفتاحية في ثلاث مواقع:

1) أمام المتحولات: وهنا تعني أن هذا المتحول نهائي لا يمكن تغيير قيمته، ويمكن أن يكون هذا المتحول النهائي من أي نوع.

مثلاً:

```
final double PI = 3.14;
```

وهنا يمكن أيضاً إضافة الكلمة static ليصبح مشتركاً بين كل كائنات الصف باعتبار أن PI هو معطى عضوي صف ما. ويكتب ذلك على الشكل:

```
final static double PI = 3.14;
```

2) أمام الصفوف: وهنا تعني أن هذا الصف نهائي أي لا يمكن تعديله.
وسنرى لاحقاً أن تعديل صف ما يعني وراثته.

3) أمام الطرق: وتعني ذلك أن هذه الطريقة أصبحت نهائية أي لا يمكن تعديلها.
وفي الحقيقة تعديل طريقة ما يعني أنه لا يمكن تطبيق مبدأ التحميل الزائد للطرق عليها عند وراثة الصف الذي يحويها.
وسنتحدث عن ذلك بالتفصيل فيما بعد.