محاضرات العملي لمقرر

البرمجة الشيئية

ITGS211

الدرس العملي رقم (3)

إعداد الأستاذة :ملاك ددش

Classes and objects

مفهوم الكلاس في جافا:

Class: نكتبها كلاس في العربية. و الكلاس عبارة عن حاوية كبيرة تستطيع أن تحتوي على كل الكود من متغيرات و دوال و كائنات إلخ..

لتعريف كلاس جديد يكفي فقط كتابة الكلمة class, ثم وضع إسم له, ثم فتح أقواس تحدد بدايته و نهايته. مثال:

```
class ClassName {

}
```

الآن سنقوم بتعريف كلاس جديد يحتوي على 4متغيرات, بالإضافة إلى دالة تعرض قيم هذه المتغيرات عندما يتم إستدعاءها:

مثال

Person.java

```
class Person {

    String name;
    String gender;
    String job;
    int age;

    void printInfo() {
        System.out.println("Name: " +name);
        System.out.println("Gender: " +gender);
        System.out.println("Job: " +job);
        System.out.println("Age: " +age);
    }

}
```

هنا قمنا بتعريف كلاس إسمه **Person** يحتوي على 4متغيرات بالإضافة إلى دالة تعرض قيم هذه المتغيرات عندما يتم إستدعاءها.

Classes and objects

مفهوم الخصائص:

أي متغيرات يتم تعريفها بداخل كلاس و خارج أي دالة تسمى خصائص (**Attributes**), و هذا يعني أن أي كائن من هذا الكلاس سيكون عنده هذه الخصائص.
تستطيع التعامل مع هذه الخصائص من الكائن مباشرةً, بينما المتغيرات العادية لا يمكنك التعامل معها من الكائن.
المتغيرات التي يتم وضعها كبارامترات أو التي يتم تعريفها بداخل الدوال تسمى متغيرات عادية.

برنامج Netbeans يلون أسماء الخصائص باللون الأخضر, لكي يساعدك في التفريق بين المتغيرات العادية و المتغيرات التي يتم إعتبارها خصائص.

و تذكر أن المتغيرات تسمى خصائص, لأن أي كائن من هذا الكلاس سيملك نسخته الخاصة منها.

مفهوم الكائن في جافا:

object: تعني **كائن** في اللغة العربية. و الكائن عبارة عن نسخة مطابقة لكلاس معين.

بما أن الكائن عبارة عن نسخة من الكلاس, يمكننا القول أنه لا يمكن إنشاء كائن إذا لم يكن هناك كلاس.
إذاً في مفهوم برمجة الكائنات نقوم بإنشاء كلاس معين يسمونه **blue print** أي (النسخة الخام أو النسخة الأصلية) , و بعدها ننشئ نسخة أو أكثر من هذا الكلاس و نفعل بها ما نريد بدون أن نغير محتويات الكلاس الأساسي و هكذا نكون حافظنا على كودات الكلاس الأساسي لأننا نعدل على النسخ و ليس عليه مباشرةً.

بما أن الكائن عبارة عن نسخة من الكلاس. لتعريف كائن من كلاس معين يجب وضع إسم الكلاس ثم وضع إسم للكائن.

مثال

```
Person ahmad = new Person();
```

هنا قمنا بتعريف كائن من الكلاس **Person** إسمه **ahmad**.
إذاً الكائن **ahmad** سيكون عنده نسخة خاصة فيه من خصائص الكلاس **Person**.

ملاحظة: الكود **new Person()** هو الذي يقوم فعلياً بتوليد كائن من الكلاس. و هو يعطي قيم أولية للخصائص الموجودة فيه و ستفهم ذلك لاحقاً.

Classes and objects

سنقوم الآن بكتابة نفس الكود السابق على مرحلتين لتحصل على كائن من الكلاس **Person**.

مثال

```
Person ahmad;          // هنا قلنا أن ahmad سيمثل كائن من الكلاس Person
ahmad = new Person();  // هنا أصبح ahmad يمثل كائن من الكلاس Person و بعدها قمنا بتخزينه في ahmad, بتوليد كائن من الكلاس
```

طريقة التعامل مع الكائنات

- نقوم بإنشاء كائن من الكلاس.
- بعدها نقوم بإدخال قيم لخصائصه, إستدعاء دواله إلخ.

لاستدعاء أي شيء موجود في الكائن الذي أنشأناه

1. نضع إسم الكائن.
2. ثم نقطة.
3. ثم الشيء الذي نريد الوصول إليه (سواء إسم متغير أو دالة).

نصائح عليك إتباعها

- يفضل إنشاء كل كلاس في ملف جافا خاص.
- إبدأ إسم الكلاس دائماً بحرف كبير.
- إبدأ إسم الكائن دائماً بحرف صغير.

علاقة الكائن بالكلاس في جافا:

الكائنات تساعد المبرمج كثيراً, فمثلاً إذا كنت تنوي إنشاء برنامج بسيط لحفظ معلومات أشخاص,

هل ستنشئ كلاس لكل شخص ؟!

Classes and objects

طبعاً لا, بل تنشئ كلاس واحد فقط يمثل شخص, و تضع فيه الأشياء الأساسية التي تريدها أن تكون موجودة عند كل شخص. ثم تنشئ منه كائنات قدر ما شئت, و عندها يصبح كل كائن من هذا الكلاس عبارة عن شخص له معلوماته الخاصة.

مثال: الآن سنقوم بإنشاء الكلاس Person و إنشاء كائنات منه في الكلاس الذي يحتوي على الدالة main().

إنتبه: يجب إنشاء الكلاس Person و الكلاس Main في نفس المجلد (أي نفس الـ package) حتى يعمل الكود بشكل صحيح.

Person.java

```java
public class Person {

    // هنا قمنا بتعريف 4 خصائص
    String name;
    String gender;
    String job;
    int age;

    // هنا قمنا بتعريف دالة تطبع محتوى كل خاصية عندما يتم استدعاءها
    void printInfo() {
        System.out.println("Name: " +name);
        System.out.println("Gender: " +gender);
        System.out.println("Job: " +job);
        System.out.println("Age: " +age);
        System.out.println();
    }

}
```

Classes and objects

```
public class Main {
public static void main(String[] args) {

هنا قمنا بإنشاء كائنات من الكلاس// Person
    Person p1 = new Person();    // الكائن p1 سيمثل محمد
    Person p2 = new Person();    // الكائن p2 سيمثل روز
    Person p3 = new Person();    // الكائن p3 سيمثل أحمد
    Person p4 = new Person();    // الكائن p4 سيمثل ربيع
// p1 هنا قمنا بتحديد خصائص الكائن
    p1.name = "Mhamad";
    p1.gender  = "Male";
    p1.job  = "Programmer";
    p1.age  = 21;
 // p2 هنا قمنا بتحديد خصائص الكائن
    p2.name = "Rose";
    p2.gender  = "Female";
    p2.job  = "Secretary";
    p2.age  = 22;
// p3 هنا قمنا بتحديد خصائص الكائن
    p3.name = "Ahmad";
    p3.gender  = "Male";
    p3.job  = "Doctor";
    p3.age  = 34;
```

```
// هنا قمنا بتحديد خصائص الكائن p4
    p4.name = "Rabih";
    p4.gender  = "Male";
    p4.job  = "Engineer";
    p4.age  = 27;
   // هنا قمنا بعرض خصائص كل كائن
    p1.printInfo();
    p2.printInfo();
    p3.printInfo();
    p4.printInfo();

  }

}
```

سنحصل على النتيجة التالية عند التشغيل

```
Name: Mhamad
Gender: Male
Job: Programmer
Age: 21

Name: Rose
Gender: Female
Job: Secretary
Age: 22

Name: Ahmad
Gender: Male
Job: Doctor
Age: 34

Name: Rabih
Gender: Male
Job: Engineer
Age: 27
```

مفهوم الكونستركتور في جافا:

Constructor: تكتب كونستركتور بالعربية.
من أهم الأشياء التي عليك التفكير بها بعد إنشاء كلاس جديد, هي تسهيل طريقة خلق كائنات من هذا الكلاس.
من هنا جائت فكرة الكونستركتور و الذي هو عبارة عن دالة لها نوع خاص, يتم إستدعائها أثناء إنشاء كائن لتوليد قيم أولية للخصائص الموجودة فيه.

بما أنه لا يمكن إنشاء كائن من كلاس إلا من خلال كونستركتور, سيقوم مترجم جافا بتوليد كونستركتور إفتراضي فارغ عنك إذا وجد أن الكلاس الذي قمت بتعريفه لا يحتوي على أي كونستركتور.

مثال:

إذا قمنا بتعريف كلاس إسمه Person و لم نقم بتعريف كونستركتور له كما في الكلاس التالي.

```
class Person {

}
```

سيقوم المترجم بإنشاء كونستركتور فارغ بشكل تلقائي عنا كالتالي

Classes and objects

```
class Person {

    public Person() {

    }

}
```

نقاط مهمة حول الكونستركتور:

- **كل كلاس يتم إنشاؤه يحتوي على كونستركتور واحد على الأقل. حتى إن لم تقم بتعريف أي كونستركتور في الكلاس فإن مترجم جافا سيقوم بإنشاء واحد إفتراضي عنك.**
- **في كل مرة يتم فيها إنشاء كائن جديد من الكلاس، معنى ذلك أنه تم استدعاء أحد الكونستركتورات الخاصة به.**
- **الكونستركتور يجب أن يحمل نفس إسم الكلاس.**
- **نوع الكونستركتور في الغالب يكون public ليكون بالإمكان الوصول إليه من أي مكان.**
- **في حال قمت بتعريف كونستركتور، لن يقوم المترجم بإنشاء واحد إفتراضي، أي لن يعود هناك كونستركتور إفتراضي.**
- **في حال قمت بإنشاء كونستركتور أو أكثر، يمكنك دائماً إنشاء كونستركتور فارغ حتى تستخدمه إن كنت لا تريد إعطاء قيم أولية محددة للخصائص عند إنشاء الكائن.**

الآن سنرجع إلى الكلاس Person, وسنضيف فيه 2 كونستركتور, واحد فارغ (أي مثل الإفتراضي), و آخر يمكننا من خلاله إدخال قيم مباشرةً في الخصائص الموجودة في الكائن بدل إستدعاء كل خاصية موجودة فيه.

مثال:

```
public class Person {


    // هنا قمنا بتعريف 4 خصائص

    String name;

    String gender;

    String job;

    int age;
```

```
// هنا قمنا بتعريف constructor فارغ, أي كأننا قمنا بتعريف constructor إفتراضي
   public Person() {


   }

/*
هنا قمنا بتعريف  constructorثاني, الهدف منه إعطاء قيم لجميع الخصائص الموجودة في الكائن عند إنشاءه مباشرةً.
*/
// عند استدعاء هذا ال constructor عليك إدخال 4 قيم من نفس النوع و بالترتيب الموضوع


  public Person(String n, String s, String j, int a) {
      name = n;   // name سيتم وضعه كقيمة للخاصية n الذي سيتم تخزينه في String الـ
      gender = s;   // gender الذي سيتم تخزينه في s سيتم وضعه كقيمة للخاصية String الـ
      job = j;   // job الذي سيتم تخزينه في j سيتم وضعه كقيمة للخاصية String الـ
      age = a;   // age الذي سيتم تخزينه في a سيتم وضعه كقيمة للخاصية int الـ
   }
// هنا قمنا بتعريف دالة تطبع محتوى كل خاصية عندما يتم استدعاءه
void printInfo() {
      System.out.println("Name: " +name);
      System.out.println("Gender: " +gender);
      System.out.println("Job: " +job);
```

```
        System.out.println("Age: " +age);
        System.out.println();
    }

}
public class Main {

    public static void main(String[] args) {
  // Person هنا قمنا بإنشاء كائنات من الكلاس
Person p1 = new Person("Mhamad", "Male", "Programmer", 21);
 // الكائن p1 يمثل الشخص محمد مع تحديد كامل خصائصه
Person p2 = new Person("Rose", "Female", "Secretary", 22);
  // الكائن p2 مع تحديد كامل خصائصه  يمثل الشخص روز //
Person p3 = new Person("Ahmad", "Male", "Doctor", 34);
 // الكائن p3 يمثل الشخص أحمد مع تحديد كامل خصائصه
Person p4 = new Person("Rabih", "Male", "Engineer", 27);
  // الكائن p4 يمثل الشخص ربيع مع تحديد كامل خصائصه
/*
      هنا قمنا بانشاء كائن جديدباستخدام الconstructorالفارغ,فاضطررنا الي ادخال قيمة لكل
      خاصية موجودة فيه .
                                                                    */
Person p5 = new Person();
    // p5 هنا قمنا بتحديد خصائص الكائن
      p5.name = "Lina";
      p5.gender  = "Female";
```

```
        p5.job  = "Graphic Designer";

        p5.age  = 24;

    // هنا قمنا بعرض خصائص كل كائن

       p1.printInfo();

        p2.printInfo();

        p3.printInfo();

        p4.printInfo();

        p5.printInfo();


    }


}
```

سنحصل على النتيجة التالية عند التشغيل.

Classes and objects

```
Name: Mhamad
Gender: Male
Job: Programmer
Age: 21

Name: Rose
Gender: Female
Job: Secretary
Age: 22

Name: Ahmad
Gender: Male
Job: Doctor
Age: 34

Name: Rabih
Gender: Male
Job: Engineer
Age: 27

Name: Lina
Gender: Female
Job: Graphic Designer
Age: 24
```

Classes and objects