

## 1. نماذج البرمجة Programming Paradigms

### 1.1 Object-oriented programming

في البرمجة الكائنية، يتم تمثيل المعلومات ككافة تصف مفاهيم مجال المشكلة ومنطق التطبيق. تحدد الفئات الطرق التي تحدد كيفية معالجة المعلومات.

### 1.2 Procedural programming

بينما في البرمجة الشئية، يتم تشكيل هيكل البرنامج من خلال البيانات التي يعالجها ، في البرمجة الإجرائية ، يتم تشكيل هيكل البرنامج من خلال الوظيفة المطلوبة للبرنامج: يعمل البرنامج كدليل خطوة بخطوة للوظيفة المطلوب أدائها.

### 1.3 Static keyword

الفرق بين Non-Static vs. Static في الدروس السابقة قمنا بتعريف دوال باستخدام static كما في المثال التالي:

```
static void printHelloWorld() {
    System.out.println("Hello World");
}
```

الفرق بين Static vs. Non-Static هو انه في الدوال من نوع static يمكننا استخدامها بدون انشاء object لهذا الكلاس بينما دوال Non-Static يجب علينا انشاء object لهذا الكلاس ثم استخدام هذه الدالة كما في المثال التالي:

```
class Animal{
    private String name;
    private int age;

    Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }

    void hunt() {
        System.out.println("We Are Hunting Now ");
    }

    static void staticHunt() {
        System.out.println("StaticHunt: We Are Hunting Now ");
    }
}

Animal cat = new Animal("Cat",4);
cat.hunt();
Animal.staticHunt();
```

## Java Access Modifiers 1.4

قبل البدء بشرح مفهوم Modifiers لاحظنا سابقا تكرار كلمة public كثيرا وهي إحدى خيارات Modifiers في لغة Java، تنقسم Modifiers الى ثلاثة يمكن تعريفها في الجدول التالي:

التعريف	نوع Modifiers
يمكن الوصول الى العناصر الخاصة بالكلاس من اي مكان	public
لا يمكن الوصول الى العناصر الخاصة بالكلاس الا من داخل الكلاس	private
يمكن الوصول الى العناصر الخاصة بالكلاس من الكلاسات الموجودة بنفس المجلد والكلاسات التي ترث منه	protected
يمكن الوصول الى العناصر الخاصة بالكلاس من الكلاسات الموجودة بنفس المجلد	default

لشرح الفكرة بشكل مبسط سوف نقوم بالتجربة على object من نوع Animal في الوقت الحالي جميع العناصر الخاصة بهذا class لم يتم تعريف Modifiers خاصة بها اي انها الان من نوع default ويتم الوصول لها من الكلاسات الموجودة بنفس المجلد.

```

class Animal{
String name;
int age;

    Animal(String name, int age)
{
    this.name = name;
this.age = age;
}
}
public class Main {
    public static void main(String[] args) {
        Animal cat = new Animal("Cat",4);
        System.out.println(cat.name);
    }
}

```

في المثال الأعلى يمكننا طباعة خاصية name الخاصة ب cat لكن لو قمنا بتغيير نوع Modifiers الخاصة بعناصر هذا الكلاس فسوف يصبح من غير الممكن طباعتها.

```

class Animal{
private String name;
private int age;

Animal(String name, int age) {
this.name = name;
this.age = age;
}
}

public class Main {
    public static void main(String[] args) {
        Animal cat = new Animal("Cat",4);
        System.out.println(cat.name);
    }
}

```

وسنحصل هنا على خطأ

java: name has private access in Animal

## 2. البرمجة الكائنية – Object Oriented Programming

تنقسم OOP الى أربع مفاهيم أساسية:

- التغليف Encapsulation
- الوراثة Inheritance
- تعدد الاشكال Polymorphism
- التجريد Abstraction

### 2.1 التغليف

معنى التغليف هو التأكد من إخفاء البيانات "الحساسة" عن المستخدمين. لتحقيق ذلك، يجب عليك:

- تعريف Attributes الخاصة بالكلاس على أنها private
  - توفير طرق الحصول العامة وتعيينها للوصول إلى قيمة المتغير الخاص وتحديثها
- تعرفنا في السابق على Modifiers ومن أجل تحقيق مفهوم Encapsulation سنحتاج إلى جعل جميع Attributes من نوع private كما في المثال التالي:

```
class Animal{
    private String name;
    private int age;

    Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }
}
```

ثم سوف نقوم بتوفير وسيلة للحصول على طريقة للوصول للعناصر الخاصة بهذا الكلاس وتسمى هذه الطريقة بـ Getters/Setters كما في المثال التالي:

```
class Animal{
    private String name;
    private int age;

    Animal(String name, int age) {
        this.name = name;
        this.age = age;
    }

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setAge(int age) {
        this.age = age;
    }
}
```

- دوال getters
    - قمنا بتعريف `getName` ووظيفتها هي الحصول على قيمة `name`
    - قمنا بتعريف `getAge` ووظيفتها هي الحصول على قيمة `age`
  - دوال setters
    - قمنا بتعريف `setName` ووظيفتها هي تغيير قيمة `name`
    - قمنا بتعريف `setAge` ووظيفتها هي تغيير قيمة `age`
- من المهم ان يكون Modifiers الخاص بهذه الدوال من نوع `public` لنتمكن من الوصول لها من اي موقع في مشروعنا.