Tipos de clases:

CLASES DENTRO DE OTRAS CLASES

- Java permite definir clases dentro de otras clases.
- A estas clases se le llaman ***anidadas***.
- Pueden ser de dos tipos, ***estáticas*** o ***no estáticas***.
- No se trata de composición de clases, sino anidamiento.
- **Pueden acceder a los atributos de la clase que le envuelve.**

Se usan para encapsular, agrupar código,…

RAZONES PARA EL USO DE CLASES ANIDADAS

- Agrupamiento lógico de clases que se utilizan en un solo lugar. Mayor cohesión.
- Aumento de la encapsulación.
- Código más legible y fácil de mantener.

CLASES INTERNAS

- Se llaman así a las clases anidadas no estáticas.
- Solo pueden existir en el marco de una instancia de la clase externa.
- Pueden acceder a sus miembros (de la clase externa).

SHADOWING EN CLASES INTERNAS

- Si definimos una variable miembro en la clase interna, con el mismo nombre otra de la clase externa, la interna oculta a la externa.
- A esto se le llama *shadowing*.

Es como una sombra para el atributo de la clase externa formada por el atributo de la interna.

CLASES LOCALES

- Clases que se definen dentro de un bloque (normalmente el cuerpo de un método)
- Afinan la cohesión del código a este nivel.

CLASES ANÓNIMAS

- Permiten definir e instanciar una clase a la vez.
- Son como clases locales sin nombre.
- Sirven para ser usadas *una vez*.
- Las podemos definir a partir de otra clase o de una interfaz.
- Podemos crearlas en el cuerpo de un método, de una clase, o como argumento de un método.

EJEMPLO:

En un mismo archivo java, tenemos estas clases:

```
 */
public class Externa {

    private int n;

    public Externa(int n) {
        this.n = n;
    }

    public int getN() {
        return n;
    }

    public void setN(int n) {
        this.n = n;
    }

}
```

```
public class Anidada {

    private String s;

    public Anidada(String s) {
        this.s = s;
    }

    public String getS() {
        return s;
    }

    public void setS(String s) {
        this.s = s;
    }

    //Como clase anidada, puede acceder a los atributos de la
    //clase externa.
    public int getN() {
        return n;
    }
```

```
public static class AnidadaEstatica {

    private String str;

    public AnidadaEstatica(String str) {
        this.str = str;
    }

    public String getStr() {
        return str;
    }

    public void setStr(String str) {
        this.str = str;
    }
```

A la propiedad n de la clase primera puede acceder la segunda, en cambio la tercera que es estática no puede acceder a n. Si n fuera estática si podría acceder.

Podemos crear instancias de las diferentes clases:

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

    //Podemos instanciar AnidadaEstatica sin necesidad de tener
    //una instancia de Externa
    Externa.AnidadaEstatica ie = new Externa.AnidadaEstatica("Hola");


    //No podemos hacer esta instanciación así
    //Externa.Interna i = new Externa.Interna("Mundo");

    //Estas son las dos formas de instanciar una clase
    //anidada NO estática
    //Creando primero la instancia externa
    Externa e = new Externa(1);
    //Y usándola para crear la anidada
    Externa.Anidada i = e.new Anidada("Mundo");
    //Creando ambas instancias a la vez
    Externa.Anidada i2 = new Externa(2).new Anidada("Mundo");
```