

EJERCICIO GUIADO. JAVA: POO. PROGRAMACIÓN DE UNA CLASE

Programación de una Clase

En este ejercicio guiado, crearemos la Clase *SalaCine*, que hemos descrito en la hoja anterior. Luego, a partir de esta clase, fabricaremos objetos representando salas de cine, y los usaremos en un proyecto Java.

Recuerda las características que hemos decidido para la Clase *SalaCine* en la hoja anterior:

CLASE SALACINE

Nombre de la Clase: *SalaCine*

Propiedades de los objetos *SalaCine*:

Aforo	- número entero (int)
Ocupadas	- número entero (int)
Película	- cadena (String)
Entrada	- número decimal (double)

Valores por defecto de los objetos del tipo *SalaCine*:

Aforo: **100**
Ocupadas: **0**
Película: **(cadena vacía)**
Entrada: **5**

Métodos de los objetos del tipo *SalaCine*:

Métodos de asignación de propiedades (set)

setAforo	- modifica la propiedad Aforo
setOcupadas	- modifica la propiedad Ocupadas
setLibres	- modifica la propiedad Ocupadas también
setPelícula	- modifica la propiedad Película
setEntrada	- modifica la propiedad Entrada

Métodos de petición de información (get)

getAforo	- devuelve el valor de la propiedad Aforo
getOcupadas	- devuelve el valor de la propiedad Ocupadas
getLibres	- devuelve el número de butacas libres
getPorcentaje	- devuelve el porcentaje de ocupación de la sala
getIngresos	- devuelve los ingresos obtenidos por la venta de entradas
getPelícula	- devuelve el valor de la propiedad Película
getEntrada	- devuelve el valor de la propiedad Entrada

Métodos de orden

Vaciar	- vacía la ocupación de la sala y borra la película
entraUno	- le indica al objeto que ha entrado una persona más en la sala

PROGRAMACIÓN DE UNA CLASE

Fichero de la Clase

La programación de una clase de objetos se realiza en un fichero aparte, cuyo nombre es exactamente el mismo que el de la propia clase, y cuya extensión es .java.

Por ejemplo, si queremos programar la clase SalaCine, esto se debe hacer en un fichero llamado:

SalaCine.java

Cuando programemos esta clase dentro de NetBeans, veremos las facilidades que nos proporciona este para la creación de la clase. De momento, solo veremos de forma teórica como hay que programar la clase. (No tiene que introducir lo que viene a continuación en ningún sitio)

Estructura básica de la Clase

Dentro del fichero de la clase, comenzará la programación de esta de la siguiente forma:

```
public class SalaCine {  
  
  
}
```

La programación de una clase comienza siempre con una línea de código como la que sigue:

```
public class NombreDeLaClase {  
  
  
}
```

Toda la programación de la clase se introducirá dentro de las dos llaves.

Propiedades de la Clase

Lo primero que se debe introducir en la clase que se está programando son las propiedades. Las propiedades de una clase son básicamente variables globales de ésta. Si introducimos las propiedades de la clase *SalaCine*, esta nos quedaría así:

```
public class SalaCine {  
  
    int Aforo;  
    int Ocupadas;  
    String Película;  
    double Entrada;  
  
}
```

Constructor de la Clase

Cuando se planteó la clase *SalaCine*, se tuvo que decidir qué valores iniciales deberían tener las propiedades de la clase. Para asignar estos valores iniciales, es necesario programar lo que se denomina el *Constructor*.

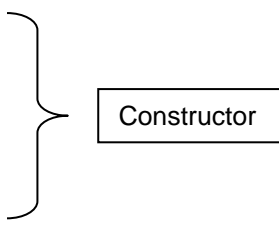
El *Constructor* de una clase es un método (un procedimiento para entendernos) un poco especial, ya que debe tener el mismo nombre de la clase y no devuelve nada, pero no lleva la palabra *void*. Dentro del constructor se inicializan las propiedades de la clase.

En general, la programación del constructor sigue la siguiente sintaxis:

```
public NombreDeLaClase() {  
    propiedad1 = valor;  
    propiedad2 = valor;  
    etc...  
}
```

La clase *SalaCine*, añadiendo el *Constructor*, tendrá el siguiente aspecto:

```
public class SalaCine {  
  
    int Aforo;  
    int Ocupadas;  
    String Película;  
    double Entrada;  
  
    //Constructor  
    public SalaCine() {  
        Aforo = 100;  
        Ocupadas = 0;  
        Película = "";  
        Entrada = 5.0;  
    }  
}
```



Observa como usamos el constructor de la clase *SalaCine* para asignar a cada propiedad los valores por defecto decididos en el diseño de la clase que se hizo en la hoja anterior.

Métodos del tipo set

Todas las clases suelen contener métodos del tipo *set*. Recuerda que estos métodos permiten asignar valores a las propiedades de la clase.

Debes tener en cuenta también que cuando se habla de método de una clase, en realidad se está hablando de un procedimiento o función, que puede recibir como parámetro determinadas variables y que puede devolver valores.

Los métodos del tipo *set* son básicamente procedimientos que reciben valores como parámetros que introducimos en las propiedades. Estos métodos no devuelven nada, así que son *void*.

Se recomienda, que el parámetro del procedimiento se llame de forma distinta a la propiedad que se asigna.

Veamos la programación del método *setAforo*, de la clase *SalaCine*:

```
public void setAforo(int afo) {  
  
    Aforo = afo;  
  
}
```

Observa este método:

- Es *void*, es decir, no devuelve nada (el significado de la palabra *public* se verá más adelante)

- El método recibe como parámetro una variable del mismo tipo que la propiedad que queremos modificar (en este caso *int*) y un nombre que se recomienda que no sea igual al de la propiedad (en nuestro caso, *afo*, de aforo)
- Puedes observar que lo que se hace simplemente en el método es asignar la variable pasada como parámetro a la propiedad.

La mayoría de los procedimientos *set* usados para introducir valores en las propiedades tienen la misma forma. Aquí tienes la programación de los demás procedimientos *set* de la clase *SalaCine*.

```
//Método setOcupadas
public void setOcupadas(int ocu) {
    Ocupadas = ocu;
}

//Método setPelícula
public void setPelícula(String peli) {
    Película = peli;
}

//Método setEntrada
public void setEntrada(double entra) {
    Entrada = entra;
}
```

Hay un método *set* de la clase *SalaCine* llamado *setLibres* cuya misión es asignar el número de localidades libres del cine. Sin embargo la clase *SalaCine* no tiene una propiedad "Libres". En realidad, este método debe modificar el número de localidades ocupadas. Observa su programación:

```
//Método setLibres
public void setLibres(int lib) {
    int ocu;

    ocu = Aforo - lib;
    Ocupadas = ocu;
}
```

Al asignar un número de localidades ocupadas, estamos asignando indirectamente el número de localidades libres. Como puedes observar en el método, lo que se hace es calcular el número de localidades ocupadas a partir de las libres, y asignar este valor a la propiedad *Ocupadas*.

No se pensó en crear una propiedad de la clase llamada *Libres* ya que en todo momento se puede saber cuantas localidades libres hay restando el Aforo menos las localidades *Ocupadas*.

La clase *SalaCine*, añadiendo los métodos *set*, quedaría de la siguiente forma:

```

public class SalaCine {

    int Aforo;
    int Ocupadas;
    String Película;
    double Entrada;

    //Constructor
    public SalaCine() {
        Aforo = 100;
        Ocupadas = 0;
        Pelicula = "";
        Entrada = 5.0;
    }

    //Métodos set

    //Método setAforo
    public void setAforo(int afo) {
        Aforo = afo;
    }

    //Método setOcupadas
    public void setOcupadas(int ocu) {
        Ocupadas = ocu;
    }

    //Método setPelicula
    public void setPelicula(String peli) {
        Pelicula = peli;
    }

    //Método setEntrada
    public void setEntrada(double entra) {
        Entrada = entra;
    }

    //Método setLibres
    public void setLibres(int lib) {
        int ocu;

        ocu = Aforo - lib;
        Ocupadas = ocu;
    }

}

```



Métodos Set

Métodos del tipo get

Al igual que los métodos *set*, los métodos *get* son muy fáciles de programar ya que suelen tener siempre la misma forma.

Estos métodos no suelen llevar parámetros y devuelven el valor de la propiedad correspondiente usando la típica instrucción *return* usada tanto en las funciones. Por tanto, un método *get* nunca es *void*. Siempre será del mismo tipo de datos que la propiedad que devuelve.

Veamos la programación del método *getAforo*:

```
//Método getAforo
public int getAforo() {
    return Aforo;
}
```

Como puedes ver el método simplemente devuelve el valor de la propiedad Aforo. Como esta propiedad es int, el método es int.

Los métodos que devuelven el resto de las propiedades son igual de sencillos de programar:

```
//Método getOcupadas
public int getOcupadas() {
    return Ocupadas;
}

//Método getPelicula
public String getPelicula() {
    return Película;
}

//Método getEntrada
public double getEntrada() {
    return Entrada;
}
```

Todos estos métodos son iguales. Solo tienes que fijarte en el tipo de datos de la propiedad que devuelven.

Existen otros métodos get que devuelven cálculos realizados con las propiedades. Estos métodos realizan algún cálculo y luego devuelven el resultado. Observa el siguiente método *get*:

```
//Método getLibres
public int getLibres() {
    int lib;
    lib = Aforo - Ocupadas;
    return lib;
}
```

No existe una propiedad *Libres*, por lo que este valor debe ser calculado a partir del Aforo y el número de localidades Ocupadas. Para ello restamos y almacenamos el valor en una variable a la que hemos llamado *lib*. Luego devolvemos dicha variable.

Los dos métodos *get* que quedan por programar de la clase *SalaCine* son parecidos:

```
//Método getPorcentaje
public double getPorcentaje() {
    double por;
    por = (double) Ocupadas / (double) Aforo * 100.0;
    return por;
}
```

Este método calcula el porcentaje de ocupación de la sala (es un valor double)

```
//Método getIngresos
public double getIngresos() {
    double ingre;
    ingre = Ocupadas * Entrada;
    return ingre;
}
```

Los ingresos se calculan multiplicando el número de entradas por lo que vale una entrada.

La clase *SalaCine* una vez introducidos los métodos get quedaría de la siguiente forma:

```
public class SalaCine {  
  
    int Aforo;  
    int Ocupadas;  
    String Película;  
    double Entrada;  
  
    //Constructor  
    public SalaCine() {  
        Aforo = 100;  
        Ocupadas = 0;  
        Pelicula = "";  
        Entrada = 5.0;  
    }  
  
    //Métodos set  
  
    //Método setAforo  
    public void setAforo(int afo) {  
        Aforo = afo;  
    }  
  
    //Método setOcupadas  
    public void setOcupadas(int ocu) {  
        Ocupadas = ocu;  
    }  
  
    //Método setPelicula  
    public void setPelicula(String peli) {  
        Pelicula = peli;  
    }  
  
    //Método setEntrada  
    public void setEntrada(double entra) {  
        Entrada = entra;  
    }  
  
    //Método setLibres  
    public void setLibres(int lib) {  
        int ocu;  
  
        ocu = Aforo - lib;  
        Ocupadas = ocu;  
    }  
  
    //Métodos get  
  
    //Método getAforo  
    public int getAforo() {  
        return Aforo;  
    }  
  
    //Método getOcupadas  
    public int getOcupadas() {  
        return Ocupadas;  
    }  
}
```



Métodos Get

```

//Método getPelicula
public String getPelicula() {
    return Película;
}

//Método getEntrada
public double getEntrada() {
    return Entrada;
}

//Método getLibres
public int getLibres() {
    int lib;
    lib = Aforo - Ocupadas;
    return lib;
}

//Método getPorcentaje
public double getPorcentaje() {
    double por;
    por = (double) Ocupadas / (double) Aforo * 100.0;
    return por;
}

//Método getIngresos
public double getIngresos() {
    double ingre;
    ingre = Ocupadas * Entrada;
    return ingre;
}

```

Métodos Get

```

}

```

Métodos de orden

Para finalizar la programación de la clase *SalaCine*, se programarán los dos métodos de orden que hemos indicado en el planteamiento de la clase. Estos métodos suelen realizar alguna tarea que involucra a las propiedades de la clase, modificándola internamente. No suelen devolver ningún valor, aunque pueden recibir parámetros.

Veamos la programación del método *Vaciar*, cuyo objetivo es vaciar la sala y quitar la película en proyección:

```

//Método Vaciar
public void Vaciar() {
    Ocupadas = 0;
    Película = "";
}

```

Como se puede observar, es un método muy sencillo, ya que simplemente cambia algunas propiedades de la clase.

El método `entraUno` es también muy sencillo de programar. Este método le indica al objeto que ha entrado un nuevo espectador. Sabiendo esto, el objeto debe aumentar en uno el número de localidades ocupadas:

```
//Método entraUno
public void entraUno() {
    Ocupadas++;
}
```

Añadiendo estos dos últimos métodos, la programación de la clase *SalaCine* quedaría finalmente como sigue:

```
public class SalaCine {
```

```
    int Aforo;
    int Ocupadas;
    String Película;
    double Entrada;
```

Propiedades (variables globales)

```
    //Constructor
    public SalaCine() {
        Aforo = 100;
        Ocupadas = 0;
        Pelicula = "";
        Entrada = 5.0;
    }
```

Constructor

```
    //Métodos set
```

```
    //Método setAforo
    public void setAforo(int afo) {
        Aforo = afo;
    }
```

```
    //Método setOcupadas
    public void setOcupadas(int ocu) {
        Ocupadas = ocu;
    }
```

```
    //Método setPelícula
    public void setPelícula(String peli) {
        Pelicula = peli;
    }
```

```
    //Método setEntrada
    public void setEntrada(double entra) {
        Entrada = entra;
    }
```

```
    //Método setLibres
    public void setLibres(int lib) {
        int ocu;

        ocu = Aforo - lib;
        Ocupadas = ocu;
    }
```

Métodos Set

```

//Métodos get

//Método getAforo
public int getAforo() {
    return Aforo;
}

//Método getOcupadas
public int getOcupadas() {
    return Ocupadas;
}

//Método getPelicula
public String getPelicula() {
    return Película;
}

//Método getEntrada
public double getEntrada() {
    return Entrada;
}

//Método getLibres
public int getLibres() {
    int lib;
    lib = Aforo - Ocupadas;
    return lib;
}

//Método getPorcentaje
public double getPorcentaje() {
    double por;
    por = (double) Ocupadas / (double) Aforo * 100.0;
    return por;
}

//Método getIngresos
public double getIngresos() {
    double ingre;
    ingre = Ocupadas * Entrada;
    return ingre;
}

```

Métodos Get

```

//Métodos de orden

//Método Vaciar
public void Vaciar() {
    Ocupadas = 0;
    Película = "";
}

//Método entraUno
public void entraUno() {
    Ocupadas++;
}

```

Métodos de orden y otros métodos.

}

EJERCICIOS RECOMENDADOS

Supongamos que tenemos una clase llamada *Rectangulo* que nos permitirá generar objetos de tipo rectángulo.

Sea el planteamiento de la clase *Rectangulo* el que sigue:

CLASE RECTANGULO

Nombre de la clase: Rectangulo

Propiedades de los objetos de la clase Rectangulo:

Base (double)
Altura (double)

Valores iniciales de las propiedades de los objetos de la clase Rectangulo:

Base – 100
Altura – 50

Métodos:

Métodos set:

setBase – permite asignar un valor a la propiedad Base.
setAltura – permite asignar un valor a la propiedad Altura.

Métodos get:

getBase – devuelve el valor de la propiedad Base
getAltura – devuelve el valor de la propiedad Altura
getArea – devuelve el área del rectángulo
getPerímetro – devuelve el perímetro del rectángulo

Otros métodos:

Cuadrar – este método debe hacer que la Altura tenga el valor de la Base.

SE PIDE:

Realiza (en papel) la programación de la clase Rectangulo a partir del planteamiento anterior.

CONCLUSIÓN

La programación de una clase se realiza en un fichero que tiene el mismo nombre que la clase y extensión .java

La estructura general de una clase es la siguiente:

```
public class NombreClase {  
  
    Propiedades (variables globales)  
  
    Constructor  
  
    Métodos set  
  
    Métodos get  
  
    Métodos de orden y otros métodos  
  
}
```

El Constructor es un procedimiento que no devuelve nada pero que no es void. El constructor debe llamarse igual que la clase y se usa para asignar los valores iniciales a las propiedades.

Los métodos set son void y reciben como parámetro un valor que se asigna a la propiedad correspondiente.

Los métodos get no tienen parámetros y devuelven el valor de una propiedad de la clase, aunque también pueden realizar cálculos y devolver sus resultados.

Los métodos de orden realizan alguna tarea específica y a veces modifican las propiedades de la clase de alguna forma.