**Universidad Tecnológica Nacional**

**Java SE8 Nivel Intermedio**

**Constructores - MVC**

Un **constructor** es un método especial de una clase que se llama automáticamente siempre que se declara un objeto de esa clase.

Su función es inicializar el objeto y sirve para asegurarnos que los objetos siempre contengan valores válidos.

Cuando se crea un objeto en Java se realizan las siguientes operaciones de forma automática:

1.  Se asigna memoria para el objeto.

2. Se inicializan los atributos de ese objeto con los valores predeterminados por el sistema.

3. Se llama al constructor de la clase que puede ser uno entre varios.

El constructor de una clase tiene las siguientes características:

Tiene el mismo nombre que la clase a la que pertenece.

En una clase puede haber varios constructores con el mismo nombre y distinto número de argumentos (se puede sobrecargar)

No se hereda.

No puede devolver ningún valor (incluyendo void).

Debe declararse público (salvo casos excepcionales) para que pueda ser invocado desde cualquier parte donde se desee crear un objeto de su clase.

**MÉTODO CONSTRUCTOR POR DEFECTO**

Si para una clase no se define ningún método constructor se crea uno automáticamente por defecto.

El **constructor por defecto** es un constructor sin parámetros que no hace nada. Los atributos del objeto son iniciados con los valores predeterminados por el sistema.

Por ejemplo, en la clase Fecha:

import java.util.\*;

public class Fecha {

    private int dia, mes, año;

    private boolean esBisiesto() {

        return ((año % 4 == 0) && (año % 100 != 0) || (año % 400 == 0));

    }

//Sobrecarga de metodo Fecha

      public void asignarFecha() {

        Calendar fechaSistema = Calendar.getInstance();

        setDia(fechaSistema.get(Calendar.DAY\_OF\_MONTH));

        setMes(fechaSistema.get(Calendar.MONTH));

        setAño(fechaSistema.get(Calendar.YEAR));

    }

    public void asignarFecha(int d) {

        Calendar fechaSistema = Calendar.getInstance();

        setDia(d);

        setMes(fechaSistema.get(Calendar.MONTH));

        setAño(fechaSistema.get(Calendar.YEAR));

    }

    public void asignarFecha(int d, int m) {

        Calendar fechaSistema = Calendar.getInstance();

        setDia(d);

        setMes(m);

        setAño(fechaSistema.get(Calendar.YEAR));

    }

    public void asignarFecha(int d, int m, int a) {

        setDia(d);

        setMes(m);

        setAño(a);

    }

//GET ANT SET

public void setDia(int d) { dia = d;   }

    public void setMes(int m) {     mes = m;   }

    public void setAño(int a) {        año = a;    }

    public int getDia() {        return dia;}

    public int getMes() {        return mes;    }

    public int getAño() {        return año;    }

    public boolean fechaCorrecta() {

        boolean diaCorrecto, mesCorrecto, anyoCorrecto;

        anyoCorrecto = (año > 0);

        mesCorrecto = (mes >= 1) && (mes <= 12);

        switch (mes) {

            case 2:

                if (esBisiesto()) {

                    diaCorrecto = (dia >= 1 && dia <= 29);

                } else {

                    diaCorrecto = (dia >= 1 && dia <= 28);

                }

                break;

            case 4:

            case 6:

            case 9:

            case 11:

                diaCorrecto = (dia >= 1 && dia <= 30);

                break;

            default:

                diaCorrecto = (dia >= 1 && dia <= 31);

        }

        return diaCorrecto && mesCorrecto && anyoCorrecto;

    }

}

no se ha definido ningún constructor, por lo que al declarar un objeto el compilador utilizará un constructor por defecto.

En este caso el método constructor por defecto es:

Fecha() { }

Vamos a añadir un constructor a la clase Fecha para poder iniciar los atributos de los nuevos objetos con valores determinados que se envían como parámetros. Si los valores corresponden a una fecha incorrecta se asigna la fecha del sistema:

public Fecha(int d, int m, int a) {

        dia = d;

        mes = m;

        año = a;

        if (!fechaCorrecta()) {

            Calendar fechaSistema = Calendar.getInstance();

            setDia(fechaSistema.get(Calendar.DAY\_OF\_MONTH));

            setMes(fechaSistema.get(Calendar.MONTH));

            setAño(fechaSistema.get(Calendar.YEAR));

        }

}

Con este método constructor podremos crear objetos de la siguiente manera:

Fecha fecha1 = new Fecha(10,2,2011);

Se crea un objeto fecha1 y se le asignan esos valores a sus atributos.

Si se crea un objeto con una fecha no válida:

Fecha fecha2 = new Fecha(10,20,2011);

A dia, mes y año se les asignará los valores del sistema.

Java crea un constructor por defecto si no hemos definido ninguno en la clase, pero si en una clase definimos un constructor ya no se crea automáticamente el constructor por defecto, por lo tanto si queremos usarlo deberemos escribirlo expresamente dentro de la clase.

En este ejemplo hemos definido un método constructor por lo que también es necesario poner el método constructor por defecto.

public Fecha(){}

Este constructor se invocará cuando se declare un objeto sin parámetros.

La clase tiene ahora dos constructores por lo que podemos crear objetos Fecha de dos formas:

Fecha f1 = new Fecha();   //se ejecuta el constructor sin parámetros

Fecha f2 = new Fecha(1,1,2010); //se ejecuta el constructor con parámetros

**INVOCAR A UN MÉTODO CONSTRUCTOR**

Un constructor se invoca cuando se crea un objeto de la clase mediante el operador new.

Si es necesario invocarlo en otras situaciones, la llamada a un constructor solo puede realizarse desde dentro de otro constructor de su misma clase y debe ser siempre la primera instrucción.

Si tenemos varios métodos constructores en la clase, podemos llamar a un constructor desde otro utilizando **this**.

Por ejemplo, si en el constructor con parámetros de la clase Fecha fuese necesario llamar al constructor sin parámetros de la misma clase escribimos:

public Fecha(int d, int m, int a) {

         this();  //llamada al constructor sin parámetros

         dia = d;

         ……….

}

**CONSTRUCTOR COPIA**

Se puede crear un objeto a partir de otro de su misma clase escribiendo un constructor llamado **constructor copia**.

Este constructor copia los atributos de un objeto existente al objeto que se está creando.

Los constructores copia tiene un solo argumento, el cual es una referencia a un objeto de la misma clase que será desde el que queremos copiar.

Por ejemplo, para la clase Fecha podemos escribir un constructor copia que permita crear objetos con los valores de otro ya existente:

Fecha fecha = new Fecha(1,1,2011);   //se invoca al constructor con parámetros

Fecha fecha1 = new Fecha(fecha);  //se invoca al constructor copia

El constructor copia que escribimos para la clase es:

//constructor copia de la clase Fecha

public Fecha(final Fecha f) {

        dia = f.dia;

        mes = f.mes;

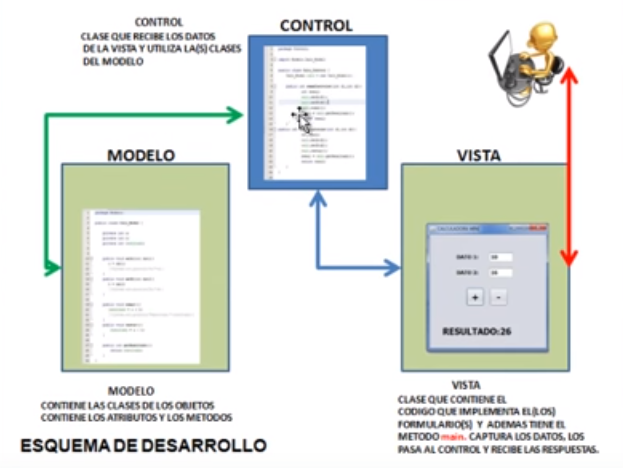
        año = f.año;

}

El constructor copia recibe una referencia al objeto a copiar y asigna sus atributos uno a uno a cada atributo del objeto creado.

Declarar el parámetro como final no es necesario pero protege al objeto a copiar de cualquier modificación accidental que se pudiera realizar sobre él.

**Patron - Model View Controller**

**Ejemplo visto en clases**

****

