Métodos que devuelven objetos

Salvador López Mendoza

Octubre 2022

Conceptos generales

Se pueden definir clases que nos ayuden a modelar el comportamiento de los objetos que participan en la solución de un problema.

Nuevos problemas:

- ¿Qué pasa si necesitamos de un método que devuelva un objeto?
- ¿Cómo hacemos para establecer una relación de orden entre dos objetos que hayamos definido?
- ¿Cómo se definen operaciones aritméticas entre objetos?

Modernización de un hospital

Problema

En el área de cuidados intensivos de un hospital se quiere automatizar la generación de horarios en los que se deben suministrar los medicamentos a los pacientes debido a que se requiere mucha precisión en esta tarea.

Para generar los horarios se debe saber para cada paciente, y de acuerdo al intervalo de minutos especificado por el médico, a qué horas debe tomar sus medicamentos o aplicar el tratamiento indicado.

El programa debe generar un listado, para cada paciente, con su horario de medicación.

Metodología de desarrollo (1)

Objetos principales:

Sustantivos: programa, hospital, horario, medicamento, paciente, tarea, médico, listado y hora.

Depuración: hospital es circunstancial, horario es el resultado esperado y medicamento es información relacionada con el paciente; tarea es lo mismo que suministrar medicamento; listado es el resultado del programa.

Objetos principales: **paciente** y **hora**.

Metodología de desarrollo (2)

Comportamiento:

- Verbos: automatizar, suministrar medicamentos.
  No hay relación con los pacientes y las horas.
- Comportamiento de las horas:
  Crear una hora inicial.
  Asignar valor a los atributos de una hora.
  Obtener el valor de los atributos de una hora.
  Sumar a una hora una cantidad de minutos.
  Imprimir la hora en un formato (`hh:mm:ss`).
  Obtener la diferencia entre dos horas.
  Determinar la relación entre dos horas.

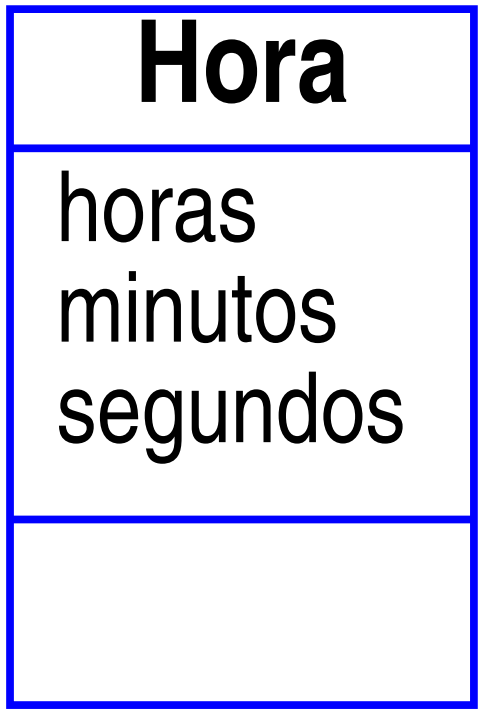
Metodología de diseño (3)

Escenario de medicación de un paciente:

1. El programa solicita los datos del paciente.
2. El usuario proporciona los datos del paciente.
3. El programa solicita el nombre del medicamento o tratamiento. Por ejemplo: cambiar suero, limpiar herida, etcétera.
4. El usuario proporciona el nombre del medicamento o tratamiento.
5. El programa solicita cada hora inicial del tratamiento.
6. El usuario proporciona la hora inicial del tratamiento.
7. El programa valida la hora inicial del tratamiento.
8. El programa solicita el intervalo de tratamiento en minutos.
9. El programa valida el intervalo de tratamiento en minutos.
10. El programa calcula las horas exactas en que se debe aplicarse el tratamiento a lo largo del día e imprime el horario.

La clase Hora

La estructura de las horas consta de tres números enteros: uno para las horas, otro para los minutos y otro más para los segundos.



La clase Hora. Encabezado

```
/**
 * Clase Hora. Define Hora en base a horas, minutos
 * y segundos
 *
 * @author Amparo Lopez Gaona
 *
 * @version 3a edicion del libro
 */

  public class Hora {
    // Guarda las horas, los minutos y los segundos
    private int horas ;
    private int minutos ;
    private int segundos ;
```

La clase Hora. Constructores

```
/**
 * Construye una Hora dandole la hora, los minutos y
     segundos
 * @param h - valor para las horas
 * @param m - valor para los minutos
 * @param s - valor para los segundos
 */

 public Hora(int h, int m, int s) {
   asignarHoras(h) ;
   asignarMinutos(m) ;
   asignarSegundos(s) ;
  }
```

La clase Hora. Constructores (II)

```
/** Construye la hora 00:00:00 por default */
public Hora() {
  this(0, 0, 0) ;
}
```

La clase Hora. Constructores (III)

```
/**
 *  Constructor de una Hora a partir de otra
 * @param h - Hora que se copiara
 */
 public Hora(Hora h) {
   this(h.horas, h.minutos, h.segundos) ;
}
```

La clase Hora. Observadores

```
/**
 * Metodo para obtener el numero de hora del dia
 *
 * @return int - las horas del objeto Hora
 */
 public int obtenerHoras() {
   return horas ;
}
/** Regresa los minutos en una hora */
public int obtenerMinutos() {
  return minutos ;
}
/** Regresa los segundos en una Hora */
public int obtenerSegundos() {
  return segundos ;
}
```

La clase Hora. Modificadores

```
/**
 * Metodo para asignar un nuevo valor a las horas
 * @param h - las horas para el objeto Hora
 *
 */
 public void asignarHoras(int h){
   if (h < 0 || h > 23) {
     System.out.println("Hora invalida") ;
     horas = 0 ;
    }
    else horas = h ;
}
```

La clase Hora. Modificadores (II)

```
/** Asigna los minutos de una hora */
public void asignarMinutos(int m) {
  if (m < 0 || m > 59) {
    System.out.println("Minuto invalido") ;
    minutos = 0 ;
  }
  minutos = m ;
}

/** Asigna los segundos de una hora */
public void asignarSegundos(int s) {
  if (s < 0 || s > 59) {
    System.out.println("Segundo invalido") ;
    segundos = 0 ;
    }
    segundos = s ;
}
```

La clase Hora. Métodos auxiliares

Métodos que ayudan a realizar operaciones, pero que no es necesario que los conozcan los usuarios de los objetos de la clase.

Se necesita de una representación de las horas que facilite las operaciones aritméticas. Representar una hora como la cantidad de segundos trancurridos hasta el momento.

```
/**
 * Metodo para convertir el objeto Hora en un numero entero
 * que equivale a esa misma hora, pero expresada en la
 * cantidad de segundos transcurridos en el dia.
 * Cada hora equivale a 3600 segundos y cada minuto a
 * 60 segundos.
 */
 private int enSegundos() {
   return this.horas * 3600 + this.minutos * 60
          + this.segundos ;
 }
```

La clase Hora. Métodos auxiliares (II)

```
/**
 * Metodo para convertir un numero entero que representa
 * los segundos transcurridos en un objeto Hora.
 *
 * @return H - objeto Hora
 */
 private Hora enHoras(int n) {
   int horas, segundos, minutos ;

   horas = n / 3600 ;
   n = n % 3600 ;
   segundos = n % 60 ;
   minutos = n / 60 ;

   return new Hora(horas, minutos, segundos) ;
}
```

La clase Hora. Métodos calculadores

```java
/**
* Metodo que suma una cantidad de segundos al objeto
    Hora
* @param mins - minutos que se sumaran a la Hora
*/

public Hora suma(int mins) {
  int segs = Math.abs(mins) * 60 ;
  int sumaSegs = this.enSegundos() + segs ;

  Hora nuevaHora = enHoras(sumaSegs) ;

  return nuevaHora ;

}
```

La clase Hora. Métodos calculadores (II)

```java
/**
* Metodo que suma una Hora al objeto Hora
* @param h - Hora que se suma a la Hora actual
*/

public Hora suma(Hora h) {
  int sumaSegs = h.enSegundos() + this.enSegundos() ;

  Hora nuevaHora = enHoras(sumaSegs) ;

  return nuevaHora ;

}
```

La clase Hora. Métodos calculadores (III)

```java
/**
* Metodo que resta una Hora al objeto Hora
* @param h - Hora que se resta a la Hora actual
*/

public Hora resta(Hora h) {

  Hora nuevaHora = null ;

  if (this.comparar(h) < 0)  {
    System.out.println("No se pueden restar," +
                       " sustraendo menor al minuendo")
                          ;
  } else {
    int restaSegs = this.enSegundos() - h.enSegundos() ;

    nuevaHora = enHoras(restaSegs) ;

  }
```

La clase Hora. Métodos calculadores (IV)

```java
/**
* Metodo para comparar dos objetsos Hora. Relacion de
    orden
* @param h - Hora contra la que se compara
* @return int - la relacion de orden (<, 0, >)
*/

public int comparar(Hora h) {
  return this.enSegundos() - h.enSegundos() ;
}
```

La clase Hora. Métodos especiales

```java
/**
* Metodo para determinar si dos Horas son iguales
* @param h - Hora contra la que se compara
* @return boolean - true si son iguales, false en otro
    caso
*/

public boolean equals(Hora h) {
  return (this.comparar(h) == 0) ;
}
```

La clase Hora. Métodos especiales (II)

```java
/**
* Metodo para convertir una Hora en un String
* @return String - la Hora en formato de cadena
*/

public String toString() {

  return this.obtenerHoras() + ":" +
         this.obtenerMinutos() + ":" +
         this.obtenerSegundos() ;
}
```