Tienda de Juegos

Pablo León Alcaide

El proyecto que quiero presentar consiste en la virtualización de una tienda de juegos y otros artículos de ocio.

El programa cuenta con un stock de artículos (ArrayList de Articulo), que puede visualizarse para su comparación (total o por categorías, por orden ascendente o descendente de precio), acceder a la descripción de los artículos, así como restringir la vista a aquellos artículos en oferta (este criterio es descrito más abajo).

La interfaz del programa permitirá que el propietario acceda para realizar modificaciones en la lista de artículos así como comprobar los pedidos realizados. Por otra parte, los clientes podrán consultar los artículos que hay en stock, acceder a los detalles, añadir elementos a la cesta y realizar pedidos de compra. Cuando un pedido contenga artículos clasificados para mayores de 18 años, se solicita el dni de un adulto.

Clases y métodos

Public class Articulo implements comparable, serializable, rebajable

Nombre (String)
Descripción (String)
Precio (double)
Descuento (double entre 0 y 1)
Estado (Enumeración)
fechaEntrada (Date)
seleccionado (boolean, determina si el artículo es añadido a la cesta)
id (int , identificador único)
cantidad (int) cantidad de este articulo en el stock

*modificarStock() → modifica la cantidad en Stock de un articulo
*calcularPrecioFinal() → precio+(precio*descuento)
*toString(nombre,id, descripcion,calcularPrecioFinal()) @Override
*cambiarDescuento() → modifica el valor de descuento. @Override
*calcularDescuento() → devuelve el valor de descuento, criterio para añadir al conjunto de ofertas. @Override

→ cambiarDescuento(double rebaja)

modifica el campo Descuento, en cada una de las clases hijas posee una restricción distinta (la base del descuento no puede sobrepasar un mínimo en cada una de ellas).

→ calcularDescuento()

En la clase Figura → Si el campo desmontable está a True, se le aplica un descuento A (constante de la clase figura), si está a False, se le aplica un descuento B (constante de la clase figura) , no obstante, se se trata de una pieza única (num_elementos=1), no se aplica descuento.
En la clase Libro → al descuento base se le añade una cantidad proporcional a partir de las 100 páginas (n.º páginas/10000)

En la clase Juego → si se trata de un juego infantil (edad <18) se le aplica el doble de descuento (Descuento base * 2)
En la clase Tablero → se aplica un descuento superior en juegos coleccionables (Descuento base * 1’3) Debido a que es más habitual que los clientes vuelvan a por expansiones de un juego que a por uno nuevo.
En la clase Cartas → sucede igual que en la clase Tablero, sólo que el descuento es menor (Descuento base * 1’15). El gasto de papél es mayor que en los juegos de tablero.
En la clase Rol → se aplica un descuento del 50% a aquellas ediciones por encima de la 2.0 (se premia a aquellos que se actualizan)

public class Figura extends Articulo tamaño
(double) peso (double)
temática (String)
desmontable (boolean)
colección (boolean)
num_elementos (int) → si no es una caja de figuras, el numero de elementos será 1
DESCUENTO_A (double)
DESCUENTO_B (double)

public class Libro extends Articulo
paginas (int)
fechaPublicacion(Date)
autor (String)
genero (String)
saga/colección (boolean)
idioma (Enumeración)
tipo (Enumeración)

public class Juego extends Articulo
duracion (int)
edad (int)

Public class Tablero extends Juego
num_piezas (int)
dimensiones (double)
num_jugadores (int)
colección (boolean)

public class Cartas extends Juego
num_cartas (int)
dificultad (Enum)
colección (boolean)

public class Rol extends Juego
genero (Enum)
material (Enum)
edicion (double)

public class Stock (envoltorio ArrayList)
ordenar()
ordenarPorPrecio()

ordenarPorCategoria()
incluirEnCesta() → boolean seleccionado=true
eliminarDeCesta() → boolean
seleccionado=false
incluirEnStock() → añade un nuevo artículo al stock si no existe (compareTo)
comprobarExistencias() → muestra artículos que escasean (ej. cantidad<5)
eliminarDeStock() → elimina un artículo del stock
descatalogar() → modifica el estado de un artículo a “Descatalogado”
mostrarDestacados() → muestra 5 artículos aleatorios del stock

A través del interfaz comparable se podrán ordenar los artículos alfabéticamente (orden natural), por el campo Precio y por su Categoría. Para ello crearemos los métodos convenientes.

Varias de las clases en la jerarquía contienen el campo boolean “coleccion” para indicar si dicho artículo pertenece a una categoría de artículos coleccionables.

Enumeraciones

Enum Estado (class Articulo)
Nuevo, Usado, Edicion_Limitada, Descatalogado

Enum Dificultad (class Cartas)
Novato, Moderado, Dificil, Veterano, Familiar

Enum Genero (class Rol)
Fantasia, Ciencia_Ficcion, Western, Terror, Pulp, Moderno,

Enum Material (class Rol)
Manual, Expansion, Dados, HojaDeJuego, Pantalla, Escenografia

Enum Tipo (class Libro)
Novela_Grafica, Comic; Novela_Aventuras, Novela_Terror, Diccionario, Guía, EligeTuAventura

Enum Idioma (class Libro)
Español, Ingles, Aleman, Frances, Italiano, Java

Interfaces:

Interface Rebajable
*cambiarDescuento()
*calcularDescuento()
La interfaz rebajable contiene dos métodos ambos relacionados con el campo “descuento”, presente en todos los artículos y que determina el precio final de los productos.

Interface Serializable
(sin métodos)

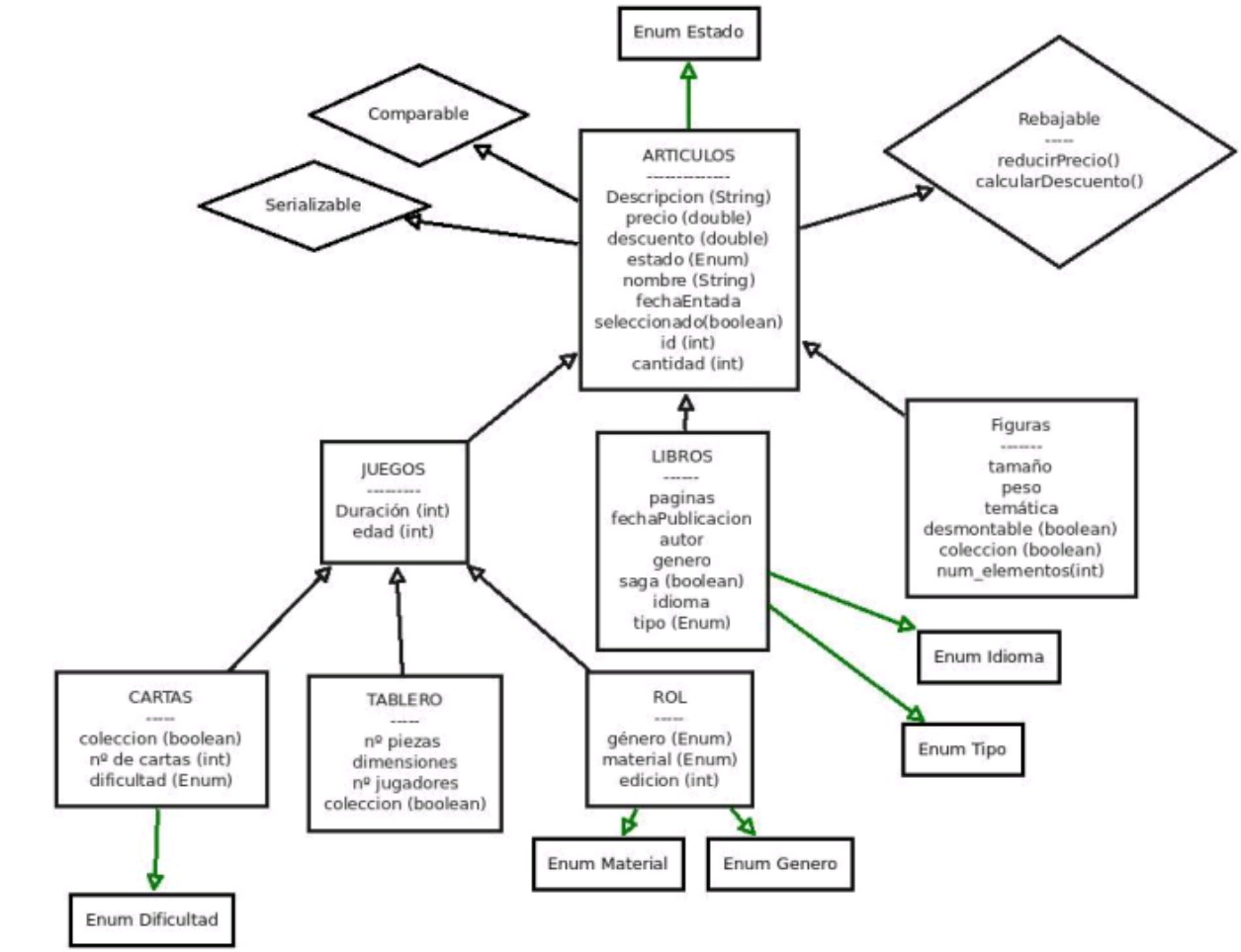
Interface Comparable

usaremos el método compareTo para controlar que no existan artículos duplicados en el stock (puede haber un artículo con una cantidad elevada, pero no múltiples artículos con el mismo nombre en el stock).

Expresiones regulares y Excepciones:

- String autor (Libros) → La cadena no puede empezar por números o símbolos, y debe tener al menos 3 caracteres. → Lanzaría NombreNoValidoException
- double precio (Articulos) → El precio no puede ser menor o igual de cero. Además, si el estado del artículo es Descatalogado, no puede ser superior a 1000, y si el estado es Edicion_Limitada, el precio no puede ser menor de 10 → Lanzaría PrecioNoValidoException
- int edad (Juegos) → la edad de juego oscila entre los 3 y los 65 años. → Lanzaría EdadNoValidaException
- double peso (Figuras) → el peso de las figuras ha de ser positivo y menor de 5000 (el coleccionismo de la tienda no incluye amantes de las esculturas pesadas) → Lanzaría PesoInvalidoException
- Date fecha Entrada / Date fechaPublicacion → la fecha debe estar dentro de un rango válido (no hay publicaciones posteriores al mes actual) → lanza FechaInvalidaException
- Campos no nulos → no se permite que al introducir elementos en el stock contenga campos nulos. → Lanzaría → FichaIncompletaException

JERARQUÍA DE CLASES


Enum Estado
Comparable
Serializable
ARTICULOS
Descripcion (String)
precio (double)
descuento (double)
estado (Enum)
nombre (String)
fechaEntada
seleccionado(boolean)
id (int)
cantidad (int)
Rebajable
reducirPrecio()
calcularDescuento()
JUEGOS
Duración (int)
edad (int)
LIBROS
paginas
fechaPublicacion
autor
genero
saga (boolean)
idioma
tipo (Enum)
Figuras
tamaño
peso
temática
desmontable (boolean)
coleccion (boolean)
num_elementos(int)
Enum Idioma
Enum Tipo
CARTAS
coleccion (boolean)
nº de cartas (int)
dificultad (Enum)
TABLERO
nº piezas
dimensiones
nº jugadores
coleccion (boolean)
ROL
género (Enum)
material (Enum)
edicion (int)
Enum Material
Enum Genero
Enum Dificultad