# Temperatura

💡 **Separación de la lógica de conversión** de Celsius a Fahrenheit fuera del main, cumpliendo con el principio de **responsabilidad única** (SRP). Bien hecho.

🧱 Clase Temperatura para encapsular datos y su comportamiento → esto es clave en orientación a objetos.

## Modificaciones

* **Inicial**

**public** **class** Main {

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

**double**[] temperaturas = **new** **double**[100];

// Toma de datos

**for** (**int** i = 0; i < temperaturas.length; i++)

temperaturas[i] = *leeSensorCelsius*();

// Cálculo con los datos en Celsius

**double** mediaCelsius = 0;

**for** (**int** i = 0; i < temperaturas.length; i++)

mediaCelsius += temperaturas[i];

mediaCelsius = mediaCelsius / temperaturas.length;

* **Final**

Clase Main:

**public** **class** Main {

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

Temperatura[] temperaturas = **new** Temperatura[100];

// Toma de datos

**for** (**int** i = 0; i < temperaturas.length; i++)

temperaturas[i] = **new** Temperatura(*leeSensorCelsius*());

// Cálculo con los datos en Celsius

**double** mediaCelsius = 0;

**for** (**int** i = 0; i < temperaturas.length; i++)

mediaCelsius += temperaturas[i].getCelsius();

mediaCelsius = mediaCelsius / temperaturas.length;

Clase Temperatura:

**public** **class** Temperatura {

**final** **double** celsius;

**public** Temperatura(**double** celsius) {

**this**.celsius = celsius;

}

**public** **double** getCelsius() {

**return** celsius;

}

**public** **double** getFaren() {

**return** celsius \* 1.8 + 32;

}

}

# Videoclub

 ✅ **Principio de Abierto/Cerrado (OCP)**: añades nuevas categorías sin tocar el código existente.

 ✅ **Responsabilidad única (SRP)**: la lógica de cálculo de importe y puntos está donde debe estar (las categorías).

 ✅ **Uso del patrón Strategy**: las categorías implementan una estrategia diferente de cálculo.

## Modficiaciones

* **Inicial**

Clase Main:

**public** **class** Main {

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

Customer raul = **new** Customer("Raúl");

Movie hotFuzz = **new** Movie("Hot Fuzz", Movie.***NEW\_RELEASE***);

Movie toyStory = **new** Movie("Toy Story", Movie.***CHILDRENS***);

Movie zombiesParty = **new** Movie("Zombies Party", Movie.***REGULAR***);

raul.addRental(**new** Rental(hotFuzz, 2));

raul.addRental(**new** Rental(toyStory, 6));

raul.addRental(**new** Rental(zombiesParty, 8));

System.***out***.println(raul.status());

}

}

Clase Rental:

**public** **class** Rental // Alquiler

{

**private** Movie movie;

**private** **int** days;

**public** Rental(Movie movie, **int** days) {

**this**.movie = movie;

**this**.days = days;

}

**public** **int** getDays() {

**return** days;

}

**public** Movie getMovie() {

**return** movie;

}

}

Clase Customer:

**public** **class** Customer {

**private** String name;

**private** List<Rental> rentals = **new** ArrayList<Rental>();

**public** Customer(String name) {

**this**.name = name;

}

**public** **void** addRental(Rental rental) {

rentals.add(rental);

}

**public** String getName() {

**return** name;

}

**public** String status() {

**double** totalPrice = 0;

**int** totalPoints = 0;

String result = "Alquileres de: " + getName() + "\n";

**for** (Rental rental : rentals) {

// 1) Calcula el importe de cada alquiler

**double** price = 0;

**switch** (rental.getMovie().getCategory()) {

**case** Movie.***NEW\_RELEASE***:

price += rental.getDays() \* 3;

**break**;

**case** Movie.***REGULAR***:

price += 2;

**if** (rental.getDays() > 2)

price += (rental.getDays() - 2) \* 1.5;

**break**;

**case** Movie.***CHILDRENS***:

price += 1.5;

**if** (rental.getDays() > 3)

price += (rental.getDays() - 3) \* 1.5;

**break**;

}

// 2) Cada alquiler da 1 punto. Punto extra para novedades alquiladas 2 o más

// días

**int** points = 1;

**if** ((rental.getMovie().getCategory() == Movie.***NEW\_RELEASE***) && rental.getDays() > 1)

points = 2;

// 3) Acumula los totales

totalPrice += price;

totalPoints += points;

// 4) Muestra el importe de esta película alquilada

result += "\t" + rental.getMovie().getTitle() + "\t" + price + " €\n";

}

result += "La deuda es de " + totalPrice + " €\n";

result += "Has obtenido " + totalPoints + " puntos en tu tarjeta";

**return** result;

}

}

Clase Movie:

**public** **class** Movie // Pel�cula

{

**private** String title;

**private** **int** category;

**public** **static** **final** **int** ***CHILDRENS*** = 2;

**public** **static** **final** **int** ***NEW\_RELEASE*** = 1;

**public** **static** **final** **int** ***REGULAR*** = 0;

**public** Movie(String title, **int** category) {

**this**.title = title;

**this**.category = category;

}

**public** **int** getCategory() {

**return** category;

}

**public** **void** setCategory(**int** category) {

**this**.category = category;

}

**public** String getTitle() {

**return** title;

}

}

* **Final**

Clase Main:

**public** **class** Main {

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

Customer raul = **new** Customer("Raúl");

Movie hotFuzz = **new** Movie("Hot Fuzz", **new** NewReleaseCategory());

Movie toyStory = **new** Movie("Toy Story", **new** ChildrenCategory());

Movie zombiesParty = **new** Movie("Zombies Party", **new** RegularCategory());

Movie documental = **new** Movie("Documental", **new** DocumentalCategory());

raul.addRental(**new** Rental(hotFuzz, 2));

raul.addRental(**new** Rental(toyStory, 6));

raul.addRental(**new** Rental(zombiesParty, 8));

raul.addRental(**new** Rental(documental,2));

System.***out***.println(raul.status());

}

}

Clase Customer:

**public** **class** Customer {

**private** String name;

**private** List<Rental> rentals = **new** ArrayList<Rental>();

**public** Customer(String name) {

**this**.name = name;

}

**public** **void** addRental(Rental rental) {

rentals.add(rental);

}

**public** String getName() {

**return** name;

}

**public** String status() {

**double** totalPrice = 0;

**int** totalPoints = 0;

String result = "Alquileres de: " + getName() + "\n";

**for** (Rental rental : rentals) {

// 1) Calcula el importe de cada alquiler

**double** price = rental.getImporte();

// 2) Cada alquiler da 1 punto. Punto extra para novedades alquiladas 2 o m��s

// d��as

**int** points = rental.getPuntos();

// 3) Acumula los totales

totalPrice += price;

totalPoints += points;

// 4) Muestra el importe de esta pel��cula alquilada

result += "\t" + rental.getMovie().getTitle() + "\t" + price + " €\n";

}

result += "La deuda es de " + totalPrice + " €\n";

result += "Has obtenido " + totalPoints + " puntos en tu tarjeta";

**return** result;

}

}

Clase Movie:

**public** **class** Movie // Pel�cula

{

**private** String title;

**private** Category category;

**public** Movie(String title, Category category) {

**this**.title = title;

**this**.category = category;

}

**public** Category getCategory() {

**return** category;

}

**public** **void** setCategory(Category category) {

**this**.category = category;

}

**public** String getTitle() {

**return** title;

}

**double** getImporte(**int** days) {

**return** category.getImporte(days);

}

**int** getPuntos(**int** days) {

**return** category.getPuntos(days);

}

}

Clase Rental:

**public** **class** Rental // Alquiler

{

**private** Movie movie;

**private** **int** days;

**public** Rental(Movie movie, **int** days) {

**this**.movie = movie;

**this**.days = days;

}

**public** **int** getDays() {

**return** days;

}

**public** Movie getMovie() {

**return** movie;

}

**int** getPuntos() {

**return** movie.getPuntos(days);

}

**double** getImporte() {

**return** movie.getImporte(days);

}

}

Interfac Category:

**public** **interface** Category {

**double** getImporte(**int** daysRental);

**int** getPuntos(**int** daysRental);

}

Clase NewReleaseCategory:

**public** **class** NewReleaseCategory **implements** Category {

@Override

**public** **double** getImporte(**int** daysRental) {

**double** price = 0;

**return** price += daysRental \* 3;

}

@Override

**public** **int** getPuntos(**int** daysRental) {

**return** (daysRental > 1) ? 2 : 1;

}

}

## Mejoras

💡 Aplicar el **patrón Factory** para la creación de películas en el main.

Clase Factoría:

**public** **class** MovieFactory {

**public** Movie create(String titulo, TipoPeliculas tipo) {

Category category = **null**;

**switch** (tipo) {

**case** ***NewRelease***:

category = **new** NewReleaseCategory();

**break**;

**case** ***Children***:

category = **new** ChildrenCategory();

**break**;

**case** ***Documental***:

category = **new** DocumentalCategory();

**break**;

**case** ***Regular***:

category = **new** RegularCategory();

**break**;

**default**:

**break**;

}

**return** **new** Movie(titulo, category);

}

}

Clase Main:

**public** **class** Main {

**public** **enum** TipoPeliculas{

***NewRelease***, ***Children***, ***Documental***, ***Regular***

}

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

Customer raul = **new** Customer("Raúl");

MovieFactory mf = **new** MovieFactory();

Movie hotFuzz1 = mf.create("Hot Fuzz", TipoPeliculas.***NewRelease***);

Movie toyStory = mf.create("Toy Story", TipoPeliculas.***Children***);

Movie zombiesParty = mf.create("Zombies Party", TipoPeliculas.***Regular***);

Movie documental = mf.create("Documental", TipoPeliculas.***Documental***);

raul.addRental(**new** Rental(hotFuzz1, 2));

raul.addRental(**new** Rental(toyStory, 6));

raul.addRental(**new** Rental(zombiesParty, 8));

raul.addRental(**new** Rental(documental,2));

System.***out***.println(raul.status());

}