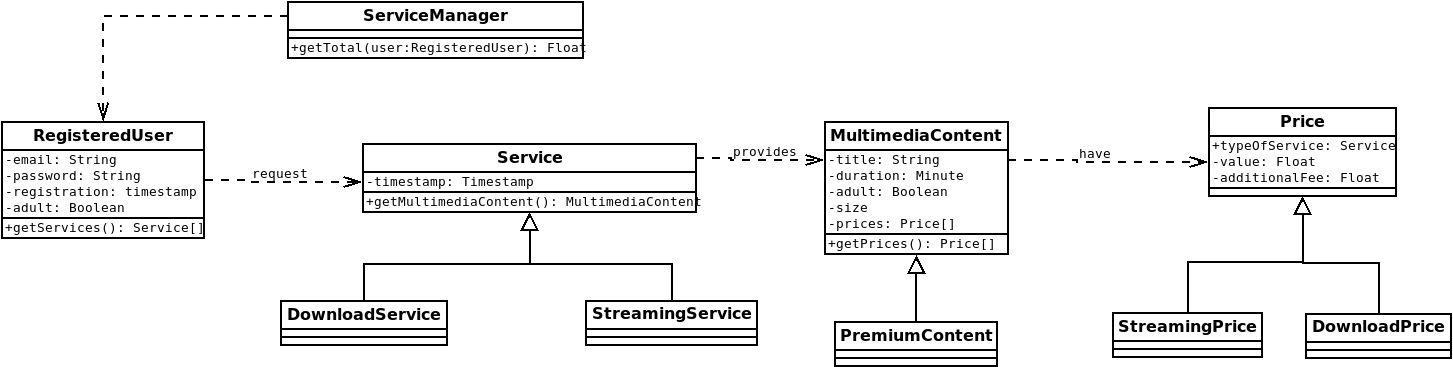
Solucion problema #1

1. Problemas:
   1. La clase RegisteredUser no deberia tener el metodo de calcular el total. Estos violaba el principio solid de “**Responsabilidad Única**”. La clase “RegisteredUser” solo debe encargarse de manejar la info y funcionalidades de un usuario registrado.
   2. El precio del fee adicional es fijo para servicio de streaming y de download. Si se quisiera especificar fee independientes para contenido premium en streaming y download, esta solución no lo permite. Además como mismo la clase “MultimediaContent” no debe tener información de su precio, la clase “PremiumContent” no debe tener información el de fee adicional. Aplicando el principio de “**Responsabilidad Única**” se puede tener la información del precio en una clase independiente.
   3. El calculo del precio violaba el principio SOLID “**Open/Close**”, ya que si se qusiera agregar un nuevo tipo de servicio, habria que modificar la clase “MultimediaContent” para agregarle un atributo con el precio del nuevo servicio, que en teoria es una extension de la clas pero el metodo del”getTotal” habria que modificarlo y esto violaria dicho principio.
   4. La clase “MultimediaContent” no deberia tener el atributo del precio asociado a ella, estaria violando el principio de “**Responsabilidad Única**” al mezclar la información de un contenido multimedia con su precio. El precio se debe manejar en una clase independiente, permitiendo en el futuro establecer precios nuevos a nuevos tipos de servicios, precios distintos por zonas, etc, sin modificar la clase “MultimediaContent”

Solucion propuesta:



Seudocodigo:

class MultimediaContent {

constructor(prices []){

this.prices = prices

}

}

class Price {

constructor(price, additionalFee = 0 ){

this.price = price,

this.additionalFee = additionalFee

}

}

class StreamingPrice {

constructor(price, additionalFee = 0) {

super(price, additionalFee)

this.typeOfService = StreamingService

}

}

class DownloadPrice {

constructor(price, additionalFee = 0) {

super(price, additionalFee)

this.typeOfService = DownloadService

}

}

class ServiceManager {

getTotal(user){

let services = user.getServices();

var total = 0;

services.forEach(service => {

let multimediaContent = service.getMultimediaContent();

let prices = multimediaContent.getPrices();

prices.forEach(price=>{

if (price.typeOfService == service){

total += price.value + price.additionalFee;

}

}

})

})

return total

}

}

Esta solucion creo que es mas flexible porque:

Cumpliendo con los principios SOLID permite:

1. El calculo del precio se hace en una clase independiente a la clase RegisteredUser por lo que si surgiera una logica nueva de como calcular el total, la clase RegistedUser no se modificaria. (Responsabilidad unica)
2. Permite agregarle precios a los contenidos multimedias sin modificar la clase de contenido multimedia. (Open/Close, Responsabilidad unica)
3. Cuando el contenido multimedia no es de tipo PremiumContent valor de additionalFee es 0 por lo que no afecta el calculo y si es de tipo PremiumContent el price que tendra asociado tendra un additionalFee distinto de 0.
4. Permite agregar mas precios al Contenido Multimedia a pesar de no crear mas servicios. Pudiera agregarseles precios a contenido de adultos, precios independientes a una seccion de contenidos, etc, y los metodos de calculo de precios se mantendrian igual. Esto cumple con el principio de sustitucion de Liskov ya que se pueden agregar muchas clases hijas a la clase Price y a la clase Service sin necesidad de modificar el codigo.