Informe Trabajo Práctico Final

Programación Orientada a Objetos II.

Integrantes:

- Camaño Valentin: elcolo619@gmail.com
- Diaz Maria Emilia: diaz.m.emilia1@gmail.com
- Rodriguez Perez Diego Jesus: oxigedo@gmail.com

Clases Modeladas:

- Usuario
- SitioWeb
- Inmueble
- Reserva
- Observer
- EstadoReserva (abstracta)
- EstadoSolicitada
- EstadoCancelada
- EstadoAceptada
- EstadoFinalizada
- FiltroCompuesto
- FiltroFecha
- FiltroPrecio
- FiltroCantHuespedes
- FiltroCiudad
- Valoracion
- Periodo
- FormaDePago (enumerativo)

Interfaces Modeladas:

- Filtro
- Propietario
- Inquilino
- PoliticaCancelacion
- MailSender
- SubscriberCancelacion
- SubscriberBajaPrecio
- SubscriberReserva

Patrones utilizados:

- Observer:
    - Para el desarrollo de la funcionalidad de notificaciones a suscriptores interesados en eventos relacionados con los inmuebles y sus reservas; se ha desarrollado un Observer encargado de manipular los diferentes maps de suscriptores y las notificaciones correspondientes.

- Adapter:
    - Para la integración de las interfaces HomePagePublisher y PopUpWindow se han integrado dos adaptadores, siendo estos, AdapterCancelacion y AdapterBajaDePrecio.
- State:
    - Para el desarrollo de los diferentes estados de una reserva se ha incluido la clase madre abstracta EstadoReserva, y sus respectivas clases de estados concretos. Estas últimas son EstadoSolicitada, EstadoAceptada, EstadoCancelada y EstadoFinalizada.
- Strategy:
    - Para el desarrollo de políticas de cancelación de un inmueble, se ha incluido la interfaz PoliticaCancelacion, implementada por las clases SinCancelacion, CancelacionBasica y CancelacionIntermedia; que responden a las diferentes políticas consignadas en el enunciado.
- Singleton:
    - Para la implementación del observer desarrollado se ha utilizado el patrón singleton, para la manipulación de una única instancia.

Decisiones de diseño:

- Todo inmueble registrado en un propietario, se encuentra publicado en el sitio web. La gestión de los inmuebles disponibles de dicho sitio se realiza a través de los inmuebles incluidos en todos sus usuarios propietarios registrados.
- Todo inmueble se encuentra publicado durante todo el año con un precio por defecto definido a la hora de registrarse. La disponibilidad de dicho inmueble está dada por las reservas aceptadas que posee el mismo.
- La clase Observer se encuentra estructurada únicamente con 3 maps, porque se asume que únicamente se podrán subscribir listeners a estos 3 eventos dados.
- Los métodos de realizarCheckOut(), cancelarReserva(), reservaCancelada(), getAntiguedadUsuario(), que reciben como parámetro una fecha, se asume que por interfaz dicha fecha corresponderá a la del día corriente.
- Una reserva únicamente puede ser cancelada si se encuentra en estado aceptada.
- Si una valoración es instanciada con puntaje menor a 1 o mayor a 5, se limita el mismo a 1 o 5 respectivamente.
- La cantidad de veces que un inmueble fue alquilado corresponde únicamente a reservas finalizadas.
- Las reservas futuras de un usuario contempla todas las reservas con un check in posterior al día corriente, sin distinción entre los estados de las mismas.
- Las categorías de valoraciones para cada entidad en el sitio web se distinguen en 3 listas diferentes porque se asumen que no se añadirán más entidades al modelo.
- Al momento de cancelarse una reserva, se ejecutará la política de cancelación consignada en el inmueble en dicho momento.