PROGRAMACIÓN CON OBJETOS II

Trabajo Integrador

Alquileres

GRUPO: TRIPLET

Nicolás Tancredi
tancredi.nikco@gmail.com
Pablo Troche
trochepablodaniel@gmail.com
Agustín Tupilojón
atupilojon@gmail.com

ESTRATEGIA DE MODELADO:

Luego de la etapa del análisis de la problemática a resolver, se define contemplar una clase **Sitio** que se encargará de administrar las funcionalidades que vinculadas a canalizar la resolución de las consignas del enunciado (búsquedas de inmuebles, visualización, reservas y cancelaciones, y puntajes). Tomamos como núcleo de la solución cuatro clases indispensables.

Sitio: Clase que se encarga de ser el concentrador de la solución. Esto significa que todos los mensajes que responden a la solución pasan a través de "Sitio". A su vez es el responsable de administrar el listado de usuarios, inmuebles y publicaciones.

Inmueble: Es la clase que modela la representación de un cualquier objeto que se pueda publicar como un producto de alquiler. Concentra toda la estructura correspondiente que puede tener un inmueble. Como, por ejemplo: tipo de inmueble, capacidad, ubicación, servicios, etc.

Usuario: Modelamos la clase "Usuario" como solución para todo usuario de la aplicación y dejamos su "rol" como inquilino o propietario según el contexto que se lo esté **relacionando**. Esta clase posee como estructura toda la información de un usuario.

Publicación: Es una clase que hace referencia a la relación entre un inmueble y un usuario con el rol de propietario. Esta se publica dentro del sitio y es capaz de tomar responsabilidades de poder responder cuando reservarse con datos propios de cada colaborador (inmueble, reservas, puntajes de propietario).

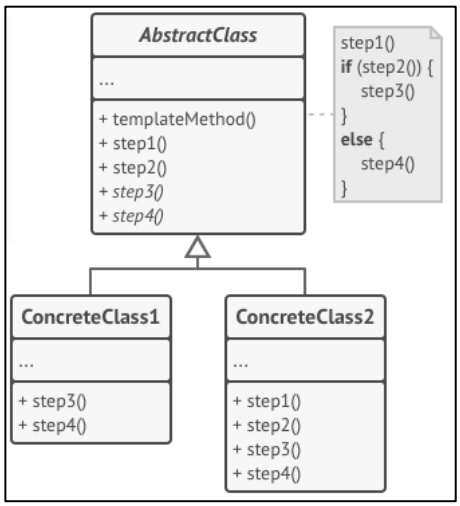
Búsquedas:

Para satisfacer el requerimiento de que el usuario pueda realizar búsquedas con criterios específicos se modelo las siguientes clases como participantes del patrón de diseño *template method, "Buscador", "Filtro", "FiltroXCiudad", etc.*

La responsabilidad de realizar el filtrado de publicaciones a mostrar es la clase colaboradora "Buscador" qué contiene en su estructura una colección de clase "Filtros" (como clase abstracta) donde cada elemento representa uno de los criterios a aplicar a la búsqueda, como por ejemplo ciudad, fechas de entrada y salida, capacidad y precio.

Filtro implementa, para la reutilización de código, un patrón de diseño del tipo *template method* en donde es definido como una clase abstracta que se encarga de aplicar el algoritmo de iteración y que delega en las clases concretas la aplicación del algoritmo de filtrado de publicaciones, retornando un conjunto de publicaciones que coincide con variables definidas en sus estructuras internas.

Por ejemplo, la clase FiltroCiudad es una subclase de Filtro dónde su estructura interna (atributo "ciudad") denota el valor del parámetro de búsqueda.



Rankings:

La administración y la posibilidad de realizar rankeos dentro de la aplicación la modelamos con las siguientes clases:

Utilizamos una Interfaz llamada “IPuntuable” para que el Sitio pueda recibir como parámetro a cualquiera de las dos clases, “Usuarios” e “Inmuebles”, ya que ambas la implementan.

La clase Puntaje posee una estructura interna que conforma de un puntaje para denotar, el valor a puntuar, un comentario y el rol del usuario de quién calificó. Este último es necesario ya que, tomando como premisa que un usuario puede ser inquino o propietario, el atributo define el rol de quién colabora.

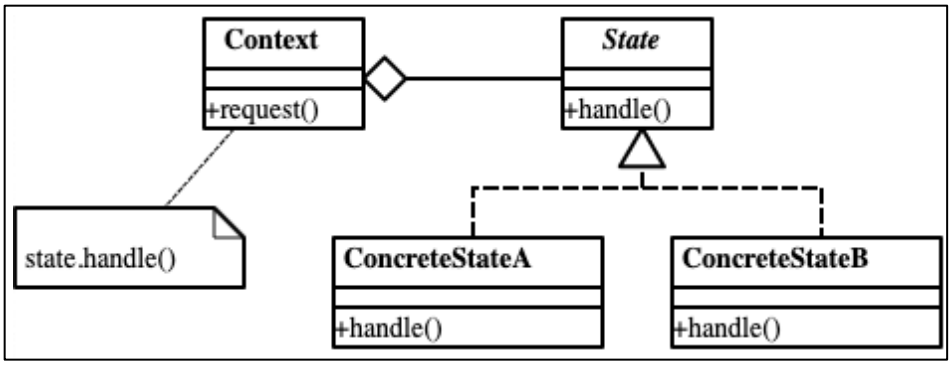
Por último, tiene como colaboración a una de las categorías pasibles de puntuación (por ejemplo, “limpieza” para inmuebles o “cordialidad” para propietarios) definidas por el “administrador del sitio”.

Reservas:

Una Publicación tiene como colaborador a una lista de Reserva que tienen la responsabilidad de representar una porción de tiempo en la cual un inquilino desea hacer uso del inmueble de la misma.

La estructura de la Reserva está compuesta por sus fechas (de entrada y salida), una colaboración de Usuario que representa a quién efectúa la reserva (en rol de inquilino) y una colaboración para representar su “estado”.

El estado de la reserva está representado por un patrón del tipo *state*.



En nuestra implementación, el rol de “contexto” lo tiene Reserva, el “estado” está representado por la clase abstracta “EstadoReserva”, y las clases concretas, a las cuales se les delega parte del comportamiento de Reserva, están definidas por EstadoSolicitud (“estado” con el que nace una Reserva), EstadoAprobado, EstadoFinalizado y EstadoCancelado.

Administración del Sitio:

Para poder responder la administración del sitio tomamos como estrategia poder crear clases concretas con un atributo de “String” dando la posibilidad de instanciar clases a partir de esa estructura y poder responder los diferentes esquemas.

Para las **categorías de rankeos** creamos una clase llamada “Categoría”, esto permite al administrador poder seguir instanciando objetos de “Categorías” como, “cortesía”, “disponibilidad”, “limpieza”, “ubicación del centro”, etc.

Para los **tipos de inmuebles** también creamos una clase que se lleva la responsabilidad de responder que tipo de inmueble va a relacionar el “Inmueble” con un “String”.

Para los **servicios** de la misma forma la clase “Servicio” posee un “String” que responde esa clasificación. Este a diferencia de los anteriores es una colaboración en con una lista para la clase “Inmueble”.

Alquileres

TripleT | Objeto 2