TRABAJO PRÁCTICO PATRONES

GONZALO FARIAS
FACUNDO RODRIGUEZ

PROGRAMACIÓN ORIENTADA A OBJETOS 3

STATE

El patrón de diseño State, que se traduce como Estado en español, es un patrón de comportamiento que se utiliza en el desarrollo de software para modelar el comportamiento de un objeto a lo largo de diferentes estados, permitiendo que el objeto cambie su comportamiento interno cuando cambia su estado.
Este patrón es particularmente útil cuando un objeto debe alterar su comportamiento en función de cambios en sus datos internos o en respuesta a eventos externos.

ELEMENTOS DE STATE

Contexto: Este es el objeto que tiene un estado interno y cuyo comportamiento cambia según el estado. El contexto contiene una referencia a un objeto de estado concreto.

Estado: Representa un estado específico y concreto en el que el contexto puede encontrarse. Los estados implementan una interfaz común que define los métodos que el contexto puede llamar para cambiar su estado o realizar acciones específicas asociadas a ese estado.

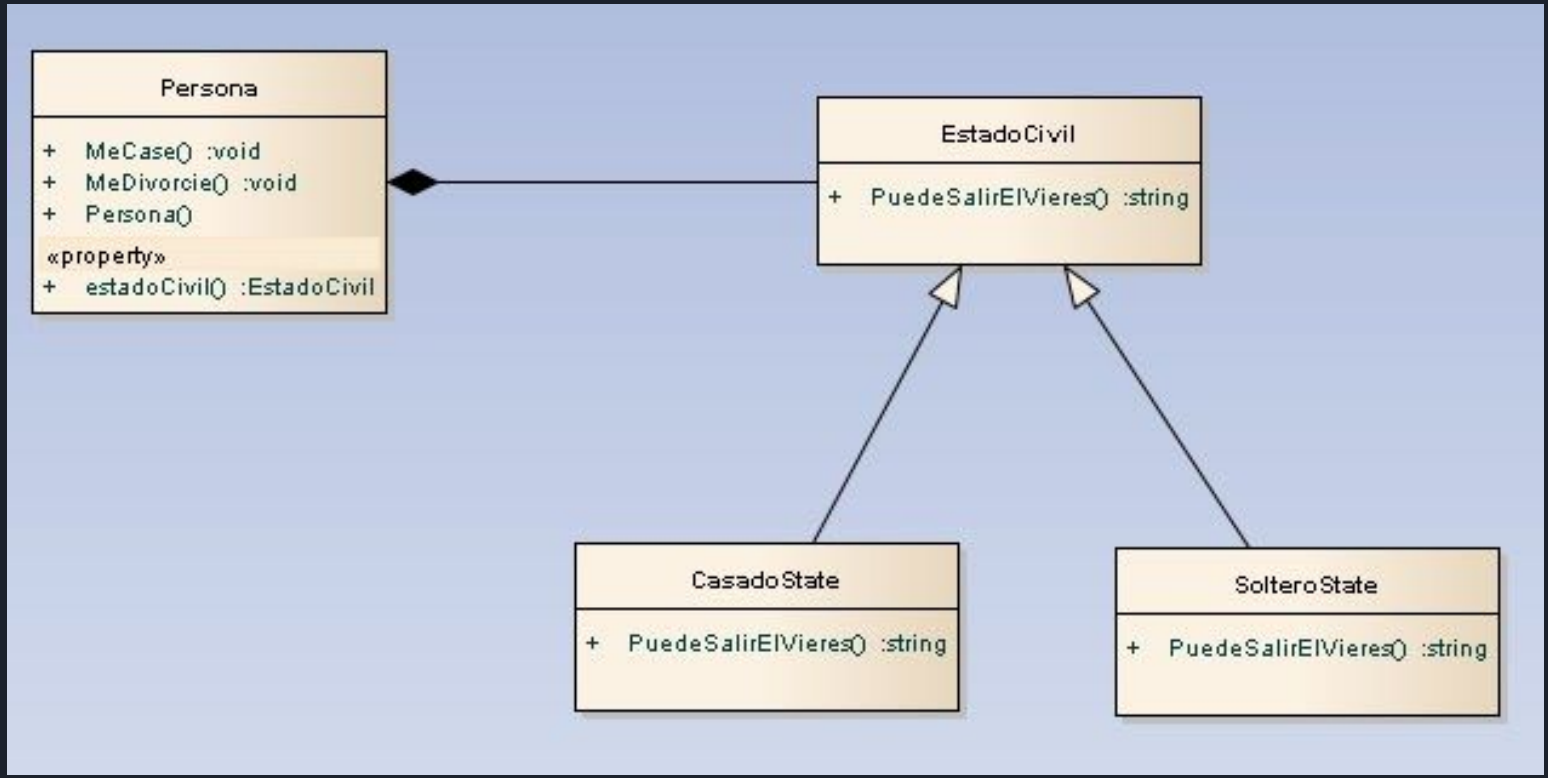
VENTAJAS DE STATE

Modularidad: El patrón State promueve la modularidad al dividir el comportamiento relacionado con el estado en clases individuales, lo que facilita la adición o modificación de estados y comportamientos.

Mantenimiento y escalabilidad: Agregar nuevos estados o cambiar el comportamiento de un objeto en función de su estado se vuelve más sencillo sin afectar otras partes del código.

Claridad: Facilita la comprensión del código, ya que cada estado se modela en su propia clase, lo que reduce la complejidad y aumenta la legibilidad.

EJEMPLO


Persona
+ MeCase() :void
+ MeDivorcie() :void
+ Persona()
«property»
+ estadoCivil() :EstadoCivil
EstadoCivil
+ PuedeSalirElVieres() :string
CasadoState
+ PuedeSalirElVieres() :string
SolteroState
+ PuedeSalirElVieres() :string

EXPLICACIÓN DE EJEMPLO

En el diagrama de clases anterior se puede observar 4 clases en primer lugar está la clase Persona que representa el contexto y es la contenedora de los estados concretos, luego está la clase EstadoCivil la cual es una clase abstracta que es una generalización de todos los estados civiles que en este caso son SolteroState y CasadoState, estos dos últimos funcionan como estados concretos que heredan de estado civil y redefinen los métodos.

MEMENTO

El patrón de diseño Memento es un patrón de comportamiento que se utiliza en el desarrollo de software para capturar y externalizar el estado interno de un objeto en un momento dado, de manera que el objeto pueda ser restaurado a ese estado más tarde. Este patrón es útil cuando se necesita implementar la capacidad de deshacer o revertir cambios en un objeto, o cuando se desea guardar y restaurar estados anteriores de un objeto.

ELEMENTOS DE MEMENTO

Originador(Originator): Este es el objeto cuyo estado se desea guardar. El originador crea un memento que contiene una copia del estado actual del objeto y puede utilizarlo para restaurar su estado más tarde.

Memento: Representa el estado capturado de un objeto originador en un momento específico. El memento contiene información sobre el estado que se va a guardar, pero no proporciona acceso directo a ese estado.

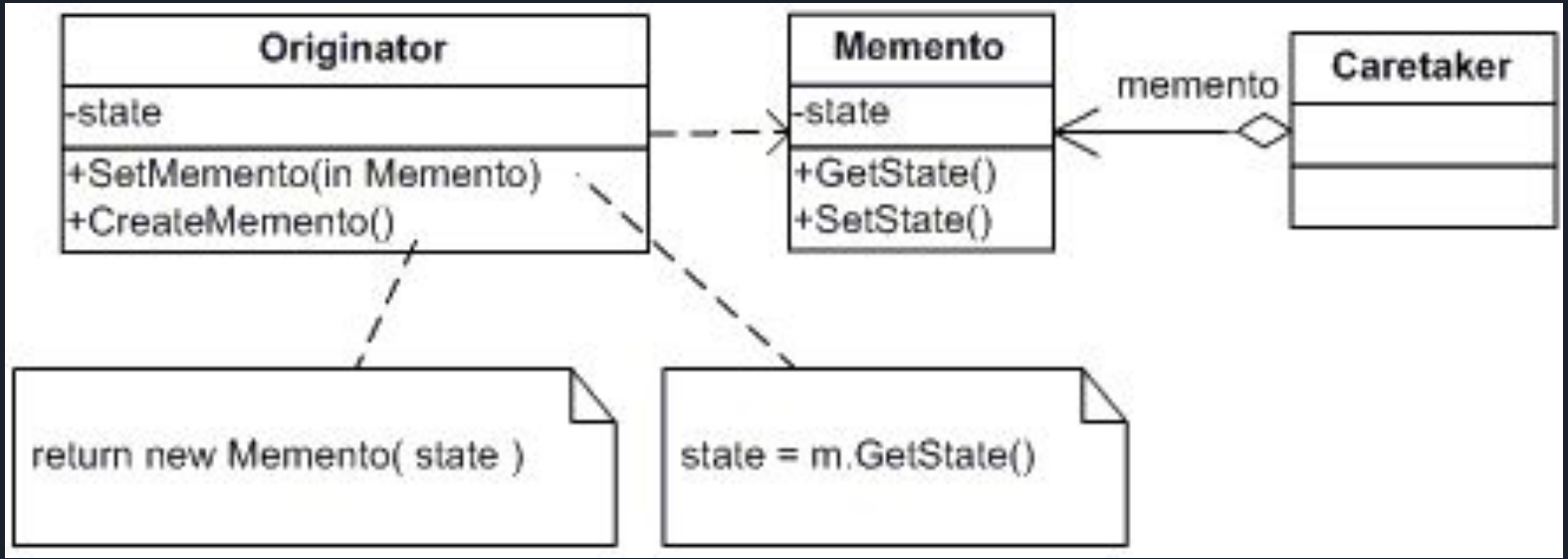
Cuidador(Caretaker): El cuidador es responsable de mantener una lista de mementos y, en algunos casos, puede ser el encargado de decidir cuándo se captura un memento y cuándo se restaura elestado del originador desde un memento específico.

VENTAJAS DE MEMENTO

Gestión de estados: Permite la captura y restauración de estados anteriores de objetos, lo que es útil para implementar funciones de deshacer (undo) o para mantener un historial de cambios.

Separación de preocupaciones: Aísla la responsabilidad de la gestión del estado, lo que permite que el originador se centre en sus operaciones principales y que el cuidador se encargue del historial de estados.

DEMOSTRACIÓN



EJEMPLO

Imaginemos una aplicación de procesamiento de texto en la que el usuario puede realizar cambios en un documento de texto y deshacer esos cambios si es necesario. El patrón Memento podría utilizarse para capturar el estado del documento antes de cada modificación y almacenarlo en una lista de mementos. Cuando el usuario desee deshacer una acción, se puede restaurar el estado anterior del documento a partir de uno de los mementos almacenados.

CONCLUSIÓN

Los patrones de diseño son soluciones probadas y documentadas para problemas comunes en el diseño de software. Ofrecen beneficios clave como modularidad, reutilización y mantenibilidad del código. Sin embargo, su aplicación debe ser selectiva y basada en las necesidades específicas del proyecto. Estos patrones representan un valioso recurso para mejorar la calidad y eficiencia del desarrollo de software.

FIN