
PATRON
FLYWEIGHT
DISEÑO DE SISTEMAS
START

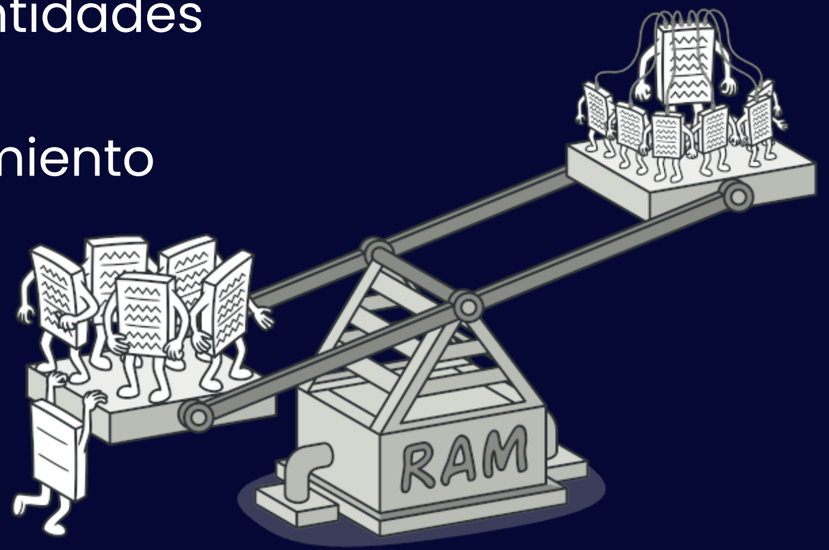
CONTENIDO

01
INTRODUCCIÓN

- Flyweight es un patrón de diseño estructural que te permite mantener más objetos dentro de la cantidad disponible de RAM compartiendo las partes comunes del estado entre varios objetos en lugar de mantener toda la información en cada objeto.

- Permite compartir información que se encuentra en pequeños objetos que existen en grandes cantidades

- Ayuda recudir las necesidades de almacenamiento



02
ESTADOS

ESTADO INTRÍNSECO

- SE PUEDE COMPARTIR A GRAN ESCALA,DISMINUYENDO EL ALMACENAMIENTO
- HACEN REFERENCIA A LOS ESTADOS COMUNES QUE TIENE EL OBJETO O GRUPO DE OBJETO A REPLICAR

ESTADO EXTRÍNSECO

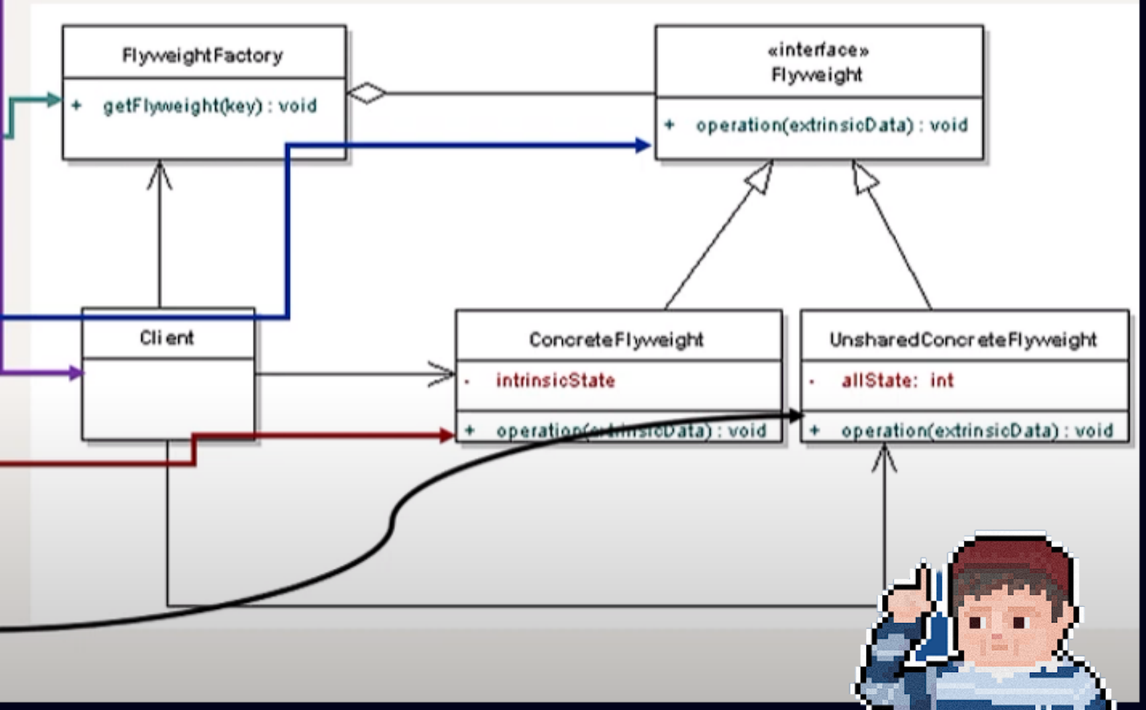
- ALUDEN A LAS CARACTERISTICAS PROPIAS DE LA INSTANCIA
- EL ESTADO SE CALCULA EN EL MOMENTO

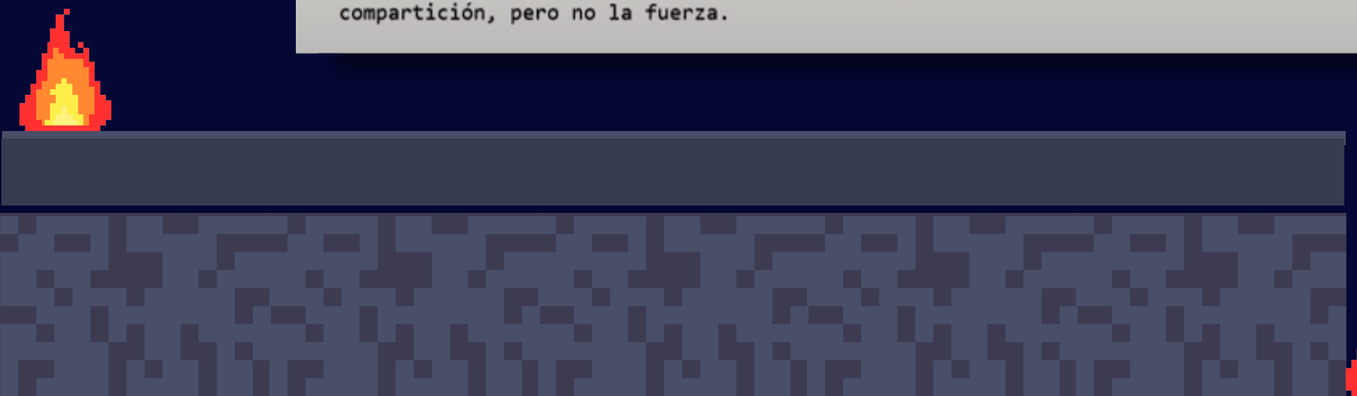
EL OBJETIVO DE ESTE PATRON ES ESTUDIAR LAS CARACTERISTICAS COMUNES DE LOS OBJETOS Y CREAR UNA CLASE EXTRINSECA. LUEGO A LA HORA DE REALIZAR LAS DIFERENTES INSTANCIAS, USA LA PARTE COMUN (INTRINSECA) Y LA COMPLEMENTA CON LOS DATOS ESPECIFICOS DE LA INSTANCIA

03

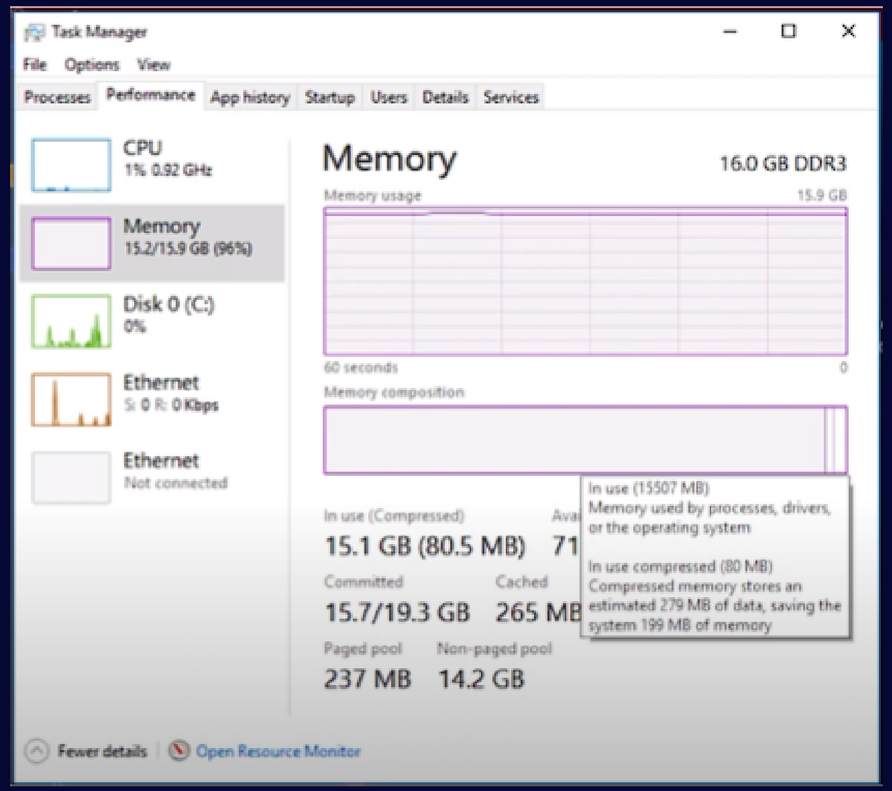
ESTRUCTURA

- Client: Trabaja con una referencia a un Flyweight. Establece los estados extrínsecos de los objEtos.
- FlyweightFactory: Crea y maneja objetos flyweight asegurándose que se compartan adecuadamente. Cuando el cliente solicita un flyweight, FlyweightFactory proporciona una instancia existente y si no existe la crea.
- Flyweight: Interfaz que contiene métodos para el establecimiento, acceso y modificación de las propiedades extrínsecas. Deberá ser implementada por las instancias del Flyweight.
- ConcreteFlyweight: Implementa la interfaz Flyweight y añade el almacenamiento de las características intrínsecas (si las hay) y deben poder ser compartidos.
- UnsharedConcreteFlyweight: Todos los Flyweight no tienen por qué ser compartidos, la implementación de la interfaz habilita la compartición, pero no la fuerza.





04 VENTAJAS

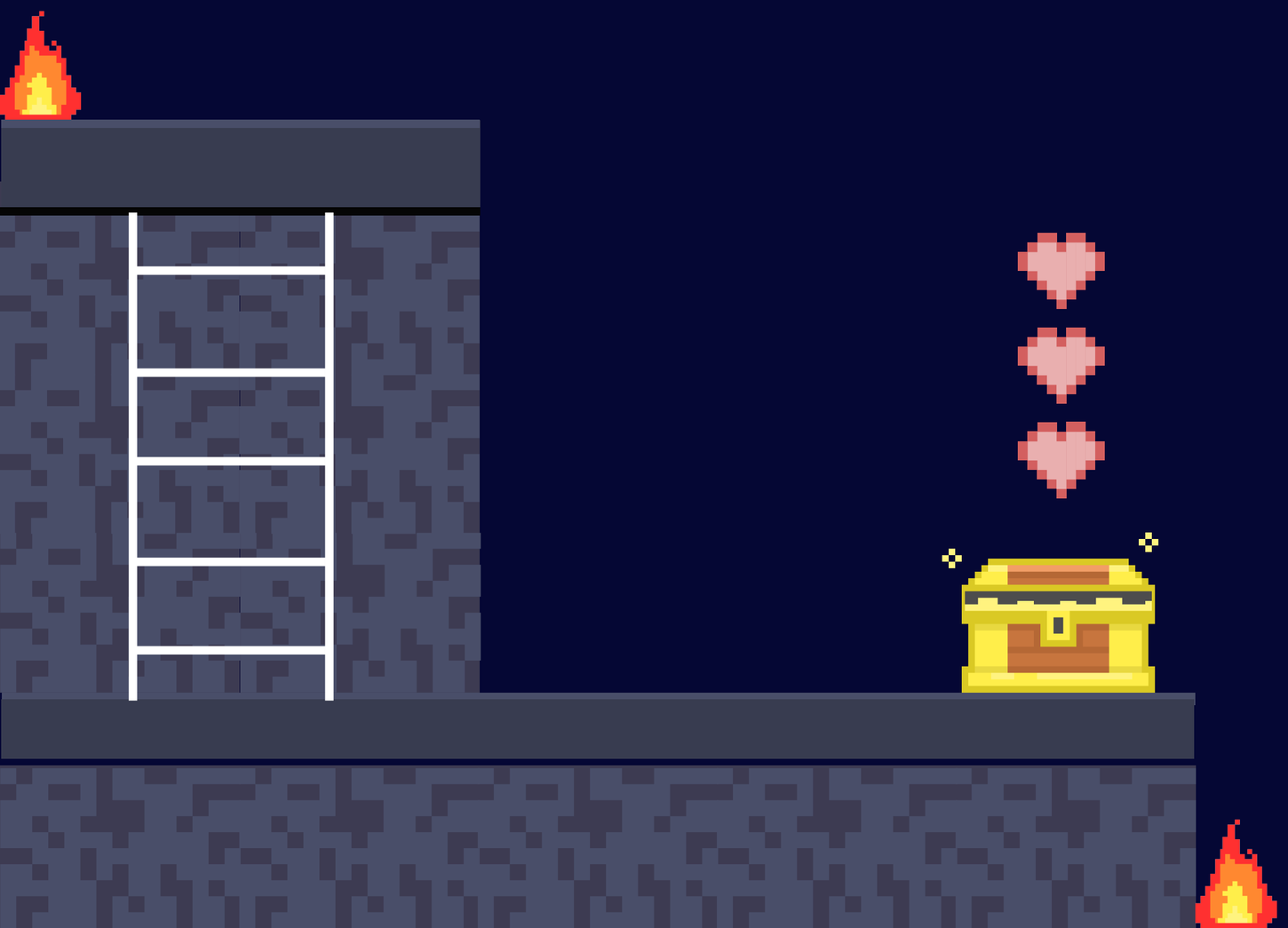


- Menos uso de memoria derivado de la reutilizacion de objetos ya existentes que comparten mismos estados intrinsecos

- Mejora en la eficiencia, sobre todo en relación con el ahorro en el almacenamiento, que aumenta paralelamente con la compartición de objetos y es función de:

(a) la reducción en el número total de instancias.
(b) la cantidad de estado intrínseco por objeto.
(c) si el estado extrínseco es computado (calculado) o almacenado.

05
DESVENTAJAS



ANTECEDENTE 01

- Se introducen costes en tiempo de ejecución asociados a las transferencias, búsquedas y/o computación del estado extrínseco (y su almacenamiento).

ANTECEDENTE 02

- En algunos casos, al mover el estado afuera del objeto puede quebrar la encapsulacion y puede ser menos eficiente que mantener el objeto completamente intrinsico

06
EJEMPLOS DE USO

Principalmente cuando no tenemos los suficientes recursos para generar una gran cantidad de objetos

Como en este caso, que se generan un millon de arboles en donde crear cada uno de los objetos va a requerir una gran cantidad de recursos

Siendo mejor utilizar el patron flyweigth para disminuir el uso de estos recursos