Universidad Nacional Autónoma de México

Facultad De Ciencias

Modelado y Programación

Trabajo de exposición: Patrón de diseño flyweight

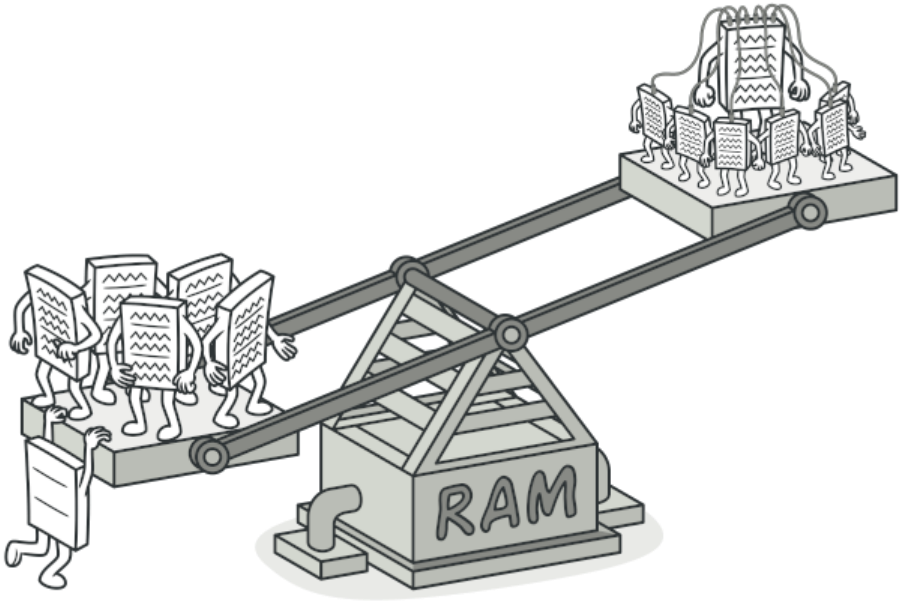
Balderas Salomón Isay Damar

419002952

¿Qué hace el patrón de diseño flyweight?

Centra su atención en la construcción de objetos ligeros, mediante la abstracción de las partes reutilizables que pueden ser compartidas con otros objetos, esto con el fin de que en lugar de crear objetos cada vez que sean requeridos, podamos reutilizar objetos creados por otras instancias logrando con ello reducir en gran medida la capacidad de memoria requerida por la aplicación.

Nos permite "acomodar" más objetos en la RAM disponible compartiendo partes en común de los objetos.



¿Para qué sirve el patrón de diseño?

Sirve para la optimización de recursos en cuanto a memoria eliminando la redundancia de objetos con propiedades idénticas.

Principalmente se ve el impacto del diseño cuando se refiere a videojuegos, ya que, por ejemplo, en un juego del género shooter, los proyectiles pueden ser objetos, y dado cierto momento del juego, llega a haber un montón de estos en la partida, haciendo que nuestro juego utilice mucha memoria RAM, lo cual puede ser conveniente y manejable para ciertas computadoras potentes en cuanto a RAM, pero no para otras, y para evitar que los usuarios con computadoras no tan potentes en cuanto a RAM tengan la misma experiencia, utilizamos este patrón de diseño.

Diagrama de clase de flyweight

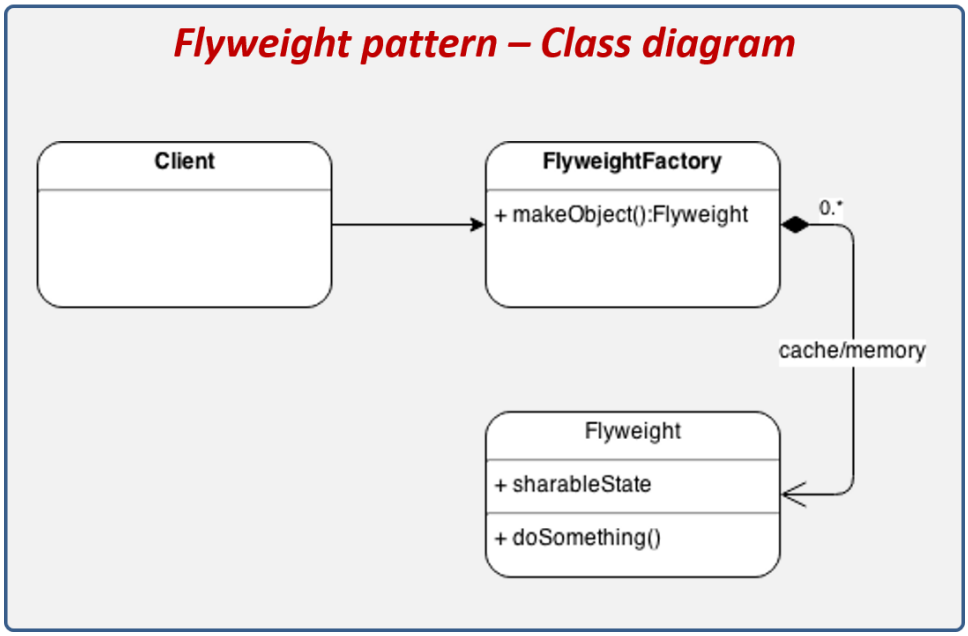
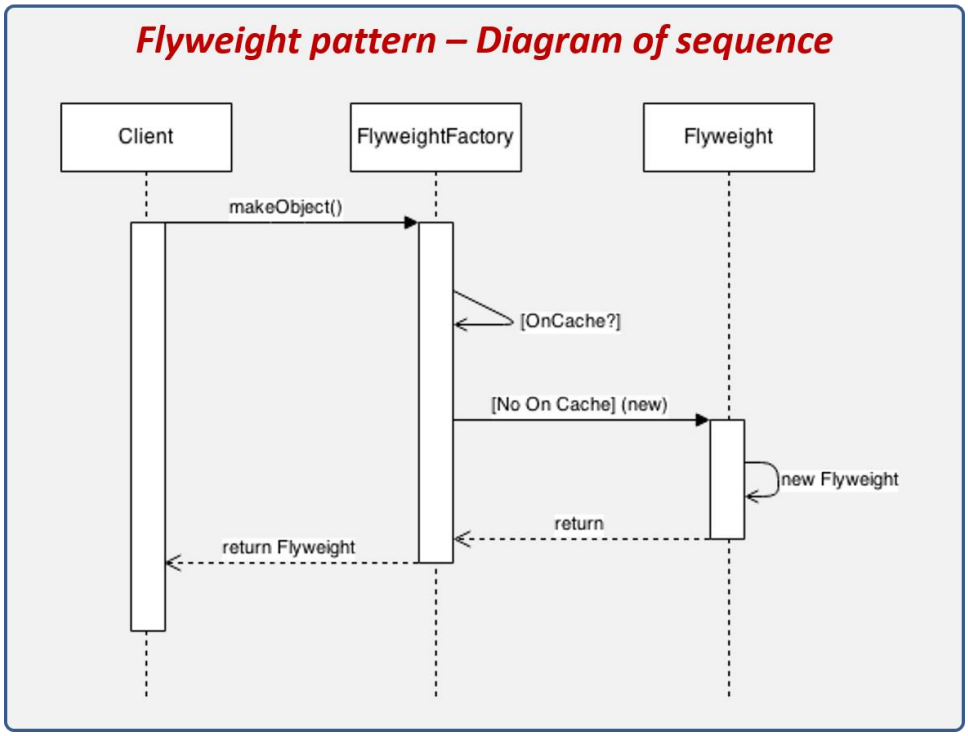


Diagrama de secuencia de flyweight



Descripción

Para el ejemplo de programa, se utilizó un ejemplo de construcción de un bosque con dos tipos de árbol, sabemos que abstractamente en un bosque (para este ejemplo) los árboles son fundamentalmente iguales en cuanto a la a su color, a su textura, etc., lo único que varía es su posición en el bosque (sus coordenadas). La utilización del patrón en este ejemplo es que sus atributos únicos de cada árbol en cuestión son sus coordenadas, mientras el color, la textura, etc., son compartidas entre los mismos tipos de árbol.

Para este ejemplo se utilizaron las bibliotecas de swing y awt, para representar gráficamente la construcción del bosque, además de eso, se utilizó un diccionario dentro de TreeFactory para ir guardando y recuperando los árboles que el programa vaya creando, siendo también esta clase nuestra Flyweight Factory.

La clase Tree son los árboles en cuestión que se crea, con su parte TreeType, que vendría siendo nuestro flyweight del programa (ya que comparte las partes idénticas de los objetos).

Forest y Demo vendrían siendo el cliente, forest lo único que hace es ir "plantando" los árboles dentro de nuestra interfaz, y en Demo el usuario dice cuáles serán los árboles, cuantos serán y el tamaño del bosque (en cuanto al tamaño de la ventana).

En cuanto a los métodos de paint y draw dentro de las diversas clases, solo son auxiliares que se utilizaron para representar gráficamente el programa y se viera más claramente el programa.

Código del ejemplo

El código se agrega en forma de archivos .java, correr Demo de los archivos.

Utilizar el comando "java –cp ./build myp.ejemplo.Demo" estando parado en el directorio que tiene a build y src

Diagrama de clase del código

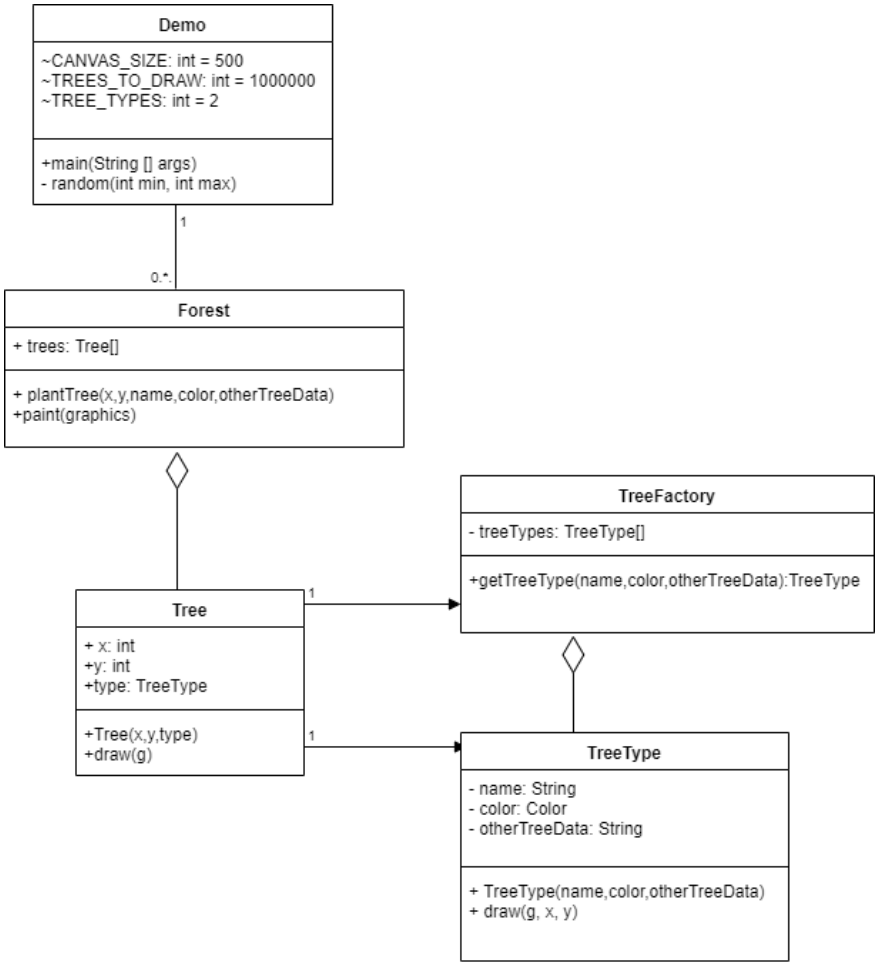


Diagrama de secuencia del código