

Tarea #1

Autores:

Josselyn Mora Santamaria

[josselyn.mora.santamaria@est.una.ac.cr](mailto:josselyn.mora.santamaria@est.una.ac.cr)

Joshua Granados Loría

[joshua.granados.loria@est.una.ac.cr](mailto:joshua.granados.loria@est.una.ac.cr)

Kevin Mora Valverde

[kevin.mora.valverde@est.una.ac.cr](mailto:kevin.mora.valverde@est.una.ac.cr)

Universidad Nacional de Costa Rica.

Curso: Programación III

II Ciclo-NCR 51172-Grupo 89

Profesor: Máster Rubén Mora Vargas

04 de octubre del 2021

Tabla de contenido

[Introducción 3](#_Toc84063250)

[Contenido 4](#_Toc84063251)

[Abtract Factory 4](#_Toc84063252)

[Flyweight 4](#_Toc84063253)

[Memento 6](#_Toc84063254)

[Conclusión 7](#_Toc84063255)

[Recomendaciones 8](#_Toc84063256)

[Bibliografías 9](#_Toc84063257)

[Anexos 10](#_Toc84063258)

# Introducción

# Contenido

## Abtract Factory

## Flyweight

Este diseño de patrón sirve para reducir el uso de memoria, esto mediante la compartición de partes en común entre varios objetos, así se reduce una gran cantidad de código al no tener que mantener la misma información en todos los objetos, este proceso va mucho de la mano con lo que es la herencia ya que cumplen una función muy similar. El diseño de patrón Flyweight tiene la siguiente estructura:

1. Se tiene que entender que Flyweight es una manera de optimizar el código de la aplicación que se este creando, se suele implementar cuando el programa creado consume mucha RAM y no se logra disminuir ese consumo de otra manera.
2. Clase Flyweight: En esta clase se encuentra los estados del objeto original del cual los objetos hijos van a heredar, este estado que se almacena dentro del objeto Flyweight se le denomina como intrínseco y los que se les pasa a sus métodos se denomina extrínseco.
3. Clase Contexto: Dentro de esta clase se encuentra el estado extrínseco, este estado es único entre todos los objetos originales, en el momento en el que un contexto se logra unir con uno de los objetos Flyweight se logra un estado completo del objeto original.
4. Cuando se invoque un método de un objeto de tipo Flyweight se debe de pasar aquellas partes del estado que sean útiles mediante los parámetros del método, además los métodos que le dan un comportamiento al objeto Flyweight se pueden utilizar en la clase de contexto.
5. Clase cliente: En esta se va a realizar todo calculo o almacenamiento del estado extrínseco de los distintos objetos Flyweight. Desde una vista del propio cliente, un Flyweight es un tipo de objeto de tipo plantilla el cual puede ser configurado durante la ejecución del programa, esto pasa porque envía información contextual dentro de los parámetros de sus distintivos métodos.

1. Clase Fabrica Flyweight: Mediante esta clase se gestionan los grupos existentes de objetos Flyweight, por lo tanto, la clase cliente no se hace cargo de crear directamente los objetos Flyweight, si no que llama a la fabrica y se le envían los parámetros que se necesite del estado intrínseco. La fábrica hace una revisión de los objetos Flyweight que se crearon previamente y devuelve uno existente que pueda coincidir con los parámetros enviados y si no llega a encontrar ninguno entonces crea uno nuevo.

En la siguiente imagen se puede observar un diagrama UML con la estructura mencionada anteriormente.

Diagrama

Descripción generada automáticamente con confianza baja

Imagen#1. Diagrama UML de la estructura del patrón de diseño Flyweight

Ventajas de utilizar Flyweight:

* El ahorro del uso de la memoria RAM.

Desventajas:

* El código puede convertirse un poco complicado de entender.
* Si hay cambios de RAM por ciclos de CPU el calculo de una nueva parte de la información se realizará cada vez que se invoque un método Flyweight.

Para implementar este patrón de diseño en un código se debe de hacer de la siguiente manera:

1. Se tienen que dividir los campos de una clase para que se convierta en la clase Flyweight, esta clase se divide en dos partes:

* Estado intrínseco: Son aquellos campos que van a tener la información única de un objeto, esta va a estar duplicada a través de todos los objetos del mismo tipo.
* Estado extrínseco: Son los campos que almacenan toda información contextual que es única en cada objeto.

1. Los campos que incorporan el estado intrínseco se tienen que dejar en la clase sin embargo estos no deben de ser alterados, se tiene que asegurar que los valores iniciales se encuentren dentro del constructor.
2. Se tienen que revisar los métodos que utilizan el estado extrínseco para agregarles un nuevo parámetro y suplantar el campo.
3. Se crea la clase fabrica para encargarse de un grupo de objetos Flyweight, una vez se encuentre esta clase lista, los clientes podrán solicitar cualquier objeto Flyweight.
4. Dentro de la clase cliente se debe guardar o calcular los valores del estado extrínseco para así lograr la invocación de los métodos de los objetos Flyweight, si se desea se puede mover el estado extrínseco a la clase contexto. (Shvets, 2019)

## Memento

# Conclusión

# Recomendaciones

# Bibliografías

Shvets, A. (2019). *Flyweight*. Refactoring.Guru. <https://refactoring.guru/es/design-patterns/flyweight>

# Anexos

En el siguiente enlace de GitHub se puede encontrar un ejemplo programado dedicado a cada patrón de diseño investigado.

<https://github.com/joshua292000/Tarea-1.git>