

**Universidad Tecnológica Centroamericana**

**Ingeniería del Software I**I

**Catedrático:** Ing. Román Pineda

**Patrones de diseño**

**Alumno: Lester Eduardo Arteaga Andino 31111820**

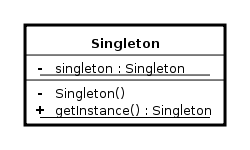
**Tegucigalpa, M. D. C.**

**Martes 01 de junio del 2020**

**Patrones de diseño**

1. **Singleton**:

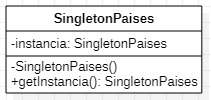
Singleton o instancia única es un patrón de diseño que permite restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase solo tenga una instancia y proporcionar un punto de acceso global a ella.



**Ejemplo de la vida real:**

Un ejemplo claro seria si queremos mostrar una lista de países. Claramente dicha lista de países es casi nula que vaya a cambiar por lo que si realizamos de forma normal este procedimiento cada usuario que se conectara al sistema generaría una nueva instancia de la clase lo cual no es eficiente ya que el sistema estaría tomando memoria innecesaria por lo que aplicando singleton la ventaja es que solo se crearía una única instancia disponible para todos los usuarios.

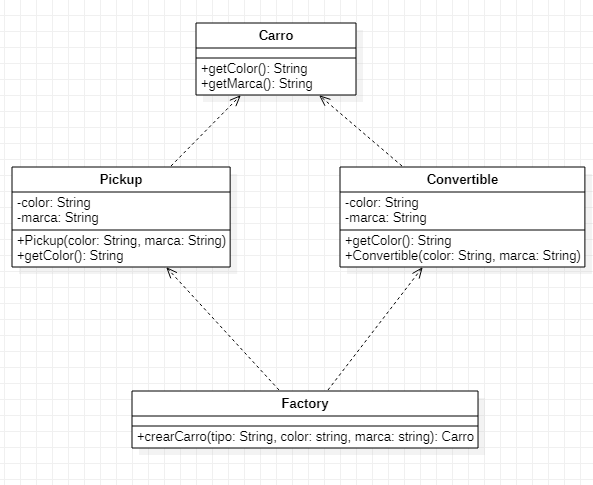
**Diagrama UML:**

****

1. **Factory**

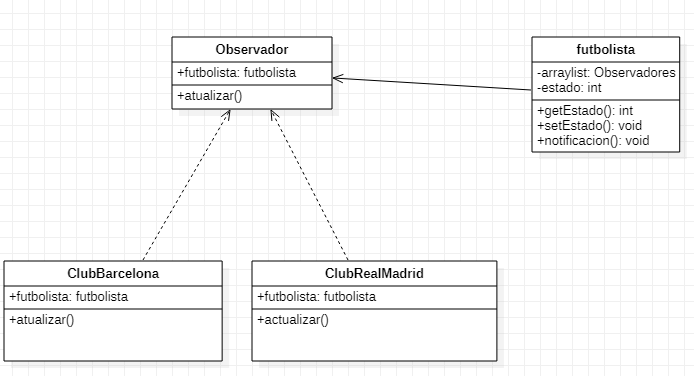
Un ejemplo real seria de este patrón sería crear un automóvil el cual se necesitan varias partes del cual además hay diferentes tipos de carros y modelos.

**Diagrama UML:**



1. **Observer**

Es un patrón muy utilizado como en frameworks como Angular. Consiste en 3 clases: sujeto, observador y el cliente.



**Ejemplo en la vida real:**

El objetivo es que cuando se hace un cambio los observadores este pendiente de los cambios que se han realizado. Mi ejemplo se basa en la contratación de un jugador entonces hay equipos que están pendientes del cambio de su precio para saber si tienen la capacidad de comprarlo o no.

1. **Adapter**
2. **Builder**
3. **State**