Patrón Singleton

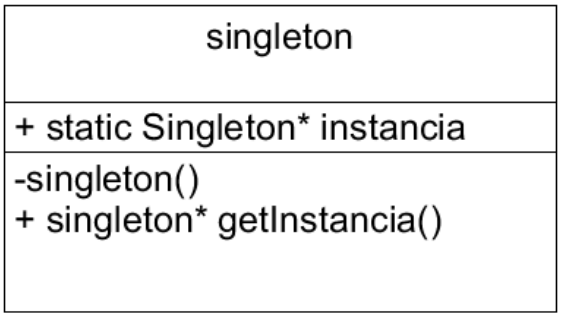
Intención:
Asegurarse de que una clase tendrá una sola instancia y proporcionar un único punto de acceso a ella. El diseño debe imposibilitar la creación accidental de varias instancias de la clase.

Ejemplo:

- ✓ Aunque puede haber muchas impresoras en un sistema, sólo puede haber una **cola de impresión**.
- ✓ El **presidente** de una compañía es un singleton, pues se supone que solo puede existir una única instancia presidente.

En este patrón la misma clase debe estar escrita de manera que sólo una instancia se puede crear, una manera común de hacer esto es:

- ✓ Declarar el constructor de clase como privado para que no sea accedido directamente.
- ✓ La propia clase es responsable de crear la única instancia.
- ✓ Se debe tener una variable global, generalmente un puntero a la única instancia.
- ✓ Se debe permitir el acceso global a dicha instancia mediante un método de clase.
- ✓ La clase tiene acceso a la variable que contiene la instancia única, y asegura que la variable se inicializa con la instancia única antes de devolver su valor. Este enfoque garantiza que un producto único se crea y se inicializa antes de su primer uso.



Observe que el constructor es privado, el método ***getInstance()*** es el que devuelve la única instancia de esta clase, o la crea si no existe.

Ejemplo N.1

Supongamos que en una empresa solo puede existir un único presidente

```
//------------------------  presidente.h ------------------------
#pragma once
#include <iostream>
#include <string>
using namespace std;

class presidente {
private:
	string    nombre;
	static presidente* instancia; // puntero a la única instancia
	presidente();//constructor privado

public:
	static presidente* getInstancia();
	string  getNombre();
	void setNombre(string n);
};


//------------------------  presidente.cpp ------------------------
#include "Presidente.h"

presidente* presidente::instancia = NULL;


presidente::presidente() {  // constructor privado
	nombre = "Sin definir";
	cout << "Se ha creado un  objeto" << endl;
}

string  presidente::getNombre() { return nombre; }
void presidente::setNombre(string n) { nombre = n; }


presidente* presidente::getInstancia() {  // método que crea la única instancia
	if (!instancia)
		instancia = new presidente();
	return instancia;

}



//------------------------  main.cpp ------------------------

#include "Presidente.h"

void  main() {
	presidente * p1 = presidente::getInstancia();
	cout << "El nombre del presidente p1: " << p1->getNombre() << '\n';
	p1->setNombre("Juan");
	cout << "El nombre del presidente p1: " << p1->getNombre() << '\n';

	presidente * p2 = presidente::getInstancia();
```

```
        cout << "El nombre del presidente p2: " << p2->getNombre() << '\n';

        p1 = presidente::getInstancia();
        p1->setNombre("Carlos");
        cout << "El nombre del presidente p1: " << p1->getNombre() << '\n';

        system("pause");
}
```