El Patrón de Diseño Singleton

Un único objeto siempre

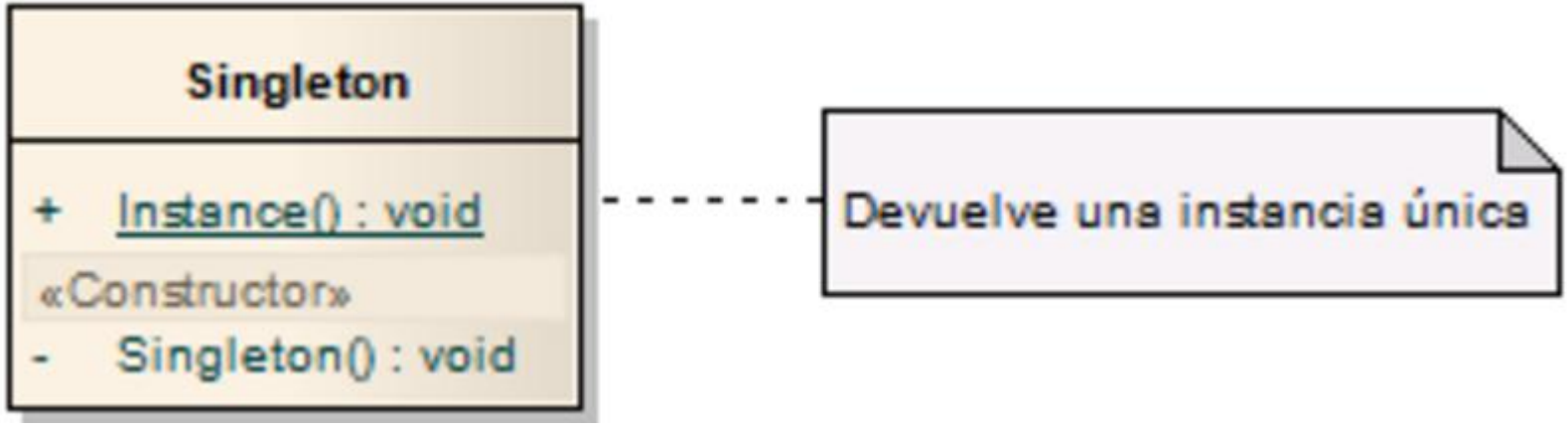
En un futuro cercano veremos la definición de Patrón de Diseño

😬

El Patrón Singleton

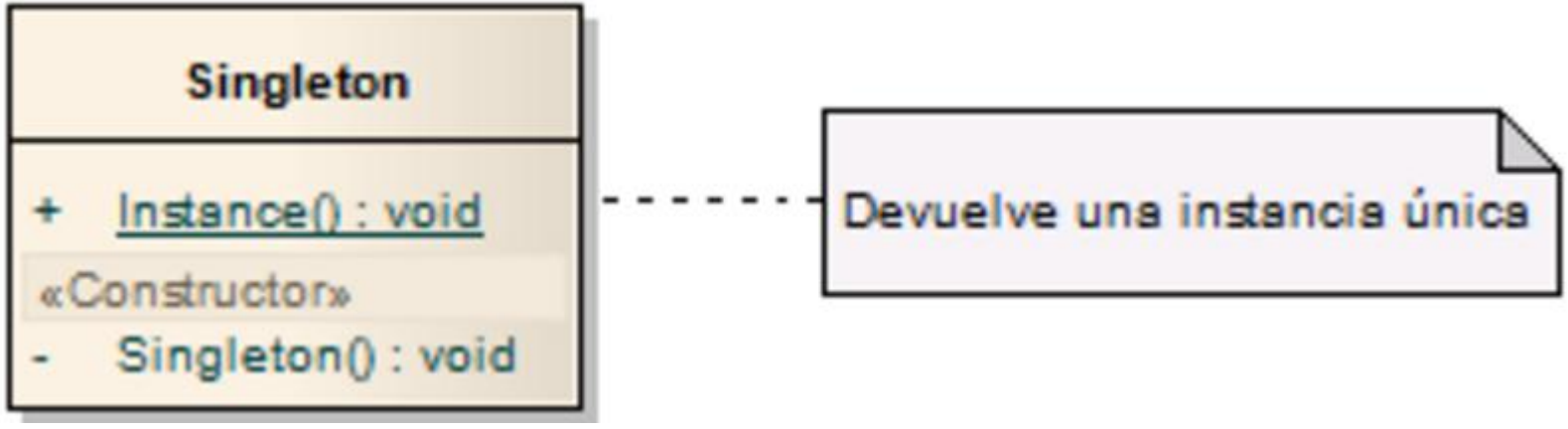
Objetivo:

Asegurarse de que una clase tiene una única instancia, proporcionando acceso global a ella.



El Patrón Singleton

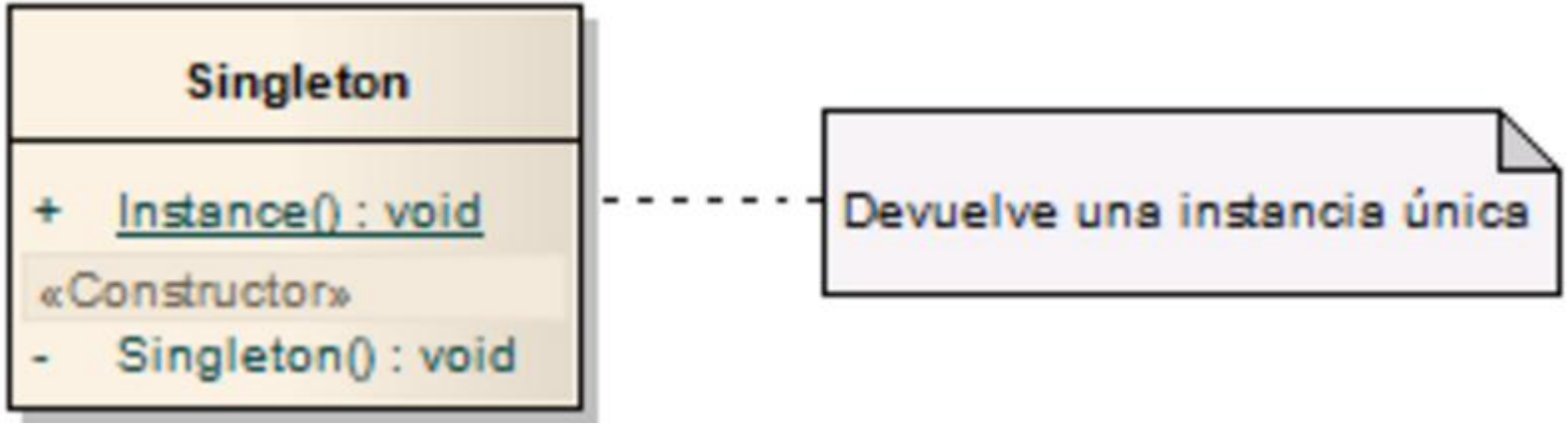
Hay clases que deben **instanciarse una única vez**. El acceso a un sistema de archivos, a la cola de impresión o al gestor de ventanas del sistema operativo debería realizarse por un único objeto, siendo labor de la propia clase el controlar que la instancia sea única. Por norma general, esta clase será accesible de forma global, y el proceso de instanciado no suele requerir parámetros.



El Patrón Singleton

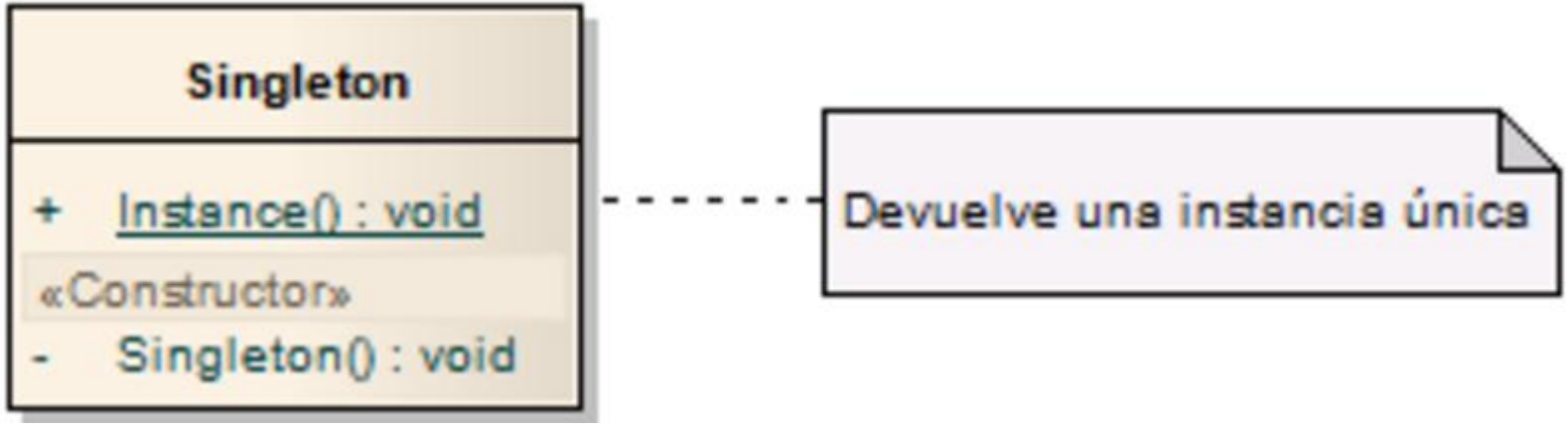
Singleton constará al menos con:

- Un método Instance() de carácter estático (método de clase) que se encargará de instanciar la clase.
- Un constructor privado que evitará que se creen nuevos objetos mediante new(), haciendo que el método Instance() sea el único que puede generar la instancia.



El Patrón Singleton

- La clase Singleton contará con un atributo de la propia clase Singleton, de carácter privado.
- El constructor (privado) se encargará de construir el objeto.
- El método estático Instance() realizará dos operaciones:
  - Comprobar si el atributo es null. En ese caso, se invocará al constructor. En caso contrario, se devolverá el objeto existente.




```java
public class Singleton
{
    // Declaramos un atributo del mismo tipo de la clase con carácter estático
    private static Singleton _instancia = null;

    public String nombre;
    public Date horaArranque;

    // Constructor privado. Únicamente puede ser invocado desde el interior
    // de la propia clase
    private Singleton()
    {
        nombre = "Patrón Singleton";
        horaArranque = new Date();
    }

    public static Singleton getInstancia()
    {
        // Si el singleton no ha sido creado previamente, se instancia.
        // En caso contrario, se devolvera el que haya sido creado previamente
        if (_instancia == null)
            _instancia = new Singleton();

        // Se devuelve la instancia
        return _instancia;
    }
```

```java
public class Prueba
{
    public static void main(String[] args)
    {
        // Instanciamos el Singleton
        Singleton s = Singleton.getInstancia();

        // Hacemos una pausa de tres segundos
        try
        {
            Thread.sleep(3000);
        }
        catch (InterruptedException e)
        {
        }

        // Intentamos instanciar un segundo Singleton
        Singleton s2 = Singleton.getInstancia();

        // Comprobamos que ambos objetos son referencias a la misma
        // instancia, que es única
        System.out.println(s.getNombre() + s.getHoraArranque());
        System.out.println(s2.getNombre() + s.getHoraArranque());

    }
}
```

```
Patrón SingletonTue May 17 09:16:53 ART 2016
Patrón SingletonTue May 17 09:16:53 ART 2016
```