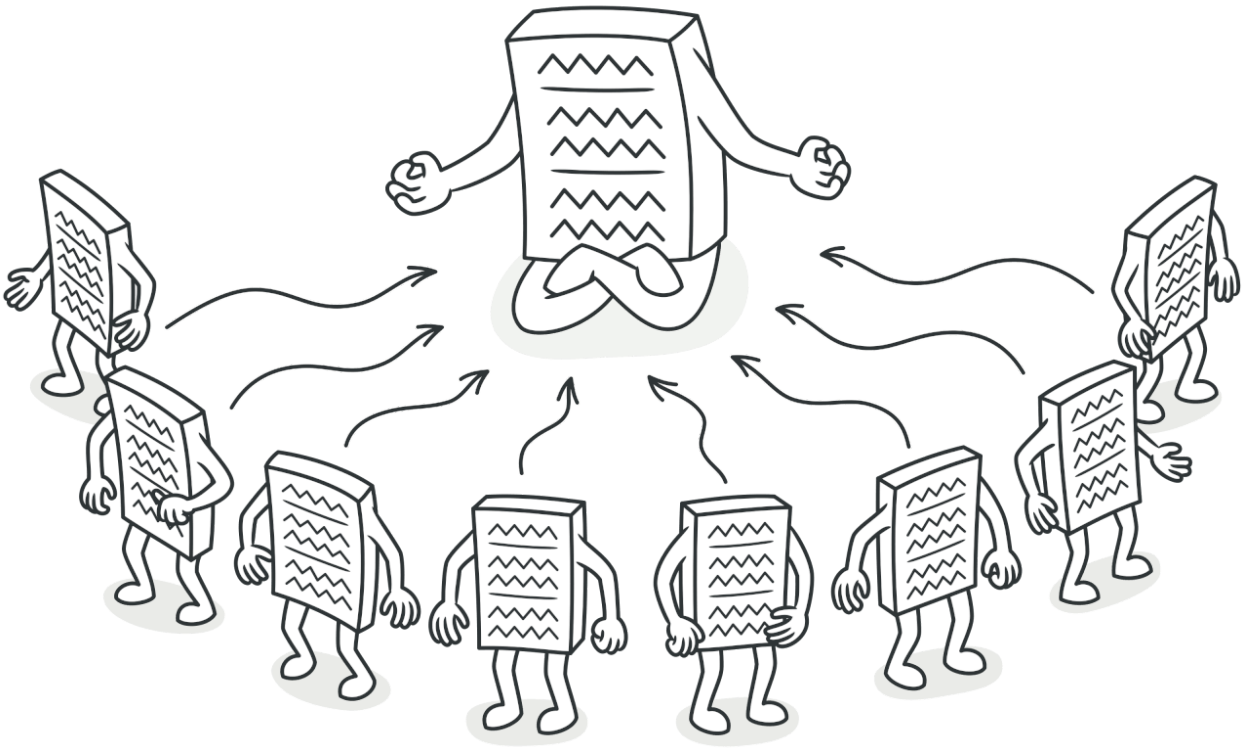
Singleton

Creational design pattern



Nona ghazizadeh

Shayan mohammadizadeh

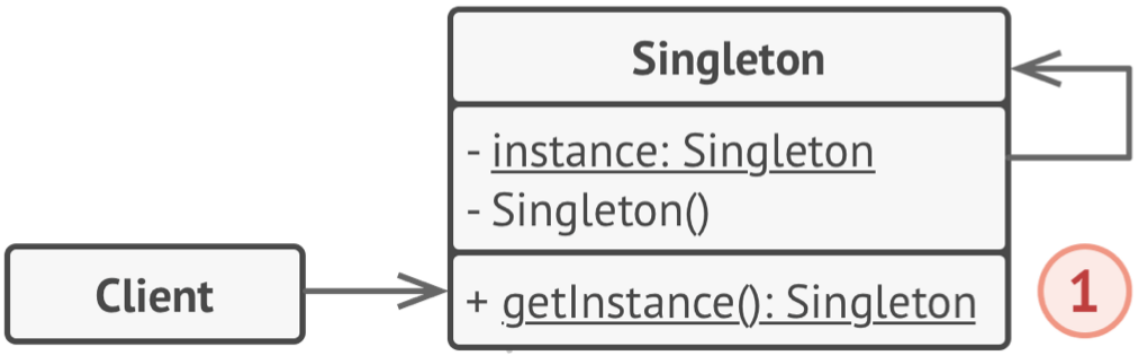
مقدمه

الگوی طراحی singleton که در لغت به معنی یگانه بودن است بدین گونه است که تنها اجازه گرفتن یک نمونه شی را به کاربر می‌دهد و به این نمونه شی گرفته شده دسترسی سراسری می‌دهد.

کاربرد

فرض کنید در یک نرم‌افزاری که در حال پیاده‌سازی هستید نیاز به ارتباط با پایگاه داده دارید بدین ترتیب یک کلاسی برای ارتباط با پایگاه داده پیاده‌سازی کرده‌اید. در صورتی که چندین instance از این کلاس بگیرید اتفاقی که می‌افتد مشابه ضرب المثل « آشپز که دو تا شد آش یا شور می‌شه یا بی‌نمک » است یعنی در صورتی که دو یا بیشتر شی از این کلاس داشته باشیم مسئله تداخل رخ می‌دهد و واضح است که تغییراتی که مطابق خواسته ما نمی‌باشد رخ می‌دهد. در کنار این مسئله می‌توان به این نکته اشاره کرد که instance گرفتن از این کلاس هزینه‌بر و زمان‌بر است. بنابراین در صورت استفاده از این طراحی برای برقراری ارتباط با پایگاه داده نیاز به تنها یکبار instance گرفتن از این کلاس داریم

ساختار



1. یک تابع استاتیک بدون ورودی به صورت public برای instance گرفتن (معمولاً با نام getInstance) از کلاس‌مان تعریف می‌کنیم
   باید constructor کلاس‌مان را private کنیم، این کار را برای آن انجام می‌دهیم که بیش از یکبار که آن هم در همان کلاس‌مان با استفاده از تابع getInstance انجام می‌دهیمقادر به instance گرفتن نباشیم و همانطور که قبل تر توضیح داده شد امکان دسترسی سراسری به این نمونه شی گرفته شده داریم.

سودو کد

```
Class Singleton:

   Private static instance;

   Private constructor():
      // ...


   Public static getInstance():
      If (this.instance == null):
         this.instance = new Singleton;

      Return this.instance;
```

خوبی(ها) و بدی(ها)

😃 شی ساخته شده از کلاس singleton تنها یک عدد است و تنها یکبار مقدار دهی اولیه می‌شود.

😃 دسترسی سراسری به شی ساخته شده داریم.

😢 دیباگ کردن کد در صورت استفاده از singleton به دلیل دسترسی سراسری که ایجاد می‌کند، دشوار است.

😢 در صورت استفاده از کلاس سینگلتون امکان پیاده‌سازی abstract class و sub class را نداریم.