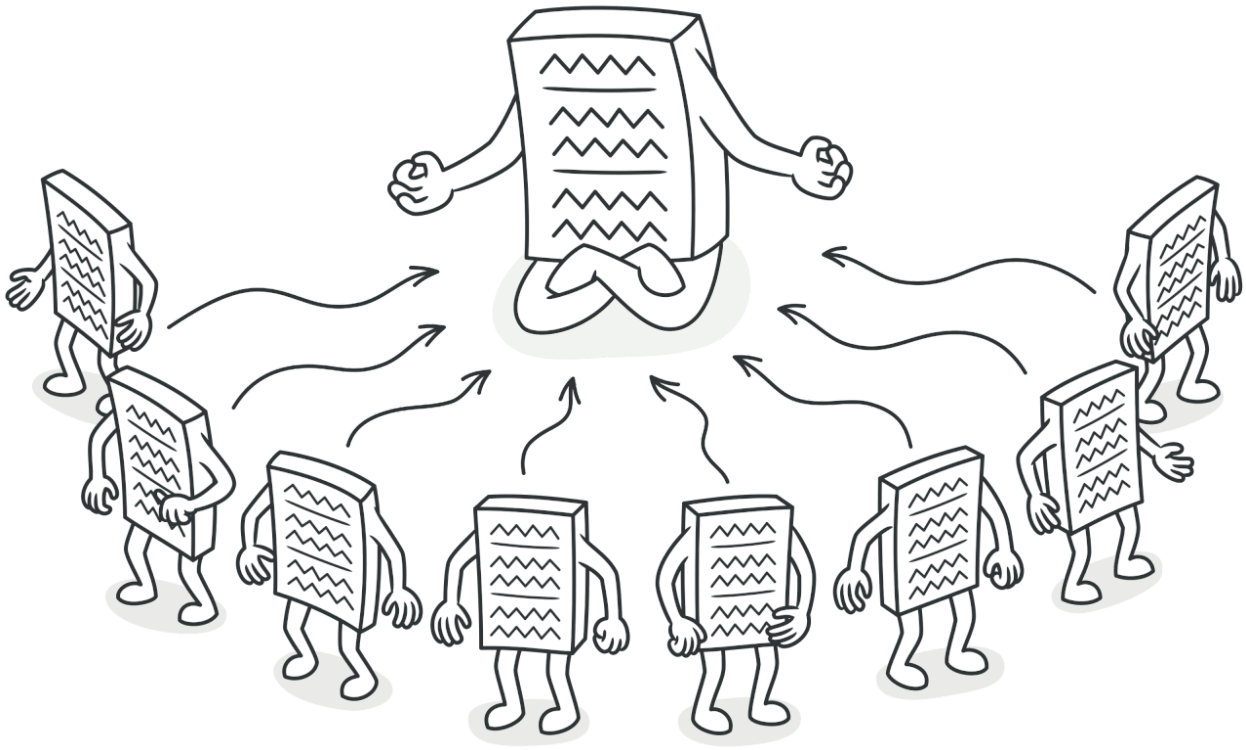
Prototype

Creational design pattern



Nona Ghazizadeh

Shayan Mohammadizadeh

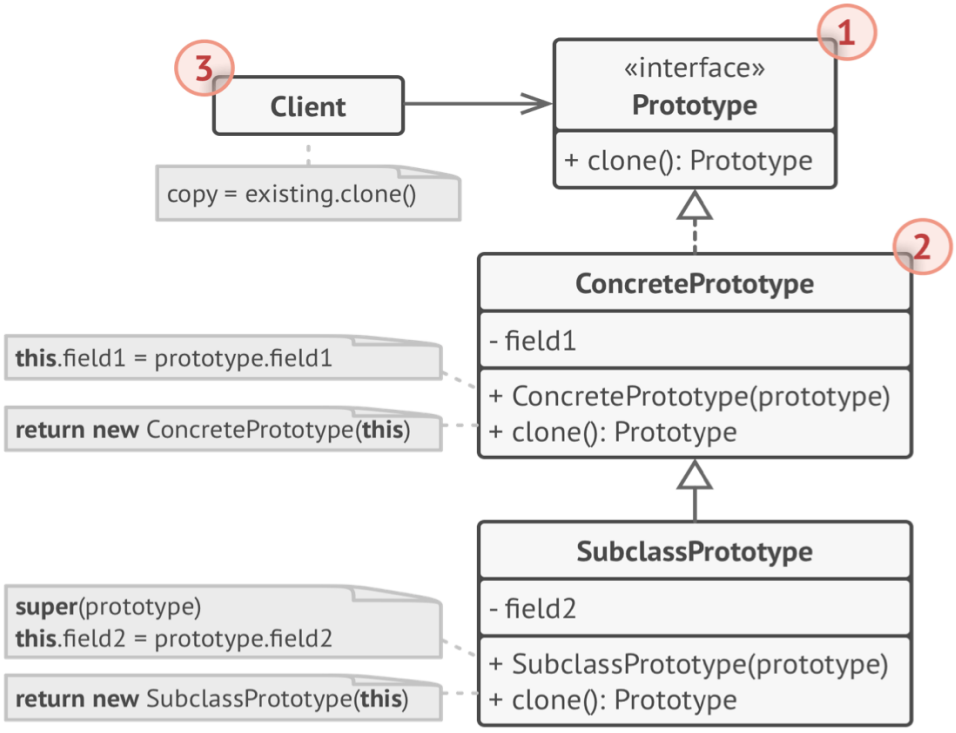
مقدمه

الگوی طراحی prototype که در لغت به معنی نمونه اولیه است بدین گونه است که به ما امکان کلون کردن از یک نمونه را می‌دهد به عبارتی دیگر نمونه‌ی جدیدی از روی نمونه‌های موجود بسازیم در این صورت دیگر نیازی به ساخت یک شیء جدید با استفاده از new نمی‌باشد.

کاربرد

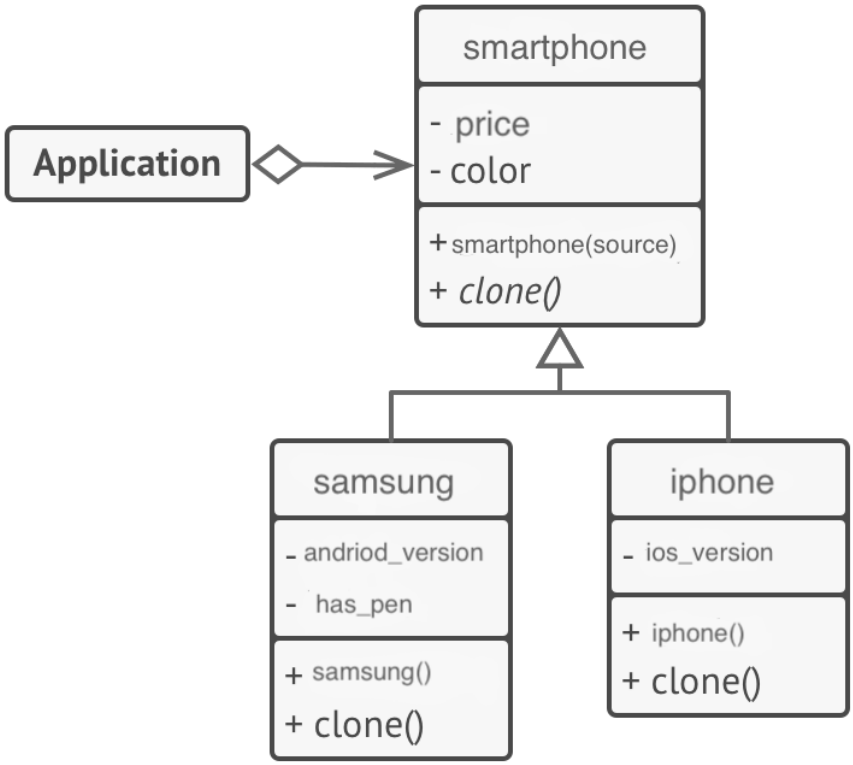
فرض کنید یک شیء تحت عنوان گوشی هوشمند دارید و می‌خواهید یک نمونه مشابه آن بسازید، بنابراین باید یک نمونه شیء جدید از کلاس ابتدایی بسازید و تمام فیلدهای شیء جدید را مشابه شیء اصلی مقداردهی کنید اما مشکل اصلی اینجاست که ممکن است برخی از این فیلدها private باشند و از بیرون نتوان به آنها دسترسی داشت که بخواهیم آنها را تغییر دهیم. یکی دیگر از مشکلاتی که هنگام کپی کردن به وجود میاید این است که ما شیء کپی شده را بر پایه کلاس شیء ابتدایی می‌سازیم بنابراین این شیء جدید به کلاس وابسته است و از آنجایی که ما همواره به دنبال کاهش وابستگی‌ها هستیم، کپی کردن مناسب نیست. بنابراین برای این که یک گوشی هوشمند مشابه شیء اولیه داشته باشیم یک interface مشترک برای تمام تلفن همراه‌ها که نیاز داریم تا کپی کنیم در نظر می‌گیریم که دارای یک تابع برای کلون کردن است بنابراین در صورتی که تمام کلاس‌های گوشی همراه از این interface استفاده کنند صرفا لازم است تا تابع کلون را در آنها پیاده‌سازی کنیم، که این همان الگوی طراحی prototype است.

ساختار



1. Interface که با نام prototype است تنها یک تابع clone را مشخص می‌کند.
2. کلاس ConcretePrototype تابع clone را پیاده‌سازی می‌کند، همچنین از رخ دادن برخی اتفاقات که مورد انتظار ما نمی‌باشد جلوگیری می‌کند.
3. client می‌تواند کپی از هر شیء که کلاسش از این interface استفاده می‌کنند، را بگیرد.

سودو کد




```
Class smartphone:
   Price: int;
   Color: string;

   constructor smartphone();
   clone():smartphone;

Class iphone extends smartphone:
   Ios_version: int;

   Func clone:
      Return new iphone(this);
```

```
Class samsung extends smartphone:
   Android_version: int;
   Has_pen: boolean;

   Func clone:
      Return new samsung(this);

Class application:
   iphone iphone_new = new iphone();
   new_iphone.ios_version = 14;

   clone_iphone = new_iphone.clone();
```

خوبی(ها) و بدی(ها)

😀 کلاس‌های پیچیده‌ای که ساخت شیء از آنها زمان‌بر و پرهزینه است دیگر برای داشتن شیء مشابه نیازی به استفاده از new نمی‌باشد.

😀 بخش client نیازی به وابسته کردن کد خود به کلاسی را ندارد

😢 کپی کردن از یک شیء که فیلدهای زیاد و وابسته به هم دارد دشوار است.