بسم الله الرحمن الرحیم

نام دوره: دوره ۴۳ مکتب جاوا آقایان

نام و نام خانوادگی: حسین حبیبی

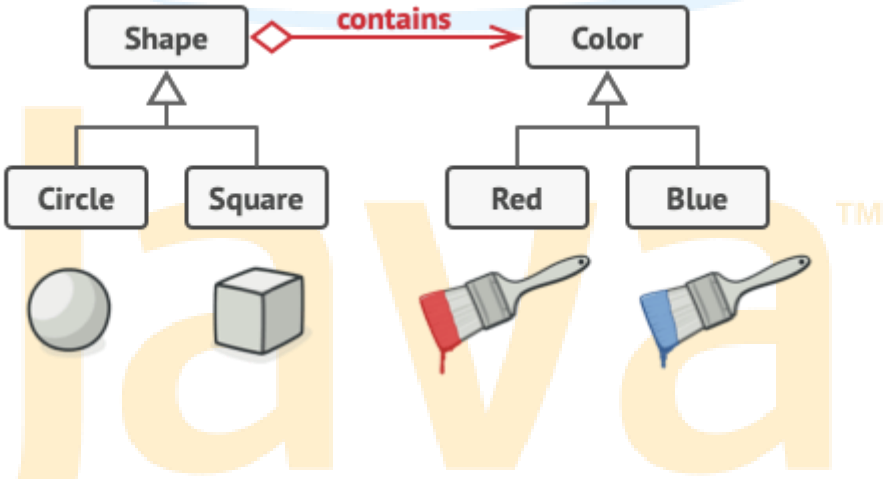
موضوع: حل تمرین سری هجدهم

Bridge design pattern

این الگو زیرمجموعه ی الگوهای ساختاری (Structural) هست. این الگو انتزاع را از پیاده سازی جدا می کند، بنابراین می توانند مستقل از هم باشند. به عبارت دیگه این نوع الگو، از اون جایی که کلاسِ پیاده سازی شده رو جداسازی و کلاسی را با ارثه ی ساختارِ Bridge بین اونا abstract میکنه، به این نام معروف شده.

در حقیقت وقتی که نیاز داریم انتزاع (Abstraction) رو از پیاده سازی (Implementation) مجزا کنیم، از Bridge استفاده میکنیم. بصورت کلی زمانی که کلاس ها کاری که انجام میدن نسبت به همدیگه تفاوتای زیادی داشته باشن، کلاس ها مدام در حال تغییر باشن، یا مواردی مشابه، از الگوی پل استفاده میکنیم. در این الگو خودِ کلاس به عنوان انتزاع در نظر گرفته و کاری که انجام میده در مرحله پیاده سازی مشخص میشه. همچنین الگوی پل میتونه به عنوان دو لایه از انتزاع در نظر گرفته بشه. این الگو از چهار قسمت اصلی تشکیل میشه:

- کلاسی Abstract که با client سرو کار داره و ارجاع یک شی از نوع Implementor را نگهداری می کند.
- کلاس RefinedAbstraction که کلاس Abstraction را توسعه میده تا چندین نسخه از متدهای abstract داشته باشه.
- اینترفیس Implementor یا Bridge که مثل یه پل یا رابط بین کلاس های abstract و کلاس های معمولی عمل میکنه.
- کلاس ConcreteImplementor که اینترفیس Implementor رو پیاده میکنه.



Observer design pattern

این الگو زیرمجموعه ی الگوهای رفتاری (Behavioral) می باشد. الگوی ناظر یا همان Observer یک الگوی طراحی نرم‌افزار است که در آن یک شی به نام موضوع (subject)، فهرست وابستگی‌هایش را با نام ناظران (observers) نگه می‌دارد و هرگونه تغییر در وضعیتش را به‌طور خودکار و معمولاً با صدا کردن یکی از روش‌های آن به اطلاع آن اشیا می‌رساند.

در حقیقت سعی داریم یه الگوی اصطلاحاً رویدادمحور رو پیاده سازی بکنیم، یعنی هربار موردی (شئ، پارامتر یا ...) تغییر میکنه بتونیم بدون صدا زدن تابع یا متد خاصی، و بصورت خودکار از این موضوع مطلع بشیم. توی این الگو، اشیائی که منتظر اعلام تغییر وضعیت هستن رو Observer و یک شئ که به بقیه اشیا تغییر رو اعلام می‌کند Subject گفته می شود.