الگوی DECORATOR: افزودن رفتار جدید به اشیا

PUBLISHED BY : FATEMEH AHMADZADEH

معرفی

DECORATOR یک الگوی ساختاری است که اجازه می‌دهد رفتار جدید به اشیا اضافه شود، بدون اینکه کلاس اصلی تغییر کند.

ویژگی‌ها:

- اشیا DECORATOR همان رابط شی اصلی را دارند.
- می‌توان چندین DECORATOR روی هم گذاشت و رفتارهای جدید اضافه کرد.
- شی اصلی هنوز همان عملکرد قبلی را دارد.

PUBLISHED BY : FATEMEH AHMADZADEH

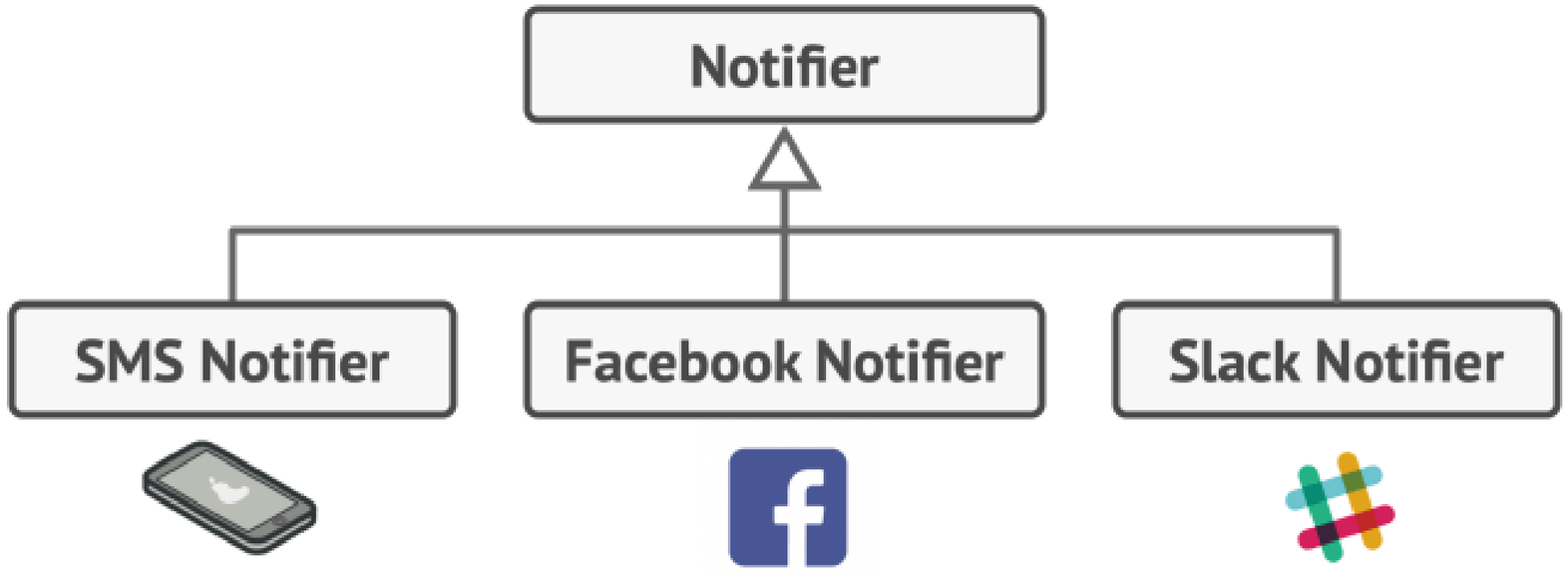
سناریو

🎬 فرض کنید یک کتابخانه اعلان داریم که پیام‌ها را به ایمیل کاربران می‌فرستد.

حالا کاربران می‌خواهند SMS، FACEBOOK و SLACK هم دریافت کنند.

- راه حل اشتباه:
  - ساخت چندین زیرکلاس: SMSNOTIFIER، FACEBOOKNOTIFIER و ...
  - اضافه کردن شرط‌های زیاد داخل کلاس اصلی

راه حل اشتباه



راه حل درست

راه‌حل DECORATOR

1. افزودن رفتار در زمان اجرا: می‌توان بدون تغییر کلاس اصلی، ویژگی یا رفتار جدید به شی اضافه کرد.
2. پیچیدن اشیا: هر شی اصلی داخل یک یا چند DECORATOR قرار می‌گیرد تا رفتار جدید داشته باشد.
3. انعطاف در ترتیب اجرا: هر DECORATOR می‌تواند قبل یا بعد از اجرای شی اصلی کاری انجام دهد.

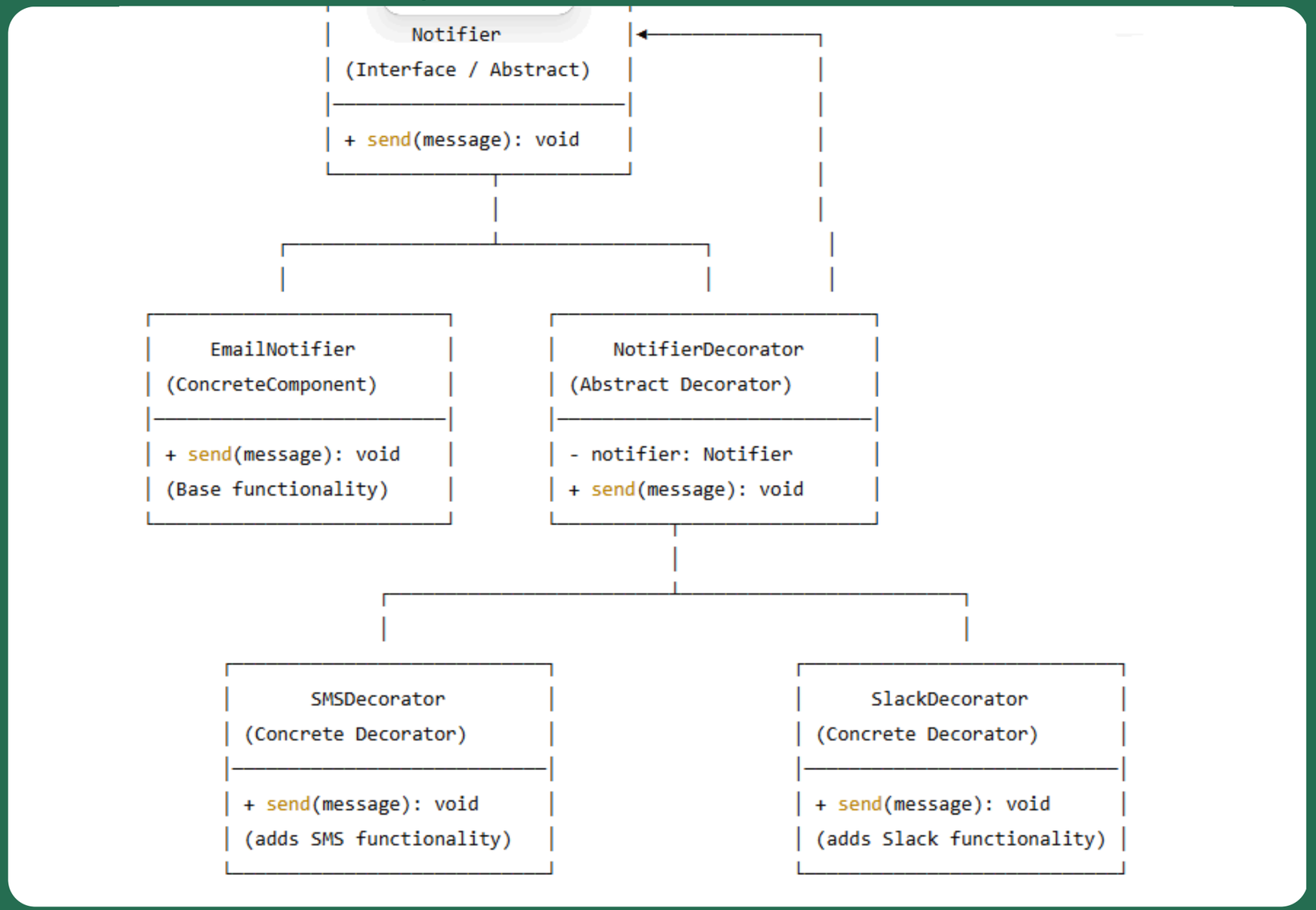
ساختار DECORATOR – مرحله به مرحله

🏗️ ساختار مرحله‌ای:

1. NOTIFIER → رابط پایه با متد SEND(MESSAGE)
2. EMAILNOTIFIER → کلاس اصلی، ارسال ایمیل
3. NOTIFIERDECORATOR → کلاس پایه DECORATOR، نگه‌دارنده یک NOTIFIER دیگر
4. SMSDECORATOR / SLACKDECORATOR → افزودن رفتار خاص

PUBLISHED BY : FATEMEH AHMADZADEH

ساختار DECORATOR – مرحله به مرحله



PUBLISHED BY : FATEMEH AHMADZADEH

کاربردهای DECORATOR

🧩 موارد کاربرد:

- افزودن رفتار در زمان اجرا :بدون تغییر کلاس اصلی، ویژگی یا قابلیت جدید اضافه می‌شود
- ساخت لایه‌های قابل استفاده مجدد :هر DECORATOR می‌تواند یک لایه رفتار جدا باشد و ترکیب‌های مختلف بسازید.

کاربردهای DECORATOR

🧩 موارد کاربرد:

- حفظ تغییر نکردن کد کلاینت : کلاس اصلی بدون تغییر، همچنان همان عملکرد قبلی را ارائه می‌دهد.
- وقتی ارث‌بری ممکن یا عملی نیست : افزودن ویژگی‌های جدید به جای ایجاد چندین زیرکلاس.

مثال‌های واقعی

👕🧥 پوشاک: تی‌شرت → ژاکت → کت

🌐 MIDDLEWARE در وب: هر MIDDLEWARE درخواست را پردازش می‌کند

📄 STREAM I/O در JAVA/PYTHON: فایل‌ها با BUFFERING، ENCRYPTION

و ... تزئین می‌شوند

مقایسه با سایر الگوها

🔗 الگو ADPATER و DECORATOR

- ADAPTER = ترجمه
- DECORATOR = افزودن قابلیت

💡 نکته:

- ADAPTER رابط را تغییر می‌دهد تا شی اصلی با محیط جدید سازگار شود.
- DECORATOR رفتار را اضافه یا تغییر می‌دهد بدون اینکه رابط اصلی تغییر کند.

مقایسه با سایر الگوها

🌳 الگو COMPOSIT و DECORATOR

- DECORATOR = بهبود یک شی
- COMPOSITE = نگهداری چند شی

💡 نکته:

- COMPOSITE برای ایجاد ساختار درختی و نگهداری چند شی استفاده می‌شود.
- DECORATOR برای افزودن یا تغییر رفتار یک شی بدون تغییر کلاس اصلی کاربرد دارد.

مقایسه با سایر الگوها

🌳 الگو COMPOSIT و DECORATOR

- DECORATOR = بهبود یک شی
- COMPOSITE = نگهداری چند شی

💡 نکته:

- COMPOSITE برای ایجاد ساختار درختی و نگهداری چند شی استفاده می‌شود.
- DECORATOR برای افزودن یا تغییر رفتار یک شی بدون تغییر کلاس اصلی کاربرد دارد.

توصیه دوستانه

🌟 اگر می‌خواهید رفتار اشیا را دینامیک، منعطف و قابل گسترش نگه دارید، از الگوی DECORATOR استفاده کنید!

💡 "هر شی، می‌تواند لایه‌ای جدید داشته باشد!"

PUBLISHED BY : FATEMEH AHMADZADEH