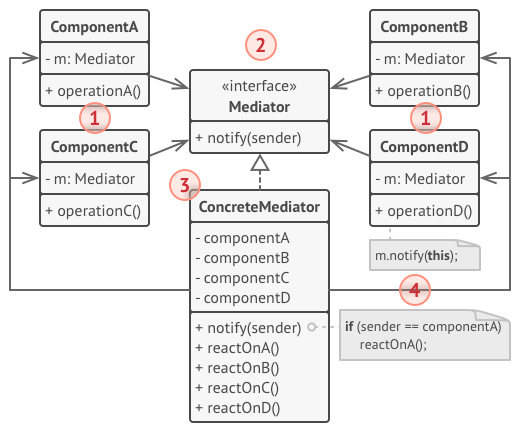
**ПОСРЕДНИК** - позволяет уменьшить связанность множества классов между собой, благодаря перемещению этих связей в один класс-посредник.

ПРИМЕНЕНИЕ:

- когда вам сложно менять некоторые классы из-за того, что они имеют множество хаотичных связей с другими классами

- когда вы не можете повторно использовать класс, поскольку он зависит от уймы других классов

- когда приходится создавать множество подклассов компонентов, чтобы использовать одни и те же компоненты в разных контекстах

****РЕАЛИЗАЦИЯ:

**1. КОМПОНЕНТЫ**: разнородные объекты, содержащие бизнес-логику программы. Каждый компонент хранит ссылку на объект посредника, но работает с ним только через абстрактный интерфейс посредников. Благодаря этому, компоненты можно повторно использовать в другой программе, связав их с посредником другого типа

**2. ПОСРЕДНИК**: определяет интерфейс для обмена информацией с компонентами. Обычно хватает одного метода, чтобы оповещать посредника о событиях, произошедших в компонентах. В параметрах этого метода можно передавать детали события: ссылку на компонент, в котором оно произошло, и любые другие данные

**3. КОНКРЕНТЫЙ ПОСРЕДНИК**: содержит код взаимодействия нескольких компонентов между собой. Часто этот объект не только хранит ссылки на все свои компоненты, но и сам их создаёт, управляя дальнейшим жизненным циклом

**4. КОМПОНЕНТЫ**: не должны общаться друг с другом напрямую. Если в компоненте происходит важное событие, он должен оповестить своего посредника, а тот сам решит — касается ли событие других компонентов, и стоит ли их оповещать. При этом компонент-отправитель не знает, кто обработает его запрос, а компонент-получатель не знает, кто его прислал

ПЛЮСЫ:

- устраняет зависимости между компонентами, позволяя повторно их использовать

- упрощает взаимодействие между компонентами

- централизует управление в одном месте

МИНУСЫ:

- может сильно раздуться