**الگوی Observer ( مشاهده گر)**

**فرض کنید کالایی میخواهید از فروشگاه آنلاینی خرید کنید که موجود نیست و الزاما شما فقط به همون کالا و برند نیاز دارید:**

* **یا شما باید هر روز به آن فروشگاه آنلاین سر بزنید که در صورت موجود شدن شما خریداری کنید**
* **یا آن فروشگاه در صورت موجود شدن به همه کاربران خودش اطلاع بده(برای سایر کاربران Spam می باشد.)**
* **میتوان توسط این الگو به افرادی که به این کالای خاص نیاز دارند را به فروشگاه معرفی کنید و در زمان موجود شدن برای همین افراد اطلاع رسانی کنیم.**

**در این الگو :**

* **Publisher :کلاسی که نقش همان فروشگاه را ایفا میکند .(همه میخواهند از تغییرات publisher مطلع باشند.)**
* **Subscribes : کلاس هایی که هستند که از تغییرات publisher مطلع میشوند.**

**نکات :**

1. **استفاده از interface به جای Abstract Class**
2. **الزام ایجاد نمونه از ConcreteSubject در ConcreteObserver نیست.**
3. **تعیین اولویت برای اطلاع رسانی**
4. **در یک کلاس میتواندی چندین بار الگوی observer را پیاده سازی کنیم**
5. **مشکل در یک مشترک میتواند ادامه کار را متوقف کند.**
6. **متد update مناسب کارهای دیتابیسی نیست**
7. **اگر اطلاع رسانی به صورت Transaction است مدیریت این transaction بر عهده subject است.**
8. **استفاده از interface برای ثبت در چنیدن Subject**
9. **استفاده از متد Notify هنگام تغییر**
10. **اگر در متد Update مشترکین ، متدی را در subject فراخوانی شود که باعث شود Notify ارسال شود باعث حلقه بینهایت میشود.**

**روش های لاگ گذاری در observer**

1. **لاگ توسط مشترکین انجام شود (در متد Update ) یا لاگ گذاری در Notify**
2. **لاگ گذاری توسط subject**

**مزایا:**

1. **اتصال سست بین publisher و subscriber**
2. **رعایت اصل open/Close**
3. **در زمان اجرا میتوانیم مشترک جدید یا حذف مشترک از publisher ها**
4. **Publisher نیازی به داشتن نوع و تعداد مشترکین ندارد**

**معایب:**

1. **در حالت عادی به ترتیب به مشترکین اطلاع رسانی می شود.**
2. **دیباگ کردن این الگو (مخصوصا در پردازش موازی )مشکل است.**

**چه زمانی از این الگو استفاده کنیم؟**

1. **زمانی که تغییر در یک شی نیاز به اعمال تغییرات در اشیا دیگر داشته باشیم.**
2. **زمانی که میخواهیم در زمان اجرا مشترکین را مدیریت کنیم**
3. **زمانی که میخواهیم بدون اتصال محکم یک شی بتوانیم اشیا مرتبط به خود را از تغییر آگاه کنیم.**