**الگوی State یا وضعیت**

**کاربرد :**

**یک شی را براساس وضعیت داخلی آن رفتارش را تغییر می دهد.**

در دنیای واقعی **دکمه قفل گوشی تلفن همراه ؛ مثلا این دکمه تو حالت عادی ( صفحه روشن باشد ) اگر این دکمه زده شود باعث قفل شدن گوشی می شود و اگر قفل باشد با زدن آن باز میشود. اگر چند ثانیه نگه دارید گوشی را خاموش میکند و اگر خاموش باشد نگه دارید گوشی روشن می شود.**

**برای مثال در سایت فروشگاهی برای هر سفارش میتوان state در نظر گرفت.**

**نکات :**

1. **Context در هر لحظه فقط یک وضعیت می تواند داشته باشد.**
2. **می توانیم از concreteState ها نمونه سازی اولیه ایجاد کنیم و در هر بخشی که نیاز بود از آن استفاده کنیم.**
3. **در Context فقط متدهای وضعیت نداریم که با تغییر وضعیت رفتار آن ها تغییر کنه می تواند در Context متد هایی داشته باشیم که برای وضعیت نباشد،می تواند از آن ها استفاده کنند.**
4. **از نظر State اهمیت ندارد که چه کسی وضعیت را تغییر میدهد،مهم این است که رفتار آن کلاس تغییر کند.**
5. **در متد Handle که در State و concreteState ها پیاده سازی شدند،میتوانیم Context را ارسال کنیم.**
6. **در context بهتر است چندین وضعیت را پیاده سازی نکنید.**
7. **اگر concreteState ها Stateless ( فقط متد داریم بدون ورودی ،Property و فیلدی هم در آن ها نیست)هستند آن ها را بصورت Singleton پیاده سازی کنید.**

**مزایا :**

1. **کد های Context را ساده سازی میکند**
2. **رعایت اصل SRP (جداسازی منطق در Context برای هر کلاس یک مسئولیت ایجاد کرده است**
3. **رعایت اصل OpenClose ( اگر State جدید اضافه کنیم نیازی نیست به ویرایش نیست و می توانیم State جدیدی را اضافه کنیم)**

**معایب:**

**اگر context وضعیت های کمی دارد و به ندرت تغییر میکند پیاده سازی State بصرفه نیست.**

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

1. **زمانی که کلاس ما دارای وضعیت های زیادی است و براساس وضعیت ها رفتار کلاس باید تغییر کند**
2. **زمانی که در کلاس خود برای تغییر رفتار آن از دستورات شرطی استفاده میکنید ،میتوانید از این الگو استفاده کنید.**
3. **نوشتن منطق تغییر رفتار به صورت سنتی باعث کد نویسی کثیف شده.**