**الگوی TemplateMethod**

**هدف : ساختار یک الگوریتم را طراحی کنیم و برخی از مراحل را به زیر کلاس بسپاریم که طبق نیاز با روش تعریف شده آن را انجام دهند. (بخشی از الگوریتم را بدون تغییر ساختار تغییر میدهند.)**

**مثلا پروژه ای که آدرس وب سایتهایی داده می شود و اطلاعات وب سایت ها را کامل بدست می آورد و روی دیتابیس SQL ذخیره میکند. بعد از مدتی مشتری درخواست این مورد را دارد که به جای SQL فرمت PDF هم داده شود و یا علاوه بر آن با فرمت XML داده شود.**

**نکات**

1. **کلاس پدر است که متدهای کلاس های فرزند را فراخوانی میکند.**
2. **متدهای Primitive بهتر است به صورت protected تعریف شوند که خارج از کلاس قابل استفاده نباشند.**
3. **متدهای hook همیشه اختیاری هستند و ممکن است بعضی از subclass ها پیاده سازی نشوند.**
4. **برای hook ها بهتر است بصورت پیشفرض پیاده سازی داشته باشند.**
5. **تا حد امکان تعداد متدهای Primitive کم باشند.**
6. **خود متد TemplateMethod نباید بصورت virtual باشد.**
7. **رعایت قواعد نامگذاری برای متد ها**

**مزایا**

1. **الگوریتم ها را میتوان به صورت داینامیک ایجاد کرد(تغییر برخی از مراحل توسط subclass ها )**
2. **به subclass اجازه داده می شود فقط قسمت های خاصی از یک الگوریتم بزرگ را طبق نیاز خود تغییر دهند و تحت تاثیر تغییرات در سایر قسمت های الگوریتم قرار نگیرد.(مثال تغییر ارسال نوتیفیکیشن )**
3. **با استفاده از این الگو کلاسهای فراوان با کدهای تکراری نداریم.**

**معایب**

1. **ممکن است ساختار ارائه شده برای الگوریتم محدودیت هایی برای subclass ها ایجاد کند.**
2. **احتمال نقض اصل Liskov**
3. **کسی که Template Method ها را توسعه میدهد،اگر سورس AbstractClass رانداشته باشد اطلاعات کافی در مورد اجرا شده متد ها ندارد.**

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

1. **زمانی که میخواهیم کلاینت برخی از مراحل الگوریتم را تغییر دهد.**
2. **زمانی که چندین کلاس داریم با تفاوت های جزئی می توانید از این الگو استفاده کنید.**

هر جایی که بتوان از تزریق وابستگی استفاده کرد نسبت به این الگو ارجعیت دارد.