دیزاین پترن ها

.

در کتاب دیزاین پترن، 23 تا آبجکت اورینتید دیزاین پترن معرفی شد در سه دسته(

Creational Patterns

Structural Patterns

Behavioral Patterns

).

Creational Pattern, The Factory

کاری به چیزی نداری، فقط میخوای که یک کلاس/فکتوری/چیز رو صدا بزنی و اون برات بسازه(مثلا کلاس رو)، مثلا یک کلاس هست که چندین متد داره و هرکدوم یک مدل خاصش رو میسازن(مثلا یک کلاسی رو).

Creational Pattern, The Builder

فکر کن یک کلاس بیلدر داری که کلاس میسازه و متد هاش اون کلاس رو تغییر میدن و در نهایت یک متد بیلد داره که اون کلاس نهایی شده رو برمیگردونه. و اینکه هرکدوم از اون متد ها، کلاس رو میگیره و تغییر رو روش اعمال میکنه و برش میگردونه.

Creational Pattern, The Singleton

Singleton is just a class that can only have a single instance of it that’s instantiated

یک کلاسی که فقط میتونه یکبار ساخته/اینیشیِیت بشه و ساخت کلاس توسط یک استاتیک متد انجام میشه و این متد برای ساخت یک نمونه/اینستنس جدید چک میکنه و شرط/if داره که اگر ساخته نشده، بسازه و در نهایت اینستنس رو برمیگردونه(این اینستنس رو تو خودش ذخیره داره که هر بار که میگی یه نمونه بساز اونو برگردونه).

Behavioral Pattern, Observer

میتونیم مثلا بهش بگیم PubSub پترن، برای درک بهتر.

مثلا یک سابجکت/پابلیشر داریم و چند آبزرور/سابسکرایبر که اون میفرسته و اینا میگیرن.

مثلا یک کلاس داریم که یک لیستی از سابسکرایبر ها داره و یک متد سابسکرایب داره که با اون بهش سابسکرایب میکنی و یم متد نوتیفای داره که به همه اعضای لیست سابسکرایبرها یک چیزی رو میفرسته.

مثلا ما تو این جا اگر در پایتون باشیم، لیست سابسکرایبر ها میشه یک لیست از آبجت های اعضا که میتونیم یک متدشونو فرابخونیم و مقداری رو هم بهش ارسال کنیم.

Behavioral Pattern, Iterator

تعیین میکنه که چجوری میشه بین آبجکت های یک کلاس ایتریت کرد و دونه دونه دیدشون، به چه صورت، به چه ترتیب، با چه ترتیب و الگوریتمی.

متد های __iter__ و __next__ در پایتون به ما این امکان رو میدن که روی یک کلاس ایتریت کنیم و حلقه بزنیم.

Behavioral Pattern, Strategy

If you want to modify or extend the behavior of a class without directly changing it

مثلا چند کلاس(فیلتر کلاس ها) داریم که از کلاس خودساخته-ابسترکت-فیلتر ارث میبرن و هرکدوم هم متدِ آیا-حذف-بشود رو به شیوه خودشون پیاده سازی میکنن، حالا یک کلاس داریم تعدادی مقدار داره و یک متد فیلتر داره که این متد یک لیستی یا یکی از اون (فیلتر کلاس ها)رو میگیره و مقدار رو میده به متد اون (فیلتر کلاس) و اگر پس شد، ادش میکنه.

Structural Pattern, Adapter

فکر کن یک پیچ داری که توی مهره جا نمیشه، یک واشری/چیزی/ادپتری بهش میچسبونی تا فیت/درست/دلخواه بشه.

مثلا یک کلاس پایتونی داری که میخوای مثل آداپتور بین دو کلاس باشه و ارتباط برقرار کنه یا تغییر بده یا اضافه کنه، مثلا از یک کلاسی ارث میبره و یک نمونه/اینستنس از یک کلاس دیگه دریافت میکنه و این دریافتی رو رفتارش رو شبیه اونی میکنه که ازش ارث برد.

Structural Pattern, Facade

The word Facade meaning, an outward appearance that is used to conceal a less pleasant or credible reality

بخش ها و قسمت ها و ساختار و سیستمِ لو لول و سطح پایین رو خیلی انتزاعی میکنه و یجورایی مثل یه اینترفیس/API میشه.

یک ظاهری رو فراهم میکنی که با اون خیلی تمیز تر و ساده تر کارهارو انجام بدی، مثلا لیست که توی پایتون همه چیز انتزاعی شده و کاری به چگونه کارکردش نداری چون انجام شده و یک ظاهر به ما داده شده که با اون کار کنیم و نگران مموری و اینها و پیاده سازی یک لیست توی رم و غیره نباشیم و استفاده کنیم.

.