**1. MVC (Model-View-Controller)**

Az MVC egy elterjedt szoftverarchitektúra-minta, amely a kódot három rétegre bontja:

* **Modell (Model):** Az adatok és az üzleti logika kezelése. Például adatbázis-interakciók.
* **Nézet (View):** A felhasználói felület megjelenítése. Például HTML, CSS, UI elemek.
* **Vezérlő (Controller):** A felhasználói interakciók kezelése, kommunikáció a Modell és a Nézet között.

**Előnyei:**

* Könnyebb karbantartás, tesztelhetőség.
* Egységként kezelhető a logika és a megjelenítés szétválasztása.

**2. Singleton**

Biztosítja, hogy egy osztálynak csak egyetlen példánya legyen, és globálisan elérhető.

* **Használat:** Konfigurációk, naplózás, adatbázis-kapcsolat.
* **Előny:** Kevesebb memóriafogyasztás, egyszerűbb kezelés.

**3. Observer (Megfigyelő)**

Egy objektum (tárgy) állapotváltozásáról értesíti a hozzá kapcsolódó megfigyelő objektumokat.

* **Használat:** Eseménykezelés, például GUI elemek frissítése.
* **Példa:** Egy gombnyomás értesíti az összes hallgatót.

**4. Factory (Gyár)**

Egy objektum létrehozásának logikáját elkülöníti a használatától.

* **Használat:** Egyes osztályok példányosítása dinamikusan történik.
* **Példa:** Egy játékban különböző karakterek (harcos, mágus) létrehozása.

**5. Strategy (Stratégia)**

Többféle algoritmus közötti választást tesz lehetővé futási időben.

* **Használat:** Számítási, keresési, rendezési algoritmusok váltása.
* **Példa:** Egy fizetési rendszer, ahol különböző fizetési módokat kezelünk (PayPal, bankkártya).

**6. Decorator (Dekorátor)**

Dinamikusan ad új funkciókat egy objektumhoz, anélkül, hogy módosítanánk az eredeti osztályát.

* **Használat:** Funkciók hozzáadása például UI elemekhez (keretek, színek).