**Tervezési minták egy objektumorientált programozási nyelvben**

**Bevezetés a tervezési mintákhoz**

A tervezési minták olyan általános megoldások, amelyek gyakran előforduló problémák megoldására nyújtanak újrahasználható struktúrákat. Nemcsak a problémák megoldásában segítenek, hanem az így megírt kód olvashatóbbá és egyszerűbbé is válik.

Ezek a minták függetlenek az adott programozási nyelvtől, így könnyen érthetőek más technológiákat használó fejlesztők számára is. Ugyanakkor ez azt is jelenti, hogy a megoldások nem "kimásolhatók és beilleszthetők" közvetlenül, hanem alkalmazkodniuk kell a konkrét projekthez.

A tervezési minták általában tartalmaznak:

* Egy problémaleírást;
* Egy újrahasználható megoldási leírást;
* Diagramokat, például struktúrális diagramokat;
* Pszeudokódot vagy kódmintákat népszerű programozási nyelveken.

**Modell-Nézet-Vezérlő (MVC)**

Ez a minta rendkívül hasznos, ha a felhasználói felületet el szeretnénk különíteni az üzleti logikától és az adatszerkezetektől. Az MVC eredetileg asztali alkalmazásoknál volt népszerű, de ma már széles körben alkalmazzák web- és mobilalkalmazások fejlesztésére.

Több nyelvnek, mint például a JavaScript, Python, Ruby, PHP, Java, C#, és Swift, van beépített MVC-keretrendszere. Az MVC alkalmazásával a teljes alkalmazás szerkezetét meghatározhatjuk, amely jóval átláthatóbbá teszi azt.

**Az MVC rétegei**

1. **Modell:** Az adatkezeléssel kapcsolatos logikát valósítja meg, és felelős az adatbázis-kezelésért.
2. **Nézet:** A felhasználói felületet biztosítja, amely dinamikusan jeleníti meg az adatokat. Egy modellhez több nézet is tartozhat.
3. **Vezérlő:** Kapcsolóelem a modell és a nézet között.

**Singleton (Egyke)**

Ez a minta biztosítja, hogy egy osztálynak legfeljebb egyetlen példánya legyen. Ezt úgy érhetjük el, hogy az osztály konstruktorát priváttá tesszük, és egy olyan metódust biztosítunk, amely csak akkor példányosítja az osztályt, ha még nem létezik példánya. Ha már létezik, akkor a meglévő példányt adja vissza.

**Gyártó metódus**

Ez a minta leegyszerűsíti az osztályok példányosítását, különösen, ha sok hasonló objektumot kell létrehoznunk, amelyeknek sok paraméterük van. A gyártó metódus lehetővé teszi az objektumok automatikus létrehozását.

**Prototípus**

Ez a minta lehetővé teszi az objektumok mély klónozását, vagyis nemcsak a hivatkozást, hanem az értékeket is másolja. Ez különösen hasznos, ha egy objektum új példányát kell létrehozni azonos értékekkel.