Tervezési minták az objektumorientált programozásban (OOP)

Definíció: Az informatikában a program-tervezési mintának (angolul Software Design Patterns) nevezik a gyakran előforduló programozási feladatokra adható általános, újrafelhasználható megoldásokat. Egy programtervezési minta rendszerint egymással együttműködő objektumok és osztályok leírása (Forrás: Wikipédia)

A programtervezési minták gyakori problémák leírásait adják meg úgy, hogy közben megadják a megoldást, amit felhasználva a konkrét rendszerhez újra és újra alkalmazható. A minták által a rendszer egyszerűbb, karbantarthatóbb és újrafelhasználhatóbb lesz, másfelől a minták ismeretével megérteni is könnyebb a rendszert, ezáltal egy-egy rész megértése a különféle nyelvek kódjaiban könnyebbé válik.

Fontos és gyakran használt tervezési minták

1. Létrehozási minták
    - Factory method : Gyártófüggvény: Interfészt definiál egy objektum létrehozásához, de az alosztályokra bízza, melyik osztályt példányosítják. A factory method megengedi az osztályoknak, hogy a példányosítást az alosztályokra ruházzák át.
    - Abstract factory: Elvont gyár: Kapcsolódó vagy egymástól függő objektumok családjának létrehozására szolgáló interfészt biztosít a konkrét osztályok megadása nélkül.
    - Singleton: Egyke: Egy osztályból csak egy példányt engedélyez, és ehhez globális hozzáférést ad.
    - Dependency Injection : Olyan programozási technika, amely egy osztályt függetlenít a függőségeitől. Ezt úgy éri el, hogy elválasztja egy objektum használatát a létrehozásától

2. Strukturális és viselkedési minták
    - Template method : egy viselkedési tervezési minta, amely meghatározza egy algoritmus vázát a szuperosztályban, de lehetővé teszi, hogy az alosztályok felülírják az algoritmus bizonyos lépéseit anélkül, hogy megváltoztatnák annak szerkezetét.
    - Strategy: a stratégiai minta (más néven irányelvminta) egy viselkedési szoftvertervezési minta , amely lehetővé teszi az algoritmus kiválasztását futás közben. Ahelyett, hogy egyetlen algoritmust közvetlenül implementálna, a kód futásidejű utasításokat kap arra vonatkozóan, hogy egy algoritmuscsalád melyikét használja.
    - Observer : **A viselkedési tervezési minták** egyike . Az Observer tervezési mintája akkor hasznos, ha érdekli egy objektum állapota, és szeretne értesítést kapni, ha bármilyen változás történik.
    - Command : Viselkedési tervezési minta, amely a kérést önálló objektummá alakítja, amely tartalmazza a kéréssel kapcsolatos összes információt
    - Memento : Szoftvertervezési minta , amely felfedi egy objektum privát belső állapotát. Ennek egyik példája az objektum korábbi állapotának visszaállítása (visszaállítás visszaállítással), a másik a verziószámítás, a másik az egyéni szerializálás.

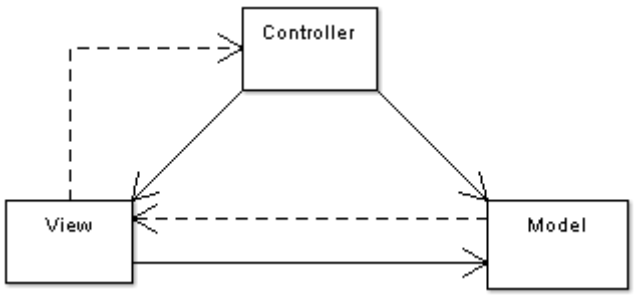
Az MVC (Model View Controller) tervezési minta

Mi az MVC?

Az MVC egy tervezési, vagy architekturális minta. A Model-View-Controller elv alapötlete az, hogy adott egy vezérlő (controller), ami megkapja a felhasználótól a kérést egy bizonyos művelet elvégzésére, majd a modellből megszerzi az adatot, amit átad a view-nak megjelenítésre. A lényege a megjelenítés leválasztása a működési logikáról.

A három szó jelentése:

- *Model (Adatok)*
- *View (Felhasználói felület)*
- *Controller (Vezérlő szerkezetek)*



Az MVC jellemzői

- Könnyű és súrlódásmentes tesztelhetőség. Kiválóan tesztelhető, bővíthető és csatlakoztatható keretrendszer
- Az MVC-mintát használó webalkalmazás-architektúra megtervezéséhez teljes irányítást biztosít a HTML és az URL-címek felett

- Használja ki az ASP.NET, JSP, Django stb. által biztosított meglévő szolgáltatásokat.
- A logika egyértelmű szétválasztása: Modell, Nézet, Vezérlő. A pályázati feladatok szétválasztása pl. üzleti logika, Ul logika és bemeneti logika
- URL-útválasztás SEO-barát URL-ekhez. Hatékony URL-leképezés az érthető és kereshető URL-ekért
- A tesztvezérelt fejlesztés (TDD) támogatása

MVC keretrendszerek:

- Ruby on Rails
- Django
- CakePHP
- Yii
- CherryPy
- Spring MVC
- Catalyst
- Rails
- Zend Framework
- CodeIgniter
- Laravel
- Fuel PHP
- Symphony