Tervezési minta

A projekt a Model-View-Controller (MVC) mintát követi, mely szétválasztja a játék logikáját, a megjelenítést, valamint a felhasználói interakciók kezelését. Ez a megközelítés biztosítsa a kód áttekinthetőségét, fenntarthatóságát és tesztelhetőségét.

1. Model: A játék logikájának és állapotának kezelése

A projekteben a Model szerepét a GameState osztály tölti be. Ez az osztály a játék belső állapotát, a játékszabályokat, valamint a győzelmi feltételeket kezeli. A főbb funkciók:

- Tárolt adatok:
  A játék tábla egy kétdimenziós tömbként (char[][]) van reprezentálva, amely minden mező állapotát tárolja
- Műveletek:
  - A játékos lépéseit a dropPiece metódus hajtja végre. Ez biztosítja, hogy a választott oszlopba a legalsó elérhető pozícióba kerül a darab, és hibát dob, ha az oszlop már tele van.
  - A checkWin metódus ellenőrzi a győzelmi feltételeket, beleértve a vízszintes, függőleges és átlós sorokat is.
  - Az isBoardFull metódus megállapítja, hogy a játék döntetlennel végződött-e, azáltal hogy ellenőrzi, van-e még üres hely a táblán.
  - A resetBoard metódus lehetőséget ad a tábla újraindítására egy új játék érdekében, az összes mezőt EMPTY állapotra állítva.

A Model kizárólag a játék belső működésével és a szabályok betartásával foglalkozik, függetlenül attól, hogyan jelenik meg a játék vagy hogyan vezérlik azt.

2. View: A játék vizuális megjelenítése

A projektben a View a felhasználói felületért felelős, amely a játékosokkal való interakció helyszíne. A View főbb funkciói:

- Játék tábla megjelenítése:
  A nézet felelős a tábla aktuális állapotának megjelenítéséért, amelyet a GameState-ből származtat. A tábla frissítése minden lépés után megtörténik, hogy tükrözze a játékosok mozgását.
- Visszajelzések megjelenítése:
  A nézet tájékoztatja a játékosokat a játék aktuális állapotáról, például hogy ki következik, győzött-e valaki, döntetlen történt-e, vagy érvénytelen lépés történt.
- Interakciók fogadása:
  A felhasználó által végrehajtott akciókat (például egy oszlopra kattintást) a nézet továbbítja a Controller-nek, amely a megfelelő logikát kezeli.

A nézet kizárólag a megjelenítési feladatokra összpontosít, és nem tartalmaz semmilyen logikát a játék működésére vonatkozóan.

3. Controller: Az interakciók és a logika összekapcsolása

A Controller köti össze a nézetet és a modellt, biztosítva, hogy a felhasználói műveletek megfelelően végrehajtódjanak. Az alábbi főbb feladatokat látja el:

- Felhasználói lépések feldolgozása:
  Amikor a játékos egy oszlopot választ, a Controller lekéri az oszlop számát a nézettől, majd meghívja a dropPiece metódust a GameState-ben, hogy végrehajtsa a lépést.
- Győzelem és döntetlen kezelése:
  A Controller a lépés után ellenőrzi a checkWin és az isBoardFull metódusok segítségével, hogy a játék véget ért-e. Ha igen, értesíti a nézetet, hogy jelenítse meg a megfelelő üzenetet.
- Hibák kezelése:
  Ha a játékos érvénytelen lépést hajt végre (pl. egy tele oszlopba próbál dobni), a Controller kezeli a kivételt, és értesíti a nézetet, hogy figyelmeztesse a játékost.

4. Interakció a komponensek között

Az MVC minta szerint a komponensek közötti interakció így történik:

- A felhasználó egy műveletet indít el a nézetben (például kiválaszt egy oszlopot).
- A nézet eseményt küld a Controllernek.
- A Controller meghívja a GameState megfelelő metódusait, hogy végrehajtsa a lépést, és ellenőrizze a győzelmi feltételeket.
- A GameState frissíti az állapotát, és visszajelzést ad a Controllernek.
- A Controller értesíti a nézetet, hogy frissítse a tábla megjelenítését és a játékosok számára szükséges visszajelzéseket.