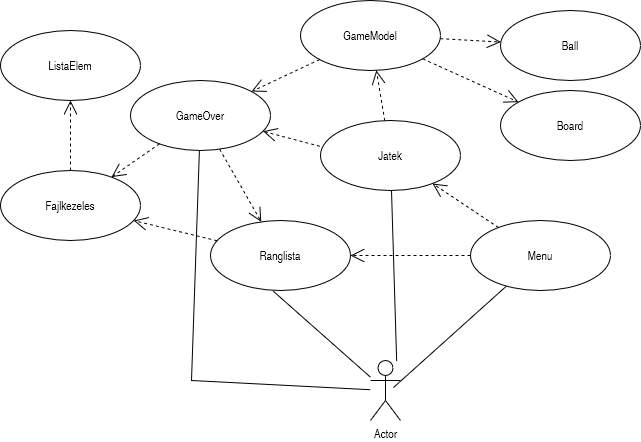
**Breakout windows forms dokumentáció**

A játék lényege, hogy egy deszkával kell úgy pattogtatni a labdát, hogy az kiüsse a téglákat. A felhasználónak három élete van, ha ezek elfogytak, a játéknak vége. Az elért pontszámokat le lehet menteni, amiket később ki is lehet listázni. A feladatot Windows Forms környezetben, C# nyelven készítettem.

**Use-case diagram**



**Tervezés**

**Nézet**

A programot háromrétegű architektúrában valósíjuk meg. A megjelenítést a formok végzik (menü, játék, game over, ranglista), míg a modell kezeli ezeknek a kapcsolatát. A fájlkezelés osztály gyakorlatilag a perzisztencia, az felelős a fájl betöltéséért, írásáért, olvasásáért és a rendezésért. Az adatkezelés feladata, hogy a játékban elért eredményeket tárolja és azokat rendezze csökkenő sorrendben, majd szükség esetén jelenítse meg.

A Menu osztályon keresztül lehet elindítani a játékot, valamint megnézni az előző játékosok elért pontszámait, ezekért egy-egy gomb felelős.

A Jatek osztály kezeli magát a játékot. Ez jeleníti meg a tégláknak használt gombokat, a deszkát és a labdát.

A GameOver Form a játék végén jelenik meg, ahol el lehet menteni az elért eredményünket, valamint vissza lehet lépni a menübe.

A Ranglista Formon lehet megnézni az elért eredményeket.

A Deszka a NumericUpDown osztályból van származtatva, ez kezeli, hogy a deszkát lehessen mozgatni.

A Labda osztály a RadioButton osztály leszármazottja, ez felelős a labda adatainak tárolásáért és a sebesség növeléséért.

**Modell:**

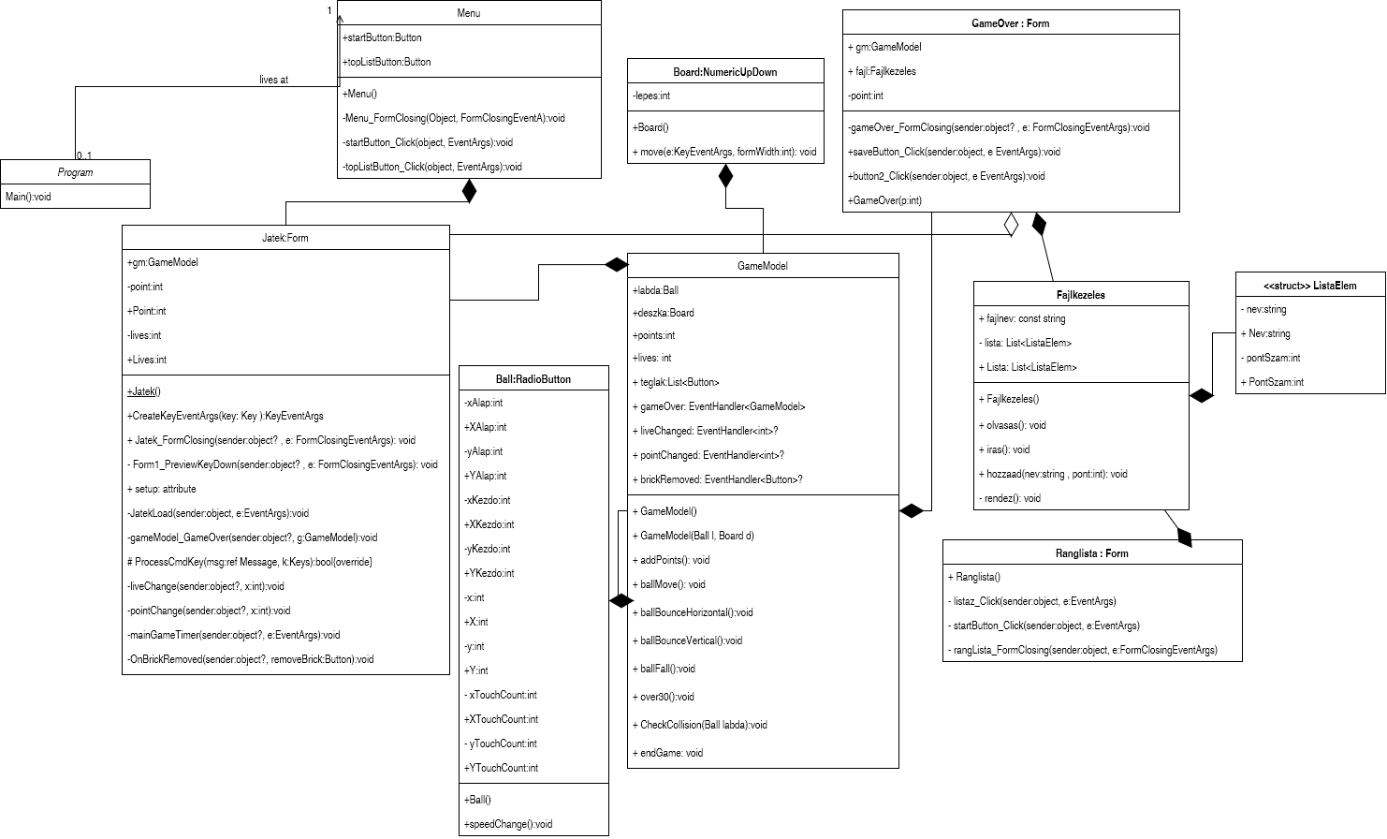
A GameModel felelős a formok mögötti folyamtokért, az kezeli az egész játék működését.

Ez kezeli a pontszámot, a labda mozgását, a téglák kiütését és a játék végét, ha leszedte az ember az összes téglát vagy elfogyott az élete.

**Perzisztencia**

A fájl kezelésesért a Fajlkezelés osztály felel. Ha a felhasználó szeretné. akkor a játék végén menti el az elért eredményeket, azokat kiírja egy fájlba. Ezt a fájlt a Ranglista formon bármikor ki lehet olvasni. Minden egyes új elem esetén rendezi a listát, így mindig csökkenő sorrendben, toplista szerűen jelennek meg az eredmények. Ehhez egy ListaElem struktúrát is segítségül hívtam, ami eltárolja a játékos nevét és az elért pontját. A fájl egy egyszerű szöveges fájl, amibe StreamWriterrel írjuk bele az adatokat és StreamReaderrel olvassuk ki. A toplista sorszámát nem tároljuk külön, azt mindig a beolvasott Lista indexe alapján határozzuk meg, hiszen a lista rendezve van, ezáltal nem kell még egy adatot eltárolni.

**UML diagram:**

****

**Tesztelés:**

A modell funkcionalitása egységtesztek segítségével lett ellenőrizve a TestProject1 osztályban.

Az alábbi tesztek kerültek megvalósításra:

* A **TestInitialLives** vizsgálja, hogy indításkor tényleg jó értéket kap-e a lives változó.
* A **TestAddPoints** a pontok hozzáadását ellenőrzi.
* A **TestBallBounceHorizontal** és a **TestBallBounceVertical** tesztek azt nézik, hogy megfelelően visszapattan-e a labda, ha ütközik valamivel.
* A **TestBallFall** a labda leesését ellenőrzi, hogy tényleg csökken-e a játékos élete.