

# Programozás alapjai 3: Házi feladat dokumentáció

Készítette: Bacskai Tamás (VHUI74)

## Felhasználói kézikönyv

A menüből a “New Game” opciót kiválasztva elindul a játék, a játék során egy virtuális kígyót kell megetetni a pályán látható piros almákkal, amik hatására megnő és a játékos pontokat kap. A kígyó magától mozog a sárga fejével egy adott irányba fix időközönként, a játékos a nyilakkal irányíthatja, amelyek hatására a kígyó feje (és majd utána a teste is) a nyíl irányába kezd el haladni. Ha a játékos kimegy a pályáról vagy pedig saját testébe ütközik, akkor vége a játéknak és a játékos visszakerül a menübe, valamint egy extra ablak megjelenik, ahol megadhatja a nevét ha szeretné és így felkerülhet a dicsőséglistára, amin a 10 legjobb játék van rajta. A játékos a menüből a “Leaderboard” opcióval meg is nézheti az előbb említett dicsőséglistát, ahol játékosneveket és pontszámokat talál majd.

## Use-case-ek leírása

Menü opciói	Eredményük
Új Játék	Indít egy új játékot
Dicsőséglista	Megnyitja a dicsőséglistát, ahol látjuk a 10 legjobb játékosok nevét és az általuk elért pontszámot
Kilépés	Kilép a programból

Játék forgatókönyvek	Történések
<b>1.</b> Snake mozgatása	A nyilak megnyomásának hatására a kígyó a nyíl irányába fordítja a fejét és elindul abba az irányba, nem tud eredeti mozgásához képest egyből az ellentétes irányba elindulni
<b>1.a</b> Snake ütközik a fallal vagy a saját testével	Ha a snake ezek valamelyikével ütközik, akkor vége a játéknak
<b>1.b</b> Snake “ütközik” az étellel (megeszi az ételt)	Ha ez megtörtént a snake megnyúlik, az őt irányító játékos pontszáma megnövekszik és egy új étel generálódik véletlenszerűen valahol a pályán
<b>1.c</b> Snake üres mezőbe lép	Ilyenkor nem történik semmi, csak haladt egy irányba a snake
<b>1.a.1</b> Snake meghalt	Ha a snake meghalt akkor a program felkínálja, hogy adjon meg egy nevet és ha a



-frame: SnakeFrame	Egy keret, akinek átadja a panelt, valamint beállítja rajta az irányítás érzékelését
-timer: Timer	Az időzítő referenciája
-apple: Apple	Az alma referenciája
-inputIsValid: boolean	Az input validitásának követésére, hogy ne tudja a kígyó egy ütem alatt “megenni magát”
<b>Függvények</b>	
-snakeSpawn()	Belső függvény, olvashatóság miatt külön szedve, a kígyó kezdeti elhelyezését végzi el a pályán
+appleEaten(int score_earned)	Egy alma megétele utáni folyamatokat bonyolítja le: új almát rak le, megnöveli a kígyó méretét egyel és az argumentumként kapott pontszámmal megnöveli a pontszámot, argumentuma esetleges egyéb ételek későbbi bevezetése miatt adható még így, alavetően az alma eggyel növeli meg a pontszámot
+addPlayerToBoard(String name)	A paraméterként kapott névvel elmenti a játékos által szerzett pontot: kiszerializálja a beolvasott Leaderboard objektumot, ha még nincs ilyen, akkor létre is hozza
+actionPerformed(ActionEvent e)	Az ütemezett cselekvések történnek itt: mozog a kígyó és újra adható valid input, ha nem halt még meg a kígyó, ha meghalt, akkor megállítja az ütemezőt létrehoz egy névbekérő ablakot, és leveszi a játék paneljét a fő ablakról valamint ráállít egy új menu panelt, valamint minden ütemben újrifesti a pálya jelenlegi helyzetét
<b>Controls belső osztály</b>	
+keyPressed(KeyEvent e)	Az irányítást kezeli le, úgy, hogy a kígyó ne tudjon magába fordulni

## Map osztály

Leítés	A pályát valósítja meg, azt közvetlen manipulálja
<b>Attribútumok</b>	
-tiles: Tile [][]	Tileekből álló pálya mátrix
-spawn_counter: int	A kígyó kezdeti elhelyezéséhez segédváltozó
-random: Random	Az alma random spawnolásához random szám generátor
-max_rows: int	Maximális sorok száma
-max_columns: int	Maximális oszlopok száma
<b>Függvények</b>	
-setNeighbours()	Pálya mezői közötti szomszédságok berendezése, belső függvény, átláthatóság és a folyamat komplexitása miatt külön szedve

+spawnSnakeBodyPart()	Kígyó egy testdarabjának (fejének kivételével!) elhelyezése a pályán
+spawnHead()	Kígyó fejének elhelyezése a pályán
+getTile(int i, int j): Tile	Az i. oszlop j. sorának mezőnek referenciáját adja vissza, szerepe leginkább csak a kirajzolásánál van
+spawnApple (Apple apple)	Elhelyezi az argumentumként kapott almát a pálya egyik olyan mezőjén, ami üres (nincs benne kígyó, de ha bővítenénk a játékot falakkal, így oda sem fogja rakni)

## Snake osztály

Leírás	A kígyót valósítja meg, a kígyó részeket egyesével rendezi, hogy ez a feladat nem a Game-re háruljon, ő csak utasítja a kígyót arra amire kell
<b>Attribútumok</b>	
-head: SnakeHead	A kígyó feje
-score: int	A kígyó jelenlegi pontszáma
-parts: LinkedList<SnakeBodyParts>	A kígyó testének részei
-alive: boolean	Azt tárolja, hogy a kígyó életben van-e vagy sem
-move_last: boolean	Azt tárolja, hogy az utolsó kígyótestrész után kerüljön-e be újabb, a kígyó megnövelésénél lesz rá szükség
<b>Függvények</b>	
+addBodyPart(SnakeBodyPart bodypart)	A paraméterként kapott testrészt hozzáfűzi a többihez
+setHead(SnakeHead head)	A paraméterként kapott fejet beállítja fejnek és hozzáfűzi a többi kígyó testrészhez
+getScore(): int	Visszaadja a kígyó pontszámát
+move()	Elvégzi a kígyó összes testrészén a mozgást és beállítja rajtuk az új irányokat (egy "kanyarodás" esetén) ha a move_last bit nincs bebillentve akkor az utolsó testrészt (ami egy újonnan megnövekedett testrész) nem mozgatja, hogy ne csússzon egymásba az utolsó és az utolsó előtti testrész
+expandSnake()	Megnöveli a kígyót: az utolsó elemen megjelöli, hogy növekedjen, valamint a move_last bitet hamisba billenti
+addScore(int a)	Megnöveli a pontot a-val és a SnakePanel pontszámát is átírja az új pontra, hogy az a frisset írja ki
+die()	Az alive bitet hamisba billenti
+isAlive(): boolean	Visszaadja, hogy a kígyó él-e vagy sem

## Apple osztály

Leírás	Az almát valósítja meg
<b>Attribútumok</b>	
-color: Color = Color.RED	Statikus tag, amely minden alma színét beállítja

-game: Game	Egy referencia a játékra, hogy tudja neki jelezni, ha megették
<b>Függvények</b>	
+eat()	Jelzi a játéknak, hogy megették és hogy mennyivel növeljék a játékos pontszámát
+draw (Graphics g, Coordinates coordinates)	Kirajzolja magát a paraméterként kapott g-vel a coordinates helyen

### Coordinates osztály

Leírás	Egy egyszerűsítő koordináta osztály, minden Tile rendelkezik vele, getterek vannak benne, valamint konstruktorában beállíthatók a koordinátái
<b>Attribútumok</b>	
-x0: int	-
-y0: int	-
-x1: int	-
-y1: int	-

### Directions enum

Leírás	4 irányt ad meg fel, balra, le, és jobbra
--------	---

### Eatable interface

Leírás	Egy interface, ami a megehető tulajdonságot kényszeríti rá az őt implementálókra: minden megehető meg lehet enni valami módon (eat()) és kirajzolható (draw())
<b>Függvények</b>	
+eat()	Megevésének következményei
+draw(Graphics g, Coordinates coordinates)	Kirajzolás, minden megehető máshogy rajzolunk ki: pls alma piros kör, kígyó testrésze zöld négyzet stb.

### GetNameFrame osztály

Leírás	Ebben az ablakban adhatja meg a játékos a nevét a dicsőséglistára való felkerüléshez
<b>Attribútumok</b>	
-game: Game	A játék referencia, aminek meghívja a dicsőséglistára helyező függvényét egy névvel, ha a játékos óhajt megadni nevet
<b>ConfirmButtonActionListener belső osztály</b>	
+actionPerformed(ActionEvent e)	A véglegesítő gomb hatására meghívja a game megfelelő függvényét

### SnakeBodyPart osztály

Leírás	A kígyó egy testrészét valósítja meg
<b>Attribútumok</b>	
#pos: tile	A testrész adott pozíciója
-d: Directions	Melyik irányba néz éppen
#snake: Snake	Melyik kígyóhoz tartozik
-expand: boolean	Növekedjen-e, a kígyó jelölheti ki rajta

<u>-color: Color = Color.GREEN</u>	Statikus tag: minden kígyótag ilyen színű lesz
<b>Függvények</b>	
+markForExpansion()	Expandot truera állítja
+move(Tile tile)	A tile mezőre lép, ha azzal nem menne ki a pályáról, illetve itt növekedik a kígyó ha meg van rá jelölve, úgy hogy a jelenlegi helyére egy új kígyó darabot hoz létre és ő pedig előre lép egyet, majd az új darabot hozzáfűzi a kígyóhoz
+eat()	Az Eatable interface hozadéka, ha a kígyó feje megeszi ezt a testrészt akkor a kígyó meghal
+draw (Graphics g, Coordinates coordinates)	Kirajzolja a kígyó testrészt a koordinátákra a statikus tagként megadott színben

### Leaderboard osztály

Leírás	A dicsőségtáblát valósítja meg
<b>Attribútumok</b>	
-top_players: LinkedList<PlayerScore>	A legjobb játékosok listája
<b>Függvények</b>	
+add(String name, int score)	Hozzáadja az új tagot, majd rendezi és levágja, ha 10-nél több tagja lenne a listának
<b>Egyéb</b>	
PlayerScore	Belső osztály a tagok kezelésére
Comparator	Található benne egy komparátor, hogy a belső egyedi osztálybeli objektumokat rendezni tudja

### LeaderboardPanel osztály

Leírás	A dicsőségtábla megjelenítését valósítja meg
<b>Attribútumok</b>	
<u>-SCREEN_WIDTH: final int = 700</u>	Szélesség
<u>-SCREEN_HEIGHT: final int = 700</u>	Magasság
-board: Leaderboard	A dicsőségtábla maga
-upper: JPanel	A felső panel a visszalépés gombnak
-main: JPanel	A középső panel a nevekkkel és a pontszámokkal
<b>Függvények</b>	
-addLeaderboardElements()	Ha létezik már dicsőséglista és azt be tudta olvasni akkor ez az ág hajtódik végre
-noLeaderboard()	Ha nem sikerült beolvasni dicsőséglistát, mert még nem mentette el senki a játékát akkor ez az ág hajtódik végre és csupa üres dicsőséglistát kapunk
<b>Egyéb</b>	
BackButtonActionListener	Belső osztály a visszagomb funkciójának megvalósítására: a keretből kitörli a

	dicsőséglista paneljét és egy új menüpanellet állít a helyébe
--	---

### MenuPanel osztály

Leírás	A menüt valósítja meg
<b>Attribútumok</b>	
<u>-SCREEN_WIDTH: final int = 700</u>	Szélesség
<u>-SCREEN_HEIGHT: final int = 700</u>	Magasság
<b>Egyéb</b>	
NewGameActionListener	Belső osztály az új játék gomb funkciójának megvalósítására: a keretből kitörli a menü paneljét és létrehoz egy új játék objektumot, ami pedig elintézi a játékkal kapcsolatos további részleteket
LeadActionListener	Belső osztály a dicsőséglista gomb funkciójának megvalósítására: a keretből kitörli a menü paneljét és létrehoz egy dicsőséglista objektumot, ami pedig elintézi a dicsőséglista megjelenítésével, beolvasásával kapcsolatos további részleteket
ExitActionListener	Belső osztály a kilépés gomb funkciójának megvalósítására: bezárja az ablakot és leáll a program

### SnakeFrame osztály

Leírás	Egy JFrame néhány előre beállított beállítással, valamint induláskor egyből létrehoz egy menüt, a Game majd ráhelyez egy KeyListenert amely érzékeli majd az inputot és továbbadja azt a Game-nek feldolgozásra
--------	---

### SnakeHead osztály

Leírás	A kígyó fejét valósítja meg
<b>Attribútumok</b>	
<u>-color: Color = Color.YELLOW</u>	Statikus tag, amely minden fej (bár csak egy van belőle) sárgásra színezi
<b>Függvények</b>	
-move()	Az őosztálya mozgását felülírja, mégpedig úgy, hogyha kimenne a pályából akkor meghal a kígyó aminek része, valamint ha olyan mezőbe lép ami nem üres, akkor megeszi az ott lévő valamit egyébként úgy mozog mint az őse
-draw (Graphics g, Coordinates coordinates)	Szokásos módon rajzol, a megadott színben egy négyzetet, ami szint érdemes a kígyótestektől különbözőre választani

### SnakePanel osztály

Leírás	A játék grafikai részét valósítja meg
<b>Attribútumok</b>	
-SCREEN_WIDTH: final int = 700	Szélesség
-SCREEN_HEIGHT: final int = 700	Magasság
-TILE_SIZE: int = 25	Egy mező mérete, osztójának kell lennie a magasságnak és a szélességnek is, másképp elcsúszik a pálya
-score: int = 0	A kijelzendő pontszám
-map: Map	Egy referencia a pályára, hogy előre annak mezőit és kirajzolhassa
<b>Függvények</b>	
-paintComponent(Graphics g)	Kirajzoló függvény

### Tile osztály

Leírás	Egy mezőt valósít meg
<b>Attribútumok</b>	
-coordinates: Coordinates	A SnakePanel Tile_Size-jára normalizált koordináták
-content: Eatable	A mezőben lévő valami ehető (kígyó rész vagy alma) lehet null is, ilyenkor üres a mező
-neighbours: Neighbours	A mező szomszédos mezői, ha egy irányba már nincs szomszédja, hanem a pálya széle van, akkor ott null a szomszéd
<b>Függvények</b>	
+Tile(int x0, int y0, int x1, int y1)	Beállítja a koordinátákat is
-setNeighbour(Directions d, Tile tile)	Beállítja szomszédnak az argumentumban kapott irányba az argumentumba kapott mezőt
-getNeighbour(Directions d): Tile	Megadja, hogy az adott mezőnek az argumentumként kapott irányba melyik mező a szomszédja (ha szélső mező akkor adhat null-t ha abba az irányba már nincs szomszédja)
-draw(Graphics g)	Kirajzolja a benne tartozkodót, aminek odaadja, majd a koordinátákat is
<b>Egyéb</b>	
Neighbours class	Belső osztály a szomszédságok kezelésére, hogy egyként lehessen kezelni