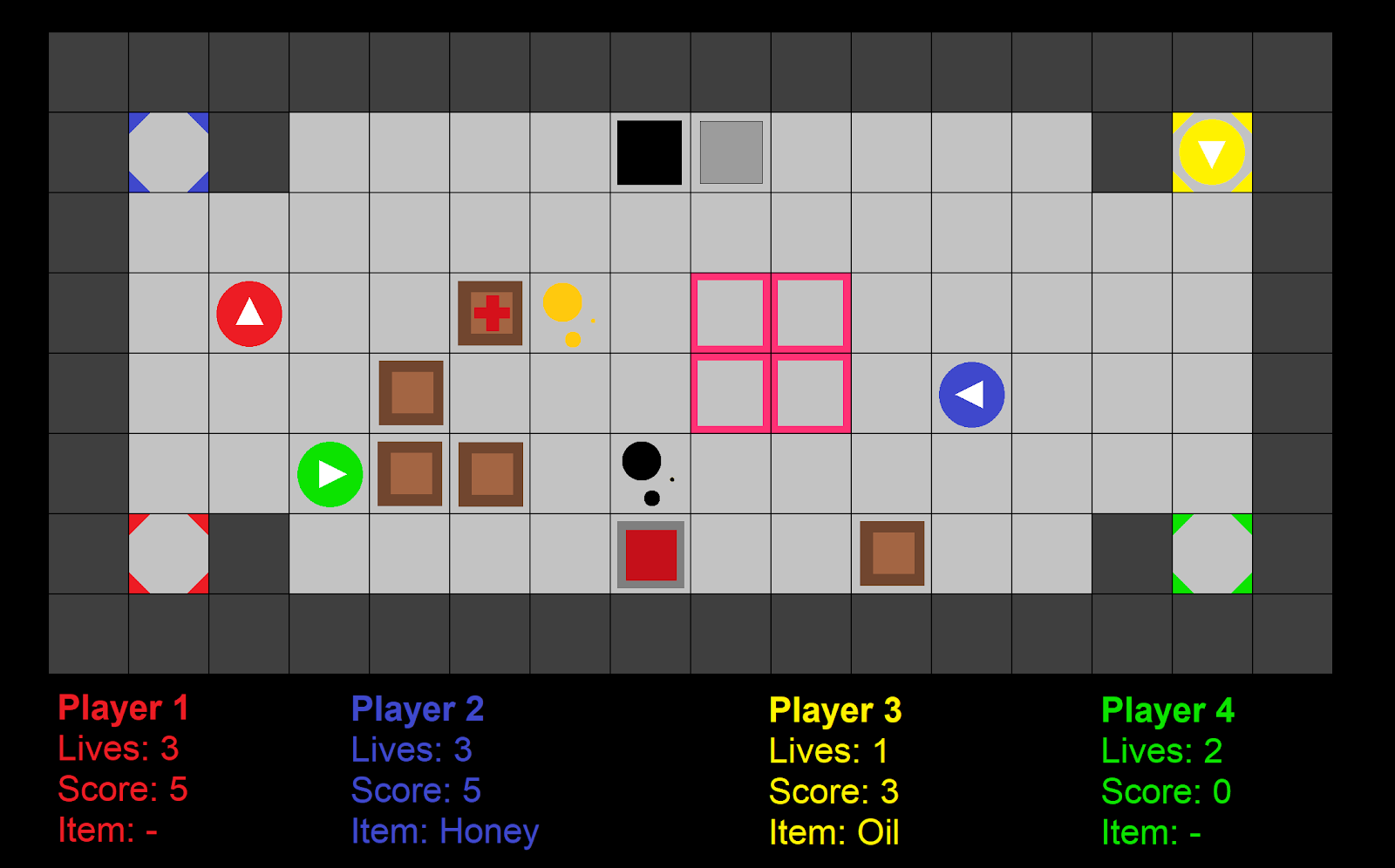
# Grafikus felület specifikációja

## A grafikus interfész



|  |  |  |  |
| --- | --- | --- | --- |
| 9d7moo8GULhgfl_wXWMv7ws-th8b5HLcr4FOVDbo6nSgUiwb9i3whVI4EU7eOQmFyHEHkyrTQZkXoBC5h1uKMy5CMgao-14PqENeKEt20vksgBZKFtauM3QeaJehM1wAgIr9v9FI | Kék dolgozó, a nyíl irányába néz | z3Td37WLSSlnKRwO9bMYEymWl4uKMVUqCkjVJAvfI3Zi-p-KyEgVAQ32uZ_uXnnAe9SLzzLMOiQzqay9XDQrM9hFURUJUe1xLEyfRwey9i2cSRZg8ohsGY2zZtAlNQD8wz-kBTpS | Kijelölt hely |
| 3Gnv1KIIQP_JVDTq-dRP3d57JPh54uOLVWAEHv40oYRRH6mNSlquSJHipkBmWqUMT_8vaP0SjsGZhVs9jTNHNHZh5Tm11V3v7xuGytCEpcd05ssguny1_sz2jBiV8UJ32W9S1JC5 | Üres padló | PER5uusOrBaiNHeZRMemiUykdCB1XF65DoBcpTw9fCjQPEcU0fGt7wul_QXW5McbERz-dAUVpY-cjvSQikARZfvVb8-ILeOQ_U9Z9vX1QkDPanM8ohi_00hS-dbWRauLi1lcrP2b | Padló, melyre mézet raktak |
| Ya2fMKPP47600y-U4IUs9SxAWf_jblWlxmWEwtYOxsSnRuUhvtrWFj9tOmLeqQl2Q6gEJ5QVtt68ez4DaZjyF04He-mOlFI0yfmVW9Hb9Y3t_SGoQ4DaswY8U4TPTNBhFMIuL-WK | Fal | iC1-r4y7PMgGM_5IcDGof8nfxiQnJnj0yzqqJw0tySU_SG7eGWEJHvtST18IDyTxtvtyMuBHjKLR1SJSJeI8bN0R8eEznAZkJTmvRLpDvoK1-38SuI3XZ-pa6cczfi2emdRTk96b | Padló, melyre olajat raktak |
| RcZEN1c3Nb_79GrZ1b5zUaVk35HGxGDXc7xyBOS7iLEkrrbcddmZcKAIoxX-cnss6hLvx1m9r8svXYfKLCp7TmGbzBPCxmzzV6ezf4yDFxQGErknqazlCNncEt4dzrKWx7wa7uCy | A kék dolgozó kiindulási helye | JKJRcgT9ZHEV2PpUcqVhKGTD7X1YtzNpZg5-uvwlBUZFMT3ELI4o84ZGrZisiUkOW4ijsQ1rXxTZQfXyDIOgXVynau3z8c3EIMfkEhS2o1fTNzoQriRj-xW0VyTUYPUwUCttpx39 | Lyuk kapcsolója |
| _sKXEwCUiC-6G6C09xGDcKbVJYG5dMvfhmDqSz6WQYBOp70QEuMPE5e2Z-pbvmwK3Fdf1YCzyVGPsZjPqks1VwuZ0NyyznXz37H0zmsgouf6Ra6xcb61vpy3ZsxJYKAkFBggjyXb | Láda | 32DTywkZ9XoG0JAeF0NR7vap-NJS56ph25yS1GLPUFyP9wc9LoqUj4Liq5QqWdCvOfvB_Z9SSRVSStlbjOB3DKHnGwZlD23ewfrJmyhjBObi7EVFOICMi_-lNLSyHHB1sYeDyBEe | Zárt lyuk |
| -ajl8UOSxc2OD9ODv8TAIo4Erxpru--WuUX6VbG4xMKmuaJaAHqWsiJorjFR1O1vmm0x3VoiBaP6L6F4Q1l-RdtboWoXPiaH2IE-7H67slWU-L_-POs60EhNM7vc62M_uCZJEq90 | Szívecskés láda | sJzwDXzqWuQtxVO5kA9vioFEa3mOiKHbQozOJ27NlFQAFUgxZ8qyDLuUoJ0Bf0B8R-Wq6724WvYXxQ8RIgGxWRLJb-85AsmYlJ8Ujr2v3zrYuO4Wx205iopbgwkIJNGROUZgGbAt | Nyitott lyuk |

## A grafikus rendszer architektúrája

### A felület működési elve

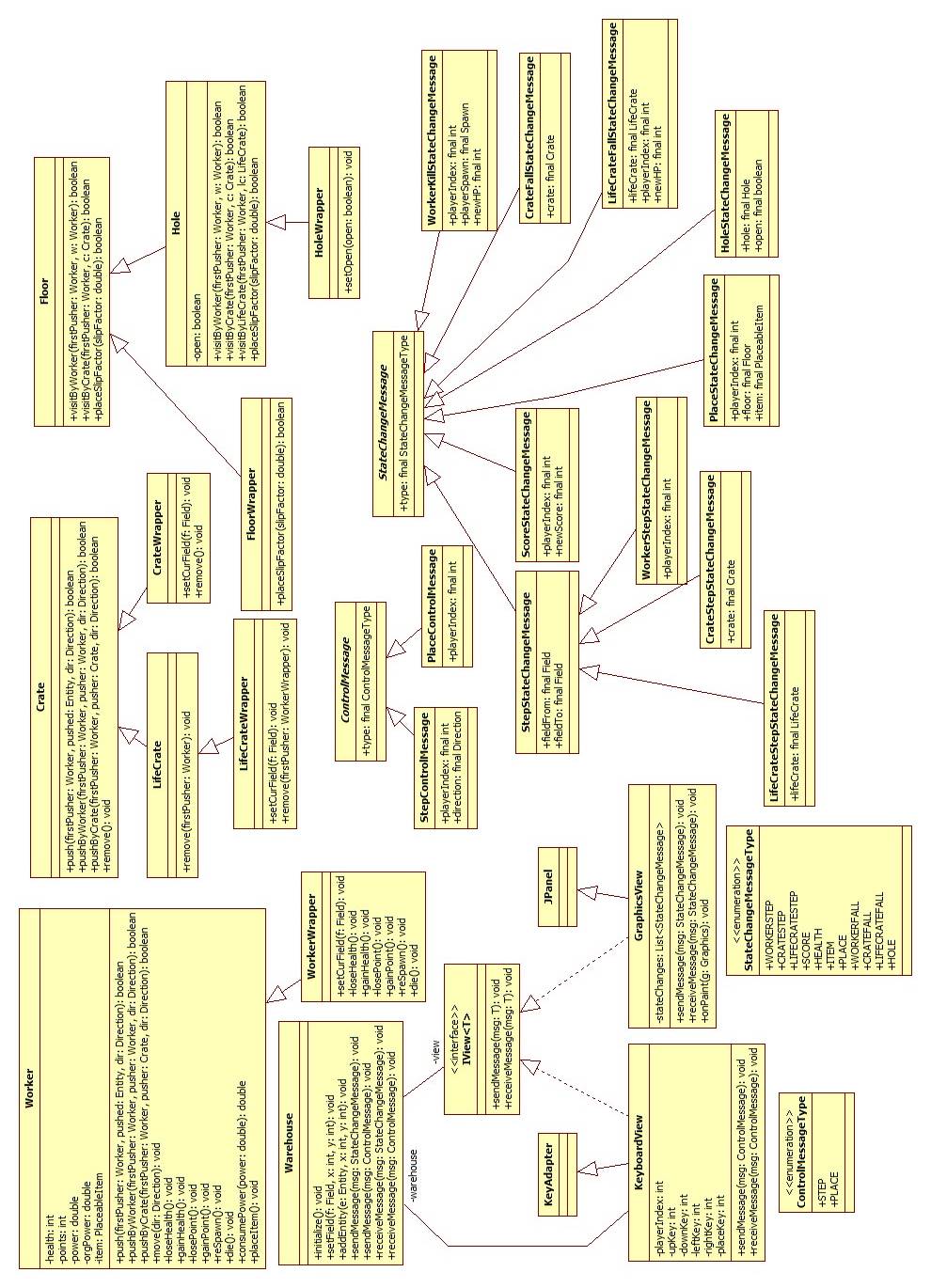
Egy push-alapú értesítési rendszer kialakítását tűztük ki célul: Minden külső elemet érintő művelet esemény értesítést küld a megfelelő komponensnek.

A **bemeneti nézetek** (billentyűzet gombjai) a megfelelő felhasználói bemenetekre felszólítják a modellt bizonyos feladatok/akciók végrehajtására (például: első játékos lépjen balra). A nézet elabsztrahálja a bemenet módját. Nem billentyűnyomás üzeneteket küld, hanem már - a modell számára értelmezhető - parancsokat, így egyszerűen kicserélhető egy billentyűzet alapú irányítás egy speciális játékvezérlőre, hang alapú irányításra, stb.

A **kimeneti nézetek** hasonlóképp tesznek, a változott állapotokról csupán nyers adatot biztosít a modell, annak értelmezése már a nézet feladata, jelen esetben ez egy grafikus megjelenítés, de lehetne akár konzol kimenet, vagy packetek küldése egy szerver-kliens alapú játéknál.

A megvalósításhoz burkolóosztályokat hozunk létre. Minden kommunikálni képes osztályt beburkolunk egy osztályba, mely felülírja azon metódusait, melyek üzenetküldést váltanak ki. Ennek hátránya, hogy az összes példányosítás ezen burkolók cseréjére szorul. Minden modell-béli objektum a Warehouse-on keresztül küldi az üzeneteit, illetve minden view-beli üzenet a Warehouse-nak küldi azt, így egyetlen objektumra szűkítjük a model-view és view-model interfészt.

### A felület osztály-struktúrája



## A grafikus objektumok felsorolása

### WorkerWrapper

#### Felelősség

Értesítést küld a Warehouse osztálynak a dolgozó olyan cselekvéseiről, melyek hatással vannak a grafikus felületre, azaz változtatnak annak állapotán. Felülírja a Worker osztály azon metódusait, melyek grafikai változtatást eredményeznek: delegálja a hívást az ősnek, majd ezután üzenetet küld a változtatás szükségességéről.

#### Ősosztályok

Worker -> Entity

#### Interfészek

-

#### Attribútumok

* **-**

#### Metódusok

* + **setCurField(f: Field): void**: A dolgozó egy új mezőre lépett, át kell állítani a figuráját oda.
* + **gainPoint(): void**: A dolgozó pontot szerzett, frissíteni kell a számlálót.
* + **losePoint(): void**: A dolgozó pontot vesztett, frissíteni kell a számlálót.
* + **gainHealth(): void**: A dolgozó életet szerzett, frissíteni kell a számlálót.
* + **loseHealth(): void**: A dolgozó életet szerzett, frissíteni kell a számlálót.
* + **reSpawn(): void:** A dolgozó újraéled, így át kell állítani a figuráját a spawn-jára.
* + **die(): void**: A dolgozó meghalt, ki kell venni a játékból.

### CrateWrapper

#### Felelősség

Értesítést küld a Warehouse osztálynak a láda olyan cselekvéseiről, melyek hatással vannak a grafikus felületre, azaz változtatnak annak állapotán. Felülírja a Crate osztály azon metódusait, melyek grafikai változtatást eredményeznek: delegálja a hívást az ősnek, majd ezután üzenetet küld a változtatás szükségességéről.

#### Ősosztályok

Crate -> Entity

#### Interfészek

-

#### Attribútumok

* **-**

#### Metódusok

* **+setCurField(f: Field):void:** A láda egy új mezőre lépett, át kell állítani a figuráját.
* **+ remove(): void:** A láda leesett, ki kell venni a játékból.

### LifeCrateWrapper

#### Felelősség

Értesítést küld a Warehouse osztálynak a szívecskés láda olyan cselekvéseiről, melyek hatással vannak a grafikus felületre, azaz változtatnak annak állapotán. Felülírja a LifeCreate osztály azon metódusait, melyek grafikai változtatást eredményeznek: delegálja a hívást az ősnek, majd ezután üzenetet küld a változtatás szükségességéről.

#### Ősosztályok

LifeCrate -> Crate -> Entity

#### Interfészek

-

#### Attribútumok

* **-**

#### Metódusok

* **+setCurField(f: Field):void:** A láda egy új mezőre lépett, át kell állítani a figuráját.
* **+ remove(firstPusher: Worker): void:** A láda leesett, ki kell venni a játékból.

### FloorWrapper

#### Felelősség

Értesítést küld a Warehouse osztálynak a padló olyan cselekvéseiről, melyek hatással vannak a grafikus felületre, azaz változtatnak annak állapotán. Felülírja a Floor osztály azon metódusait, melyek grafikai változtatást eredményeznek: delegálja a hívást az ősnek, majd ezután üzenetet küld a változtatás szükségességéről.

#### Ősosztályok

Floor -> Field

#### Interfészek

-

#### Attribútumok

* **-**

#### Metódusok

* **+placeSlipFactor(double slipFactor):boolean**: A padlóra valamilyen kenőanyag került, frissíteni kell a kinézetét.

### HoleWrapper

#### Felelősség

Értesítést küld a Warehouse osztálynak a lyuk olyan cselekvéseiről, melyek hatással vannak a grafikus felületre, azaz változtatnak annak állapotán. Felülírja a Hole osztály azon metódusait, melyek grafikai változtatást eredményeznek: delegálja a hívást az ősnek, majd ezután üzenetet küld a változtatás szükségességéről.

#### Ősosztályok

Hoel -> Floor -> Field

#### Interfészek

-

#### Attribútumok

* **-**

#### Metódusok

* + **setOpen(boolean open):void**: A lyuk állapota megváltozik, át kell rajzolni a kinézetét.

### *IView<T>*

#### Felelősség

Implementálja minden nem modellhez tartozó, de a modellel interakcióba lépő osztály.

#### Ősosztályok

-

#### Metódusok

* **+ sendMessage (msg: T): void:** Üzenetet küld a modellnek.
* **+ receiveMessage (msg: T): void:** Üzenetet fogad a modelltől.

### KeyboardView

#### Felelősség

Egy grafikus elemhez kapcsolva figyeli a billentyűzet lenyomott gombjait. Ha a lenyomott gomb a nézethez tartozó játékos egyik irányításához tartozó gombja, üzenettel szólítja fel a modellt - a Warehouse-on keresztül - a megfelelő akció végrehajtására. Például 4 játékos esetén 4 darab keyboard view van.

#### Ősosztályok

KeyAdapter

#### Interfészek

IView<ControlMessage>

#### Attribútumok

* - **playerIndex: int**: A felügyelt játékos indexe.
* - **upKey: int**: A játékos felfele lépés gombjának kódja.
* - **downKey: int**: A játékos lefele lépés gombjának kódja.
* - **leftKey: int**: A játékos balra lépés gombjának kódja.
* - **rightKey: int**: A játékos jobbra lépés gombjának kódja.
* - **placeKey: int**: A játékos tárgy-lerakás gombjának kódja.
* - **warehouse: Warehouse**: A játékteret képező raktár, mely felelős az üzenetek továbbításáért.

#### Metódusok

* + **sendMessage (msg: ControlMessage): void**: Ha a lenyomott billentyű irányítja valamire a felügyelt játékost, a megfelelő ControlMessage-re fordítja és elküldi a modellnek.
* + **receiveMessage (msg: ControlMessage): void**: Nem szükséges.

### GraphicsView

#### Felelősség

Az állapotváltozás üzenetekből állít elő egy grafikus ábrát a hozzá tartozó ablakba.

#### Ősosztályok

JPanel

#### Interfészek

IView<StateChangeMessage>

#### Attribútumok

* - **stateChanges: ArrayList<StateChangeMessage>**: Tárolja az üzeneteket a kirajzolás pillanatáig. (onPaint() kiüríti)

#### Metódusok

* + **sendMessage (msg: StateChangeMessage): void**: Nem szükséges.
* + **receiveMessage (msg: StateChangeMessage): void**: Eltárolja az üzenetet a *stateChanges* listába.
* + **onPaint (Graphics g): void**: Kirajzolja az állapotváltozások alapján az új megjelenítendő képet, üríti a *stateChanges* listát.

### ControlMessageType <<Enumeration>>

#### Felelősség

Mivel az üzenetek POD-ok, a típuslekérdezés a legegyszerűbb és legmegfelelőbb módja a konkrét üzenettípus meghatározására, ez az enumeráció a ControlMessage-k pontos típusait írja le: STEP, PLACE.

### *ControlMessage*

#### Felelősség

Az irányítás üzenetek őse, a munkásokat utasítja.

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

* + **type: final MessageType**: Az kontroll üzenet konkrét típusa.

#### Metódusok

* -

### StepControlMessage

#### Felelősség

Lépésre utasítja az egyik munkást.

#### Ősosztályok

ControlMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A lépésre utasított munkás indexe.
* + **direction: final Direction**: Az irány, melybe a munkást lépésre utasítja.

#### Metódusok

* -

### PlaceControlMessage

#### Felelősség

A munkásnál levő tárgy letevésére utasítja őt.

#### Ősosztályok

ControlMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A letevésre utasított munkás indexe.

#### Metódusok

* -

### StateChangeMessageType <<Enumeration>>

#### Felelősség

Mivel az üzenetek POD-ok, a típuslekérdezés a legegyszerűbb és legmegfelelőbb módja a konkrét üzenettípus meghatározására, ez az enumeráció a StateChangeMessage-k pontos típusait írja le: WORKERSTEP, CRATESTEP, LIFECRATESTEP, SCORE, HEALTH, ITEM, PLACE, WORKERFALL, CRATEFALL, LIFECRATEFALL, HOLE.

### *StateChangeMessage*

#### Felelősség

Az irányítás üzenetek őse.

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

* + **type: final StateChangeMessageType**: Az állapotváltozás üzenet konkrét típusa

#### Metódusok

* -

### StepStateChangeMessage

#### Felelősség

Közös ős ami entitások kirajzolásáért felelős.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A lépő dolgozó indexe.
* + **fieldFrom: final Field**: A mező, melyikről a dolgozó lelépett.
* + **fieldTo: final Field**: A mező, melyikre a dolgozó lép.

#### Metódusok

* -

### WorkerStepStateChangeMessage

#### Felelősség

Egy dolgozót rajzol ki a felületen egy megadott pozícióba, ezzel animálva egy lépést.

#### Ősosztályok

StepStateChangeMessage -> StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A lépő dolgozó indexe.

#### Metódusok

* -

### CrateStepStateChangeMessage

#### Felelősség

Egy ládát rajzol ki a felületen egy megadott pozícióba, ezzel animálva egy lépést.

#### Ősosztályok

StepStateChangeMessage -> StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **crate: final Crate**: A tolt láda.

#### Metódusok

* -

### LifeCrateStepStateChangeMessage

#### Felelősség

Egy szívecskés ládát rajzol ki a felületen egy megadott pozícióba, ezzel animálva egy lépést.

#### Ősosztályok

StepStateChangeMessage -> StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **lifeCrate: final LifeCrate**: A tolt szívecskés láda.

#### Metódusok

* -

### CrateFallStateChangeMessage

#### Felelősség

Egy láda lyukba esését animálja.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **crate: final Crate**: A leesett láda.

#### Metódusok

* -

### LifeCrateFallStateChangeMessage

#### Felelősség

Egy szívecskés láda lyukba esését animálja.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + lifeCrate: final LifeCrate: A leesett szívecskés láda.

#### Metódusok

* -

### ScoreStateChangeMessage

#### Felelősség

Egy dolgozót pontszámát változtatja a kijelző felületén.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A pontot kapó vagy veszítő játékos indexe.
* + **newScore: final int:** A játékos új pontszáma.

#### Metódusok

* -

### HealthStateChangeMessage

#### Felelősség

Egy dolgozót életeinek számát változtatja a kijelző felületén.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A pontot kapó vagy veszítő játékos indexe.
* + **newScore: final int:** A játékos új pontszáma.

#### Metódusok

* -

### ItemStateChangeMessage

#### Felelősség

Egy dolgozónál lévő tárgyat változtatja a kijelző felületén.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: Az tárgyat kapó vagy lerakó játékos indexe.
* + **item: final PlaceableItem:** A játékos dolgozójánál lévő tárgy.

#### Metódusok

* -

### PlaceStateChangeMessage

#### Felelősség

Egy tárgy lerakását animálja.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **floor: final Floor**: A padló, amelyre lerakjuk a tárgyat.
* + **item: final PlaceableItem:** A padlóra lerakandó tárgy.

#### Metódusok

* -

### HoleStateChangeMessage

#### Felelősség

Egy lyuk állapotváltozását reprezentálja.

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **hole: final Hole**: A pontot kapó vagy veszítő játékos indexe.
* + **open: final boolean:** A lyuk új állapota.

#### Metódusok

* -

### WorkerKillStateChangeMessage

#### Felelősség

A játékos meghalását reprezentálja (lyukba esi .

#### Ősosztályok

StateChangeMessage

#### Interfészek

-

#### Attribútumok

* + **playerIndex: final int**: A leesett dolgozóhoz tartozó játékos indexe.
* + **playerSpawn: final Spawn:** A leesett dolgozóhoz tartozó kiindulási hely.
* + **newHP: final int:** A leesett dolgozóhoz új csökkentett élete.

#### Metódusok

* -

### Warehouse

#### Felelősség

Az osztály abban változik, hogy ő felelős az üzenetek továbbításáért a csomagoló osztályok és a View-k között.

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

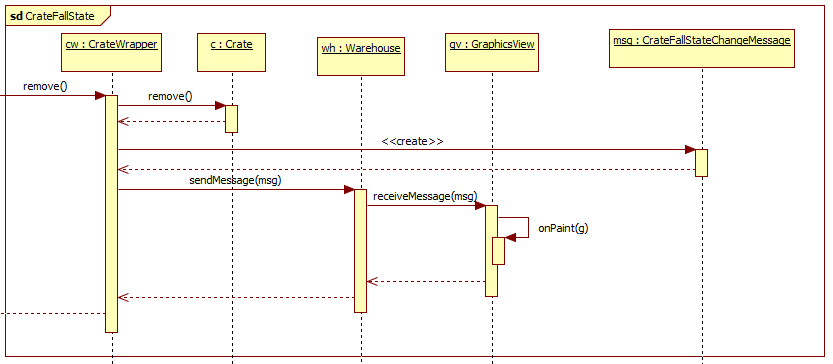
* **- view: IView<T>**

#### Metódusok

* + **sendMessage (msg: StateChangeMessage): void**: A raktár üzenetet továbbít.
* **+ sendMessage (msg: ControlMessage): void**: A raktár üzenetet továbbít.
* **+ receiveMessage (msg: StateChangeMessage): void**: A raktár üzenetet fogad.
* **+ receiveMessage (msg: ControlMessage): void**: A raktár üzenetet fogad.

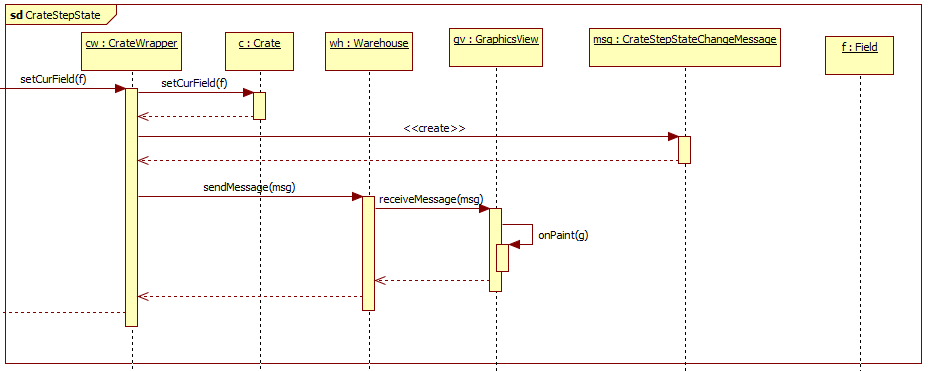
## Kapcsolat az alkalmazói rendszerrel

### CrateFallState



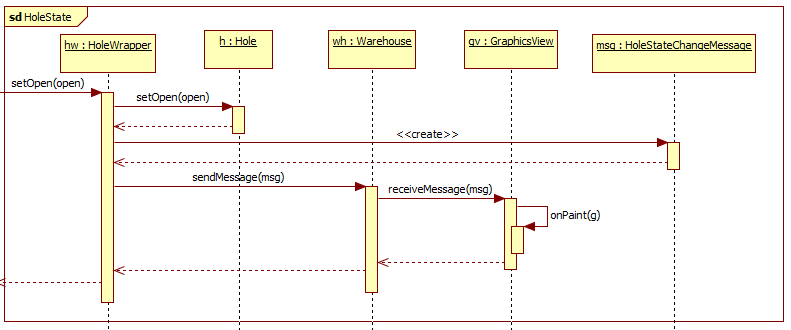
*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### CrateStepState



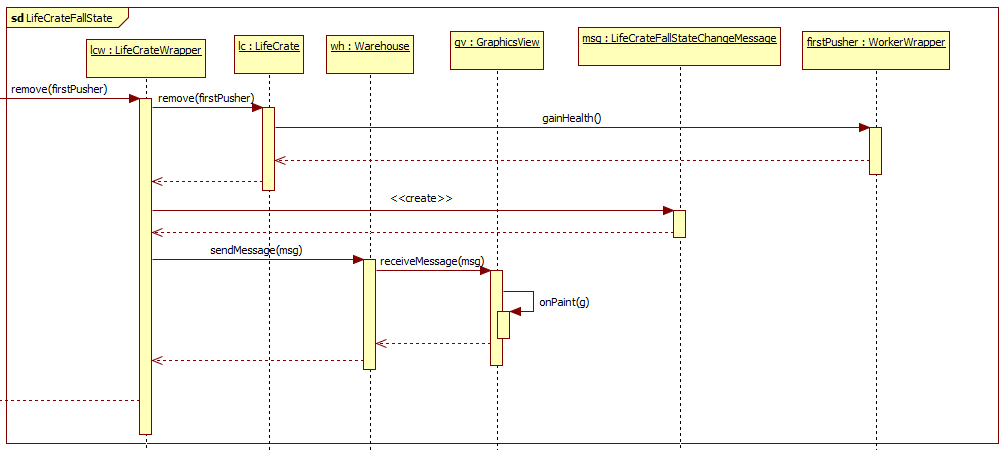
*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### HoleState



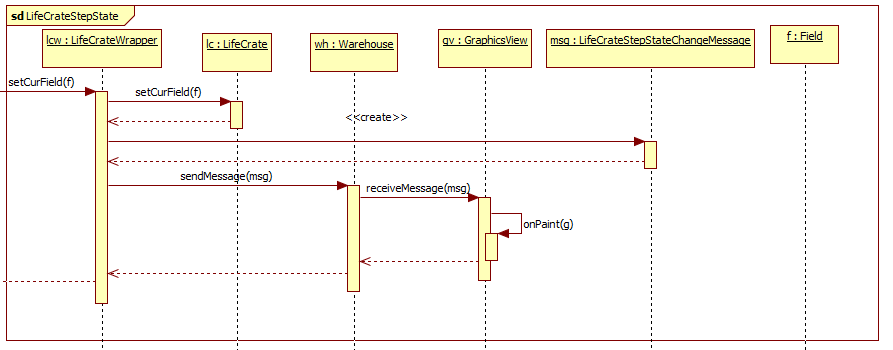
*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### LifeCrateFallState



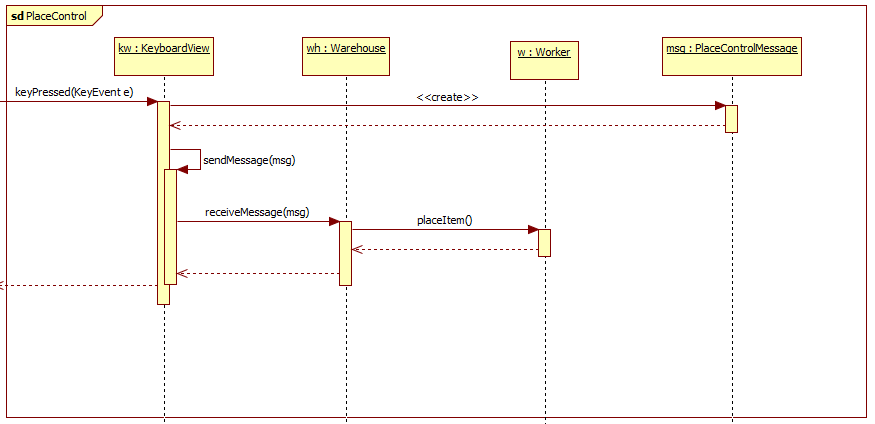
*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### LifeCrateStepState

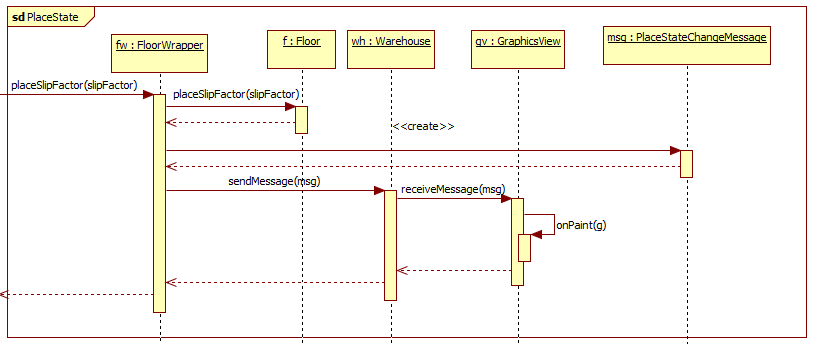


*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### PlaceControl

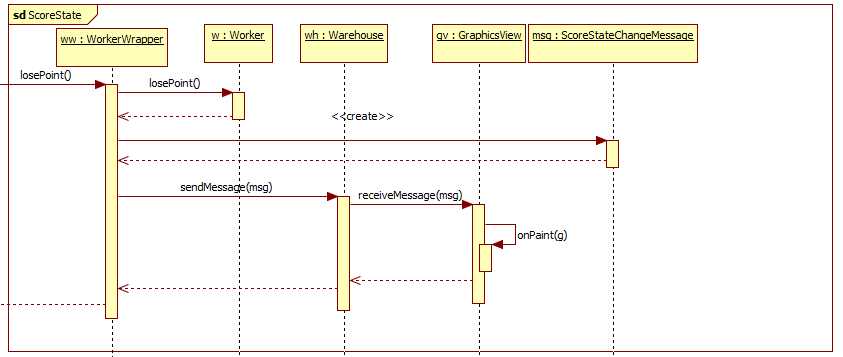


### PlaceState



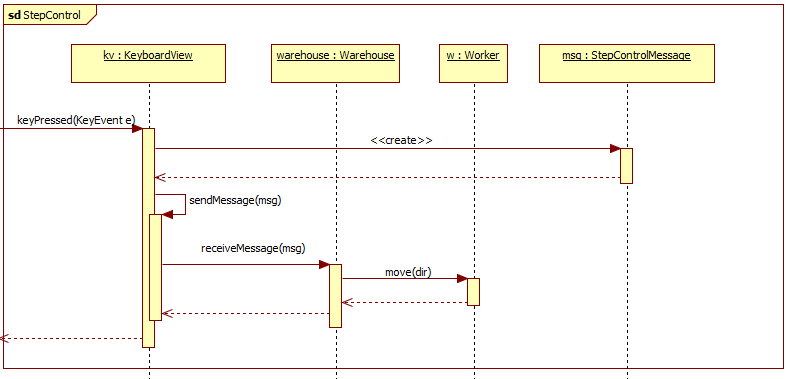
*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### ScoreState

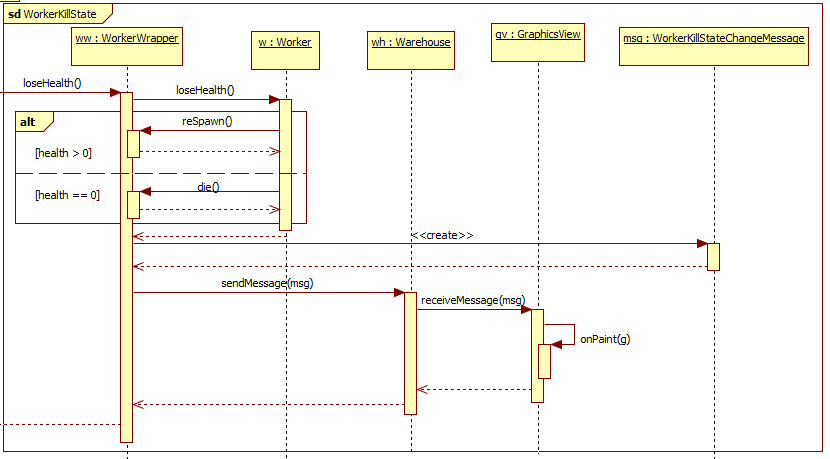


*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### StepControl

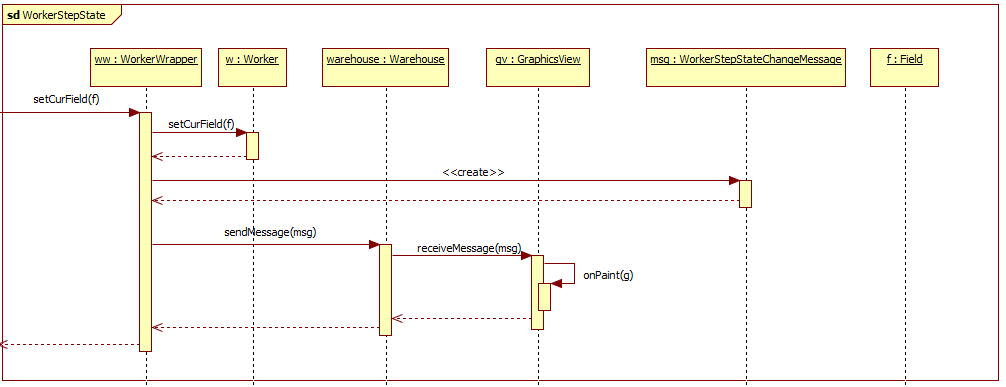


### WorkerKillState



*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

### WorkerStepState



*Repaint függvényt hívunk, és annak hatására fog meghívódni az onPaint, nem közvetlen mi hívjuk.*

## Napló

|  |  |  |  |
| --- | --- | --- | --- |
| **Dátum** | **Időtartam** | **Résztvevők** | **Leírás** |
| 2018.01.04-05 | 3 óra | JANI | Osztályok leírása. |
| 2018.01.04-05 | 3 óra | SZAKÁLLAS | Szekvenciadiagramok. |
| 2018.01.04-05 | 3 óra | CSANÁDY | Osztályok leírása. |
| 2018.01.04-05 | 3 óra | LENKEFI | Grafikus interfész és a felület működési elve. |
| 2018.01.04-05 | 3 óra | LAKATOS | Osztálydiagram, osztályok leírása, adminisztráció. |