Fungorium

68 – nullpointerexception

Konzulens:

Simon Balázs

8. Részletes tervek

Csapattagok

Fórián-Szabó Bernát Zsigmond	ABIVEP	forianzsiga@gmail.com
Gyárfás Réka	AFVLHM	gyarfas.reka@gmail.com
Kemecsei Kornél	HDB6X9	kornel.kemecsei@edu.bme.hu
Kuzmin Iván Georgijevics	U0725D	ikuzmin@edu.bme.hu
Tóth Mihály Balázs	OAYOF1	toth.misi05@gmail.com

2025.04.14.

8.1.1 App

Felelősség

Az App osztály a Fungorium prototípus fő belépési pontja, amely kezeli az alkalmazás futását, a felhasználói parancsok értelmezését és végrehajtását. Inicializálja a játékosokat (InsectPlayer, FungusPlayer), a térképet (Map) és a parancslistát (CommandList). A run() metódus egy konzolos eseménykezelő ciklust indít, amely feldolgozza a felhasználó parancsait és véletlenszerű kilépési üzenettel zárja le a programot.

Ősosztályok

java.lang.Object

Interfészek

Nem valósít meg interfészt.

Attribútumok

- running: boolean
Az alkalmazás fő ciklusának vezérlésére szolgáló logikai változó. A run() metódus akkor áll le, ha ez false.
- commands: CommandList
A parancsokat tartalmazó objektum, amely felelős a parancsok leképezéséért és végrehajtásáért.
- map: Map
A játékhoz használt térkép reprezentációja.
- insectPlayer: InsectPlayer
A rovar játékos példánya, amely a UseCase loggolásához is regisztrálva van.
- fungusPlayer: FungusPlayer
A gombás játékos példánya, hasonlóan inicializálva és regisztrálva, mint az insectPlayer.

Metódusok

+ App(): void
Konstruktor, amely inicializálja a játékosokat, és logolási szempontból regisztrálja őket a UseCase.logger segítségével.
+ InsectPlayer getInsectPlayer(): InsectPlayer
Visszaadja az insectPlayer példányt.
Láthatóság: **public (+)**
+ FungusPlayer getFungusPlayer(): FungusPlayer
Visszaadja a fungusPlayer példányt.
Láthatóság: **public (+)**
+ Map getMap(): Map
Visszaadja a jelenlegi térképet.
Láthatóság: **public (+)**
+ CommandList getCommands(): CommandList
Visszaadja a commands példányt.
Láthatóság: **public (+)**
+ void setMap(Map map): void
Beállítja az aktuális térképet.
Láthatóság: **public (+)**
+ void run(): void
Az alkalmazás fő ciklusát vezérli. Inicializálja a CommandList-et, majd egy végtelen ciklusban parancsokat olvas be és hajt végre, amíg a running attribútum true. A ciklus végén

egy véletlenszerű, humoros kilépőüzenetet ír ki.
Láthatóság: **public (+)**

8.1.2 AcidTile

Felelősség

Az AcidTile osztály a térkép egyik speciális mezőtípusa, amely savas tulajdonságokkal rendelkezik, és minden rajta tartózkodó entitást (GameEntity) folyamatosan sebez a megadott sebzési rátával minden ciklusban. Örökli az alap Tile viselkedést, de kiegészíti azt entitások sebezésével.

Ősosztályok

map.Tile

Interfészek

Nem valósít meg interfészt.

• Attribútumok

- - damageRate: int
  Meghatározza, hogy az adott mezőn található entitások mennyi sebzést kapnak minden frissítési ciklusban. A damage() metódus hívása damageRate-szer ismétlődik minden entitásra.

• Konstruktor

- + AcidTile(int growthRate, int maxMycelium, Tekton parentTekton, int damageRate)
  Inicializálja az AcidTile mezőt az adott növekedési rátával, maximális micéliummennyiséggel, a szülő Tekton példánnyal és sebzési rátával.
  Meghívja a szülő (Tile) konstruktorát az első három paraméterrel, majd beállítja a damageRate attribútumot.

• Metódusok

- - void damageEntities()
  A mezőn található minden GameEntity példányra annyiszor hívja meg a damage() metódust, ahányat a damageRate meghatároz.
  Láthatóság: **private (-)**
- @Override + void update()
  Először meghívja a damageEntities() metódust, majd a szülőosztály update() metódusát. Ez biztosítja, hogy az entitások sebzése minden frissítési ciklusban megtörténjen, mielőtt az alap frissítési logika lefutna.
  Láthatóság: **public (+)**
  Felülírja a Tile osztály update() metódusát.

- ### CannotGrowBodyOnTekton

Felelősség

A **CannotGrowBodyOnTekton** osztály a **UseCase** absztrakt osztályból származik, és azt a szituációt modellezi, amikor egy **FungusPlayer** nem tud testet (gomba testet) növeszteni egy **Tekton**-on. Az osztály az adott feltételek mellett ellenőrzi, hogy a **FungusPlayer** képes-e növeszteni a gomba testet a **Tekton** adott területén (szomszédos **Tile**-ok).

Ősosztályok

- **UseCase**: Az osztály a **UseCase** absztrakt osztályból öröklődik, és megvalósítja annak **execute()** metódusát.

Attribútumok

- Az osztálynak nincs egyéb attribútuma az örökölt **id**, **name**, és **logger** mellett.
- **Konstruktor**
- **CannotGrowBodyOnTekton()**: Az osztály konstruktora, amely az ősosztály konstruktorát hívja meg az ID és a név paraméterekkel (ID: 16, név: "Cannot Grow Body on Tekton"). Ez a konstruktor inicializálja az osztályt.

Metódusok

- **execute()**: A **CannotGrowBodyOnTekton** konkrét implementációja a **UseCase** absztrakt metódusának. Ez a metódus az alábbi lépéseket hajtja végre:
    1. **Színpad inicializálása**: Nyomtatja a "Initializing scene..." üzenetet, és beállítja a megfelelő irányt és behúzást.
    2. **Térkép inicializálása**: Létrehoz egy új **Map** objektumot.
    3. **Tekton inicializálása**: Létrehoz egy új **Tekton** objektumot, amely hozzáadódik a térképhez.
    4. **Csempék inicializálása**: Két **Tile** objektumot hoz létre (t1 és t2), amelyeket a **Tekton** objektumhoz ad, és beállítja a szülő tektonját a csempékhez.
    5. **Játékos inicializálása**: Létrehoz egy új **FungusPlayer** objektumot.
    6. **Test növesztése**: A **FungusPlayer** megpróbál gomba testet növeszteni az első csempén (t1).
    7. **Második próbálkozás**: A **FungusPlayer** megpróbál gomba testet növeszteni a második csempén (t2) ugyanazon **Tekton** területén, ami nem sikerül, mert az szabályellenes.

8.1.3 Command (absztrakt osztály)

Felelősség

A Command osztály az alkalmazásban futtatható parancsok absztrakt ősosztálya. Feladata a parancsok nevének, leírásának, használatának tárolása, bemeneti érvényesítések és az entitások ID alapú keresésének biztosítása. Minden konkrét parancs (pl. CreateMapCommand, SpawnFungusCommand, stb.) ebből az osztályból származik.

Attribútumok

- name – A parancs neve (pl. "create", "spawn").
- description – A parancs rövid leírása.
- usage – A parancs szintaktikája, ha eltér a névtől.
- app – Az aktuális App példány, amelyen keresztül a térkép és más erőforrások elérhetők.
- scanner – A bemeneti adatokat beolvasó Scanner.
- entityId – Statikus azonosítószámláló az entitásokhoz (GameEntity példányokhoz).

Metódusok

- abstract boolean execute(String[] args)
  A konkrét parancsoknak ezt a metódust kell implementálniuk, hogy végrehajtsák a parancs logikáját.
- boolean isWrongNumberOfArgs(int expected, int got)
  Ellenőrzi, hogy a kapott argumentumszám megegyezik-e az elvárt számmal.

- boolean isMapUninitialized()
  Ellenőrzi, hogy van-e már példányosítva térkép (Map), különben hibaüzenetet ad.
- GameEntity assignId(GameEntity entity)
  Egyedi ID-t rendel egy entitáshoz, és be is jelenti a konzolra.
- Integer parsePositiveNumber(String input, String forWhat)
  Pozitív egész számként próbál értelmezni egy bemenetet.
- <T extends GameEntity> T findEntityWithId(int id, String forWhat)
  Megkeresi az adott ID-val rendelkező entitást a térképen.
- <T extends GameEntity> T parseEntityId(String input, String forWhat)
  Kombinált metódus: értelmezi az ID-t és megkeresi az entitást.
- Tekton parseTekton(String input, String forWhat)
  Egy Tekton objektumot keres ID alapján.
- TektonAndTile parseTektonAndTile(String inputTekton, String inputTile)
  Kombinált metódus, amely egy Tekton-t és annak egy Tile-ját adja vissza ID alapján.
- setApp(App app) és setScanner(Scanner scanner)
  Beállítók az alkalmazás- és bemeneti környezethez.

Belső osztály: TektonAndTile
Egy egyszerű adatstruktúra, amely egy Tekton-t és egy Tile-t együttesen tárol, hogy könnyebben lehessen hivatkozni rájuk együtt.

8.1.4 CommandList

Felelősség

A CommandList osztály felelős a programban elérhető összes parancs regisztrálásáért és tárolásáért, valamint a parancsok lekérdezéséért név alapján. Az osztály biztosítja, hogy minden parancs hozzáférjen az alkalmazás állapotához (App) és a bemeneti folyamhoz (Scanner).

Attribútumok

- Map<String, Command> commands
  Egy HashMap, amely a parancsokat tárolja: a kulcs a parancs neve (String), az érték maga a parancs (Command).

Konstruktor

A konstruktor során:
1. Minden támogatott parancs példányosítása megtörténik.
2. A commands térképen keresztül regisztrálásra kerülnek a name mező alapján.
3. Beállításra kerül az aktuális App és Scanner példány minden parancshoz.

Metódusok

- private void add(Command cmd)
  Egy parancsot ad a commands térképhez a neve (getName()) alapján.
- public Command get(String command)
  Visszaadja a parancs objektumot a parancs nevének (String) megadása alapján.
- public Collection<Command> getCommands()
  Visszaadja az összes regisztrált parancsot egy gyűjteményként (hasznos pl. list parancshoz).

8.1.5 Create

Felelősség

Az Create osztály felelőssége új entitások vagy térképelemek létrehozása a játékban. Az osztály lehetővé teszi különböző típusú objektumok, például szporák, testek, és térképelemek (mint például AcidTile, HealTile, Map, stb.) létrehozását és hozzáadását a térképhez. A parancs végrehajtása során a felhasználó különböző adatokat ad meg az új elemek létrehozásához, például a térképhez tartozó információkat vagy egyéb attribútumokat, mint például a növekedési sebesség.

Ősosztályok

- **Object → Command**

Interfészek

- Nem implementál interfészeket.

Attribútumok

- **validTypes**: A típusokat tartalmazó lista, amelyek alapján új entitásokat vagy térképelemeket lehet létrehozni. (láthatóság: private, típusa: String[])

Metódusok

- **Create()**: Az osztály konstruktora, amely inicializálja a Create parancsot, beállítva annak nevét, leírását és szintaxisát. (láthatóság: public)
- **promptForPositiveInteger(String forWhat): Integer**: Kér egy pozitív egész számot a felhasználótól egy adott kontextusban. (láthatóság: private)
- **promptForTekton(String forWhat): Tekton**: Megkérdezi a felhasználótól egy tektonikus lemez ID-ját és visszaadja a megfelelő Tekton objektumot. (láthatóság: private)
- **promptForTektonAndTile(): TektonAndTile**: Megkérdezi a felhasználótól a tektonikus lemez és a hozzá tartozó tile ID-ját, majd visszaadja a megfelelő TektonAndTile objektumot. (láthatóság: private)
- **promptForTileData(): TileData**: Kérdéseket tesz fel a felhasználónak a TileData attribútumok megadásához, beleértve a növekedési sebességet és a maximális myceliumot, majd visszaadja a megfelelő TileData objektumot. (láthatóság: private)
- **execute(String[] args): boolean**: Végrehajtja a parancsot a megadott argumentumok alapján. A parancs létrehoz különböző típusú entitásokat vagy térképelemeket a paramétereknek megfelelően. A metódus a következő típusokat támogatja: map, cutspore, freezespore, slowspore, speedupspore, fungusbody, insect, mycelium, tekton, tile, acidtile, healtile, monotile. (láthatóság: public)

Osztályok

- **TileData**: Az osztály egy belső osztály, amely három attribútummal rendelkezik: growthRate, maxMycelium, és parentTekton. Az osztály célja, hogy azokat az adatokat tárolja, amelyek egy Tile létrehozásához szükségesek. (láthatóság: public final class)
    - **getGrowthRate(): int**: Visszaadja a növekedési sebességet. (láthatóság: public)
    - **getMaxMycelium(): int**: Visszaadja a maximális myceliumot. (láthatóság: public)
    - **getParentTekton(): Tekton**: Visszaadja a szülő tektonikus lemezt. (láthatóság: public)

- **TektonAndTile**: Egy segédosztály, amely a Tekton és Tile objektumokat tárolja, hogy azokat együttesen kezelhessük.

Áttekintés

A Create osztály végrehajtja a parancsokat a felhasználó által megadott argumentumok alapján, és különböző típusú entitásokat vagy térképelemeket hoz létre a játékban. Az osztály kéri a szükséges adatokat a felhasználótól (például ID-k, sebesség, életerő), és ezen információk alapján hajtja végre a megfelelő műveleteket.

8.1.6 CutSpore

Felelősség

A CutSpore osztály a Spore osztályból származik, és annak specifikus típusaként működik. Felelőssége a spóra hatásának alkalmazása egy rovarra, amely elnyomja a rovar vágó képességét, miután elfogyasztja. A CutSpore hatása a rovar vágási képességét ideiglenesen blokkolja, amíg a hatás érvényben van.

Ősosztályok

- **Object**
  - **Spore**
    - **CutSpore**

Interfészek

- A CutSpore nem implementál semmilyen interfészt.

Attribútumok

- id: A spóra egyedi azonosítója. Publikus (+), típus: int.
- currentTile: Az a térkép elem, ahol a spóra található. Publikus (+), típus: Tile.
- nutrientValue: A spóra tápláló értéke. Publikus (+), típus: int.
- lifetime: A spóra élettartama, ami meghatározza, meddig marad aktív. Publikus (+), típus: int.
- effectTime: A hatás időtartama, ami azt jelzi, hogy mennyi ideig befolyásolja a rovar vágó képességét. Publikus (+), típus: int.
- isConsumed: A spóra fogyasztásának állapota. Privát (-), típus: boolean.

Metódusok

- public CutSpore(int id, Tile currentTile, int nutrientValue, int lifetime, int effectTime): A konstruktor, amely inicializálja a spórát az adott értékekkel. Az osztály ősosztályának konstruktorát hívja meg a szükséges paraméterekkel.
- public CutSpore(): Alapértelmezett konstruktor, amely inicializálja a CutSpore objektumot alapértelmezett értékekkel és elvégzi a kezdeti regisztrációt a UseCase.replace(this) segítségével. A UseCase.printWrapper metódus használatával naplózza az objektum inicializálását.
- public void getEaten(Insect i): Felüldefiniált metódus, amely akkor kerül meghívásra, amikor egy rovar megeszi a spórát. A metódus beállítja a isConsumed attribútumot igazra, letiltja a rovar vágó képességét (i.setCut(false)), hozzáadja a spórát a rovar listájához, és eltávolítja a spórát a térképről.
- public void removeEffect(Insect i): Felüldefiniált metódus, amely visszaállítja a rovar vágó képességét (i.setCut(true)) a spóra hatásának eltávolításakor.

8.1.7 Destroy

Felelősség

A Destroy osztály felelős egy adott játékelem eltávolításáért a térképről a parancs végrehajtása során. A Command osztályból öröklődve biztosítja a parancs megfelelő szintaxist és funkciót, hogy egy játékelem eltávolítható legyen a játéktérképről, az entity id alapján.

Ősosztályok

- **Object**
  - **Command**
    - **Destroy**

Interfészek

- A Destroy osztály nem implementál semmilyen interfészt.

Attribútumok

- Nincsenek egyedi attribútumok az osztályban, mivel az osztály a szülőosztály konstruktorát használja a szükséges információk megadásához.

Metódusok

- public Destroy(): Konstruktor, amely beállítja a parancs nevét, leírását és szintaxisát. A szülőosztály (Command) konstruktorát hívja meg, és definiálja a parancs nevét ("destroy"), leírását ("Destroy a given entity"), és szintaxisát ("destroy <entity id>").
- public boolean execute(String[] args): Az execute metódus végrehajtja a "destroy" parancsot. A metódus végrehajtásakor a következő lépéseket hajtja végre:
  1. Ellenőrzi, hogy a paraméterek száma megfelelő-e (két paraméterre van szükség: a parancs és az entity id).
  2. Ellenőrzi, hogy a térkép inicializálva van-e.
  3. Az entity id alapján megpróbálja kinyerni a játékelem objektumot.
  4. Ha a játékelem megtalálható, eltávolítja azt a térképről a getCurrentTile().removeEntity(entity) segítségével.

Ha bármelyik lépés nem sikeres, a metódus false értéket ad vissza, jelezve, hogy a parancs nem hajtódott végre. Az utolsó sorban lévő return false biztosítja, hogy a parancs végrehajtása után a visszatérés biztosított, ha bármilyen probléma történt.

8.1.8 DetachedMyceliumDies

Felelősség

A DetachedMyceliumDies osztály felelős egy mycelium elválásának és halálának szimulálásáért, miközben a térkép és a tekton objektumok kezelésére is kiterjed. Az osztály végrehajtja a mycelium életciklusát, beleértve a frissítéseket, a halált, és végül eltávolítja azt a térképről.

Ősosztályok

- **Object**
  - **UseCase**
    - **DetachedMyceliumDies**

Interfészek

- A DetachedMyceliumDies osztály nem implementál semmilyen interfészt.

Attribútumok

- Nincsenek egyedi attribútumok az osztályban, mivel az osztály a szülőosztály konstruktorát használja a szükséges információk megadásához.

Metódusok

- public DetachedMyceliumDies(): Konstruktor, amely meghívja a szülőosztály (UseCase) konstruktorát, és beállítja az id-t és a nevet, ezzel azonosítva a konkrét use case-t: "Detached Mycelium Dies" az id 11 értékkel.
- public void execute(): Az execute metódus végrehajtja a mycelium elválásának és halálának szimulálását a következő lépésekben:
    1. **Színpad inicializálása**: Az aktuális jelenet inicializálása történik, és a rendszer logolja ezt a műveletet.
    2. **Térkép és Tekton létrehozása**: Létrejön egy új Map objektum, amelyhez egy Tekton is hozzáadásra kerül.
    3. **Tile létrehozása**: Létrejön egy új Tile objektum, és hozzárendelésre kerül a Tekton szülő.
    4. **Mycelium létrehozása és hozzárendelése**: Létrejön egy új Mycelium objektum, amelyet a tile-hoz rendelnek, hogy kölcsönösen hivatkozzanak egymásra.
    5. **Tick szimulálása**: Az mycelium.update() metódust 5 alkalommal meghívják, hogy szimulálják a mycelium állapotának változását, beleértve az egészségi állapotát.
    6. **Mycelium halála**: Ha a mycelium egészségi állapota 0 vagy annál kisebb, akkor a die() metódus meghívásra kerül, ami a mycelium halálát szimulálja.
    7. **Eltávolítás**: A mycelium eltávolításra kerül a tile-ról, és a térkép egy újabb tick()-et hajt végre, amely végül a mycelium leválásának és kapcsolódásának kezelését is tartalmazza.

8.1.9 EatingCutSpore

Felelősség

Az EatingCutSpore osztály felelős annak a szimulálásáért, amikor egy rovar fogyasztja el a CutSpore-t, amely hatással van a rovar vágási képességére. Az osztály inicializálja a játékbeli jelenetet, létrehozza a szükséges entitásokat, például a rovar játékost, a tektonokat és a különböző spórákat, majd a rovar megeszi a CutSpore-t, amely hatással lesz annak képességeire.

Ősosztályok

- **Object**
    - **UseCase**
        - **EatingCutSpore**

Interfészek

- Az EatingCutSpore osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály nem tartalmaz egyedi attribútumokat, mivel az osztály a szülőosztály (UseCase) konstruktorát használja a szükséges információk megadásához.

Metódusok

- public EatingCutSpore(): Konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja az id-t és a nevet, ezzel azonosítva a konkrét use case-t: "Cut spore affecting insect" az id 7 értékkel.
- public void execute(): Az execute metódus végrehajtja a következő lépéseket:
    1. **Színpad inicializálása**: A játékbeli jelenet inicializálása történik, és a rendszer logolja ezt a műveletet.
    2. **InsectPlayer létrehozása**: Létrejön egy új InsectPlayer objektum, amely felelős a rovarok irányításáért.
    3. **Tekton létrehozása**: Létrejön egy új Tekton objektum, amely a térképet és annak elemeit kezeli.
    4. **Tile létrehozása**: Létrejön egy új Tile objektum, amely a rovar és a spóra helyét reprezentálja.
    5. **Insect létrehozása**: Létrejön egy új Insect objektum, amely a rovar tulajdonságait tárolja.
    6. **CutSpore létrehozása**: Létrejön egy új CutSpore objektum, amely hatással van a rovar képességeire, ha megeszi.
    7. **InsectPlayer-hez adás**: A rovar hozzáadásra kerül az InsectPlayer-hez, hogy irányíthatóvá váljon.
    8. **Tile és Tekton összekapcsolása**: A Tile-t a Tekton-hoz rendelik, hogy a térképen való helye érvényes legyen.
    9. **Entitások hozzáadása a Tile-hoz**: A Tile-hoz hozzáadódik a rovar és a spóra.
    10. **Spóra elfogyasztása**: A rovar megeszi a CutSpore-t, amely hatással lesz annak vágási képességére.

8.1.10 EatingFreezingSpore

Felelősség

Az EatingFreezingSpore osztály felelős annak a szimulálásáért, amikor egy rovar elfogyasztja a FreezeSpore-t, amely hatással van a rovar mozgására, vagy más jellemzőire a játékban. Az osztály inicializálja a szükséges entitásokat, például a rovar játékost, a tektonokat, a spórát, és lehetővé teszi, hogy a rovar elfogyassza a spórát, ami hatással lesz annak képességeire.

Ősosztályok

- **Object**
    - **UseCase**
        - **EatingFreezingSpore**

Interfészek

- Az EatingFreezingSpore osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály nem tartalmaz egyedi attribútumokat, mivel az osztály a szülőosztály (UseCase) konstruktorát használja a szükséges információk megadásához.

Metódusok

- public EatingFreezingSpore(): Konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja az id-t és a nevet, ezzel azonosítva a konkrét use case-t: "Freezing spore affecting insect" az id 6 értékkel.
- public void execute(): Az execute metódus végrehajtja a következő lépéseket:
    1. **Színpad inicializálása**: A játékbeli jelenet inicializálása történik, és a rendszer logolja ezt a műveletet.

2. **InsectPlayer létrehozása**: Létrejön egy új InsectPlayer objektum, amely felelős a rovarok irányításáért.
3. **Tekton létrehozása**: Létrejön egy új Tekton objektum, amely a térképet és annak elemeit kezeli.
4. **Tile létrehozása**: Létrejön egy új Tile objektum, amely a rovar és a spóra helyét reprezentálja.
5. **Insect létrehozása**: Létrejön egy új Insect objektum, amely a rovar tulajdonságait tárolja.
6. **FreezeSpore létrehozása**: Létrejön egy új FreezeSpore objektum, amely hatással van a rovar képességeire, ha megeszi.
7. **InsectPlayer-hez adás**: A rovar hozzáadásra kerül az InsectPlayer-hez, hogy irányíthatóvá váljon.
8. **Tile és Tekton összekapcsolása**: A Tile-t a Tekton-hoz rendelik, hogy a térképen való helye érvényes legyen.
9. **Entitások hozzáadása a Tile-hoz**: A Tile-hoz hozzáadódik a rovar és a spóra.
10. **Spóra elfogyasztása**: A rovar megeszi a FreezeSpore-t, amely hatással lesz annak mozgására vagy más jellemzőire.

8.1.11 EatingSlowSpore

Felelősség

Az EatingSlowSpore osztály felelős annak a szimulálásáért, amikor egy rovar elfogyasztja a SlowSpore-t, amely lassítja a rovar mozgását a játékban. Az osztály inicializálja a szükséges entitásokat, például a rovar játékost, a tektonokat, a spórát, és lehetővé teszi, hogy a rovar elfogyassza a spórát, ami hatással lesz a rovar képességeire.

Ősosztályok

- **Object**
  - **UseCase**
    - **EatingSlowSpore**

Interfészek

- Az EatingSlowSpore osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály nem tartalmaz egyedi attribútumokat, mivel az osztály a szülőosztály (UseCase) konstruktorát használja a szükséges információk megadásához.

Metódusok

- public EatingSlowSpore(): Konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja az id-t és a nevet, ezzel azonosítva a konkrét use case-t: "Slowing spore affecting insect" az id 5 értékkel.
- public void execute(): Az execute metódus végrehajtja a következő lépéseket:
  1. **Színpad inicializálása**: A játékbeli jelenet inicializálása történik, és a rendszer logolja ezt a műveletet.
  2. **InsectPlayer létrehozása**: Létrejön egy új InsectPlayer objektum, amely felelős a rovarok irányításáért.
  3. **Tekton létrehozása**: Létrejön egy új Tekton objektum, amely a térképet és annak elemeit kezeli.
  4. **Tile létrehozása**: Létrejön egy új Tile objektum, amely a rovar és a spóra helyét reprezentálja.

5. **Insect létrehozása**: Létrejön egy új Insect objektum, amely a rovar tulajdonságait tárolja.
6. **SlowSpore létrehozása**: Létrejön egy új SlowSpore objektum, amely lassítja a rovar mozgását.
7. **InsectPlayer-hez adás**: A rovar hozzáadásra kerül az InsectPlayer-hez, hogy irányíthatóvá váljon.
8. **Tile és Tekton összekapcsolása**: A Tile-t a Tekton-hoz rendelik, hogy a térképen való helye érvényes legyen.
9. **Entitások hozzáadása a Tile-hoz**: A Tile-hoz hozzáadódik a rovar és a spóra.
10. **Spóra elfogyasztása**: A rovar megeszi a SlowSpore-t, amely hatással lesz annak mozgására.

8.1.12 EatingSpeedupSpore

Felelősség

Az EatingSpeedupSpore osztály felelős annak a szimulálásáért, amikor egy rovar elfogyasztja a SpeedUpSpore-t, amely felgyorsítja a rovar mozgását a játékban. Az osztály inicializálja a szükséges entitásokat, például a rovar játékost, a tektonokat, a spórát, és lehetővé teszi, hogy a rovar elfogyassza a spórát, ami hatással lesz a rovar képességeire.

Ősosztályok

- **Object**
  - **UseCase**
    - **EatingSpeedupSpore**

Interfészek

- Az EatingSpeedupSpore osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály nem tartalmaz egyedi attribútumokat, mivel az osztály a szülőosztály (UseCase) konstruktorát használja a szükséges információk megadásához.

Metódusok

- public EatingSpeedupSpore(): Konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja az id-t és a nevet, ezzel azonosítva a konkrét use case-t: "Speedup spore affecting insect" az id 4 értékkel.
- public void execute(): Az execute metódus végrehajtja a következő lépéseket:
  1. **Színpad inicializálása**: A játékbeli jelenet inicializálása történik, és a rendszer logolja ezt a műveletet.
  2. **InsectPlayer létrehozása**: Létrejön egy új InsectPlayer objektum, amely felelős a rovarok irányításáért.
  3. **Tekton létrehozása**: Létrejön egy új Tekton objektum, amely a térképet és annak elemeit kezeli.
  4. **Tile létrehozása**: Létrejön egy új Tile objektum, amely a rovar és a spóra helyét reprezentálja.
  5. **Insect létrehozása**: Létrejön egy új Insect objektum, amely a rovar tulajdonságait tárolja.
  6. **SpeedUpSpore létrehozása**: Létrejön egy új SpeedUpSpore objektum, amely felgyorsítja a rovar mozgását.
  7. **InsectPlayer-hez adás**: A rovar hozzáadásra kerül az InsectPlayer-hez, hogy irányíthatóvá váljon.

8. **Tile és Tekton összekapcsolása**: A Tile-t a Tekton-hoz rendelik, hogy a térképen való helye érvényes legyen.
9. **Entitások hozzáadása a Tile-hoz**: A Tile-hoz hozzáadódik a rovar és a spóra.
10. **Spóra elfogyasztása**: A rovar megeszi a SpeedUpSpore-t, amely hatással lesz annak mozgására.

8.1.13 EatingSpore

Felelősség

Az EatingSpore osztály a spórák elfogyasztásának szimulációját célozza meg, de a metódus jelenleg még nem implementált. Az osztály öröklődik a UseCase osztálytól, amely az alapvető use case logikát biztosítja. Az execute metódus szintén jelenleg nem tartalmaz valódi végrehajtást, de a helyére implementálni kell a spórák elfogyasztását.

Ősosztályok

- **Object**
  - **UseCase**
    - **EatingSpore**

Interfészek

- Az EatingSpore osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály nem tartalmaz egyedi attribútumokat, mivel az osztály a szülőosztály (UseCase) konstruktorát használja a szükséges információk megadásához.

Metódusok

- public EatingSpore(): Konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja az id-t és a nevet, ezzel azonosítva a konkrét use case-t: "Eating Spore" az id 3 értékkel.
- public void execute(): A spóra elfogyasztása.

8.1.14 Exit Command

Felelősség

Az Exit osztály a program kilépését kezeli, és az alap Command osztálytól öröklődik. A execute metódus akkor hívódik meg, amikor a felhasználó a exit parancsot adja ki. Az osztály a program futását leállítja, jelezve, hogy a parancs végrehajtása sikeres volt.

Ősosztályok

- **Object**
  - **Command**
    - **Exit**

Interfészek

- Az Exit osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály nem tartalmaz egyedi attribútumokat, mivel a parancsokat a szülőosztály (Command) kezelni.

Metódusok

- public Exit(): Konstruktor, amely meghívja a szülőosztály konstruktorát, beállítva a parancs nevét és leírását. Az exit parancs a program kilépését szolgálja.
- public boolean execute(String[] args): A parancs végrehajtása. Az execute metódus visszatérési értéke true, jelezve, hogy a program sikeresen leállt. Ez a metódus nem használ semmilyen bemeneti argumentumot (args), és az exit parancs végrehajtásakor a program kilép.

Működés

A parancs végrehajtása a program kilépését kezdeményezi. Az execute metódus a következőképpen működik:

- A metódus visszaadja a true értéket, amely jelezheti a program számára, hogy a kilépési parancsot sikeresen végrehajtották.

8.1.15 FreezeSpore osztály

Felelősség

A FreezeSpore osztály egy spóra, amely az elfogyasztása után megakadályozza egy rovar mozgását. Ez a hatás azáltal érhető el, hogy a rovar sebességét -100%-ra állítja, így nem tud mozogni, amíg a hatás tart.

Ősosztályok

- **Object**
  - **Spore**
    - **FreezeSpore**

Interfészek

- Az osztály nem implementál semmilyen interfészt.

Attribútumok

- Az osztály az alap Spore osztály öröklött attribútumait használja, mint például a spóra azonosító, az aktuális tile, a tápanyag értéke, élettartam és hatás időtartam. A konkrét FreezeSpore osztály nem tartalmaz saját attribútumokat, hanem a szülőosztálytól örökli őket.

Metódusok

- **Konstruktorok:**
  - public FreezeSpore(int id, Tile currentTile, int nutrientValue, int lifetime, int effectTime): A konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja a spóra alapértékeit.
  - public FreezeSpore(): A konstruktor, amely a super() hívásával beállítja az alapértelmezett értékeket, valamint a UseCase.replace(this) metódust is hívja, ami valószínűleg nyilvántartja vagy inicializálja a spórát a rendszerben. Ezen kívül egy log üzenetet is kiír, amely tájékoztatja a felhasználót a FreezeSpore inicializálásáról.
- **getEaten(Insect i)**: Ez a metódus felelős azért, hogy amikor a rovar elfogyasztja a spórát, a rovar sebessége -100%-ra csökken, és a rovar nem tud mozogni. A spóra eltávolításra kerül az aktuális tile-ról, és hozzáadódik a rovar spóráihoz.
  - Paraméterek: Insect i – a rovar, amely elfogyasztja a spórát.
  - Hatás: Beállítja a rovar sebességét -100%-ra, jelezve a mozgás letiltását.
- **removeEffect(Insect i)**: Ez a metódus visszaállítja a rovar sebességét alapértékére (0), azaz lehetővé teszi, hogy újra mozogjon.

- Paraméterek: Insect i – a rovar, amelyre alkalmazni kell a hatást.
- Hatás: Visszaállítja a rovar sebességét 0%-ra, ami a normál mozgási sebességet jelenti.

Működés

- **getEaten**: Amikor egy rovar elfogyasztja a FreezeSpore-t, a rovar sebessége -100%-ra csökken, így az nem tud mozogni. Ezen kívül a spóra eltávolításra kerül a tile-ról, és a rovar hozzáadódik a spórához, hogy nyilvántartásban legyen.
- **removeEffect**: Ha a hatás időtartama lejár, vagy valamilyen más okból szükséges, a spóra hatása eltávolításra kerül, és a rovar sebessége visszaáll az alapértékre, lehetővé téve a mozgást.

8.1.16 Fungus osztály

Felelősség

A Fungus osztály egy absztrakt osztály, amely a gombák (fungus) alapvető jellemzőit és viselkedését definiálja. A Fungus osztály a GameEntity szülőosztálytól öröklődik, és biztosítja a gombák egészségét, valamint a haláluk kezelését.

Ősosztályok

- **Object**
  - **GameEntity**
    - **Fungus**

Interfészek

- Az osztály nem implementál semmilyen interfészt.

Attribútumok

- health: A gomba egészségét jelző egész szám, amely meghatározza, hogy mennyi életpontja van a gombának. Ez az érték a gomba "életrevalóságát" reprezentálja.
- **Konstruktorok**
- **protected Fungus(int id, int health, Tile currentTile)**: A konstruktor, amely inicializálja a gomba azonosítóját, egészségét, valamint az aktuális tile-t (helyszínt), ahol a gomba található.
  - Paraméterek:
    - int id: A gomba egyedi azonosítója.
    - int health: A gomba életerejét meghatározó érték.
    - Tile currentTile: Az a térkép (tile), amelyen a gomba található.
- **protected Fungus()**: Az alapértelmezett konstruktor, amely meghívja a szülőosztály konstruktorát, és alapértelmezett értékeket állít be.

Metódusok

- **public void die()**: A die metódus felelős a gomba halálának kezeléséért. A halálkor egy üzenet kerül kiírásra, amely tartalmazza a gomba típusát és a memóriacímét. A halál lekezelését a szülőosztály (GameEntity) nem végzi el, hanem a leszármazott osztályoknak kell meghatározniuk.
  - Hatás: Kiírja a gomba haláláról szóló üzenetet.
- **public int getHealth()**: Ez a metódus visszaadja a gomba aktuális egészségét.
  - Visszatérési érték: int – a gomba életerejét (health) reprezentáló érték.
- **Működés**

- **die**: A die metódus a gomba halálának kezelésére szolgál, és a UseCase.printWrapper segítségével kiírja, hogy a gomba milyen típusú, valamint az objektum memória címét, miközben az életbe lép a halál hatása.
- **getHealth**: A getHealth metódus lehetővé teszi a gomba életerejének lekérdezését, amelyet később más logikai elemek használhatnak a gomba állapotának kezelésére.

8.1.17 FungusBody

Felelősség

- A FungusBody osztály a gomba testét reprezentálja. Ez az osztály felelős a gomba myceliumának növesztéséért, a spóra töltet kezeléséért, valamint spóraködök létrehozásáért, amelyek hatással vannak a térkép egyes területeire. A testnek saját spóratöltete és szabályozott regenerálódási rendszere van, amely a játék dinamikáját segíti elő.

Ősosztályok

- Legősebb osztály: Fungus (a FungusBody a Fungus osztályból származik)

Interfészek

- Nincs interfész, amit a FungusBody közvetlenül implementál.

Attribútumok

- private static final int MAX_SPORE_CHARGE: A maximális spóratöltet, amelyet a test tárolhat. A szokásos játékmenet során a spórák gyűjtése a test fejlődését segíti elő.
- private int sporeCharge: A gomba aktuális spóratöltete, amely minden frissítéskor növekszik.
- private FungusPlayer player: A gomba tulajdonosa, aki felelős a gomba irányításáért.

Metódusok

- public void update(): A gomba spóratöltetét növeli a CHARGE_PER_TICK konstans értékével, figyelve arra, hogy ne haladja meg a MAX_SPORE_CHARGE értékét. Publikus metódus.
- public void decrementSporeCharge(): Csökkenti a spóratöltetet 1 egységgel. Publikus metódus.
- public void sporeCloud(int size): Spóraköd létrehozása a megadott mérettel. A spóraköd a gomba körüli területeken random elhelyezett spórákat hoz létre, amelyek különböző hatásokkal rendelkezhetnek, mint például lassítás, fagyasztás vagy sebzés. Publikus metódus.
- public void damage(): A gomba életpontját csökkenti, ha a gomba életereje 0 alá csökken, meghívja a die() metódust, hogy kezelje a halálát. Publikus metódus.

8.1.18 FungusBodyGrow

Felelősség

- Az osztály felelőssége, hogy kezelje egy gomba test növekedését a játék térképén, egy meghatározott helyszínen. Ez az osztály az execute metódusban feldolgozza a parancsot, és ha a megfelelő paraméterek rendelkezésre állnak, akkor a gomba testét egy meghatározott tektonikus lemezen és azon egy meghatározott csempén növeszti.

Az osztály célja, hogy a gomba testét bővítse a térképen, amely által új területek válhatnak a gomba hatáskörébe.

Ősosztályok

- Legősebb osztály: Command
- Az osztály a Command osztályból származik, amely a parancsok feldolgozásáért felelős absztrakt osztály.

Interfészek

- Nincs interfész, amit a FungusBodyGrow közvetlenül implementál.

Attribútumok

- Nincs új attribútum az osztályban, mivel a parancs végrehajtásának logikája csak az execute metódusban kerül implementálásra.

Metódusok

- boolean execute(String[] args): A parancs végrehajtását végző metódus, amely a következő feladatokat hajtja végre:
  - Ellenőrzi, hogy a bemeneti argumentumok száma helyes-e, és a térkép inicializálva van-e.
  - Kiszedi a gomba testét az args[1] alapján az parseEntityId metódus segítségével.
  - Kiszedi a céltábla és csempe információkat az args[2] és args[3] alapján a parseTektonAndTile metódus segítségével.
  - Meghívja a gomba játékos growBody metódusát a megadott csempén a gomba növesztésére.
- **Paraméterek**
  - String[] args: A parancs argumentumai, amelyek tartalmazzák a gomba test azonosítóját, a célzott tektonikus lemez azonosítóját és a célzott csempe azonosítóját.
- **Visszatérési érték**
  - boolean: A metódus igazat ad vissza, ha a parancs sikeresen végrehajtódott, és hamisat, ha valamilyen érvénytelen bemenet vagy egyéb hiba történt.

A FungusBodyGrow osztály a gomba testét növeszti a megadott helyen a játék térképén, ezzel dinamikusan bővítve a gomba befolyását az új területekre.

8.1.19 FungusBodyReleaseSporeCloud

Felelősség

- Az osztály felelőssége, hogy egy gomba testet aktiváljon, hogy spórákat szórjon szét a játék térképén. Ez a parancs egy meghatározott gomba test számára kiadja a spórák kibocsátását, amelyeket a sporeCloud metódus segítségével hajt végre. A parancs célja, hogy a gomba biológiai működését szimulálja, és lehetővé teszi a spórák eloszlását a játékban, potenciálisan befolyásolva a térkép további eseményeit.

Ősosztályok

- Legősebb osztály: Command
- Az osztály a Command osztályból származik, amely a parancsok végrehajtásáért felelős absztrakt osztály.

Interfészek

- Nincs interfész, amit a FungusBodyReleaseSporeCloud közvetlenül implementál.

Attribútumok

- Nincs új attribútum az osztályban, mivel az osztály kizárólag a parancs végrehajtását kezeli.

Metódusok

- boolean execute(String[] args): A parancs végrehajtásáért felelős metódus, amely a következő lépéseket hajtja végre:
  - Ellenőrzi, hogy a bemeneti argumentumok száma helyes-e, és hogy a térkép inicializálva van-e.
  - Kiszedi a gomba testet az args[1] alapján az parseEntityId metódus segítségével.
  - Ha a gomba test megtalálható, meghívja a gomba játékos sporeCloud metódusát, és 5-ös erősséggel elindítja a spórák kibocsátását.
- **Paraméterek**
  - String[] args: A parancs argumentumai, amelyek tartalmazzák a gomba test azonosítóját.
- **Visszatérési érték**
  - boolean: A metódus hamisat ad vissza, ha a parancs végrehajtása hibával találkozik, például ha a bemeneti paraméterek nem megfelelőek vagy a térkép nincs inicializálva.

8.1.20 FungusEatsInsect

Felelősség

- Az osztály felelőssége egy gomba játékos (FungusPlayer) és egy rovar (Insect) közötti interakciót szimulálni, ahol a gomba megeszi a rovar. A parancs inicializálja a térképet, hozzáadja a különböző elemeket (tekton, térkép, gomba test, rovar), és végül végrehajtja a gomba fogyasztási műveletét. A rovar sebességét csökkenti, hogy az étkezési folyamatot modellezze, majd a gomba megeszi a rovart.

Ősosztályok

- Legősebb osztály: UseCase
- Az osztály a UseCase osztályból származik, amely a különböző játékbeli esetek végrehajtásáért felelős absztrakt osztály.

Interfészek

- Nincs interfész, amit a FungusEatsInsect osztály közvetlenül implementál.

Attribútumok

- Nincs új attribútum az osztályban, mivel az osztály kizárólag a gomba és rovar közötti interakció végrehajtását kezeli.

Metódusok

- void execute(): A metódus a következő lépéseket hajtja végre:
  - Inicializálja a térképet (Map m) és egy Tekton objektumot, amelyet hozzáad a térképhez.
  - Létrehoz egy HealTile objektumot, amelyet hozzáad a Tekton-hoz és beállítja annak szülőjét.

- Létrehozza a FungusPlayer objektumot és növeszt egy gomba testet a HealTile-ra.
- Létrehozza az InsectPlayer-t és hozzáad egy Insect objektumot, amely a térképen található.
- A rovar sebességét -1-re állítja, szimulálva ezzel a teljes lelassulást.
- Végül a gomba megeszi a rovart a consumeInsect() metódus segítségével.

- **Paraméterek**
  - Nincsenek paraméterek, mivel az osztály a execute() metóduson belül végzi el a szükséges objektumok inicializálását és a parancsot.
- **Visszatérési érték**
  - void: A metódus nem ad vissza értéket, csak végrehajtja az összes lépést, amely magában foglalja a térkép, a gomba és rovar elemek kezelését és a gomba evési műveletét.

8.1.21 FungusGrowingAMushroom

Felelősség

- Az osztály felelőssége, hogy szimulálja a gomba játékos (FungusPlayer) gombatest növesztését egy megadott térképi mezőn. A parancs a játék világát inicializálja, létrehozza a szükséges objektumokat, mint a térképet (Map), tektonikus lemezt (Tekton), és egy térképi négyzetet (Tile), majd a gomba játékos megpróbál egy gombát növeszteni a kiválasztott térképi mezőn.

Ősosztályok

- Legősebb osztály: UseCase
- Az osztály a UseCase osztályból származik, amely a különböző játékműveletek végrehajtásáért felelős alap osztály.

Interfészek

- Nincs interfész, amit a FungusGrowingAMushroom osztály közvetlenül implementál.

Attribútumok

- Az osztály nem tartalmaz új attribútumokat. Az osztály a műveletet egy metódusban, az execute()-ban hajtja végre, így nincs szükség specifikus attribútumok tárolására.

Metódusok

- void execute(): A metódus a következő lépéseket hajtja végre:
  1. A parancs végrehajtásának elején egy "scene inicializálása..." üzenetet nyomtat ki.
  2. Létrehoz egy új Map objektumot, amely reprezentálja a játék világát.
  3. Létrehozza a Tekton objektumot, amely egy tektonikus lemezt reprezentál.
  4. A Tekton objektumot hozzáadja a térképhez.
  5. Létrehoz egy új Tile objektumot, amely a térképen található egy mezőt reprezentálja.
  6. A Tile objektumot hozzárendeli a Tekton-hoz szülőként.
  7. Létrehozza a FungusPlayer objektumot, amely a gomba játékost reprezentálja.

8. A gomba játékos megpróbál egy gombatestet növeszteni a Tile objektumra a growBody() metódus segítségével.
- **Paraméterek**
  - Nincsenek paraméterek, mivel az osztály az execute() metódusban hozza létre az összes szükséges objektumot.
- **Visszatérési érték**
  - void: A metódus nem ad vissza értéket, csak végrehajtja a szimuláció lépéseit és inicializálja azokat az objektumokat, amelyek szükségesek a gombatest növesztéséhez.

8.1.22 FungusPlayer

Felelősség

A **FungusPlayer** osztály a játékos karakterét képviseli, amely gombafonalakat (mycelium) és gombatesteket (fungus body) növeszt, valamint rovarokat fogyaszt, ha azok megbénultak és szomszédosak valamely gombafonalhoz vagy gombatesthez. Az osztály felelőssége továbbá a spórák használata, például spóraködök létrehozása, és a gombatestek kezelése a játék állapotának változásai szerint.

Ősosztályok

- **Player → FungusPlayer**

Interfészek

- Nincs interfész, amelyet az osztály megvalósít.

Attribútumok

- **fungusBodies**: Lista, amely a gombatesteket tárolja. Láthatóság: **private**, Típus: **List<FungusBody>**
- **mycelia**: Lista, amely a gombafonalakat tárolja. Láthatóság: **private**, Típus: **List<Mycelium>**

Metódusok

- **consumeInsect(Insect insect)**:
  - Leírás: A metódus egy rovar elfogyasztására szolgál, amennyiben a rovar bénult és a gombafonal vagy gombatest szomszédságában található.
  - Láthatóság: **public**
  - Paraméterek:
    - **Insect insect**: A rovar, amelyet a FungusPlayer próbál elfogyasztani.
  - Algoritmus:
    - Először ellenőrzi, hogy a rovar bénult-e (sebessége 0).
    - Ha igen, ellenőrzi, hogy a rovar szomszédos-e a gombafonalakkal vagy gombatestekkel.
    - Ha a rovar szomszédos, meghal és helyén egy új gombatestet növeszt.
  - Activity diagram:

Paralyzált rovar ellenőrzése
Szomszédság ellenőrzése gombafonalakkal és gombatestekkel
Rovar elfogyasztása és gombatest növesztése

- **growMycelium(Tile tile)**:
  - Leírás: A metódus megpróbál egy gombafonalat növeszteni egy adott tile-on, ha a tile szomszédos egy élő gombafonalhoz vagy gombatesthez.
  - Láthatóság: **public**

    - Paraméterek:
        - **Tile tile**: A tile, amelyen gombafonalat próbálunk növeszteni.
    - Algoritmus:
        - Ellenőrzi, hogy a megadott tile szomszédos-e valamely élő gombafonalhoz vagy gombatesthez.
        - Ha igen, akkor a tile-on gombafonalat növeszt.
    - Activity diagram:

Tile szomszédság ellenőrzése gombafonalakkal és gombatestekkel
Gombafonal növesztése, ha szomszédos

- **growBody(Tile tile)**:
    - Leírás: A metódus megpróbál egy gombatestet növeszteni a megadott tile-on, ha annak nincs már gombatestje, és ha a tekton rendelkezik elegendő spórával.
    - Láthatóság: **public**
    - Paraméterek:
        - **Tile tile**: A tile, amelyen gombatestet próbálunk növeszteni.
    - Algoritmus:
        - Ellenőrzi, hogy a tile már nem tartalmaz gombatestet.
        - Ellenőrzi, hogy a tekton elegendő spórával rendelkezik-e a gombatest növesztéséhez.
        - Ha a tile szomszédos egy gombafonalhoz, akkor a gombatestet növeszti.
    - Activity diagram:

Tile és tekton ellenőrzése
Gombatest növesztés

- **sporeCloud(FungusBody target, int size)**:
    - Leírás: A metódus egy spóraköd létrehozására szolgál egy célzott gombatest körül, ha a célzott gombatest elegendő spórával rendelkezik.
    - Láthatóság: **public**
    - Paraméterek:
        - **FungusBody target**: A célzott gombatest, amely körül a spóraködöt létrehozzuk.
        - **int size**: A spóraköd mérete, amely meghatározza, hogy hány spórát használnak el.
    - Algoritmus:
        - Ellenőrzi, hogy a célzott gombatest rendelkezik-e elegendő spórával.
        - Ha elegendő spóra van, létrehozza a spóraködöt.
        - Ha nem elegendő a spóra, üzenetet küld a játékosnak.
    - Activity diagram:

Ellenőrzés: elegendő spóra
Spóraköd létrehozása

- **addFungusBody(FungusBody fb)**:
    - Leírás: A metódus egy gombatestet ad hozzá a játékos gombatestjeinek listájához.
    - Láthatóság: **public**
    - Paraméterek:
        - **FungusBody fb**: A hozzáadott gombatest.
    - Algoritmus:
        - A gombatestet hozzáadja a **fungusBodies** listához.
- **removeFungusBody(FungusBody fb)**:

    - Leírás: A metódus eltávolít egy gombatestet a játékos gombatestjeinek listájából.
    - Láthatóság: **public**
    - Paraméterek:
        - **FungusBody fb**: Az eltávolítandó gombatest.
    - Algoritmus:
        - A gombatestet eltávolítja a **fungusBodies** listából.
- **getFungusBodies()**:
    - Leírás: A metódus visszaadja a játékos gombatestjeinek listáját.
    - Láthatóság: **public**
    - Visszatérési érték: **List<FungusBody>**
- **addMycelium(Mycelium myc)**:
    - Leírás: A metódus egy gombafonalat ad hozzá a játékos gombafonalaihoz.
    - Láthatóság: **public**
    - Paraméterek:
        - **Mycelium myc**: A hozzáadott gombafonal.
- **removeMycelium(Mycelium myc)**:
    - Leírás: A metódus eltávolít egy gombafonalat a játékos gombafonalai közül.
    - Láthatóság: **public**
    - Paraméterek:
        - **Mycelium myc**: Az eltávolítandó gombafonal.
- **getMycelia()**:
    - Leírás: A metódus visszaadja a játékos gombafonalainak listáját.
    - Láthatóság: **public**
    - Visszatérési érték: **List<Mycelium>**

8.1.23 FungusSpreadingSpores

Felelősség

A **FungusSpreadingSpores** osztály a játék egyik használati esetét valósítja meg, amelynek célja, hogy a játékos (FungusPlayer) spóraködöt hozzon létre egy adott gombatest (FungusBody) körül. Az osztály előkészíti a megfelelő térképet, elhelyezi a Tekton-t és a gombafonalakat, majd elindítja a spórák szórását. A metódusok és osztályok, mint a **Map**, **Tekton**, **Tile**, **FungusPlayer**, és **FungusBody** inicializálása szükséges a folyamat során. A **sporeCloud** metódus hívásával végrehajtja a gombatest körüli spóraköd létrehozását, amely alapvető a játék mechanikájában.

Ősosztályok

- **UseCase → FungusSpreadingSpores**

Interfészek

- Nincs interfész, amelyet az osztály megvalósít.

Attribútumok

- **m**: Térkép, amely a játék környezetét reprezentálja. Láthatóság: **private**, Típus: **Map**
- **tek**: A játékos által irányított Tekton karakter. Láthatóság: **private**, Típus: **Tekton**
- **t**: A tile, amelyre a gombafonalat vagy gombatestet helyezzük. Láthatóság: **private**, Típus: **Tile**
- **fp**: A játékos FungusPlayer karaktere, amely a gombafonalakat és gombatesteket kezel. Láthatóság: **private**, Típus: **FungusPlayer**

- **fb**: Gombatest, amelyre a spóraködöt szórjuk. Láthatóság: **private**, Típus: **FungusBody**

Metódusok

- **FungusSpreadingSpores()**:
  - Leírás: Konstruktor, amely beállítja a használati eset nevét és azonosítóját.
  - Láthatóság: **public**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Szerepe: A konstruktor beállítja a használati eset alapvető paramétereit.
- **execute()**:
  - Leírás: A metódus végrehajtja a FungusSpreadingSpores használati esetet, amely tartalmazza a térkép, Tekton, Tile, FungusPlayer, és FungusBody objektumok inicializálását, majd elindítja a gombatest körüli spóraköd szórását.
  - Láthatóság: **public**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Algoritmus:
    1. Inicializálás:
       - Térkép inicializálása (**Map m**)
       - Tekton karakter inicializálása és hozzáadása a térképhez
       - Tile inicializálása és hozzárendelése a Tektonhoz
       - FungusPlayer és FungusBody objektumok inicializálása
    2. A FungusPlayer a **sporeCloud** metódust hívja a gombatest körüli spóraköd létrehozására.
  - Activity diagram:
    1. Térkép inicializálása
    2. Tekton inicializálása és hozzáadása a térképhez
    3. Tile inicializálása
    4. FungusPlayer és FungusBody inicializálása
    5. Spóraköd szórása
- **sporeCloud(FungusBody target, int size)**:
  - Leírás: A metódus egy spóraköd létrehozására szolgál egy célzott gombatest körül, ha a célzott gombatest elegendő spórával rendelkezik.
  - Láthatóság: **public**
  - Paraméterek:
    - **FungusBody target**: A célzott gombatest, amely körül a spóraködöt létrehozzuk.
    - **int size**: A spóraköd mérete, amely meghatározza, hogy hány spórát használnak el.
  - Algoritmus:
    - Ellenőrzi, hogy a célzott gombatest rendelkezik-e elegendő spórával.
    - Ha elegendő spóra van, létrehozza a spóraködöt.
    - Ha nem elegendő a spóra, üzenetet küld a játékosnak.

8.1.24 GameEntity

Felelősség

A **GameEntity** osztály az összes játékelem alap osztálya, amely közös tulajdonságokat és metódusokat biztosít a játékelemek számára. Ezek közé tartozik az egyedi azonosító, a

jelenlegi mező (Tile), valamint az egészségi állapot kezelése. Az osztály felelőssége a játékelemek helyének kezelése és az alapvető műveletek végrehajtása, mint például sebzés, gyógyítás, halál, és a mező beállítása. Az osztály különböző származtatott osztályok számára biztosít bővítési lehetőségeket a részletes implementációkhoz.

Ősosztályok

- Nincs ősosztálya, mivel a **GameEntity** osztály a játékelemek alapját képezi.

Interfészek

- Nincs interfész, amelyet az osztály megvalósít.

Attribútumok

- **id**: Egyedi azonosító az entitás számára. Láthatóság: **private**, Típus: **int**
- **currentTile**: Az a mező, amelyen az entitás jelenleg áll. Láthatóság: **private**, Típus: **Tile**
- **idCounter**: Statikus számláló, amely az egyedi azonosítók generálásáért felelős. Láthatóság: **private static**, Típus: **int**

Metódusok

- **GameEntity(int id, Tile currentTile)**:
  - Leírás: Konstruktor, amely az entitás egyedi azonosítóját és aktuális mezőjét állítja be.
  - Láthatóság: **protected**
  - Paraméterek:
    - **id**: Az entitás egyedi azonosítója.
    - **currentTile**: Az a mező, ahol az entitás éppen áll.
  - Visszatérési érték: Nincs
  - Szerepe: Inicializálja az entitást az adott id-val és mezővel.
- **GameEntity()**:
  - Leírás: Paraméter nélküli konstruktor, amely alapértelmezett beállításokkal hozza létre az entitást.
  - Láthatóság: **protected**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Szerepe: Az alapértelmezett konstruktor, amely lehetővé teszi az entitások későbbi inicializálását.
- **setTile(Tile tile)**:
  - Leírás: Beállítja az entitás aktuális mezőjét.
  - Láthatóság: **public**
  - Paraméterek:
    - **tile**: Az a mező, amelyet beállítunk az entitásnak.
  - Visszatérési érték: Nincs
  - Szerepe: Az entitás aktuális mezőjének frissítése.
- **getCurrentTile()**:
  - Leírás: Visszaadja az entitás aktuális mezőjét.
  - Láthatóság: **public**
  - Paraméterek: Nincs
  - Visszatérési érték: **Tile**
  - Szerepe: Az aktuális mező lekérdezése.
- **update()**:

    - Leírás: Az entitás frissítése. A metódus implementációja a származtatott osztályoktól függ.
    - Láthatóság: **public**
    - Paraméterek: Nincs
    - Visszatérési érték: Nincs
    - Szerepe: Frissíti az entitást (a konkrét implementáció a származtatott osztályokban található).
- **damage()**:
    - Leírás: Sebzés végrehajtása az entitáson. A metódus implementációja a származtatott osztályoktól függ.
    - Láthatóság: **public**
    - Paraméterek: Nincs
    - Visszatérési érték: Nincs
    - Szerepe: Sebzés alkalmazása (a konkrét implementáció a származtatott osztályokban található).
- **die()**:
    - Leírás: Az entitás halálának kezelésére szolgáló metódus. A metódus logot generál és a konkrét implementációk az osztályokban történnek.
    - Láthatóság: **public**
    - Paraméterek: Nincs
    - Visszatérési érték: Nincs
    - Szerepe: Az entitás halálának szimulálása és annak kezelése.
- **heal()**:
    - Leírás: A gyógyítási műveletet végrehajtja az entitáson. A metódus implementációja a származtatott osztályoktól függ.
    - Láthatóság: **public**
    - Paraméterek: Nincs
    - Visszatérési érték: Nincs
    - Szerepe: Az entitás gyógyítása (a konkrét implementáció a származtatott osztályokban található).
- **setId(int id)**:
    - Leírás: Az entitás egyedi azonosítójának beállítása.
    - Láthatóság: **public**
    - Paraméterek:
        - **id**: Az új egyedi azonosító.
    - Visszatérési érték: Nincs
    - Szerepe: Az entitás egyedi azonosítójának beállítása.
- **getId()**:
    - Leírás: Visszaadja az entitás egyedi azonosítóját.
    - Láthatóság: **public**
    - Paraméterek: Nincs
    - Visszatérési érték: **int**
    - Szerepe: Az entitás egyedi azonosítójának lekérdezése.
- **getCut()**:
    - Leírás: Az entitás vágásának kezelésére szolgáló metódus. Jelenleg az "insect vs mycelium" és "insect vs insect" helyzetekre használható.
    - Láthatóság: **public**
    - Paraméterek: Nincs
    - Visszatérési érték: Nincs
    - Szerepe: Az entitás vágásának kezelése, amely egyes helyzetekben szükséges.

8.1.25 HealTile

Felelősség

A **HealTile** osztály a **Tile** osztály leszármazottja, és egy olyan speciális típusú mezőt képvisel, amely képes gyógyítani az azon lévő entitásokat. A **HealTile** osztály a mező frissítésekor gyógyítja az összes entitást, amely az adott mezőn tartózkodik.

Ősosztály

- **Tile**: A **HealTile** osztály a **Tile** osztályból öröklődik, így annak összes alapfunkcióját örökli, de kiterjeszti a gyógyítás képességével.

Interfészek

- Nincs interfész, amelyet az osztály megvalósít.

Attribútumok

- **growthRate**: Az alapértelmezett növekedési sebesség (szülő osztályban).
- **maxMycelium**: A maximális mikélium mennyiség (szülő osztályban).
- **parentTekton**: A szülő **Tekton** entitás, amely irányítja a térképet.

Metódusok

- **HealTile(int growthRate, int maxMycelium, Tekton parentTekton)**:
  - Leírás: Konstruktor, amely inicializálja a **HealTile** mezőt a szülő osztály konstruktorával, és beállítja a növekedési sebességet, a maximális mikéliumot és a szülő **Tekton** entitást.
  - Láthatóság: **public**
  - Paraméterek:
    - **growthRate**: A növekedési sebesség értéke.
    - **maxMycelium**: A maximális mikélium mennyiség.
    - **parentTekton**: A szülő **Tekton** entitás, amely a térképet kezeli.
  - Visszatérési érték: Nincs
  - Szerepe: Az osztály inicializálása a szülő osztály által meghatározott paraméterekkel.
- **healEntities()**:
  - Leírás: A metódus végigiterál a mezőn található összes entitáson, és meghívja a **heal()** metódust minden egyes entitáson, hogy gyógyuljanak.
  - Láthatóság: **private**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Szerepe: A mezőn található entitások gyógyítása.
- **update()**:
  - Leírás: Az osztály **update()** metódusa, amely először meghívja a **healEntities()** metódust az entitások gyógyításához, majd meghívja a szülő osztály **update()** metódusát a többi frissítési művelet végrehajtásához.
  - Láthatóság: **public**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Szerepe: Az entitások gyógyítása és a szülő osztály frissítési logikájának meghívása.

8.1.26 HealTileKeepsAlive

Felelősség

A **HealTileKeepsAlive** osztály a **UseCase** osztályból származik, és egy olyan használati esetet modellez, ahol egy **HealTile** típusú mező képes életben tartani az azon lévő **Mycelium** entitást, biztosítva annak túlélését. Az osztály a **Mycelium** entitás frissítésével demonstrálja, hogy a gyógyító mező hogyan hat a játék entitásaira.

Ősosztály

- **UseCase**: A **HealTileKeepsAlive** osztály a **UseCase** osztályból öröklődik, és a szülő osztály által biztosított alap funkciókat használja.

Interfészek

- Nincs interfész, amelyet az osztály megvalósít.

Attribútumok

Nincs külön attribútum, amit közvetlenül az osztály deklarálna, mivel minden szükséges objektumot a metóduson belül hoz létre. Az attribútumok az alosztályok (pl. **Map**, **Tekton**, **HealTile**, **Mycelium**) kezelésére szolgálnak.

Metódusok

- **HealTileKeepsAlive()**:
  - Leírás: Konstruktor, amely inicializálja az osztályt és beállítja az alapértelmezett azonosítót és nevet a szülő osztályban található konstruktor segítségével.
  - Láthatóság: **public**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Szerepe: Az osztály kezdeti beállításainak végrehajtása, beleértve a szülő osztály konstruktorának meghívását.
- **execute()**:
  - Leírás: A **execute** metódus a játék egyes elemeinek létrehozásáért és azok interakciójának kezeléséért felelős. A metódus során létrejön egy **Map** és egy **Tekton**, majd hozzáadódik egy **HealTile**. Egy **Mycelium** entitás is létrejön, amely a gyógyító mezőhöz van rendelve. Végül, a gyógyító mező nem teljes játéknál történő frissítése révén a **Mycelium** entitás frissítése történik meg.
  - Láthatóság: **public**
  - Paraméterek: Nincs
  - Visszatérési érték: Nincs
  - Szerepe: Az összes szükséges entitás inicializálása, és az, hogy a **Mycelium** frissítése révén a gyógyító mező hatása láthatóvá váljon. A frissítési folyamat szimulálása történik a valódi játékmenet frissítése helyett.

8.1.27 Insect

Felelősség

Az **Insect** osztály a játékban található rovarokat reprezentálja. A rovaroknak van egy sebességük, képesek a myceliumot vágni, és egy játékos (InsectPlayer) irányítja őket. A rovarok különböző spórákkal befolyásolhatják a játékot, például spórák elfogyasztásával, vágással és mozgással.

Attribútumok

- **int speed**: A rovar sebessége.

- **boolean canCut**: Megadja, hogy a rovar képes-e vágni a myceliumot.
- **InsectPlayer controlledBy**: A rovar irányító játékos.
- **List<Spore> underInfluence**: A rovarra ható spórák listája.
- **Konstruktorok**
- **Insect(int id, Tile currentTile, InsectPlayer player)**:
    - Leírás: Létrehozza a rovar objektumot, beállítja annak sebességét, vágási képességét, a játékos irányítja, és hozzáadja a játékos által irányított rovarok listájához. A rovar ezen kívül egy üres spóra listával is rendelkezik.
    - Paraméterek:
        - **id**: A rovar azonosítója.
        - **currentTile**: A rovar jelenlegi helye a térképen.
        - **player**: Az **InsectPlayer** objektum, amely irányítja a rovart.

Metódusok

- **split()**:
    - Leírás: Létrehoz egy új rovart ugyanazzal a játékossal, és elhelyezi azt a jelenlegi helyére. Az új rovar klónozott példány lesz.
    - Paraméterek: Nincs.
    - Visszatérési érték: Egy új **Insect** példány.
- **die()**:
    - Leírás: A rovar meghal, eltávolítja magát a térképről és a játékos irányította rovarok listájából.
    - Paraméterek: Nincs.
    - Visszatérési érték: Nincs.
- **update()**:
    - Leírás: Frissíti a rovar állapotát. Ez a metódus iterál a rovarra ható spórák listáján, frissíti azokat, és eltávolítja azokat, melyek hatása lejárt. Beállítja a vágási képességet is.
    - Paraméterek: Nincs.
    - Visszatérési érték: Nincs.
- **eat(Spore target)**:
    - Leírás: A rovar megeszi a megadott spórát, és frissíti a játékos pontjait. Ha a spóra egy **SplitSpore**, akkor a rovar szétválik.
    - Paraméterek:
        - **target**: A megenni kívánt spóra.
    - Visszatérési érték: Nincs.
- **step(Tile target)**:
    - Leírás: A rovar mozog a megadott cél tile-ra. A mozgás során eltávolítja magát az aktuális tile-ról és hozzáadja a célnak.
    - Paraméterek:
        - **target**: A cél tile.
    - Visszatérési érték: Nincs.
- **cut(Tile target)**:
    - Leírás: A rovar levágja a myceliumot a megadott tile-on. Ez a művelet csak akkor történhet meg, ha a rovar képes vágni a myceliumot.
    - Paraméterek:
        - **target**: A vágni kívánt tile.
    - Visszatérési érték: Nincs.
- **addSpore(Spore spore)**:
    - Leírás: Hozzáad egy spórát a rovarra ható spórák listájához.

    - Paraméterek:
        - **spore**: A hozzáadni kívánt spóra.
    - Visszatérési érték: Nincs.
- **setSpeedPercent(int percent)**:
    - Leírás: Beállítja a rovar sebességét a megadott százalékos értéknek megfelelően. Ha a százalék 0, akkor a sebesség visszaáll 100-ra.
    - Paraméterek:
        - **percent**: A sebesség új értéke százalékos formában.
    - Visszatérési érték: Nincs.
- **getSpeed()**:
    - Leírás: Visszaadja a rovar aktuális sebességét.
    - Paraméterek: Nincs.
    - Visszatérési érték: **int** — a rovar sebessége.
- **setCut(boolean canCut)**:
    - Leírás: Beállítja, hogy a rovar képes-e vágni a myceliumot.
    - Paraméterek:
        - **canCut**: Ha **true**, akkor a rovar képes vágni a myceliumot.
    - Visszatérési érték: Nincs.

8.1.28 InsectCantCut

Felelősség

Az InsectCantCut osztály egy parancs, amely megakadályozza, hogy az összes rovar képes legyen myceliumot vágni egy adott, megadott tektonikus lemezhez kapcsolódó területen. A parancs az adott lemezhez tartozó összes rovar vágási képességét letiltja.

Ősosztályok

- **Command** (legőső osztály)

Interfészek

- Az InsectCantCut osztály nem valósít meg interfészeket.

Attribútumok

- Az InsectCantCut osztálynak nincs publikus attribútuma.

Metódusok

- **InsectCantCut()**: Konstruktor, amely inicializálja az osztályt, beállítja a parancs nevét, leírását és használati szintaxisát. (láthatóság: public)
- **boolean execute(String[] args)**: A parancs végrehajtása, amely letiltja a rovarok vágási képességét, miután megtalálta a megadott tektonikus lemezt és annak kapcsolódó myceliumait. Ha a lemez nem található, vagy a bemeneti paraméterek érvénytelenek, a parancs hibát jelez. (láthatóság: public)

8.1.29 InsectCut

Felelősség

Az InsectCut osztály egy parancsot képvisel, amely lehetővé teszi egy rovar számára, hogy vágjon egy myceliumot egy adott térképi mezőn. A parancs végrehajtásakor az osztály ellenőrzi, hogy a bemeneti paraméterek megfelelőek-e (helyes számú argumentum, a térkép inicializált-e, léteznek-e a rovar és mycelium objektumok), majd a megfelelő rovar elvégzi a mycelium vágási műveletét.

Ősosztályok

- **Command** (legőső osztály)

Interfészek

- Az InsectCut osztály nem valósít meg közvetlenül interfészeket.

Attribútumok

- Az InsectCut osztálynak nincs publikus attribútuma.

Metódusok

- **InsectCut()**: Konstruktor, amely inicializálja az osztályt, beállítja a parancs nevét, leírását és a használati szintaxisát. (láthatóság: public)
- **boolean execute(String[] args)**: A parancs végrehajtása, amely az alábbi lépéseket hajtja végre:
    1. Ellenőrzi, hogy a bemeneti argumentumok száma helyes-e. Ha nem, akkor visszatér false értékkel.
    2. Ellenőrzi, hogy a térkép inicializált-e. Ha nem, akkor visszatér false értékkel.
    3. Megpróbálja azonosítani az Insect objektumot az adott rovar ID alapján. Ha nem található, visszatér false értékkel.
    4. Megpróbálja azonosítani a Mycelium objektumot az adott mycelium ID alapján. Ha nem található, visszatér false értékkel.
    5. Hozzáadja a rovar vezérlését az InsectPlayer-hez.
    6. A rovar végrehajtja a mycelium vágását a megfelelő térképi mezőn a cut metódus segítségével.
    7. A parancs végrehajtása után false értéket ad vissza. (láthatóság: public)

8.1.30 InsectCutMycelium

Felelősség

Az InsectCutMycelium osztály egy használati esetet (use case) reprezentál, ahol egy rovar (Insect) egy myceliumot (Mycelium) vág le egy adott térképi mezőn (Tile). Az osztály inicializálja az összes szükséges entitást, beleértve a rovar játékost (InsectPlayer), a térképi tektonot (Tekton), a térképi mezőt (Tile), a myceliumot (Mycelium) és a rovart (Insect), majd végrehajtja a mycelium vágási műveletet.

Ősosztályok

- **UseCase** (legőső osztály)

Interfészek

- Az InsectCutMycelium osztály nem valósít meg közvetlenül interfészeket.

Attribútumok

- Az InsectCutMycelium osztálynak nincs publikus attribútuma.

Metódusok

- **InsectCutMycelium()**: Konstruktor, amely inicializálja az osztályt, és beállítja az osztály nevét és a használt paramétereket. (láthatóság: public)
- **void execute()**: A használati eset végrehajtása, amely az alábbi lépéseket hajtja végre:
    1. Kiírja az "Initializing scene..." üzenetet.
    2. Létrehoz egy InsectPlayer objektumot.
    3. Létrehoz egy Tekton objektumot, amely tárolja a térképi mezőt.

4. Létrehoz egy Tile objektumot, amely a térképi mezőt reprezentálja.
5. Létrehoz egy Mycelium objektumot.
6. Létrehoz egy Insect objektumot, amely egy rovar, és hozzárendeli a Tile-hoz és az InsectPlayer-hez.
7. A Tile objektumot hozzáadja a Tekton-hoz.
8. A Tile-hoz hozzáadja a Mycelium és Insect objektumokat.
9. Az InsectPlayer-hez hozzáadja a rovar vezérlését.
10. A cut() metódus segítségével a rovar levágja a myceliumot a térképi mezőn.

(láthatóság: public)

8.1.31 InsectEatSpore

Felelősség

Az InsectEatSpore osztály egy parancsot reprezentál, amely lehetővé teszi, hogy egy rovar (Insect) elfogyassza a megadott spórát (Spore). Az osztály a parancs végrehajtásához szükséges entitások azonosítását végzi el, és biztosítja, hogy a rovar képes legyen végrehajtani az étkezési műveletet.

Ősosztályok

- **Command** (legőső osztály)

Interfészek

- Az InsectEatSpore osztály nem valósít meg közvetlenül interfészeket.

Attribútumok

- Az InsectEatSpore osztálynak nincs publikus attribútuma.

Metódusok

- **InsectEatSpore()**: Konstruktor, amely beállítja a parancs nevét, a leírását és a szintaxist. A parancs neve insect_eat_spore, és két paramétert vár: a rovar azonosítóját és a spóra azonosítóját.
  - (láthatóság: public)
- **boolean execute(String[] args)**: A parancs végrehajtása. Az alábbi lépéseket hajtja végre:

1. Ellenőrzi, hogy a paraméterek száma helyes-e.
2. Ellenőrzi, hogy a térkép inicializálva van-e.
3. Parse-olja a rovar objektumot az azonosítója alapján.
4. Ha a rovar nem létezik, visszatér false-szal.
5. Parse-olja a spóra objektumot az azonosítója alapján.
6. Ha a spóra nem létezik, visszatér false-szal.
7. Hozzáadja a rovart a InsectPlayer vezérlő rovarjaihoz.
8. Meghívja a eat metódust, hogy a rovar elfogyassza a megadott spórát.
9. Visszatér false-szal, jelezve, hogy a parancs végrehajtása befejeződött.
   - (láthatóság: public)

8.1.32 InsectFreeze

Felelősség

Az InsectFreeze osztály egy parancsot reprezentál, amely lehetővé teszi egy rovar (Insect) megbénítását. A parancs egy FreezeSpore spórát generál, és azt hozzáadja a rovar aktuális

csempéjéhez. Ez a spóra a rovar fogyasztásakor hatással van annak mozgására, megbénítva azt.

Ősosztályok

- **Command** (legőső osztály)

Interfészek

- Az InsectFreeze osztály nem valósít meg közvetlenül interfészeket.

Attribútumok

- Az InsectFreeze osztálynak nincs publikus attribútuma.

Metódusok

- **InsectFreeze()**: Konstruktor, amely beállítja a parancs nevét, a leírását és a szintaxist. A parancs neve insect_freeze, és egy paramétert vár: a rovar azonosítóját.
  - (láthatóság: public)
- **boolean execute(String[] args)**: A parancs végrehajtása. Az alábbi lépéseket hajtja végre:
  1. Ellenőrzi, hogy a paraméterek száma helyes-e (két paraméter).
  2. Ellenőrzi, hogy a térkép inicializálva van-e.
  3. Parse-olja a rovar objektumot az azonosítója alapján.
  4. Ha a rovar nem létezik, visszatér false-szal.
  5. Létrehoz egy új FreezeSpore típusú spórát.
  6. Hozzáadja a spórát a rovar aktuális csempéjéhez.
  7. Hozzáadja a rovart a InsectPlayer vezérlő rovarjaihoz.
  8. Meghívja a eat metódust, hogy a rovar elfogyassza a spórát, ami megbénítja.
  9. Visszatér false-szal, jelezve, hogy a parancs végrehajtása befejeződött.
  - (láthatóság: public)

8.1.33 InsectFreezeTimesOut

Felelősség

Az InsectFreezeTimesOut osztály egy használati esetet (use case) reprezentál, amely a rovar (Insect) megbénulásának időtúllépését modellezi. A parancs nem lett teljesen implementálva, de alapvetően egy rovar megbénítását szimulálja a FreezeSpore elfogyasztásával. Az osztály jelenleg csak a színpad előkészítését hajtja végre, azaz a térkép, tekton, csempe, rovar és spóra objektumokat létrehozza, de a megbénulás valódi hatása még nincs implementálva.

Ősosztályok

- **UseCase** (legőső osztály)

Interfészek

- Az InsectFreezeTimesOut osztály nem valósít meg közvetlenül interfészeket.

Attribútumok

- Az osztály nem rendelkezik saját attribútumokkal, a szükséges objektumokat a execute metódus hozza létre.

Metódusok

- **InsectFreezeTimesOut()**: Konstruktor, amely beállítja a használati eset nevét és az azonosítóját. A használati eset neve Insect freeze times out.

    - (láthatóság: public)
- **void execute()**: A használati eset végrehajtása, amely az alábbi lépéseket hajtja végre:
  1. Színpad előkészítése: A térkép, tekton, csempe, rovar és spóra objektumok létrehozása.
  2. Térkép létrehozása: Egy új Map objektumot hoz létre.
  3. Tekton létrehozása: A térképen egy új Tekton objektumot hoz létre, amelyet a térképhez rendel.
  4. Csempe létrehozása: Egy új Tile objektumot hoz létre, amelyet a tektonhoz rendel.
  5. Kölcsönös hivatkozás: A csempe és a tekton egymásra hivatkozik.
  6. RovarPlayer létrehozása: Egy új InsectPlayer objektumot hoz létre, amely a rovarokat vezérli.
  7. Rovar létrehozása: Egy új Insect objektumot hoz létre, amely a csempén tartózkodik.
  8. FreezeSpore létrehozása: Egy új FreezeSpore objektumot hoz létre, és hozzáadja a csempéhez.
  9. Rovar eszik FreezeSpore: A rovar elfogyasztja a FreezeSpore spórát, de a spóra hatásai nincsenek még implementálva.

8.1.34 InsectMove

Felelősség

Az InsectMove osztály egy használati esetet (use case) képvisel, amely a rovar (Insect) mozgatását modellezi a játékban. A rovar egy kiinduló csempéről (currentTile) a célcsempére (targetTile) mozog. Az osztály felelős az összes szükséges objektum létrehozásáért és a rovar mozgásának inicializálásáért, bár maga a mozgás logika nincs részletezve. Ez a használati eset alapvetően előkészíti a környezetet a rovar mozgásának szimulálásához.

Ősosztályok

- **UseCase** (legőső osztály)

Interfészek

- Az InsectMove osztály nem valósít meg közvetlenül interfészeket.

Attribútumok

- Az osztály nem rendelkezik saját attribútumokkal, a szükséges objektumokat a execute metódus hozza létre.

Metódusok

- **InsectMove()**: Konstruktor, amely beállítja a használati eset nevét és az azonosítóját. A használati eset neve Insect player moving to tile.
    - (láthatóság: public)
- **void execute()**: A használati eset végrehajtása, amely az alábbi lépéseket hajtja végre:
  1. Színpad előkészítése: A térkép, tekton, csempe és rovar objektumok létrehozása.
  2. RovarPlayer létrehozása: Egy új InsectPlayer objektumot hoz létre, amely a rovarokat vezérli.
  3. Tekton létrehozása: A térképhez tartozó Tekton objektum létrehozása.
  4. Csempe létrehozása: Két Tile objektumot hoz létre: az egyik a kiinduló csempe (currentTile), a másik a célcsempe (targetTile).

5. Rovar létrehozása: Egy új Insect objektumot hoz létre, amely a kiinduló csempén tartózkodik.
6. Csempék hozzáadása a Tektonhoz: A csempéket a Tekton objektumhoz rendeli.
7. Rovar hozzáadása a csempéhez: A rovar hozzáadásra kerül a kiinduló csempéhez.
8. Rovar hozzáadása a RovarPlayerhez: A rovar hozzáadásra kerül az InsectPlayer objektumhoz, hogy azt vezérelhesse.
9. Rovar mozgása: A InsectPlayer objektum utasítja a rovart, hogy mozogjon a célcsempére (targetTile).
   - (láthatóság: public)

8.1.35 InsectPlayer

Felelősség

Az InsectPlayer osztály egy olyan játékos (player), aki rovarokat irányít a játékban. Az osztály felelős a rovarok kezeléséért, a mozgásuk irányításáért, az etetésükért, és a vágásuk kezeléséért. Az osztály különböző akciókat hajt végre, amelyek a rovarokat érintik, mint például mozgás egy adott csempére, egy spóra elfogyasztása, vagy egy csempe vágása.

Ősosztályok

- **Player**: Az InsectPlayer az Player osztályból származik, így örökli annak funkcionalitását, például a játékos alapvető tulajdonságait és képességeit.

Attribútumok

- **List<Insect> controlledInsects**: A controlledInsects egy lista, amely azokat a rovarokat tartalmazza, amelyeket az InsectPlayer irányít. Ezen rovarok állapotait és mozgását kezeli az osztály.
- **Konstruktor**
- **InsectPlayer()**: A konstruktor inicializálja az InsectPlayer objektumot, és hozzáadja azt a UseCase-hez a játék megfelelő inicializálásához. A konstruktor beállítja az üzenetlogot és inicializálja a controlledInsects listát.

Metódusok

- **void addControlledInsect(Insect controlledInsect)**: Egy új rovar hozzáadása az InsectPlayer irányítása alá. Ez a metódus a rovar hozzáadására szolgál a listához.
- **List<Insect> getControlledInsects()**: Visszaadja a játékos által irányított rovarok listáját.
- **void moveTo(Tile tile)**: Az alapértelmezett moveTo metódus, amely a játékos első rovarát mozgósítja a célcsempére (tile). Ez a metódus az moveTo(Tile tile, Insect controlledInsect) metódus egyszerűsített verziója.
- **void moveTo(Tile tile, Insect controlledInsect)**: A rovar mozgása a megadott célcsempére. Az alábbi lépéseket hajtja végre:
  1. Ellenőrzi, hogy a rovar a játékos irányítása alatt áll-e.
  2. Ellenőrzi, hogy a rovar képes-e mozogni.
  3. Ellenőrzi, hogy a célcsempe a rovar szomszédja-e.
  4. Ha a csempe ugyanazon a Tekton-on van, vagy ha van egy híd, a rovar lép a célcsempére.
  5. Ha egyik feltétel sem teljesül, nem történik mozgás.

- **void eat(Spore s)**: Az alapértelmezett eat metódus, amely az első rovar számára eteti meg a spórát (Spore).
- **void eat(Spore s, Insect controlledInsect)**: A rovar etetése egy adott spórával. Az alábbi lépéseket hajtja végre:
    1. Ellenőrzi, hogy a rovar a játékos irányítása alatt áll-e.
    2. Meghívja a rovar eat metódusát a spóra feldolgozásához.
- **void cut(Tile tile)**: Az alapértelmezett cut metódus, amely az első rovar számára engedélyezi a vágást a megadott csempén.
- **void cut(Tile tile, Insect controlledInsect)**: A rovar vágása a megadott csempén. A metódus ellenőrzi, hogy a rovar a játékos irányítása alatt áll-e, és meghívja a rovar cut metódusát.
- **void removeControlledInsect(Insect controlledInsect)**: A rovar eltávolítása a játékos irányítása alól, ha már nem irányítja azt (például ha elpusztul).

8.1.36 InsectSlowDown

Felelősség

Az InsectSlowDown osztály egy parancsot definiál, amely alkalmazza a lassító hatást egy adott rovarra. A parancs lehetővé teszi a játékos számára, hogy egy rovar sebességét lecsökkentse egy megadott százalékra, ezáltal lassítva annak mozgását a játék során. A parancs a rovar számára létrehoz egy lassító spórát, amit azután elfogyaszt.

Ősosztály

- **Command**: Az InsectSlowDown az Command osztályból öröklődik, tehát egy parancsot képvisel, amely a játék eseményeit manipulálja. A Command osztály biztosítja a parancsok alapvető struktúráját, és az execute metódus segítségével végrehajtja azokat.

Attribútumok

- **Nincs specifikus attribútum az osztályban**.
- **Konstruktor**
- **InsectSlowDown()**: Az osztály konstruktora, amely beállítja a parancs nevét, leírását és a szintaxisát:
    - **Név**: "insect_slow_down"
    - **Leírás**: A parancs leírása, ami elmagyarázza, hogy a parancs hogyan alkalmazza a lassítást egy rovarra.
    - **Szintaxis**: "insect_slow_down <insect id> <speed percentage>", ahol <insect id> a rovar azonosítóját, <speed percentage> pedig a kívánt sebesség csökkentést jelöli.

Metódusok

- **boolean execute(String[] args)**: Az execute metódus felelős a parancs végrehajtásáért. Az alábbi lépéseket hajtja végre:
    1. **Argumentumok ellenőrzése**: Ellenőrzi, hogy a parancs megfelelő számú argumentumot kapott-e, és hogy a térkép inicializálva van-e.
    2. **Rovar keresése**: Az argumentumokban szereplő rovar azonosítóját használja a rovar keresésére. Ha a rovar nem található, a parancs hibát jelez.
    3. **Sebesség ellenőrzése**: Az argumentumokban szereplő sebesség százalékos értéket ellenőrzi, hogy pozitív szám-e. Ha nem, akkor hibát jelez.
    4. **Lassító spóra létrehozása**: Ha minden feltétel teljesül, létrehoz egy SlowSpore objektumot a kívánt sebességgel.

5. **Spóra hozzáadása és elfogyasztása**: A lassító spórát hozzáadja a rovar aktuális csempéjéhez, majd a játékos elfogyasztja azt, amely alkalmazza a lassítást a rovaron.

- **Metódus végrehajtásának logikája**
  1. Argumentumok validálása:
  - Az isWrongNumberOfArgs() metódus biztosítja, hogy a parancs megfelelő számú argumentumot kapott.
  - Az isMapUninitialized() metódus ellenőrzi, hogy a térkép inicializálva van-e.
  2. Rovar keresése:
  - A parseEntityId() metódus segítségével a parancs megpróbálja megtalálni a rovar objektumot a megadott azonosítóval. Ha nem találja, visszatér a hibával.
  3. Sebesség érték ellenőrzése:
  - A parsePositiveNumber() metódus ellenőrzi, hogy a sebesség egy pozitív szám-e, amely az új sebesség százalékos értéke lesz.
  4. Spóra létrehozása és alkalmazása:
  - Ha minden validáció sikeres, létrehozásra kerül a SlowSpore objektum, ami tartalmazza a lassítás mértékét.
  - A spórát hozzáadják a rovar aktuális csempéjéhez, majd a játékos elfogyasztja azt, hogy a rovar sebességét lecsökkentsék.

8.1.37 InsectSpeedUp

Felelősség

Az InsectSpeedUp osztály egy parancsot definiál, amely alkalmazza a sebességnövelő hatást egy adott rovarra. A parancs lehetővé teszi a játékos számára, hogy egy rovar sebességét megnövelje egy megadott százalékra, így gyorsítva annak mozgását a játék során. A parancs a rovar számára létrehoz egy sebességnövelő spórát, amit azután elfogyaszt.

Ősosztály

- **Command**: Az InsectSpeedUp az Command osztályból öröklődik, tehát egy parancsot képvisel, amely a játék eseményeit manipulálja. A Command osztály biztosítja a parancsok alapvető struktúráját, és az execute metódus segítségével végrehajtja azokat.

Attribútumok

- **Nincs specifikus attribútum az osztályban**.

Metódusok

- **InsectSpeedUp()**: Az osztály konstruktora, amely beállítja a parancs nevét, leírását és a szintaxisát:
  - **Név**: "insect_speed_up"
  - **Leírás**: A parancs leírása, amely elmagyarázza, hogy a parancs hogyan alkalmazza a sebességnövelést egy rovarra.
  - **Szintaxis**: "insect_speed_up <insect id> <speed percentage>", ahol <insect id> a rovar azonosítóját, <speed percentage> pedig a kívánt sebesség növelést jelöli.
- **boolean execute(String[] args)**: Az execute metódus felelős a parancs végrehajtásáért. Az alábbi lépéseket hajtja végre:
  1. Argumentumok ellenőrzése: Ellenőrzi, hogy a parancs megfelelő számú argumentumot kapott-e, és hogy a térkép inicializálva van-e.

2. Rovar keresése: Az argumentumokban szereplő rovar azonosítóját használja a rovar keresésére. Ha a rovar nem található, a parancs hibát jelez.
3. Sebesség ellenőrzése: Az argumentumokban szereplő sebesség százalékos értéket ellenőrzi, hogy pozitív szám-e. Ha nem, akkor hibát jelez.
4. Sebesség növelő spóra létrehozása: Ha minden feltétel teljesül, létrehoz egy SpeedUpSpore objektumot a kívánt sebességgel.
5. Spóra hozzáadása és elfogyasztása: A sebességnövelő spórát hozzáadják a rovar aktuális csempéjéhez, majd a játékos elfogyasztja azt, amely alkalmazza a sebességnövelést a rovaron.

- **Metódus végrehajtásának logikája**
- **Konstruktor**
- **InsectSpeedUp()**: Az osztály konstruktora, amely beállítja a parancs nevét, leírását és a szintaxisát:
  - **Név**: "insect_speed_up"
  - **Leírás**: A parancs leírása, amely elmagyarázza, hogy a parancs hogyan alkalmazza a sebességnövelést egy rovarra.
  - **Szintaxis**: "insect_speed_up <insect id> <speed percentage>", ahol <insect id> a rovar azonosítóját, <speed percentage> pedig a kívánt sebesség növelést jelöli.

8.1.38 InsectSplitSpore

Felelősség

Az InsectSplitSpore osztály egy use case-t valósít meg, amely a rovarok számára lehetőséget ad egy "split spore" (hasadási spóra) elfogyasztására, amely új rovarokat hozhat létre. Az osztály egy példa a játék különböző entitásainak (mint a Tekton, Tile, Insect, Spore) kezdeti állapotának előkészítésére és az azok közötti interakciók demonstrálására.

Ősosztály

- **UseCase**: Az InsectSplitSpore az UseCase osztályból öröklődik. Az UseCase biztosítja az alapvető struktúrát, amely tartalmazza a szintetikus események kezelését és végrehajtását.

Attribútumok

Metódusok

- **InsectSplitSpore()**: A konstruktor a szülő osztály konstruktorát hívja meg, és beállítja a use case azonosítóját (20) és nevét ("Insect Split Spore").
- **void execute()**: Az execute metódus tartalmazza az összes lépést, amelyet a "split spore" használata igényel. A következő eseményeket hajtja végre:
  1. Színpad inicializálása: A jelenet inicializálása egy egyszerű üzenettel, amely a kiindulási állapotot mutatja.
  2. Térkép létrehozása: Létrehoz egy új Map objektumot.
  3. Tekton inicializálása: A Tekton objektum inicializálása és hozzárendelése a térképhez.
  4. Csempe inicializálása: Létrehoz egy új Tile objektumot, és hozzárendeli a Tekton-hoz.
  5. InsectPlayer létrehozása: Az InsectPlayer objektum inicializálása.
  6. Rovar létrehozása: A rovar (Insect) inicializálása és hozzárendelése az InsectPlayer-hez.

7. Spóra hozzáadása: A "split spore" (SplitSpore) létrehozása, amely hozzáadásra kerül a csempére.
8. Spóra elfogyasztása: Az InsectPlayer megpróbálja elfogyasztani a "split spore"-t a rovar segítségével, ami a rovar viselkedését befolyásolhatja.

Logika

- Map és Tekton inicializálása: A térkép és a tekton inicializálása és összekapcsolása. A Tekton a játék egyik fontos entitása, amelyhez kapcsolódóan térképi elemeket és entitásokat lehet hozzáadni.
- Csempe és entitások kezelése: A rovar számára hozzáadott csempe tartalmazza a "split spore"-t, amelyet a rovar végül elfogyaszthat.
- InsectPlayer és Rovar: A InsectPlayer objektum lesz felelős a rovarok irányításáért. A rovar a InsectPlayer-hez kerül hozzáadásra, és az eat metódus alkalmazásával a rovar elfogyasztja a spórát.
- Hasadási spóra: A hasadási spóra hatása nem szerepel explicit módon ebben a kódban, de feltételezhető, hogy a spóra hatására a rovar szétválhat vagy új rovarokat generálhat.

8.1.39 InsectStep

Felelősség

Az InsectStep parancs lehetővé teszi egy rovar számára, hogy egy adott célcsempére lépjen. A parancs bemeneti paramétereként meg kell adni a rovar ID-ját, a cél tektonikus lemezt és a cél csempét. A parancs végrehajtásakor a rovar eljut a megadott helyre.

Ősosztály

- **Command**: Az InsectStep osztály a Command osztályból öröklődik. A Command osztály biztosítja az alapvető szerkezetet és funkcionalitást, amely lehetővé teszi különböző parancsok végrehajtását a játékban.

Attribútumok

- **Nincs specifikus attribútum az osztályban**.
- **Konstruktor**
- **InsectStep()**: A konstruktor a szülő osztály konstruktorát hívja meg, és beállítja a parancs nevét (insect_step), a leírást, és a szintaxist. Ez a parancs arra szolgál, hogy a rovarokat célzott helyekre mozgassa.

Metódusok

- **boolean execute(String[] args)**: Az execute metódus végrehajtja a parancsot, és biztosítja, hogy minden szükséges paraméter helyesen van megadva és a térkép inicializálva van. Az alábbi lépéseket hajtja végre:
    1. Helyes számú paraméterek ellenőrzése: Ha a parancs argumentumainak száma nem megfelelő, a metódus visszatér false-szal.
    2. Térkép inicializálásának ellenőrzése: Ha a térkép nincs inicializálva, a metódus false-szal tér vissza.
    3. Rovar ellenőrzése: A rovar ID-ja alapján a parseEntityId metódus segítségével próbálja meg betölteni az adott rovart. Ha nem található, a metódus false-szal tér vissza.
    4. Céltektonikus lemez és csempe ellenőrzése: A parseTektonAndTile metódus segítségével ellenőrzi, hogy a megadott tektonikus lemez és csempe helyes-e. Ha nem, visszatér false-szal.

5. Rovar mozgása: A rovar hozzáadása az InsectPlayer-hez és a moveTo metódus végrehajtása a célcsempére.

Logika

- **Helyes paraméterek biztosítása**: A parancs előtt biztosítja, hogy az összes szükséges információ (rovar ID, cél tektonikus lemez és célcsempe) helyesen van megadva. Ez a validáció megelőzi a hibás adatbevitelből eredő problémákat.
- **Rovar mozgása**: Az InsectPlayer felelős a rovarok irányításáért. A rovar tehát először hozzáadódik az InsectPlayer-hez, majd a moveTo metódus segítségével végrehajtja a mozgást a megadott célcsempére.
- **Célcsempe meghatározása**: A parseTektonAndTile metódus felelős azért, hogy a megadott tektonikus lemez és csempe alapján helyesen azonosítsa a célt, ahová a rovarnak lépnie kell.

8.1.40 InsectUnfreeze

Felelősség

Az InsectUnfreeze parancs egy rovar "felszabadítását" végzi el, amely elérhetőséget ad számára a mozgásához. A parancs a rovarnak ad egy SpeedUpSpore spórát, ami a rovar sebességét visszaállítja a normál állapotra.

Ősosztály

- **Command**: Az InsectUnfreeze osztály a Command osztályból öröklődik. A Command osztály biztosítja az alapvető szerkezetet és funkcionalitást a különböző parancsok végrehajtásához.

Attribútumok

Metódusok

- **InsectUnfreeze()**: A konstruktor a szülő osztály konstruktorát hívja meg, és beállítja a parancs nevét (insect_unfreeze), leírását, és a szintaxist. Ez a parancs lehetővé teszi egy rovar "feloldását" a mozgáskorlátozottságból, ha a rovarot korábban valamilyen módon lelassították vagy megfagyasztották.
- **boolean execute(String[] args)**: Az execute metódus végrehajtja a parancsot, és biztosítja, hogy minden szükséges paraméter helyesen van megadva és a térkép inicializálva van. Az alábbi lépéseket hajtja végre:
    1. **Helyes számú paraméterek ellenőrzése**: Ha a parancs argumentumainak száma nem megfelelő, a metódus visszatér false-szal.
    2. **Térkép inicializálásának ellenőrzése**: Ha a térkép nincs inicializálva, a metódus false-szal tér vissza.
    3. **Rovar ellenőrzése**: A rovar ID-ja alapján a parseEntityId metódus segítségével próbálja meg betölteni az adott rovart. Ha nem található, a metódus false-szal tér vissza.
    4. **Felszabadító spóra hozzáadása**: A SpeedUpSpore egy sebesség-növelő spóra, amelyet a rovar tile-jához adunk. Ezzel a spórával "felszabadítjuk" a rovart a lelassított vagy fagyott állapotból.
    5. **Rovar mozgása**: Az InsectPlayer felelős a rovarok irányításáért. A rovar tehát először hozzáadódik az InsectPlayer-hez, majd a eat metódus végrehajtásával a rovar "megeszi" a spórát, visszaállítva a normál sebességét.

Logika

- **Helyes paraméterek biztosítása**: A parancs előtt biztosítja, hogy az összes szükséges információ (rovar ID) helyesen van megadva. Ez a validáció megelőzi a hibás adatbevitelből eredő problémákat.
- **Spóra hozzáadása**: A spóra, amelyet a rovar megeszik, nem egy valódi "fagyasztó" spóra, hanem inkább egy olyan spóra, amely a rovar sebességét helyreállítja. Ez lehet egy egyszerű módja a rovar állapotának helyreállítására anélkül, hogy egy valódi "fagyasztó" spórát kellene bevezetni.
- **Rovar mozgásának irányítása**: A rovar hozzáadása az InsectPlayer-hez és a spóra "elfogyasztása" a rovar sebességét visszaállítja.

8.1.41 ListCmd

Felelősség

A ListCmd parancs egy olyan parancs, amely kilistázza az összes elérhető parancsot, valamint azok leírásait és használatát a rendszerben. A parancs célja, hogy a felhasználó megismerkedhessen a rendszerben elérhető összes parancs részleteivel.

Ősosztály

- **Command**: Az ListCmd osztály a Command osztályból öröklődik, így biztosítja az alapvető funkcionalitást, mint a parancsok neve, leírása és szintaxisa.

Attribútumok

Metódusok

- **ListCmd()**: A konstruktor a szülő osztály konstruktorát hívja meg, beállítva a parancs nevét (list), leírását, valamint a szintaxist. A parancs a rendszerben elérhető összes parancsot listázza ki.

- **boolean execute(String[] args)**: Az execute metódus felelős a parancs végrehajtásáért. Az alábbi lépéseket hajtja végre:
    1. Fejléc megjelenítése: A parancs először kiírja a fejlécet, amely az oszlopokat ("Command" és "Description") tartalmazza, és egy vonalat húz a jobb olvashatóság érdekében.
    2. Parancsok listájának lekérése és rendezése: A parancsokat az app.getCommands().getCommands() segítségével lekéri, majd egy ArrayList-be rendezi őket. Ezt követően a listát az Collections.sort() metódussal rendezi a parancsok neve szerint.
    3. Parancsok kiírása: A parancsok neve és leírása a System.out.printf() segítségével szépen formázva kerül kiírásra a felhasználónak.

Logika

- **Parancsok rendezése**: A parancsok névszerinti rendezése biztosítja, hogy a lista átlátható legyen, és könnyen kereshetőek legyenek a parancsok.
- **Formázás**: A kiírt lista formázása egy táblázatot eredményez, ahol a parancsok neve és leírása jól elkülönül, és könnyen olvasható.

8.1.42 Load

Felelősség

Az Load osztály felelőssége a játékállapot betöltése a megadott fájlból. A parancs végrehajtása után a játék aktuális állapota visszaáll a fájlban tárolt adatok szerint, lehetővé téve a játékos számára, hogy folytassa a játékot a korábbi állapotból.

Ősosztályok

- **Command**: Az Load osztály a Command osztályból öröklődik, amely biztosítja a parancsok kezelését és a parancsok végrehajtásának lehetőségét.

Interfészek

- Az Load osztály nem implementál semmilyen interfészt.

Attribútumok

- **Nincs speciális attribútum az osztályban.** Az osztály minden szükséges adatot az execute metóduson keresztül dolgoz fel.

Metódusok

- **boolean execute(String[] args)**:
  - **Leírás**: Az execute metódus felelős a játékállapot betöltéséért a megadott fájlból. A fájlnév az args paraméterben található. Végrehajtja a fileban lévő parancsokat ezzel a kívánt állapotba állítva az alkalmazást.
  - **Láthatóság**: public
  - **Paraméterek**:
    - args: A parancs argumentumai, amelyek tartalmazzák a fájl nevét, amelyből a játékállapotot be kell tölteni.
  - **Visszatérési érték**: boolean – A metódus jelenleg kivételt dob, mivel a funkció még nincs implementálva.

8.1.43 Map

Felelősség

A Map osztály felelőssége a játék térképének reprezentálása, amely tartalmaz egy listát a tektonokból és egy kétdimenziós tömböt a gyors hozzáférés érdekében. A térkép biztosítja a tektonok hozzáadását, eltávolítását, a játékelemek frissítését, valamint a térkép logikai frissítését (tick) a játékmenet során.

Ősosztályok

- **Nincs őssosztály**: A Map osztály nem öröklődik más osztályoktól.

Interfészek

- Az Map osztály nem implementál semmilyen interfészt.

Attribútumok

- **tektons**: Listája a térképen található tektonoknak. Láthatósága: private, típusa: List<Tekton>.
- **tiles**: Kétdimenziós tömb, amely a térképen található összes tile-t tárolja. Láthatósága: private, típusa: Tile[][].
- **width**: A térkép szélessége. Láthatósága: private, típusa: int.
- **height**: A térkép magassága. Láthatósága: private, típusa: int.

Metódusok

- **Map(int width, int height)**:
  - **Leírás**: Konstruktor, amely inicializálja a térképet a megadott szélességgel és magassággal. Inicializálja a tektons listát és a tiles kétdimenziós tömböt.
  - **Láthatóság**: public
  - **Paraméterek**:

        - width: A térkép szélessége (int).
        - height: A térkép magassága (int).
    - **Visszatérési érték**: Nincs.
- **Map()**:
    - **Leírás**: Alapértelmezett konstruktor, amely 100x100-as térképet hoz létre.
    - **Láthatóság**: public
    - **Paraméterek**: Nincs.
    - **Visszatérési érték**: Nincs.
- **int addTekton(Tekton tekton)**:
    - **Leírás**: Hozzáad egy tektont a térképhez és visszaadja annak indexét.
    - **Láthatóság**: public
    - **Paraméterek**:
        - tekton: A hozzáadandó tekton.
    - **Visszatérési érték**: Az új tekton indexe (int).
- **List<Tekton> getTektons()**:
    - **Leírás**: Visszaadja a térképen található összes tektont tartalmazó listát.
    - **Láthatóság**: public
    - **Paraméterek**: Nincs.
    - **Visszatérési érték**: List<Tekton> – A tektonok listája.
- **void removeTekton(Tekton tekton)**:
    - **Leírás**: Eltávolít egy tektonot a térképről.
    - **Láthatóság**: public
    - **Paraméterek**:
        - tekton: A térképről eltávolítandó tekton.
    - **Visszatérési érték**: Nincs.
- **void update(GameEntity e)**:
    - **Leírás**: Frissíti a megadott játékelemeket (GameEntity).
    - **Láthatóság**: public
    - **Paraméterek**:
        - e: A frissítendő játékelem.
    - **Visszatérési érték**: Nincs.
- **void tick()**:
    - **Leírás**: A tick metódus a térképet frissíti, és elvégzi a játék logikáját. Jelenleg nem implementált, de a jövőben a tektonok és azok myceliuma kapcsolódásait kezelheti.
    - **Láthatóság**: public
    - **Paraméterek**: Nincs.
    - **Visszatérési érték**: Nincs.
- **Tile getTile(int x, int y)**:
    - **Leírás**: Visszaadja a térkép egy adott koordinátáján lévő tile-t, ha az érvényes. Ha a koordináták kívül esnek a térképen, akkor null-t ad vissza.
    - **Láthatóság**: public
    - **Paraméterek**:
        - x: A keresett tile X koordinátája.
        - y: A keresett tile Y koordinátája.
    - **Visszatérési érték**: A megtalált tile (Tile) vagy null, ha a koordináták kívül esnek a térképen.

8.1.44 MonoTile

Felelősség

A MonoTile osztály a térkép egy speciális típusú tile-ját képviseli, amely a mycelium növekedésére és elfoglaltságára van beállítva. A MonoTile alapvetően a növekedési ütemet és a tekton szülőjét kezeli, valamint meghatározza, hogy a tile elfoglalt-e a mycelium által.

Ősosztályok

- **Tile**: A MonoTile a Tile osztályból származik, és örökli annak tulajdonságait és viselkedését.

Interfészek

- Az MonoTile osztály nem implementál semmilyen interfészt.

Attribútumok

A MonoTile osztály az örökölt attribútumokat használja a Tile osztályból, valamint tartalmaz egy speciális metódust, amely a tile elfoglaltságát határozza meg.

- **myceliumSpace**: Az attribútum a tile-nál található mycelium területet jelöli, amit a szülő tekton határoz meg. Láthatósága: protected, típusa: int. (Ez az attribútum a Tile osztályból származik.)

Metódusok

MonoTile(int growthRate, Tekton parentTekton):

  - **Leírás**: Konstruktor, amely létrehozza a MonoTile-t egy adott növekedési ütemmel és a szülő tektonnal.
  - **Láthatóság**: public
  - **Paraméterek**:
    - growthRate: A tile növekedési üteme (int).
    - parentTekton: A tekton, amely a tile szülője (Tekton).
  - **Visszatérési érték**: Nincs.
- **boolean isTaken()**:
  - **Leírás**: Visszaadja, hogy a tile elfoglalt-e a mycelium által. A metódus akkor ad true értéket, ha a mycelium terület értéke 0, jelezve, hogy a tile üres és szabad. Ha nem 0, akkor a tile foglalt.
  - **Láthatóság**: public
  - **Paraméterek**: Nincs.
  - **Visszatérési érték**: boolean – Ha a tile elfoglalt, akkor true, egyébként false.

8.1.45 MultipleMyceliumGrowingOnTekton

Felelősség

A MultipleMyceliumGrowingOnTekton osztály a játékmenet egy szcenárióját modellezi, ahol több mycelium növekszik egy tekton felületén. A execute metódus segítségével inicializálja a játékosokat, a térképet, a tektonokat és a tile-okat, majd lehetővé teszi a mycelium növekedését több játékos számára.

Ősosztályok

- **UseCase**: A MultipleMyceliumGrowingOnTekton a UseCase osztályból származik, így örökli annak működését és funkcióit.

Interfészek

- Az osztály nem implementál semmilyen interfészt.

Attribútumok

A MultipleMyceliumGrowingOnTekton osztály nem tartalmaz attribútumokat, mivel a működése csak az execute metódusra koncentrál.

Metódusok

- **MultipleMyceliumGrowingOnTekton()**:
  - **Leírás**: Konstruktor, amely meghívja a szülőosztály konstruktorát, és beállítja az aktuális használati esetet (use case).
  - **Láthatóság**: public
  - **Visszatérési érték**: Nincs.
- **void execute()**:
  - **Leírás**: A metódus inicializálja a játék szcenárióját, amelyben a következő lépések történnek:
    1. Inicializálja a játék térképét (Map), a tektonot (Tekton), valamint a tile-okat (Tile).
    2. Hozzáadja a tektonot a térképhez és a tile-okat a tektonhoz.
    3. Beállítja a tile-ok szülő tektonját.
    4. Inicializál két játékost (FungusPlayer), majd mindkét játékos próbál myceliumot növeszteni egy-egy tile-on.
  - **Láthatóság**: public
  - **Paraméterek**: Nincs.
  - **Visszatérési érték**: Nincs.

8.1.46 Mycelium

Felelősség

A Mycelium osztály a gombafonalakat reprezentálja, amelyek a játék világában a különböző gombák közötti kapcsolatokat és kommunikációt biztosítják. Feladata a mycelium hálózatok kezelése, a gombatestekkel való kapcsolódás, az egészség fenntartása, és a környezeti hatásokra (pl. gyógyító területek) reagálás. A mycelium a kapcsolódó gombák épségét és növekedését biztosítja a játéknál. Az osztály kezelni tudja a mycelium halálát, leválasztását és a kapcsolatok keresését más myceliumokkal és gombatestekkel.

Ősosztályok

- Legősebb osztály → Fungus → Mycelium

Interfészek

- A Mycelium nem valósít meg külön interfészeket, mivel örökli a Fungus osztályt.

Attribútumok

- **maxHealth: int**: A mycelium maximális életereje. Láthatóság: private, típusa: int
- **player: FungusPlayer**: Az a játékos, aki birtokolja a myceliumot. Láthatóság: private, típusa: FungusPlayer
- **connectedBodies: List<FungusBody>**: A mycelium által összekapcsolt gombatestek listája. Láthatóság: private, típusa: List<FungusBody>
- **connectedMycelia: List<Mycelium>**: A mycelium által összekapcsolt más myceliumok listája. Láthatóság: private, típusa: List<Mycelium>

Metódusok

- **Mycelium(int id, int health, Tile currentTile, FungusPlayer player)**
  - Leírás: Konstruktor, amely inicializálja a myceliumot a megadott paraméterekkel, például az id, health, currentTile, és a player objektummal.

    - Láthatóság: public
    - Típus: void
- **Mycelium()**
    - Leírás: Alapértelmezett konstruktor, amely a szülő konstruktorát hívja meg és inicializálja a myceliumot.
    - Láthatóság: public
    - Típus: void
- **update()**
    - Leírás: A mycelium állapotának frissítése, a kapcsolat keresése más myceliumokkal és gombatestekkel, gyógyulás gyógyító területen, és sebzés kezelése, ha nincs kapcsolat.
    - Láthatóság: public
    - Típus: void
- **connect()**
    - Leírás: Kapcsolódás más myceliumokhoz és gombatestekhez. Ha kapcsolatot talál, a mycelium gyógyul.
    - Láthatóság: public
    - Típus: void
- **detach()**
    - Leírás: Leválasztja a myceliumot a hálózatról, eltávolítja a kapcsolódó myceliumokat és gombatesteket.
    - Láthatóság: public
    - Típus: void
- **searchConnection()**
    - Leírás: Keresés más myceliumok és gombatestek után a szomszédos cellákban. Ha kapcsolatot talál, frissíti a kapcsolódó myceliumok és gombatestek listáját.
    - Láthatóság: private
    - Típus: boolean
- **addConnectedMycelium(Mycelium myc)**
    - Leírás: Hozzáadja a myceliumot a kapcsolódó myceliumok listájához, és rekurzívan frissíti az összes kapcsolódó myceliumot.
    - Láthatóság: private
    - Típus: void
- **addConnectedBody(FungusBody fb)**
    - Leírás: Hozzáadja a gombatestet a kapcsolódó testek listájához, és rekurzívan frissíti az összes kapcsolódó myceliumot.
    - Láthatóság: private
    - Típus: void
- **die()**
    - Leírás: A mycelium halála, leválasztás a hálózatról, eltávolítás a játékos listájából és a térképről.
    - Láthatóság: public
    - Típus: void
- **getCut()**
    - Leírás: A mycelium levágása, halálának kiváltása.
    - Láthatóság: public
    - Típus: void
- **damage()**
    - Leírás: Sebzés alkalmazása a myceliumra. Ha az életerő nullára csökken, meghívja a die() metódust.

    - Láthatóság: public
    - Típus: void
- **heal()**
    - Leírás: A mycelium gyógyulása, ha nem éri el a maximális életerőt. Gyógyítás csak gyógyító területen történhet.
    - Láthatóság: public
    - Típus: void

8.1.47 MyceliumDie

Felelősség

A MyceliumDie osztály a játékban lévő myceliumok halálának kezelésére szolgál. Az osztály a Command osztályt örökli, és az egyik specifikus parancsot implementálja, amely lehetővé teszi a játékos számára, hogy egy adott myceliumot "meghaljon". A parancs a mycelium azonosítóját várja bemeneti paraméterként, és végrehajtja a megfelelő műveleteket annak eltávolítására a térképről.

Ősosztályok

- Legősebb osztály → Object → Command

Interfészek

- A MyceliumDie osztály nem valósít meg interfészeket.

Attribútumok

- **commandName: String**: A parancs neve. Láthatóság: protected, típusa: String
- **commandDescription: String**: A parancs leírása, amely megmagyarázza, hogy mit csinál a parancs. Láthatóság: protected, típusa: String
- **commandSyntax: String**: A parancs szintaxisa, amely bemutatja a helyes formátumot, amelyet a felhasználónak követnie kell. Láthatóság: protected, típusa: String

Metódusok

- **MyceliumDie()**
    - Leírás: Konstruktor, amely beállítja a parancs nevét, leírását és szintaxisát.
    - Láthatóság: public
    - Típus: void
- **execute(String[] args)**
    - Leírás: A mycelium_die parancs végrehajtása. A metódus először ellenőrzi, hogy a bemeneti paraméterek helyesek-e, majd a mycelium azonosítót keresve meghívja a mycelium halálának műveletét. Ha a mycelium nem található, a parancs hibát jelez.
    - Láthatóság: public
    - Típus: boolean
    - **Algoritmus:**
        1. Ellenőrzi, hogy helyes számú paramétert kaptunk-e (2).
        2. Ha a térkép nincs inicializálva, visszatér false-al.
        3. A parseEntityId metódussal megpróbálja kikeresni a myceliumot az azonosító alapján.
        4. Ha a mycelium null, visszatér false-al.
        5. Meghívja a mycelium.die() metódust, hogy meghaljon a mycelium.

6. Eltávolítja a myceliumot a jelenlegi térképről (getCurrentTile().removeEntity).
7. Visszatér false-al, jelezve, hogy a parancs végrehajtása befejeződött.

8.1.48 MyceliumGrow

Felelősség

A MyceliumGrow osztály a mycelium növekedését kezelő parancsot valósít meg. A parancs célja, hogy a myceliumot egy meghatározott térképi helyre terjessze ki, és bővítse a mycelium hálózatot egy új téglalap (tektonikus lemez) és annak egy meghatározott csempéje szerint. A parancsot a játékosok használhatják a mycelium hálózatuk kiterjesztésére.

Ősosztályok

- Legősebb osztály → Object → Command

Interfészek

- A MyceliumGrow osztály nem valósít meg interfészeket.

Attribútumok

- **commandName: String**: A parancs neve. Láthatóság: protected, típusa: String
- **commandDescription: String**: A parancs leírása, amely megmagyarázza, hogy mit csinál a parancs. Láthatóság: protected, típusa: String
- **commandSyntax: String**: A parancs szintaxisa, amely bemutatja a helyes formátumot, amelyet a felhasználónak követnie kell. Láthatóság: protected, típusa: String

Metódusok

- **MyceliumGrow()**
  - Leírás: Konstruktor, amely beállítja a parancs nevét, leírását és szintaxisát.
  - Láthatóság: public
  - Típus: void
- **execute(String[] args)**
  - Leírás: A mycelium_grow parancs végrehajtása. A metódus először ellenőrzi, hogy a bemeneti paraméterek helyesek-e, majd a mycelium azonosítót és a célzott tektonikus lemezt és csempét keresi. Ha minden valid, akkor a játékos mycelium hálózatát növeli a megfelelő helyen.
  - Láthatóság: public
  - Típus: boolean
  - **Algoritmus:**
    1. Ellenőrzi, hogy helyes számú paramétert kaptunk-e (4).
    2. Ha a térkép nincs inicializálva, visszatér false-al.
    3. A parseEntityId metódussal megpróbálja kikeresni a myceliumot az azonosító alapján.
    4. Ha a mycelium null, visszatér false-al.
    5. A parseTektonAndTile metódussal megpróbálja kikeresni a célzott tektonikus lemezt és csempét.
    6. Ha a tektonikus lemez és csempe invalid, visszatér false-al.
    7. Meghívja az app.getFungusPlayer().growMycelium(tektonTile.getTile()) metódust, hogy bővítse a mycelium hálózatot a célzott helyre.
    8. Visszatér false-al, jelezve, hogy a parancs végrehajtása befejeződött.

8.1.49 MyceliumGrowing

Felelősség

A MyceliumGrowing osztály a mycelium növekedését szimuláló use case-t valósít meg. Ennek a use case-nek a feladata, hogy kezdeményezze a mycelium növekedését egy meghatározott csempe (Tile) számára a megfelelő tektonikus lemezen (Tekton). A folyamat során az osztály létrehozza a szükséges objektumokat, beállítja azokat és elindítja a mycelium növekedését a játékos (FungusPlayer) irányításával.

Ősosztályok

- Legősebb osztály → Object → UseCase

Interfészek

- A MyceliumGrowing osztály nem valósít meg interfészeket.

Attribútumok

- **id: int**: A use case azonosítója, amely a szintaxis szerint a 9-es értéket képviseli. Láthatóság: protected, típusa: int
- **name: String**: A use case neve, amely "Mycelium growing". Láthatóság: protected, típusa: String

Metódusok

- **MyceliumGrowing()**
  - Leírás: Konstruktor, amely beállítja a use case azonosítóját és nevét.
  - Láthatóság: public
  - Típus: void
- **execute()**
  - Leírás: A MyceliumGrowing use case végrehajtása. A metódus a következő lépéseket hajtja végre:
    1. Először kiírja, hogy "Initializing scene...", és beállítja a nyíl irányát.
    2. Létrehozza a Tekton objektumot.
    3. Létrehozza a Tile objektumot.
    4. Létrehozza a FungusPlayer objektumot.
    5. A Tile-hoz hozzárendeli a Tekton szülőt.
    6. A FungusPlayer objektumot használva elindítja a mycelium növekedését a csempe számára.
  - Láthatóság: public
  - Típus: void
  - **Algoritmus:**
    1. A printWrapper metódus segítségével információt ír ki a kezdeti állapotról.
    2. A Tekton objektum létrehozása a new Tekton(null) formában történik, bár nem kerül megadásra konkrét érték.
    3. A Tile objektumot a new Tile() segítségével hozza létre.
    4. A FungusPlayer objektumot a new FungusPlayer() metódus segítségével hozza létre.
    5. A setParentTekton metódussal a Tile szülőjeként hozzárendeli a Tekton-t.
    6. A growMycelium metódus segítségével a FungusPlayer elindítja a mycelium növekedését a Tile-ra.

7. A logger minden lépéshez adatokat rögzít (tek, t, fp), segítve a folyamat követését.

8.1.50 MyceliumGrowingWithSpore

Felelősség

A MyceliumGrowingWithSpore osztály a mycelium növekedésének szimulálását valósítja meg spórák segítségével. Ez a use case a mycelium növekedését egy újabb mechanizmussal bővíti, ahol a növekedés a spórák által terjedő módon történik. Az osztály a mycelium növekedésének színhelyét és a szükséges objektumokat (pl. tektonikus lemez, csempe, játékos) hozza létre, majd elindítja a növekedési folyamatot.

Ősosztályok

- Legősebb osztály → Object → UseCase

Interfészek

- A MyceliumGrowingWithSpore osztály nem valósít meg interfészeket.

Attribútumok

- **id: int**: A use case azonosítója, amely a szintaxis szerint a 10-es értéket képviseli. Láthatóság: protected, típusa: int
- **name: String**: A use case neve, amely "Mycelium growing with spore". Láthatóság: protected, típusa: String

Metódusok

- **MyceliumGrowingWithSpore()**
  - Leírás: Konstruktor, amely beállítja a use case azonosítóját és nevét.
  - Láthatóság: public
  - Típus: void
- **execute()**
  - Leírás: A MyceliumGrowingWithSpore use case végrehajtása. A metódus a következő lépéseket hajtja végre:
    1. Először kiírja, hogy "Initializing scene...", és beállítja a nyíl irányát.
    2. Létrehozza a Tekton objektumot.
    3. Létrehozza a Tile objektumot.
    4. Létrehozza a FungusPlayer objektumot.
    5. A Tile-hoz hozzárendeli a Tekton szülőt.
    6. A FungusPlayer objektumot használva elindítja a mycelium növekedését a csempe számára.
  - Láthatóság: public
  - Típus: void
  - **Algoritmus:**
    1. A printWrapper metódus segítségével információt ír ki a kezdeti állapotról.
    2. A Tekton objektum létrehozása a new Tekton(null) formában történik, bár nem kerül megadásra konkrét érték.
    3. A Tile objektumot a new Tile() segítségével hozza létre.
    4. A FungusPlayer objektumot a new FungusPlayer() metódus segítségével hozza létre.
    5. A setParentTekton metódussal a Tile szülőjeként hozzárendeli a Tekton-t.

6. A growMycelium metódus segítségével a FungusPlayer elindítja a mycelium növekedését a Tile-ra.
7. A logger minden lépéshez adatokat rögzít (tek, t, fp), segítve a folyamat követését.

8.1.51 OnlyOneMyceliumGrowingOnTekton

Felelősség

Az OnlyOneMyceliumGrowingOnTekton osztály egy olyan használati esetet valósít meg, ahol a mycelium csak egyetlen helyen növekedhet egy tektonikus lemezen (tekton), biztosítva, hogy ne növekedjen mycelium több csempén ugyanazon tektonikus lemezen. A játékos megpróbál egy myceliumot növeszteni két különböző csempén ugyanazon a tektonikus lemezen, de a második kísérletet el kell utasítani, mivel a szabályok szerint csak egyetlen mycelium növekedhet egy tektonikus lemezen.

Ősosztályok

- Legősebb osztály → Object → UseCase

Interfészek

- Az OnlyOneMyceliumGrowingOnTekton osztály nem valósít meg interfészeket.

Attribútumok

- **id: int**: A use case azonosítója, amely a 17-es értéket képviseli. Láthatóság: protected, típusa: int
- **name: String**: A use case neve, amely "Only One Mycelium Growing on Tekton". Láthatóság: protected, típusa: String

Metódusok

- **OnlyOneMyceliumGrowingOnTekton()**
  - Leírás: Konstruktor, amely beállítja a use case azonosítóját és nevét.
  - Láthatóság: public
  - Típus: void
- **execute()**
  - Leírás: A OnlyOneMyceliumGrowingOnTekton use case végrehajtása. A metódus a következő lépéseket hajtja végre:
    1. A színhely inicializálása és információ kiírása.
    2. Létrehozza a Map objektumot, amely a térképet reprezentálja.
    3. Létrehozza a Tekton objektumot és hozzáadja a térképhez.
    4. Két Tile objektumot hoz létre, majd ezeket hozzáadja a tektonikus lemezhez.
    5. A csempék szülőjeként hozzárendeli a tektonikus lemezt.
    6. Létrehozza a FungusPlayer objektumot.
    7. A játékos megpróbál myceliumot növeszteni az egyik csempén.
    8. A játékos megpróbál myceliumot növeszteni a második csempén, ugyanazon tektonikus lemezen, de ez nem sikerül a szabályoknak megfelelően.
  - Láthatóság: public
  - Típus: void
  - **Algoritmus:**
    1. A printWrapper metódus segítségével információt ír ki a kezdeti állapotról.

2. A Map objektumot a new Map() segítségével hozza létre.
3. A Tekton objektumot a new Tekton(m) metódussal hozzák létre, ahol m a térkép.
4. A addTekton metódus segítségével a tekton hozzáadódik a térképhez.
5. A két Tile objektumot a new Tile() metódussal hozzák létre.
6. A addTile metódus segítségével a csempék hozzáadódnak a tektonikus lemezhez.
7. A setParentTekton metódussal mindkét csempének szülőjeként hozzárendelik a tektonikus lemezt.
8. A FungusPlayer objektumot a new FungusPlayer() metódussal hozzák létre.
9. Az első mycelium növesztési próbálkozás sikeres lesz, hiszen az első csempe rendelkezik szülő tektonikus lemezzel.
10. A második próbálkozás ugyanazon tektonikus lemezen nem fog sikerülni, mivel a szabályok szerint egy tektonikus lemezen csak egy mycelium növekedhet.

8.1.52 Player

Felelősség

A Player osztály egy általános absztrakt osztály, amely egy játékos alapvető jellemzőit és működését definiálja, például a pontszámot, az akciópontokat és az akciók végrehajtását a játék során. Az osztály továbbá tartalmazza azokat a metódusokat, amelyek az alapvető játékos interakciókat (például az akciók számolása, a pontok frissítése) végzik el.

Ősosztályok

- Legősebb osztály → Object → Player

Attribútumok

- **score: int**: A játékos aktuális pontszáma.
  - Láthatóság: protected
  - Típus: int
- **actionPoints: int**: A játékos aktuális akciópontjai, amelyeket minden egyes játék kör (tick) alatt frissítenek.
  - Láthatóság: protected
  - Típus: int
- **ACTIONS_PER_TICK: final int**: Az a konstans érték, amely meghatározza, hány akciópontot kap a játékos minden egyes körben (tick).
  - Láthatóság: public static final
  - Típus: int

Metódusok

- **protected Player()**
  - Leírás: A Player osztály konstruktora, amely inicializálja a score és actionPoints attribútumokat. A score kezdetben 0-ra van állítva, míg az actionPoints is 0-ra.
  - Láthatóság: protected
  - Típus: void
- **public void tick()**
  - Leírás: A tick metódus minden egyes játék körben frissíti a játékos akciópontjait az ACTIONS_PER_TICK értékének megfelelően.

    - Láthatóság: public
    - Típus: void
- **public int getActionPoints()**
    - Leírás: Visszaadja a játékos jelenlegi akciópontjait.
    - Láthatóság: public
    - Típus: int
- **public int getScore()**
    - Leírás: Visszaadja a játékos aktuális pontszámát.
    - Láthatóság: public
    - Típus: int
- **public void setActionPoints(int actionPoints)**
    - Leírás: Beállítja a játékos akciópontjainak értékét.
    - Láthatóság: public
    - Típus: void
- **public void updateScore(int score)**
    - Leírás: Frissíti a játékos pontszámát az argumentumban megadott értékkel, hozzáadva azt a jelenlegi pontszámhoz.
    - Láthatóság: public
    - Típus: void
- **public abstract void pickStartingTile(Tile tile)**
    - Leírás: Absztrakt metódus, amelyet az osztályt öröklő alosztályoknak kell implementálniuk. A metódus feladata, hogy kiválassza a kezdő csempét.
    - Láthatóság: public abstract
    - Típus: void

8.1.53 Save

Felelősség

A Save osztály egy parancsot reprezentál, amely a játék állapotát egy fájlba próbálná menteni. Az osztály a Command absztrakt osztályból öröklődik, és annak az implementálásával rendelkezik, hogy miként kell kezelni a fájlba történő mentést.

Ősosztályok

- Legősebb osztály → Object → Command → Save

Metódusok

- **public Save()**
    - Leírás: A konstruktor inicializálja a parancs nevét, leírását és a szintaxisát. A parancs neve save, amely a játék állapotát mentené egy fájlba, és a fájlnevet várja argumentumként.
    - Paraméterek:
        - name: String → save
        - description: String → "Write the game state to the specified file"
        - syntax: String → "save <filename>"
- **public boolean execute(String[] args)**
    - Leírás: Az execute metódus az a hely, ahol az összes parancs végrehajtása történik. A Save parancs esetében ez a metódus felelős a játék állapotának fájlba mentéséért.
    - Paraméterek:
        - args: String[] → Az argumentumokat tartalmazó tömb, amely tartalmazza a fájl nevét, ahova menteni kell.

- Visszatérési érték:
  - boolean → Sikeres volt-e a mentés.

8.1.54 SetTileParentTekton

Felelősség

A SetTileParentTekton osztály egy parancsot reprezentál, amely arra szolgál, hogy beállítsa a térkép egyik Tile objektumának szülő tektonikus lemezét (Tekton). Az osztály a Command absztrakt osztályból öröklődik, és annak implementálásával rendelkezik, hogy miként kell kezelni a szülő tektonikus lemez beállítását.

Ősosztályok

- Legősebb osztály → Object → Command → SetTileParentTekton

Konstruktorok

- **public SetTileParentTekton()**
  - Leírás: A konstruktor inicializálja a parancs nevét, leírását és a szintaxisát. A parancs neve set_tile_parent_tekton, amely egy Tile szülő tektonikus lemezét állítja be, és a parancs két argumentumot vár: a tile id-t és a parent tectonic plate id-t.
  - Paraméterek:
    - name: String → set_tile_parent_tekton
    - description: String → "Set the parent tectonic plate of the tile"
    - syntax: String → "set_tile_parent_tekton <tile id> <parent tectonic plate id>"

Metódusok

- **public boolean execute(String[] args)**
  - Leírás: Az execute metódus az a hely, ahol az összes parancs végrehajtása történik. A SetTileParentTekton parancs esetében ez a metódus a felelős a Tile szülő tektonikus lemezének beállításáért.
  - Paraméterek:
    - args: String[] → Az argumentumokat tartalmazó tömb, amely tartalmazza a tile id-t és a parent tectonic plate id-t.
  - Visszatérési érték:
    - boolean → Sikeres volt-e a művelet

8.1.55 SetTileType

• Felelősség

A SetTileType osztály a játék térképén lévő egyes tile-ok típusainak beállításáért felelős. A parancs három típusú tile-t kezel: MonoTile, HealTile, és AcidTile. A parancs az argumentumok alapján módosítja egy adott Tile típusát egy Tekton lemezen, a megfelelő ID-k és típus megadásával. Az osztály az execute metódust implementálja, amely végrehajtja a parancsot, ha a megfelelő paraméterek érvényesek, és végrehajtja a megfelelő tile típusú objektumok létrehozását a térképen.

Ősosztályok

- Legősebb osztály → Object → Command → SetTileType

Interfészek

A SetTileType osztály nem implementál közvetlenül interfészeket.

Attribútumok

Nincsenek külön attribútumai, mivel az osztály a Command osztálytól örökli a paramétereket és a működést. Az execute metódusban átadott argumentumok és a map térkép állapota határozza meg a működését.

Metódusok

- **public boolean execute(String[] args)**
  - Leírás: Az execute metódus a parancs végrehajtásáért felelős. Ellenőrzi az argumentumok helyességét, a térkép inicializáltságát, majd a Tekton és Tile objektumot frissíti a megadott típusra (Mono, Heal, Acid). Ha érvénytelen típus van megadva, hibaüzenetet jelenít meg.
  - Láthatóság: public
  - Paraméterek:
    - args: String[]: A parancs argumentumai (tectonic plate ID, tile ID, tile type)
  - Visszatérési érték: boolean: Visszaadja a végrehajtás eredményét (ebben az esetben mindig false).
  - Algoritmus:

1. Ellenőrzi az argumentumok számát és a térkép állapotát.
2. Az ID-k alapján megtalálja a megfelelő Tekton és Tile objektumot.
3. A switch szerkezet segítségével beállítja a tile típusát.
4. Hibaüzenetet ír ki, ha a típus érvénytelen.

8.1.56 SlowSpore

Felelősség

A **SlowSpore** osztály felelőssége, hogy lassító hatást gyakoroljon azokra az rovarokra, amelyek megeszik. A lassító spóra azzal csökkenti az rovarok sebességét, hogy beállítja annak sebességét egy adott százalékos értékre, amelyet az **effectValue** attribútum határoz meg. A spóra élettartama alatt az érintett rovar lassabbá válik, amíg a spóra hatása nem szűnik meg.

Ősosztályok

- **Object** (legősebb osztály)
- **Spore** (Ősosztály2)

Interfészek

- Az **Insect** osztályt érintő metódusok miatt nem implementál interfészt, de felüldefiniálja a **Spore** osztályban definiált metódusokat, így viselkedése megegyezik a szülőosztályéval.

Attribútumok

- **private boolean isConsumed**: A spóra elfogyasztásának állapotát jelző változó. Láthatósága: -, típusa: boolean
- **private int effectValue**: A lassító spóra hatásának mértéke, amely a rovar sebességét százalékban csökkenti. Láthatósága: -, típusa: int
- **private Tile currentTile**: A spóra jelenlegi helye a játéktáblán. Láthatósága: -, típusa: Tile
- **private int lifetime**: A spóra élettartama, amíg hatása aktív. Láthatósága: -, típusa: int

- **private int effectTime**: Az idő, ameddig a spóra hatása tart. Láthatósága: -, típusa: int
- **private int nutrientValue**: A spóra tápanyag értéke. Láthatósága: -, típusa: int

Metódusok

- **public SlowSpore(int id, Tile currentTile, int nutrientValue, int lifetime, int effectTime, int effectValue)**
  A konstruktora a **Spore** szülőosztály konstruktorát hívja meg, és inicializálja a lassító spóra attribútumait.
  Leírás: A konstruktor segítségével hozzuk létre a **SlowSpore** objektumot, amely a spóra egyedi azonosítóját, elhelyezkedését, tápanyag értékét, élettartamát, hatás idejét és sebességcsökkentés értékét állítja be.
  Láthatósága: +, típusa: void
- **public SlowSpore(int effectValue)**
  Egy alternatív konstruktor, amely csak a **effectValue** értéket állítja be, a többi attribútum alapértelmezett értékre kerül.
  Leírás: A spóra effektív értékének megadása, valamint a **UseCase** osztályban való regisztrálás történik. Ezen kívül a spóra inicializálásáról naplózási információt is generál.
  Láthatósága: +, típusa: void
- **public void getEaten(Insect i)**
  A metódus akkor kerül végrehajtásra, amikor az rovar megeszi a spórát. A metódus beállítja a rovar sebességét a **effectValue** százalékos értékre, és eltávolítja a spórát a jelenlegi helyéről.
  Leírás: A metódus beállítja az **isConsumed** attribútumot igazra, alkalmazza a lassító hatást a rovar sebességére, és eltávolítja a spórát a játéktábláról.
  Láthatósága: +, típusa: void
- **public void removeEffect(Insect i)**
  A metódus visszaállítja a rovar sebességét az alapértékre, törölve a lassító spóra hatását.
  Leírás: A spóra hatásának eltávolítása után a rovar sebességét 0-ra állítja, így visszatér a normál mozgásához.
  Láthatósága: +, típusa: void

8.1.57 SpeedUpSpore

Felelősség

A **SpeedUpSpore** osztály felelőssége, hogy felgyorsítsa azokat az rovarokat, amelyek megeszik. A gyorsító spóra az **effectValue** százalékos értékének megfelelően növeli az rovar sebességét, így gyorsabbá téve a mozgást a spóra hatásának időtartama alatt. Ez a hatás addig tart, amíg a spóra érvényben van.

Ősosztályok

- **Object** (legősebb osztály)
- **Spore** (Ősosztály2)
- **• Interfészek**
- Az **Insect** osztályt érintő metódusok miatt nem implementál interfészt, de felüldefiniálja a **Spore** osztályban definiált metódusokat, így viselkedése megegyezik a szülőosztályéval.

Attribútumok

- **private boolean isConsumed**: A spóra elfogyasztásának állapotát jelző változó. Láthatósága: -, típusa: boolean
- **private int effectValue**: A gyorsító spóra hatásának mértéke, amely a rovar sebességét százalékban növeli. Láthatósága: -, típusa: int
- **private Tile currentTile**: A spóra jelenlegi helye a játéktáblán. Láthatósága: -, típusa: Tile
- **private int lifetime**: A spóra élettartama, amíg hatása aktív. Láthatósága: -, típusa: int
- **private int effectTime**: Az idő, ameddig a spóra hatása tart. Láthatósága: -, típusa: int
- **private int nutrientValue**: A spóra tápanyag értéke. Láthatósága: -, típusa: int

Metódusok

- **public SpeedUpSpore(int id, Tile currentTile, int nutrientValue, int lifetime, int effectTime, int effectValue)**
  A konstruktora a **Spore** szülőosztály konstruktorát hívja meg, és inicializálja a gyorsító spóra attribútumait.
  Leírás: A konstruktor segítségével hozzuk létre a **SpeedUpSpore** objektumot, amely a spóra egyedi azonosítóját, elhelyezkedését, tápanyag értékét, élettartamát, hatás idejét és sebességnövelési értékét állítja be.
  Láthatósága: +, típusa: void
- **public SpeedUpSpore(int effectValue)**
  Egy alternatív konstruktor, amely csak a **effectValue** értéket állítja be, a többi attribútum alapértelmezett értékre kerül.
  Leírás: A spóra effektív értékének megadása, valamint a **UseCase** osztályban való regisztrálás történik. Ezen kívül a spóra inicializálásáról naplózási információt is generál.
  Láthatósága: +, típusa: void
- **public void getEaten(Insect i)**
  A metódus akkor kerül végrehajtásra, amikor az rovar megeszi a spórát. A metódus beállítja a rovar sebességét a **effectValue** százalékos értékre, és eltávolítja a spórát a jelenlegi helyéről.
  Leírás: A metódus beállítja az **isConsumed** attribútumot igazra, alkalmazza a gyorsító hatást a rovar sebességére, és eltávolítja a spórát a játéktábláról.
  Láthatósága: +, típusa: void
- **public void removeEffect(Insect i)**
  A metódus visszaállítja a rovar sebességét az alapértékre, törölve a gyorsító spóra hatását.
  Leírás: A spóra hatásának eltávolítása után a rovar sebességét 0-ra állítja, így visszatér a normál mozgásához.
  Láthatósága: +, típusa: void

8.1.58 SplitSpore

Felelősség

A SplitSpore osztály a Spore osztály egy speciális változata, amelyet egy rovar (Insect) elfogyasztása után egy új rovar (Insect) keletkezésével reagál. A SplitSpore osztály felelős a spórák kezeléséért, amelyeket rovarok fogyaszthatnak, és a fogyasztás hatására új rovarok szaporodnak. Az osztály továbbá a UseCase osztály segítségével naplózza az inicializálást és helyettesíti az aktuális entitást.

Ősosztályok

- Legősebb osztály → Object → Spore → SplitSpore

Interfészek

- A SplitSpore osztály nem implementál interfészeket.

Attribútumok

A SplitSpore osztály az örökölt attribútumokat használja a Spore osztályból, amely az alábbiakat tartalmazza:

- id: A spóra egyedi azonosítója.
- currentTile: A spóra jelenlegi helye a térképen.
- nutrientValue: A spóra tápláló értéke, amely meghatározza, hogy mennyi tápanyagot ad a rovaroknak.
- lifetime: A spóra élettartama.
- isConsumed: A spóra elfogyasztott állapota.

Metódusok

- **public SplitSpore()**
  - Leírás: Az alapértelmezett konstruktor, amely meghívja a szülőosztály konstruktorát és végrehajtja a spóra inicializálását. A UseCase.replace() metódus segítségével a spórát lecseréli egy másik entitásra, majd a UseCase.printWrapper() segítségével naplózza az inicializálást.
  - Láthatóság: public
  - Paraméterek: Nincs.
  - Visszatérési érték: Nincs.
  - Algoritmus:
    1. Meghívja a szülőosztály konstruktorát.
    2. Lecseréli a spórát a UseCase.replace() segítségével.
    3. Naplózza az inicializálást a UseCase.printWrapper() segítségével.
- **public SplitSpore(int id, Tile currentTile, int nutrientValue, int lifetime)**
  - Leírás: A paraméterezett konstruktor, amely lehetővé teszi egy új SplitSpore objektum létrehozását az id, currentTile, nutrientValue és lifetime paraméterekkel.
  - Láthatóság: public
  - Paraméterek:
    - id: int: A spóra egyedi azonosítója.
    - currentTile: Tile: A spóra aktuális helye a térképen.
    - nutrientValue: int: A spóra tápláló értéke.
    - lifetime: int: A spóra élettartama.
  - Visszatérési érték: Nincs.
  - Algoritmus:
    1. Meghívja a szülőosztály konstruktorát a paraméterekkel.
- **public void getEaten(Insect i)**
  - Leírás: Ha egy rovar elfogyasztja a spórát, a getEaten metódus végrehajtja a spóra fogyasztásának logikáját, beállítva az isConsumed változót igazra, meghívja az Insect.split() metódust, és eltávolítja a spórát a jelenlegi térképről.
  - Láthatóság: public
  - Paraméterek:
    - i: Insect: Az a rovar, amely elfogyasztja a spórát.
  - Visszatérési érték: Nincs.
  - Algoritmus:
    1. Beállítja az isConsumed változót igazra.
    2. Meghívja a rovar split() metódusát.

3. Eltávolítja a spórát a jelenlegi tile-ról a currentTile.removeEntity(this) segítségével.

8.1.59 Spore

Felelősség

A Spore osztály egy absztrakt osztály, amely a játékban található spórákat reprezentálja. Közös tulajdonságokat és metódusokat tartalmaz minden spóra típus számára. A spóra képes rovarok általi fogyasztásra, valamint hatások alkalmazására a rovarokra. Az osztály kezeli a spóra élettartamát, hatásainak idejét, és azt, hogy a spóra elfogyott-e már.

Ősosztályok

- Legősebb osztály → Object → GameEntity → Spore

Interfészek

- A Spore osztály nem implementál interfészeket.

Attribútumok

- nutrientValue: A spóra tápanyagértéke. Megadja, hogy mennyi tápanyagot biztosít, ha egy rovar megeszi.
  - Láthatóság: protected
  - Típus: int
- lifetime: A spóra élettartama, azaz a játékban hány körig marad életben.
  - Láthatóság: protected
  - Típus: int
- effectTime: Az a szám, amely megadja, hány körig alkalmazza a spóra a hatását a rovarokra.
  - Láthatóság: protected
  - Típus: int
- effectValue: A spóra hatásának erőssége, amely meghatározza a hatás mértékét a rovarra.
  - Láthatóság: protected
  - Típus: int
- isConsumed: Egy boolean változó, amely jelzi, hogy a spóra elfogyott-e már egy rovar által.
  - Láthatóság: protected
  - Típus: boolean

Metódusok

- **protected Spore(int id, Tile currentTile, int nutrientValue, int lifetime, int effectTime, int effectValue)**
  - Leírás: A spóra paraméterezett konstruktora, amely inicializálja a spóra azonosítóját, helyét, tápanyagértékét, élettartamát, hatásának idejét és erejét. Az inicializálás után a spóra hozzáadásra kerül a megadott térképhez.
  - Láthatóság: protected
  - Paraméterek:
    - id: int: A spóra egyedi azonosítója.
    - currentTile: Tile: A spóra aktuális helye a térképen.
    - nutrientValue: int: A spóra tápanyagértéke.
    - lifetime: int: A spóra élettartama.
    - effectTime: int: A hatás időtartama.

        - effectValue: int: A hatás ereje.
    - Visszatérési érték: Nincs.
    - Algoritmus:
        1. Meghívja a szülőosztály konstruktorát.
        2. Inicializálja az attribútumokat.
        3. Hozzáadja a spórát a térképhez.
- **protected Spore()**
    - Leírás: Az alapértelmezett konstruktor, amely a szülőosztály alapértelmezett konstruktorát hívja meg.
    - Láthatóság: protected
    - Paraméterek: Nincs.
    - Visszatérési érték: Nincs.
    - Algoritmus:
        1. Meghívja a szülőosztály alapértelmezett konstruktorát.
- **public void getEaten(Insect i)**
    - Leírás: Ez a metódus akkor kerül végrehajtásra, amikor egy rovar elfogyasztja a spórát. A spóra elfogyott státuszra kerül, és az osztályokban implementált logika alapján hatásokat alkalmazhat a rovarra.
    - Láthatóság: public
    - Paraméterek:
        - i: Insect: Az a rovar, amely elfogyasztja a spórát.
    - Visszatérési érték: Nincs.
    - Algoritmus:
        1. Beállítja az isConsumed változót igazra.
        2. A spóra hatásainak alkalmazását a származtatott osztályok valósítják meg.
- **public void removeEffect(Insect i)**
    - Leírás: Ezt a metódust a származtatott osztályok implementálják, hogy eltávolítsák a spóra hatásait a rovarról.
    - Láthatóság: public
    - Paraméterek:
        - i: Insect: Az a rovar, akiről el kell távolítani a spóra hatásait.
    - Visszatérési érték: Nincs.
    - Algoritmus:
        1. A hatások eltávolítását a származtatott osztályok valósítják meg.
- **public void update()**
    - Leírás: Ez a metódus minden egyes körben csökkenti a spóra élettartamát, ha még nem lett elfogyasztva. Ha a spórát megevett egy rovar, akkor a hatás idejét csökkenti.
    - Láthatóság: public
    - Paraméterek: Nincs.
    - Visszatérési érték: Nincs.
    - Algoritmus:
        1. Ha a spóra elfogyott, csökkenti a effectTime értékét.
        2. Ha a spóra nem lett elfogyasztva, csökkenti a lifetime értékét.
- **public int getLifetime()**
    - Leírás: A spóra élettartamának lekérdezése.
    - Láthatóság: public
    - Paraméterek: Nincs.
    - Visszatérési érték: int

    - Algoritmus:
        1. Visszaadja a spóra lifetime értékét.
- **public int getEffectTime()**
    - Leírás: A spóra hatásidejének lekérdezése.
    - Láthatóság: public
    - Paraméterek: Nincs.
    - Visszatérési érték: int
    - Algoritmus:
        1. Visszaadja a spóra effectTime értékét.
- **public int getNutrientValue()**
    - Leírás: A spóra tápanyagértékének lekérdezése.
    - Láthatóság: public
    - Paraméterek: Nincs.
    - Visszatérési érték: int
    - Algoritmus:
        1. Visszaadja a spóra nutrientValue értékét.

8.1.60 Tekton Osztály

Felelősség:

A **Tekton** osztály felelőssége egy terület (tektonikus lemez) kezelésének, amely a térképen található csempék (Tile) gyűjteményét reprezentálja. Az osztály kezeli a tektonikus lemezre eső csempék adatait, és különböző műveleteket hajt végre, például a lemez kettévágását a középső vonalon egy törés mentén. Továbbá, felelős a játékosok számára elérhető spórák nyilvántartásáért, és különböző események, például a törés és a játékosok kölcsönhatásai kezeléséért.

Ősosztályok:

- **Object** (Legősebb osztály)

Interfészek:

- Nincs közvetlenül megvalósított interfész.

Attribútumok:

- **breakChance**: A tektonikus lemez törésének esélye, láthatósága: **private**, típusa: int
- **tiles**: A tektonikus lemezhez tartozó csempék listája, láthatósága: **private**, típusa: List<Tile>
- **fungusBody**: A tektonikus lemezen található gomba testet reprezentáló objektum, láthatósága: **private**, típusa: FungusBody
- **map**: A térkép, amelyhez a tektonikus lemez tartozik, láthatósága: **private**, típusa: Map
- **playerSpores**: A játékosokhoz rendelt spórák száma, láthatósága: **private**, típusa: HashMap<FungusPlayer, Integer>

Metódusok:

- **Tekton(Map map)**: Konstruktor, amely inicializálja a tektonikus lemezt és hozzáadja azt a térképhez.
- **addTile(Tile tile)**: Hozzáad egy csempét a tektonikus lemezhez és beállítja a csempe szülőjének a tektonikus lemezt.
- **getTiles()**: Visszaadja a tektonikus lemezhez tartozó csempék listáját. Láthatósága: **public**, típusa: List<Tile>

- **getMap()**: Visszaadja a tektonikus lemezhez tartozó térképet. Láthatósága: **public**, típusa: Map
- **getPlayerSpores(FungusPlayer player)**: Visszaadja a játékoshoz tartozó spórák számát. Ha a játékos nincs nyilvántartva, 0-át ad vissza. Láthatósága: **public**, típusa: int
- **addPlayerSpore(FungusPlayer player)**: Növeli a játékoshoz tartozó spórák számát 1-el, ha már nyilvántartásra került, vagy hozzáadja a játékost a nyilvántartáshoz 1 spóra értékkel. Láthatósága: **public**
- **hasFungusBody()**: Ellenőrzi, hogy a tektonikus lemezen van-e gomba test. Láthatósága: **public**, típusa: boolean
- **breakTekton()**: A tektonikus lemezt kettévágja egy törésvonal mentén, és két új tektonikus lemezt hoz létre. A törésvonal meghatározása véletlenszerűen történik a lemez közepére, figyelembe véve a szélességet és a magasságot. A metódus visszaadja a két új tektonikus lemezt egy listában. Láthatósága: **public**, típusa: ArrayList<Tekton>
- **increaseChance(int amount)**: Növeli a tektonikus lemez törésének esélyét az amount értékkel. Láthatósága: **public**
- **getLeftmostTile()**: Visszaadja a legbaloldalibb csempét a tektonikus lemezen. Láthatósága: **private**, típusa: Tile
- **getRightmostTile()**: Visszaadja a legjobboldalibb csempét a tektonikus lemezen. Láthatósága: **private**, típusa: Tile
- **getTopmostTile()**: Visszaadja a legfelső csempét a tektonikus lemezen. Láthatósága: **private**, típusa: Tile
- **getBottommostTile()**: Visszaadja a legalacsonyabb csempét a tektonikus lemezen. Láthatósága: **private**, típusa: Tile
- **findWidth()**: A tektonikus lemez szélességét számítja ki, amely a baloldali és jobboldali csempék közötti távolság. Láthatósága: **private**, típusa: int
- **findHeight()**: A tektonikus lemez magasságát számítja ki, amely a felső és alsó csempék közötti távolság. Láthatósága: **private**, típusa: int
- **faultLine()**: Kiszámítja a törésvonalat a tektonikus lemezen, amely a lemez közepére esik, és egy véletlenszerű eltolást alkalmaz. A számított törésvonal alapján dönti el, hogy a tektonikus lemez vízszintesen vagy függőlegesen törik el. Láthatósága: **private**, típusa: int

8.1.61 TektonBreaking

Felelősség:

A **TektonBreaking** osztály felelőssége a tektonikus lemez (Tekton) törésének szimulálása a térképen (Map). Az osztályban az eljárás kezdetekor inicializálásra kerül a térkép és a tektonikus lemez, majd szimulálja, hogy egy térkép frissítése következtében a tektonikus lemez eltörik. Az osztály a Tekton osztály breakTekton() metódusának hívásával kezdeményezi a törést, és ennek eredményét (új tektonikus lemezeket) egy listában tárolja.

Ősosztályok:

- **UseCase** (Legősebb osztály): A TektonBreaking osztály örökli a UseCase osztályt, amely valószínűleg más műveletekhez hasonlóan általános használati esetekkel dolgozik, például különböző térképinterakciók szimulálása.

Interfészek:

- Nincs közvetlenül megvalósított interfész.

Attribútumok:

- **logger**: A naplózásért felelős eszköz, láthatósága: **protected**, típusa: Logger
- **map**: A térkép, amelyhez a tektonikus lemez tartozik, láthatósága: **private**, típusa: Map
- **tekton**: A térképen található tektonikus lemez, amelynek törését szimuláljuk, láthatósága: **private**, típusa: Tekton

Metódusok:

- **TektonBreaking()**: Konstruktor, amely inicializálja a szimulációval kapcsolatos alapvető információkat, mint például az azonosítót (15) és a nevet ("Tekton breaking") a szülőosztályban (UseCase).
- **execute()**: A metódus végrehajtja a tektonikus lemez törésének szimulációját. Az alábbi lépéseket tartalmazza:
    1. A szimuláció kezdete után a térkép és a tektonikus lemez inicializálása történik.
    2. A térképen hozzáadódik a tektonikus lemez.
    3. A tektonikus lemez törése egy frissítés következtében történik meg, és az új tektonikus lemezeket egy ArrayList<Tekton> típusú változóban tárolja.

  Láthatósága: public

Algoritmus:

- A metódus elsőként a printWrapper segítségével üzenetet jelenít meg az "Initializing scene..." szöveggel.
- A térkép (Map) objektumot és a tektonikus lemez (Tekton) objektumot hoz létre, majd hozzáadja a térképhez.
- A törés szimulációja után a breakTekton() metódus segítségével a tektonikus lemez két új tektonikus lemezként hasítódik és tárolódik egy listában.

8.1.62 TektonBreakingMycelium

Felelősség:

A **TektonBreakingMycelium** osztály felelőssége a tektonikus lemez törésének szimulálása, miközben figyelembe veszi a gombafonal (mycelium) növekedését és annak eltávolítását, ha a tektonikus lemez eltörik. A szimulációban a gombafonal a tektonikus lemezekhez tartozó csempéken (tile) nő, és a tekton törésének következményeként a hozzá kapcsolódó mycelium elpusztul.

Ősosztályok:

- **UseCase** (Legősebb osztály): A TektonBreakingMycelium osztály örökli a UseCase osztályt, amely különböző használati esetek (use case) kezelésére szolgál, például térképi interakciók szimulálására.

Interfészek:

- Nincs közvetlenül megvalósított interfész.

Attribútumok:

- **logger**: A naplózáshoz használt eszköz, láthatósága: **protected**, típusa: Logger
- **map**: A térkép objektum, amely a szimuláció alapját képezi, láthatósága: **private**, típusa: Map
- **tekton**: A térképen található tektonikus lemez, amely a törés szimulálásában szerepel, láthatósága: **private**, típusa: Tekton

- **fungusPlayer**: A gombás játékos, aki a gombafonalakat növeszti a tektonikus lemez csempéin, láthatósága: **private**, típusa: FungusPlayer

Metódusok:

- **TektonBreakingMycelium()**: Konstruktor, amely inicializálja a szimulációhoz szükséges alapvető adatokat, mint például az azonosítót (19) és a nevet ("Tekton breaking with mycelium") a szülőosztályban (UseCase).
- **execute()**: A metódus végrehajtja a tektonikus lemez törésének szimulációját a gombafonalakkal együtt. Az alábbi lépéseket hajtja végre:
    1. A térkép és a tektonikus lemez inicializálása, valamint hozzáadása a térképhez.
    2. Két csempe (Tile) hozzáadása a tektonikus lemezhez.
    3. A gombás játékos létrehozása és a gombafonal növesztése az első csempén (t1), majd a második csempén (t2).
    4. A tektonikus lemez törése, amely új tektonikus lemezeket generál.
    5. A gombafonalak eltávolítása a törés után a kapcsolódó szabályoknak megfelelően, azaz a tektonikus lemezekhez tartozó csempéken található myceliumok elpusztítása.

  Láthatósága: public

Algoritmus:

- A metódus elsőként üzenetet jelenít meg a szimuláció kezdetéről a printWrapper segítségével.
- Létrehoz egy térképet és tektonikus lemezt, hozzáadva azokat a térképhez.
- Két csempét (tile) hoz létre és kapcsolja őket a tektonikus lemezhez.
- Egy FungusPlayer objektumot is létrehoz, amely gombafonalat növeszt az első csempére, majd a második csempére is.
- A tektonikus lemez törése a breakTekton() metódus segítségével történik, és az új tektonikus lemezeket egy listában tárolja.
- A szimuláció végén eltávolítja a gombafonalakat a tektonikus lemezekhez tartozó csempékről.

8.1.63 TektonBreaks

Felelősség:

A **TektonBreaks** osztály felelőssége a tektonikus lemez törésének végrehajtása és a hozzá tartozó gombafonalak eltávolítása. A parancs végrehajtása után a tektonikus lemez eltűnik a térképről, és új, törött tektonikus lemezek jelennek meg a térképen. Emellett minden hozzá kapcsolódó gombafonalat (mycelium) eltávolít, amint a tektonikus lemez törik.

Ősosztály:

- **Command**: A TektonBreaks osztály örökli a Command osztályt, amely a játékban végrehajtható parancsok alapvető funkcionalitását biztosítja.

Attribútumok:

- **logger**: A naplózáshoz használt eszköz, láthatósága: **protected**, típusa: Logger
- **args**: A parancs végrehajtásához szükséges argumentumok, láthatósága: **private**, típusa: String[]
- **app**: A játék objektum, amely elérhetővé teszi a térképet és a kapcsolódó funkciókat, láthatósága: **private**, típusa: App

Metódusok

- **TektonBreaks()**: Konstruktor, amely inicializálja a parancs nevét, leírását és a szintaxisát:
- **execute(String[] args)**: Ez a metódus végrehajtja a tektonikus lemez törését és a hozzá tartozó mycelium eltávolítását. A következő lépéseket hajtja végre:
    1. Ellenőrzi, hogy az argumentumok helyesek-e.
    2. Ellenőrzi, hogy a térkép inicializálva van-e.
    3. Megkeresi a tektonikus lemezt az args alapján.
    4. A tektonikus lemez törésével új tektonikus lemezeket hoz létre.
    5. Keres egy listát a myceliumokból, amelyek a tektonikus lemezhez tartozó csempéken vannak.
    6. A myceliumokat eltávolítja a játékból.
    7. Eltávolítja a tektonikus lemezt a térképről, és hozzáadja az új, törött tektonikus lemezeket.

Láthatóság: public

Algoritmus:

- Az első lépés ellenőrzi, hogy a parancs helyes számú argumentumot kapott-e, és ha nem, visszatér **false**-szal.
- Ezután ellenőrzi, hogy a térkép inicializálva van-e, és ha nem, visszatér **false**-szal.
- A **parseTekton** metódust használja a tektonikus lemez megtalálásához a térképen, az argumentumban szereplő azonosítóval.
- Ha a tektonikus lemez nem található, a metódus visszatér **false**-szal.
- Ha a tektonikus lemez megtalálható, akkor meghívja a breakTekton() metódust, amely létrehozza az új tektonikus lemezeket.
- A metódus ezután keres egy listát a mycelium entitásokból a tektonikus lemezhez tartozó csempéken, és eltávolítja őket.
- A tektonikus lemez eltávolításra kerül a térképről, és az új tektonikus lemezek hozzáadásra kerülnek.

8.1.64 TektonCantGrowFungus

Felelősség

A **TektonCantGrowFungus** osztály felelőssége, hogy blokkolja a gombatestek növekedését egy adott tektonikus lemezen. Ez a parancs azt biztosítja, hogy semmilyen gombatest ne növekedhessen a tektonikus lemezen, amikor a felhasználó aktiválja a megfelelő parancsot. A parancs alapvetően a jövőbeni funkcionalitásra van előkészítve, de jelenleg nem implementált.

Ősosztályok

- **Command**
    - A **TektonCantGrowFungus** osztály a **Command** osztályból származik, amely az összes parancs közös ősosztálya. A **Command** osztály felelős a parancsok végrehajtásáért, és az általános felületet biztosít az összes parancs számára.

Interfészek

- **Nincs interfész**: A **TektonCantGrowFungus** osztály nem implementál semmilyen interfészt.

Attribútumok

- **logger**: A **logger** attribútum a naplózást végzi a parancsokhoz kapcsolódóan. Láthatóság: **protected**, Típus: Logger.
- **args**: Az argumentumok, amelyeket a parancs végrehajtásához használnak. Láthatóság: **private**, Típus: String[].
- **app**: A játék objektum, amely elérhetővé teszi a térképet és a kapcsolódó funkciókat. Láthatóság: **private**, Típus: App.

Metódusok

- **execute(String[] args)**:
  - **Leírás**: A **execute** metódus végrehajtja a parancsot. Jelenleg nem implementált, és UnsupportedOperationException kivételt dob. Az implementáció célja, hogy blokkolja a gombatestek növekedését egy adott tektonikus lemezen.
  - **Láthatóság**: **public**
  - **Paraméterek**:
    - String[] args: Az argumentumok, amelyek tartalmazzák a parancs végrehajtásához szükséges információkat, például a tektonikus lemez azonosítóját.
  - **Visszatérési érték**:
    - A metódus **void**, de ha nem implementálják, kivételt dob.
  - **Aktuális implementáció**:
    - A metódus jelenleg kivételt dob:

8.1.65 TektonMultipleMycelium

Felelősség

A **TektonMultipleMycelium** osztály felelőssége, hogy lehetővé tegye több mycelium növekedését egy adott tektonikus lemezen. Ez a parancs elméletileg lehetővé tenné a myceliumok szaporodását egy tektonikus lemezen, de a funkció jelenleg nincs implementálva. Az osztály későbbi implementációja ezt a funkciót fogja ellátni.

Ősosztályok

- **Command**
  - A **TektonMultipleMycelium** osztály a **Command** osztályból származik, amely az összes parancs közös ősosztálya. A **Command** osztály felelős a parancsok végrehajtásáért, és biztosítja az általános felületet az összes parancs számára.

Interfészek

- **Nincs interfész**: A **TektonMultipleMycelium** osztály nem implementál semmilyen interfészt.

Attribútumok

- **logger**: A **logger** attribútum a naplózást végzi a parancsokhoz kapcsolódóan. Láthatóság: **protected**, Típus: Logger.
- **args**: Az argumentumok, amelyeket a parancs végrehajtásához használnak. Láthatóság: **private**, Típus: String[].
- **app**: A játék objektum, amely elérhetővé teszi a térképet és a kapcsolódó funkciókat. Láthatóság: **private**, Típus: App.

Metódusok

- **execute(String[] args)**:
    - **Leírás**: A **execute** metódus a parancs végrehajtásáért felelős. A parancs jelenlegi állapotában nem implementált, és a metódus **UnsupportedOperationException** kivételt dob. Az elméleti implementáció célja, hogy több myceliumot engedjen növekedni egy adott tektonikus lemezen.
    - **Láthatóság**: **public**
    - **Paraméterek**:
        - String[] args: Az argumentumok, amelyek tartalmazzák a parancs végrehajtásához szükséges információkat, például a tektonikus lemez azonosítóját.
    - **Visszatérési érték**:
        - A metódus **void**, de ha nem implementált, kivételt dob.

8.1.66 TektonOneMycelium

Felelősség

A **TektonOneMycelium** osztály felelőssége, hogy biztosítsa, hogy csak egyetlen mycelium növekedhessen egy adott tektonikus lemezen. Ez a parancs lehetővé tenné, hogy egy tektonikus lemezen csak egy mycelium található legyen, de jelenleg a funkció nincs implementálva. A későbbiekben a parancs implementálása fogja kezelni ennek a logikának a megvalósítását.

Ősosztályok

- **Command**
    - A **TektonOneMycelium** osztály a **Command** osztályból származik. A **Command** osztály biztosítja az összes parancs közös felületét és végrehajtási logikáját.

Interfészek

- **Nincs interfész**: A **TektonOneMycelium** osztály nem valósít meg semmilyen interfészt.

Attribútumok

- **logger**: A **logger** attribútum felelős a naplózási feladatok elvégzéséért. Láthatóság: **protected**, Típus: Logger.
- **args**: Az argumentumokat tartalmazó tömb, amelyet a parancs végrehajtása során használnak. Láthatóság: **private**, Típus: String[].
- **app**: Az alkalmazás objektum, amely lehetővé teszi a térképhez való hozzáférést és más fontos funkciók elérését. Láthatóság: **private**, Típus: App.

Metódusok

- **execute(String[] args)**:
    - **Leírás**: A **execute** metódus a parancs végrehajtásáért felelős. Jelenleg a metódus nem implementált, és egy **UnsupportedOperationException** kivételt dob. A funkció végrehajtása, amely biztosítja, hogy csak egy mycelium növekedhessen, a jövőbeli implementációval válik elérhetővé.
    - **Láthatóság**: **public**
    - **Paraméterek**:
        - String[] args: A parancs végrehajtásához szükséges argumentumok. Tartalmazza a tektonikus lemez azonosítóját.

- **Visszatérési érték**:
  - A metódus **void** típusú, de jelenleg kivételt dob.

8.1.67 Tick

Felelősség

A **Tick** osztály felelőssége, hogy végrehajtsa a kívánt számú kört (round) a térképen. Minden egyes kör egy új "tick" műveletet jelent, amely frissíti a térkép állapotát. Ez az osztály lehetővé teszi, hogy a játék előrehaladjon és a különböző entitások működjenek a megadott számú kört követően.

Ősosztályok

- **Command**
  - A **Tick** osztály a **Command** osztályból származik. A **Command** osztály biztosítja a közös felületet az összes parancs számára, lehetővé téve azok végrehajtását.

Interfészek

- **Nincs interfész**: A **Tick** osztály nem valósít meg semmilyen interfészt.

Attribútumok

- **logger**: A **logger** attribútum biztosítja a naplózási funkciókat, és segíti a parancsok végrehajtásának követését. Láthatóság: **protected**, Típus: Logger.
- **args**: A **args** attribútum tartalmazza a parancs végrehajtásához szükséges argumentumokat, például a kört. Láthatóság: **private**, Típus: String[].
- **app**: Az **app** objektum biztosítja a hozzáférést az alkalmazás térképhez és más szükséges funkciókhoz. Láthatóság: **private**, Típus: App.

Metódusok

- **execute(String[] args)**:
  - **Leírás**: A **execute** metódus felelős a parancs végrehajtásáért. Ellenőrzi az argumentumok számát, a térkép inicializáltságát, és ha minden rendben van, végrehajtja a megadott számú kört a térképen.
  - **Láthatóság**: **public**
  - **Paraméterek**:
    - String[] args: A parancs végrehajtásához szükséges argumentumok. Tartalmazza a körök számát.
  - **Visszatérési érték**:
    - **boolean**: A metódus mindig **false** értékkel tér vissza.
  -

8.1.68 Tile Osztály

Felelősség

A **Tile** osztály egy térképen található cellát reprezentál. Minden egyes **Tile** tartalmazza a koordinátáit, a mycelium növekedését, a hozzá rendelt entitásokat, valamint azt, hogy milyen kapcsolódó **Tekton** lemezhez tartozik. Ezen kívül kezeli a környezetében lévő más **Tile** cellákkal való kapcsolatokat, például a szomszédos cellákat, illetve a hidakat, amelyek más tektonikus lemezekre vezetnek.

Ősosztályok

- **Nincs ősosztály**: A **Tile** osztály nem származik más osztályból.

Interfészek

- **Nincs interfész**: A **Tile** osztály nem valósít meg semmilyen interfészt.

Attribútumok

- **x** és **y**: A **Tile** koordinátái a térképen.
- **growthRate**: A **Tile** növekedési üteme.
- **myceliumSpace**: A **Tile**-on lévő mycelium objektumok maximális száma.
- **parentTekton**: A **Tile**-hoz tartozó **Tekton** lemez, amely meghatározza a földrajzi elhelyezkedést.
- **entities**: A **Tile**-on lévő összes entitás (pl. mycelium, gombatest).
- **bridges**: A más **Tile**-okra vezető hidak listája.

Metódusok

- **addEntity(GameEntity entity)**: Az entitás hozzáadása a **Tile**-hoz és a **Tile** beállítása az entitás számára.
- **getEntities()**: Visszaadja a **Tile**-on található entitások listáját.
- **removeEntity(GameEntity entity)**: Eltávolít egy entitást a **Tile**-ról és naplózza az eseményt.
- **getParentTekton()**: Visszaadja a **Tile**-hoz tartozó **Tekton** lemezt.
- **getGrowthRate()**: Visszaadja a **Tile** növekedési ütemét.
- **getMaxMycelium()**: Visszaadja, hány mycelium objektumot képes a **Tile** tárolni.
- **setParentTekton(Tekton parentTekton)**: Beállítja a **Tile**-hoz tartozó **Tekton** lemezt.
- **isNeighbor(Tile tile)**: Ellenőrzi, hogy a megadott **Tile** szomszédos-e az aktuális **Tile**-lal (vizsgálja a vízszintes, függőleges és átlós szomszédokat).
- **getNeighbors()**: Visszaadja az összes szomszédos **Tile**-t.
- **growMycelium(FungusPlayer player)**: Mycelium növesztése a **Tile**-on, ha van hely a növekedésre.
- **growBody(FungusPlayer player)**: Gombatest növesztése a **Tile**-on.
- **update()**: Frissíti az összes entitást a **Tile**-on.
- **getX()** és **getY()**: Visszaadják a **Tile** koordinátáit.
- **hasBridge(Tile tile)**: Ellenőrzi, hogy van-e híd a megadott **Tile**-hoz.

8.1.69 UseCase

Felelősség

A **UseCase** osztály a rendszerben végrehajtandó műveletek (use case-ek) alapját képezi. Ez az absztrakt osztály biztosítja az alapvető működést az összes specifikus művelethez, mint az azonosító kezelés, üzenetformázás és naplózás. Az osztály figyelmet fordít a bemeneti paraméterekkel való interakcióra, valamint a bejövő üzenetek kezelésére az indentálás (behúzás) és irány szerinti kijelzés révén.

Ősosztályok

- **Nincs ősosztály**: Az osztály közvetlenül nem származik más osztályból.

Interfészek

- **Nincs interfész**: A **UseCase** osztály nem valósít meg semmilyen interfészt.

Attribútumok

- **id**: Az azonosító, amely egyedi módon azonosítja a műveletet.

- **name**: A művelet neve.
- **persistentIndentDepth**: A végrehajtott művelet behúzásának mélysége, amely az üzenetek megjelenítéséhez szükséges.
- **persistentDir**: Az irány, amely meghatározza, hogy a nyíl milyen irányba mutatjon az üzenet megjelenítésekor (balra vagy jobbra).
- **logger**: Egy statikus **HashMap**, amely a különböző objektumokhoz rendelt naplózott üzeneteket tárolja.
- **Enumok**
- **Indent**: Enum típus, amely három értéket tartalmaz:
  - **UNINDENT**: Csökkenti az indentálás mélységét.
  - **INDENT**: Növeli az indentálás mélységét.
  - **KEEP**: Megőrzi a jelenlegi indentálás mélységét.
- **ArrowDirection**: Enum típus, amely meghatározza az üzenet irányát:
  - **LEFT**: A nyíl balra mutat.
  - **RIGHT**: A nyíl jobbra mutat.
- **Konstruktor**
- **UseCase(int id, String name)**: Az osztály konstruktora, amely beállítja az **id** és **name** attribútumokat. A konstruktor törli a **logger**-ben tárolt adatokat.

Metódusok

- **replace(Object o)**: Áthelyezi az objektum naplózott üzenetét a **logger** térképbe, az előző objektum bejegyzését eltávolítva.
- **getID()**: Visszaadja a művelet azonosítóját.
- **getName()**: Visszaadja a művelet nevét.
- **printWrapper(String message, ArrowDirection dir, int indentDepth)**: Kiírja az üzenetet a megfelelő nyíl irányával és a behúzás mélységével. A behúzást a **\t** karakterek segítségével valósítja meg.
- **printWrapper(String message, ArrowDirection dir)**: Ugyanaz, mint az előző, de itt a behúzási mélységet a legutolsó beállítás alapján alkalmazza.
- **printWrapper(String message)**: Az előző változatokkal azonos működésű, de itt a nyíl irányát és a behúzást az utolsó használt értékek szerint alkalmazza.
- **printWrapper(String message, ArrowDirection dir, Indent indentDir)**: A metódus kezeli a behúzást és az üzenet irányát, az indentálás növelését vagy csökkentését. Az **indentDir** paraméter határozza meg, hogy az indentálás növekedjen, csökkenjen, vagy változatlan maradjon a kiírás után.
- **execute()**: Ez egy absztrakt metódus, amelyet minden konkrét **UseCase** osztálynak implementálnia kell, hogy végrehajtsa a műveletet.

8.1.70 UseCaseList

Felelősség

A **UseCaseList** osztály egy gyűjteményt tartalmaz, amely a különböző **UseCase** objektumokat tárolja. Az osztály biztosítja a **UseCase** példányok rendezését és a duplikált azonosítók, illetve duplikált nevekkel kapcsolatos hibák ellenőrzését.

Ősosztályok

- **Nincs ősosztály**: Az osztály közvetlenül nem származik más osztályból.

Interfészek

- **Nincs interfész**: A **UseCaseList** osztály nem valósít meg semmilyen interfészt.

Attribútumok

- **useCases**: Egy **ArrayList** típusú attribútum, amely tartalmazza az összes **UseCase** objektumot. A műveletek tárolása és kezelése ezen az adatstruktúrán keresztül történik.
- **Konstruktor**
- **UseCaseList()**: A konstruktor feltölti a **useCases** listát az előre meghatározott **UseCase** objektumokkal. Minden **UseCase** osztály egy külön műveletet reprezentál, amely a listában szerepel. Az egyes műveletek azonosítói és nevei különbözőek, és mindegyik művelethez van egy egyedi azonosító.

Metódusok

- **getUseCases()**: Visszaadja a **useCases** listát, amely tartalmazza az összes **UseCase** objektumot. Ez a metódus lehetővé teszi a műveletek listájának elérését.
- **sortUseCases()**: Rendezési funkció, amely először ellenőrzi az **UseCase** listában található elemeket a következő szempontok alapján:
    - **Duplikált azonosítók**: Ha bármely két **UseCase** objektum azonos azonosítóval rendelkezik, az **IllegalArgumentException** kivételt dob.
    - **Duplikált nevek**: A műveletek nevei között figyeli, hogy van-e olyan, amelyik név egyezik (nem érzékeny a kis- és nagybetűkre). Ha igen, szintén kivételt dob.
    - **Rendezés ID alapján**: A műveleteket az **ID** szerint rendezi növekvő sorrendbe.

8.2 A tesztek részletes tervei, leírásuk a teszt nyelvén

8.2.1 Teszteset1

- **Leírás**

Az insect átlép egyik Tile-ról a másikra.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a rovar mozgását két szomszédos tekton között és a mycelium által biztosított összeköttetést. Ellenőrzi, hogy a rovar helyes mezőn léphet.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Insect 0 0
insect_step 0 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t1”,
„speed”: 2,
„canCut”: true,
}

8.2.2 Teszteset2

- **Leírás**

Az insect átlép elvágja a Mycelium-ot.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a rovar képes-e elvágni a fonalat, megszüntetve az összeköttetést. Ellenőrzi, hogy a mycelium állapota helyesen frissül.

- **Bemenet**

create Map

```
create Tekton 1 1
create Tile 0 1 2
create Insect 0 0
create Mycelium 0 0
insect_cut 0 1
save out.txt
```

- **Elvárt kimenet**

```
„T0”: {
        „Típus”: „normal”,
        „Tiles”: [t0],
        „breakChance”: 1,
        „maxSporeCount”: 1
        },
„t0”: {
        „parentTekton”: T0,
        „growthRate”: 1
        „maxMycelium”: 2
        },
„insect_0”: {
        „currentTile”: „t0”,
        „speed”: 2,
        „canCut”: true,
   },
„mycelium_1”: {
        „currentTile”: t0,
        „health”: 100,
        „isDying”: true
}
```

8.2.3 Teszteset3

- **Leírás**

Az insect megeszik egy gyorsító spórát és flegyorsul.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a gyorsító hatás érvényesülését a rovar mozgásában. Ellenőrzi, hogy a rovar képes-e több mezőt megtenni egy lépésben.

- **Bemenet**

```
create Map
create Tekton 1 1
create Tile 0 1 2
create Insect 0 0
create SpeedUpSpore 0 0
insect_eat_spore 0 1
save out.txt
```

- **Elvárt kimenet**

```
„T0”: {
        „Típus”: „normal”,
        „Tiles”: [t0],
        „breakChance”: 1,
```

„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 4,
„canCut”: true,
},
„speedupspore_1”: {
„currentTile”: t0,
„isConsumed”: true
}

8.2.4 Teszteset4

- **Leírás**

Az insect megeszik egy lassító spórát és lelassul.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a lassító hatás érvényesülését a rovar mozgásában. Ellenőrzi, hogy a rovar kevesebb mezőt tud megtenni egy lépésben.

- **Bemenet**

```
create Map
create Tekton 1 1
create Tile 0 1 2
create Insect 0 0
create SlowSpore 0 0
insect_eat_spore 0 1
save out.txt
```

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 1,
„canCut”: true,
},
„slowspore_1”: {

„currentTile”: t0,
„isConsumed”: true
}

8.2.5 Teszteset5

- **Leírás**

Az insect megeszik egy freezespore-t és megbénul.

- **Ellenőrzött funkcionalitás, várható hibahelyek**
- Teszteli, hogy a bénító hatás helyesen leállítja a rovar mozgását. Ellenőrzi, hogy a rovar nem tud lépni, amíg a hatás tart.
- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Insect 0 0
create FreezeSpore 0 0
insect_eat_spore 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 0,
„canCut”: true,
},
„freezespore_1”: {
„currentTile”: t0,
„isConsumed”: true
}

8.2.6 Teszteset6

- **Leírás**

Az insect megeszik egy CutSpore-t és emiatt nem tud vágni.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a rovar nem tud fonalat vágni, ha nincs elegendő feltétel (pl. nincs gombatest). Ellenőrzi a fonál vágásának hiányát.

- **Bemenet**

create Map

create Tekton 1 1
create Tile 0 1 2
create Insect 0 0
create CutSpore 0 0
insect_eat_spore 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„sporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 4,
„canCut”: false,
},
„cutspore_1”: {
„currentTile”: t0,
„isConsumed”: true
}

8.2.7 Teszteset7

- **Leírás**

Az insect megeszik egy FreezeSpore-t utána újra mozogni kezd.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a bénító hatás lejárta után a rovar képes újra lépni. Ellenőrzi, hogy a rovar mozgása helyreáll-e.

- **Bemenet**

create Map
create Tekton 1 1
Create Tile 0 1 2
create Insect 0 0
create FreezeSpore 0 0
insect_eat_spore 0 1
insect_unfreeze 0 0 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,

„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 4,
„canCut”: true,
}

8.2.8 Teszteset8

- **Leírás**

A gombafonal növekszik.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a mycelium növekedését, ha egy gombatestből új fonál növekszik. Ellenőrzi, hogy a növekedés megfelelő irányban történik-e.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create FungusBody 0 0 1 1
create Mycelium 0 0
mycelium_grow 1 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1

},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.9 Teszteset9

- **Leírás**

A gombafonal olyan tektonon növekszik, ahol spóra van.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a spórák hatását a mycelium növekedésére. Ellenőrzi, hogy a spórák gyorsítják-e a növekedést és több mezőt elérhet-e egyszerre a fonál.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create FungusBody 0 0 1 1
create Mycelium 0 0
create SpeedUpSpore 0 1
mycelium_grow 1 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1

},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.10 Teszteset10

- **Leírás**

Gombától elszakított fonál elhal.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a fonál elhal-e, ha már nem kapcsolódik gombatesthez. Ellenőrzi, hogy a fonál élete csökken-e és végül elhal-e.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Mycelium 0 0
mycelium_die 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
}

8.2.11 Teszteset11

- **Leírás**

Egy gombatest nő.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a gombatest növekedését. Ellenőrzi, hogy a megfelelő feltételek megléte esetén a gombatest új mezőn nőhet.

- **Bemenet**

create Map
create Tekton 1 1

create Tile 0 1 2
fungus_body_grow 0 0 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
}

8.2.12 Teszteset12

- **Leírás**

Egy gombatest spórát szór.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a gombatest képes-e spórákat szórni, és ezek a szomszédos tektonon megjelennek-e. Ellenőrzi a spórák helyes terjedését.

- **Bemenet**

create Map
create Tekton 1 2
create Tile 0 1 2
create Tile 0 1 2
create FungusBody 0 0 1 1
fungus_body_release_spore_cloud 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 2
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {

„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 0
},
„mycelium_1”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.13 Teszteset13

- **Leírás**

A gombatest elpusztul.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a gombatest halálát. Ellenőrzi, hogy a gombatest élete nullára csökken-e és végül elhal-e.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create FungusBody 0 0 1 1
fungus_body_die 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},

8.2.14 Teszteset14

- **Leírás**

Egy tekton kettétörik és a törés elvágja a fonalat.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli a tekton törését és annak hatását a fonálra. Ellenőrzi, hogy a fonál elhal-e, miután megszűnt az összeköttetés.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create FungusBody 0 0 1 1
create Mycelium 0 0
create Mycelium 0 1
tekton_breaks 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„T1”: {
„Típus”: „normal”,
„Tiles”: [t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T1,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: true
}

8.2.15 Teszteset15

- **Leírás**

A tektonon nem tud gombatest nőni.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a gombatest nem nőhet, ha a tekton nem alkalmas a növekedéshez. Ellenőrzi, hogy a megfelelő feltételek hiányában nem nő gombatest.

- **Bemenet**

create Map
create Tekton 1 1
tekton_cant_grow_fungus 0
create Tile 0 1 2
fungus_body_grow 0 0 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „cantgrow”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
}

8.2.16 Teszteset16

- **Leírás**

A tektonon csak egyféle fonál nőhet.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a tekton nem engedi meg kétféle fonál növekedését. Ellenőrzi, hogy a tektonon csak egyféle mycelium nőhet.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tekton 1 1
create Tile 1 1 2
create Tekton 1 1
tekton_one_mycelium 2
create MonoTile 2 1 1
create FungusBody 0 0 1 1
create Mycelium 0 0
create FungusBody 0 1 1 1
create Mycelium 0 1
mycelium_grow 1 2 2
mycelium_grow 3 2 2
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„T1”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„T2”: {
„Típus”: „mono”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t0”: {
„parentTekton”: T1,
„growthRate”: 1
„maxMycelium”: 2
},
„t0”: {
„parentTekton”: T2
„growthRate”: 1
„maxMycelium”: 1
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}
„fungusbody_2”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_3”: {
„currentTile”: t0,

„health”: 100,
„isDying”: false
},
„mycelium_4”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}

8.2.17 Teszteset17

- **Leírás**

A tektonon többféle fonál is nő.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a tektonon többféle mycelium növekedhet-e. Ellenőrzi, hogy a szomszédos tektonok képesek-e többféle fonál növesztésére ugyanazon tektonra.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Tile 0 1 2
create FungusBody 0 0 1 1
create Mycelium 0 0
create FungusBody 0 2 1 1
create Mycelium 0 2
mycelium_grow 1 0 1
mycelium_grow 3 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1,t2],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t2”: {
„parentTekton”: T0
„growthRate”: 1
„maxMycelium”: 2

},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}
„fungusbody_2”: {
„currentTile”: t2,
„health”: 1,
„sporecharge”: 1
},
„mycelium_3”: {
„currentTile”: t2,
„health”: 100,
„isDying”: false
},
„mycelium_4”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}
„mycelium_5”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.18 Teszteset18

- **Leírás**

A tektonon annak típusa miatt elhal a fonál.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy egy savas tekton képes-e elpusztítani a myceliumot. Ellenőrzi, hogy a fonál elhal-e, ha acid tektonon található.

- **Bemenet**

```
create Map
create Tekton 1 1
create Tekton 1 1
create Tile 0 1 2
create AcidTile 1 1 2
create FungusBody 0 0 1 1
create Mycelium 0 0
create Mycelium 1 1
tick 1
save out.txt
```

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„T1”: {
„Típus”: „acid”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t0”: {
„parentTekton”: T1,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}

8.2.19 Teszteset19

- **Leírás**

A rovar egy osztódó spórát eszik meg.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, az osztódást és az osztódott rovarok irányíthatóságát.

- **Bemenet**

```
create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Insect 0 0
create SplitSpore 0 0
intsect_eat_spore 0 1
save out.txt
```

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T1,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 4,
„canCut”: true,
}
„insect_1”: {
„currentTile”: „t1”,
„speed”: 4,
„canCut”: true,
}

8.2.20 Teszteset20

- **Leírás**

Egy gyógyító típusú tile nem hagyja a gombatesttől elvágott fonalat meghalni.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, a gyógyító tile életben tartja-e a gombatesttől elválasztott myceliumot.

- **Bemenet**

create Map
create Tekton 1 1
create HealTile 0 1 2
create Mycelium 0 0
tick 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„sporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1

„maxMycelium”: 2
},
„mycelium_0”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.21 Teszteset21

- **Leírás**

Gomba megeszik egy bénult rovart és a helyén egy gombatestet növeszt.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, a húsevő gomba működését.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Tile 0 1 2
create FungusdBody 0 0 1 1
create Mycelium 0 0
create Mycelium 0 1
create Mycelium 0 2
create Insect 0 2
insect_freeze 4
consume_insect 3 4
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1,t2],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t2”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {

„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
},
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
},
„mycelium_3”: {
„currentTile”: t2,
„health”: 100,
„isDying”: false
},
„fungusbody_4”: {
„currentTile”: t2,
„health”: 1,
„sporecharge”: 1
}

8.2.22 Teszteset22

- **Leírás**

A rovar megeszik egy spórát.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

Teszteli, hogy a rovar képes elfogyasztani a spórát és annak hatása megfelelően érvényesül (pl. gyorsítás, lassítás). Ellenőrzi a pontszám növekedését.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Insect 0 0
create SlowSpore 0 0
insect_eat_spore 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1

„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 1,
„canCut”: true,
},
„slowspore_1”: {
„currentTile”: t0,
„isConsumed”: true
}

8.2.23 Teszteset23

- **Leírás**

Létrehoz több különböző állapotot melyet utána egy mentési file-ba kiment.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A mentés működését elkülönítve is vizsgáljuk a hibakeresés megkönnyítésének érdekében.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create FungusdBody 0 0 1 1
create Mycelium 0 0
create SpeedUpSpore 0 1
mycelium_grow 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {

„currentTile”: t0,
„health”: 100,
„isDying”: false
}
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.24 Teszteset24

- **Leírás**

Több előre létrehozott mentési file-t is betölt és ellenőrzi a helyességét.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A betöltés működését elkülönítve is vizsgáljuk a hibakeresés megkönnyítésének érdekében.

- **Bemenet**

load 1_InsectStepsAcrossMycelium.txt
save out.txt
load 2_InsectCutsMycelium.txt
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t1”,
„speed”: 2,
„canCut”: true,
}

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {

„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 2,
„canCut”: true,
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: true
}

8.2.25 Teszteset25

- **Leírás**

A különböző objektumok létrehozását teszteli.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A létrehozást elkülönítve is vizsgáljuk a hibakeresés megkönnyítésének érdekében.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create FungusdBody 0 0 1 1
create Mycelium 0 0
create SpeedUpSpore 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„maxSporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_0”: {
„currentTile”: t0,
„health”: 1,
„sporecharge”: 1

},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
}

8.2.26 Teszteset26

- **Leírás**

A tekton létrehozását, törését és egyéb funkcióit vizsgálja.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A tekton részletes tesztelése, hogy olyan hibákat is megfogjunk melyek a szimpla use-case tesztekkel nem jönnek elő.

- **Bemenet**

create Map
create Tekton 1 1
tekton_breaks 0
tekton_breaks 1
tekton_cant_grow_fungus 0
tekton_one_mycelium 1
tekton_multiple_mycelium 2
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „cantgrow”,
„Tiles”: [],
„breakChance”: 1,
„maxSporeCount”: 1
},
„T1”: {
„Típus”: „mono”,
„Tiles”: [],
„breakChance”: 1,
„maxSporeCount”: 1
},
„T2”: {
„Típus”: „normal”,
„Tiles”: [],
„breakChance”: 1,
„maxSporeCount”: 1
}

8.2.27 Teszteset27

- **Leírás**

A tile létrehozását, tektonnal való összekötését és egyéb funkcióit vizsgálja.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A tile részletes tesztelése, hogy olyan hibákat is megfogjunk melyek a szimpla use-case tesztekkel nem jönnek elő.

- **Bemenet**

create Map
create Tekton 1 1
create Tekton 1 1
create Tile 0 1 2
set_tile_type 0 acid
sat_tile_parent_tekton 0 1
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „acid”,
„Tiles”: [],
„breakChance”: 1,
„sporeCount”: 1
},
„T1”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„sporeCount”: 1
},

„t0”: {
„parentTekton”: T1,
„growthRate”: 1
„maxMycelium”: 2
}

8.2.28 Teszteset28

- **Leírás**

A rovar létrehozását, evését és egyéb funkcióit vizsgálja.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A rovar részletes tesztelése, hogy olyan hibákat is megfogjunk melyek a szimpla use-case tesztekkel nem jönnek elő.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Insect 0 0
create Insect 0 1
insect_slow_down 0 100
insect_speed_up 1 100
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„sporeCount”: 1
},

„T1”: {
„Típus”: „normal”,
„Tiles”: [t0],
„breakChance”: 1,
„sporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t0”: {
„parentTekton”: T1,
„growthRate”: 1
„maxMycelium”: 2
},
„insect_0”: {
„currentTile”: „t0”,
„speed”: 1,
„canCut”: true,
},
„insect_1”: {
„currentTile”: „t0”,
„speed”: 3,
„canCut”: true,
},

8.2.29 Teszteset29

- **Leírás**

A gombász tulajdonságait, létrehozását használatát teszteli részletesen.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A gombász részletes tesztelése, hogy olyan hibákat is megfogjunk melyek a szimpla use-case tesztekkel nem jönnek elő.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Fungus 0 0
create Mycelium 0 0
create Mycelium 0 1
fungus_body_grow 0 0 0
fungus_body_die 0
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,

„sporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_3”: {
„currentTile”: t1,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
},
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

8.2.30 Teszteset30

- **Leírás**

A gombász tulajdonságait, létrehozását használatát teszteli részletesen.

- **Ellenőrzött funkcionalitás, várható hibahelyek**

A gombász részletes tesztelése, hogy olyan hibákat is megfogjunk melyek a szimpla use-case tesztekkel nem jönnek elő.

- **Bemenet**

create Map
create Tekton 1 1
create Tile 0 1 2
create Tile 0 1 2
create Fungus 0 0
create Mycelium 0 0
mycelium_grow 1 0 1
save out.txt
mycelium_die 2
save out.txt

- **Elvárt kimenet**

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],

„breakChance”: 1,
„sporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_3”: {
„currentTile”: t1,
„health”: 1,
„sporecharge”: 1
},
„mycelium_1”: {
„currentTile”: t0,
„health”: 100,
„isDying”: false
},
„mycelium_2”: {
„currentTile”: t1,
„health”: 100,
„isDying”: false
}

„T0”: {
„Típus”: „normal”,
„Tiles”: [t0,t1],
„breakChance”: 1,
„sporeCount”: 1
},
„t0”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„t1”: {
„parentTekton”: T0,
„growthRate”: 1
„maxMycelium”: 2
},
„fungusbody_3”: {
„currentTile”: t1,
„health”: 1,
„sporecharge”: 1
},

```
„mycelium_1”: {
        „currentTile”: t0,
        „health”: 100,
        „isDying”: false
}
```

8.3 A tesztelést támogató programok tervei

Tetszeléshez a CompilerForDummies.bat fájlt kell futtatni, ami fordítja és futtatja a programot. Ott a test parancsal ki lehet választani, hogy melyik vagy az összes tesztet futtassuk melyek kimenetelét egy expected outputhoz hasonlítja össze a program, hogy sikeres volt-e a test.

8.4 Napló

Kezdet	Időtartam	Résztvevők	Leírás
2025.04.05. 19:00	0,5 óra	Fórián Gyárfás Kemecsei Kuzmin Tóth	Megbeszélés: konzultációs információk, feladatok átbeszélése
2025.04.08. 14:00	4 óra	Tóth	Tevékenység: maradék osztályok implementálása
2025.04.12. 14:00	1 óra	Kemecsei	Tevékenység: verziókezelés, merge conflict-ok kezelése, program debugolás, project management, bemeneti nyelv dokumentáció javítása
2025.04.12. 17:00	4 óra	Fórián	Tevékenység: fordító futtató program elkészítése, dokumentálása
2025.04.13. 08:00	6 óra	Kemecsei	Tevékenység: Osztályleírások, tervek megírása
2025.04.13. 18:00	8 óra	Gyárfás	Tevékenység: Tesztesetek megírása, dokumentálása