8. Részletes tervek – Függelék

Javított oldalak 1-20

8. Részletes tervek

[A dokumentum célja, hogy pontosan specifikálja az implementálandó osztályokat, beleértve a privát attribútumokat és metódusokat, ezek definícióját is.
A dokumentum második fele részletesen be kell mutassa a korábban definiált be- és kimeneti nyelv szintakszisát felhasználva, hogy mely tesztekkel lesz a prototípus ellenőrizve.]

8.1 Osztályok és metódusok tervei.

8.1.1 *TectonImpl*

- **Felelősség**

  Az absztrakt Tecton osztály implementációja. Kezeli a gombatestek és gombafonalak fenntartását, rajta lévő objektumok tárolását. Szomszédos tektonok tárolását.

- **Interfészek**

  Tecton, RoundBeginSubscriber

- **Attribútumok**
  - **-breakTimer**: int
    Az időzítő amely ha elér 0-ra a tekton eltörik
  - ***-neighbours**: **List<Tecton>***
    A tektonnal szomszédos tektonok listája
  - ***-myceliumCapacity**: **int***
    Maximum ennyi gombafonál lehet az adott tektonon
  - ***-spores**: **Queue<Spore>***
    Az adott tektonon lévő spórák listája
  - ***-mushroomBody**: **MushroomBody***
    Itt van eltárolva ha az adott tektonon van-e gombatest
  - ***-mycelia**: **Queue<Mycelium>***
    Az adott tektonon lévő gombafonalak listája
  - ***-occupants**: **List<Insect>***
    Az adott tektonon lévő bogarak listája
  - ***-notSustained: Set<Tecton>***
    A tektonok összessége, aminek fonalai nincsenek összeköttetésben gombatestel, ezért el fognap pusztulni

- **Metódusok**
  - ***+getBreakTimer(): int***
    A tektontörés getterje
  - ***+setBreakTimer(breakTimer: int): void***
    A tektontörés setterje. A kapott paraméter az új körök száma a törésig
  - ***+getNeighbours(): List<Tecton>***
    A tekton szomszédlistájának getterje
  - ***+setNeighbours(neighbours: List<Tecton>): void***
    A tekton szomszédlistájának setterje. A kapott paraméter az új szomszédok listája
  - ***+getMyceliaCapacity(): int***

A maximális gombafonál szám

- ***+setMyceliaCapacity(myceliaCapacity: int): void***
  A maximális gombafonál szám setterje
- ***+getSpores(): Queue<Spore>***
  A tektonon lévő spórák getterje
- ***+setSpores(spores: Queue<Spore>): void***
  A tektonon lévő spórák setterje
- ***+getMushroomBody(): MushroomBody***
  A tektonon lévő gombatest getterje
- ***+setMushroomBody(mushroomBody: MushroomBody): void***
  A tektonon lévő gombatest setterje
- ***+getMycelia(): Queue<Mycelium>***
  A tektonon lévő gombafonalak getterje
- ***+setMycelia(mycelia: Queue<Mycelium>): void***
  A tektonon lévő gombafonalak setterje
- ***+distance(tecton: Tecton): int***
  A függvény megadja, hogy milyen messze van egy cél tekton a jelenlegi tektontól (ezt a metódust a spórák kilövés miatt kell használni, hogy az adott gombatest tudja, hogy melyik tektonra szabad, vagy nem, spórát lőjön)

```
procedure distance(target_tecton):

// Inicializáció
 distances ← CREATE new Map (to store Tecton → Integer distance)
 queue ← CREATE new Queue (to store Tectons to visit)

// BFS elindítása az adott Tectonrol
 SET distances[this] ← 0       // Távolság a kezdeti Tectontol önmagáig
 ENQUEUE this INTO queue       // a queue-ba belerakni a kezdeti Tectont

 //BFS
 WHILE queue IS NOT EMPTY DO
   // Lekérni a következő Tectont
     current_tecton ← DEQUEUE from queue
     // Lekérni a távolságát
     current_distance ← GET distances[current_tecton]

    // Megnézni, hogy elértük-e a cél Tectont
     IF current_tecton IS target_tecton THEN
        RETURN current_distance
     END IF

     // Meglátogatni a szomszédos Tectonokat
     FOR EACH neighbour IN current_tecton.neighbours DO
    // Ha meg nem volt ez a Tecton meglatogatva
        IF distances DOES NOT CONTAIN neighbour THEN
       // Beállítani mint látogatott, elmenteni távolságát es a queue-ba rakni
            SET distances[neighbour] ← current_distance + 1
            ENQUEUE neighbour INTO queue
        END IF
     // Különben nem csinálunk semmit
     END FOR EACH
 END WHILE

// Ha nem találtuk meg a cél Tectont
 RETURN -1
END
```

- ***-neighboursWithMycelia(): List<Tecton>***
  Azok a szomszédok összege, amelyen van gombafonál vagy gombatest
- ***-myceliaCheckSustain(): void***
  A függvény megnézi, hogy a tekton és velük gombafonállal összekötött tektonok még összekötésben állnak-e gombatestel

```
procedure myceliaCheckSustain:

// Inicializáció
connected ← CREATE new Set<Tecton>
queue ← CREATE new Queue<Tecton>
visited ← CREATE new Set<Tecton>
isSustaining = false

ADD this To visited
ENQUEUE this ONTO queue

 //BFS
 WHILE queue IS NOT EMPTY DO
   // Lekerni a kovetkezo Tectont
     current_tecton ← DEQUEUE from queue
   ADD current_tecton TO connected
      IF current_tecton.sustaining THEN BEGIN
      isSustaining = true
      END

      FOR EACH neighbour IN neighboursWithMycelia(current_tecton) DO BEGIN
      IF ADD neighbour TO visited returns True THEN BEGIN
            ENQUEUE neighbour ONTO queue
      END
      END FOR EACH
 END WHILE
 IF isSustaining IS False THEN BEGIN
      ADD ALL elements FROM connected TO notSustained
 END
END
```

- ***+checkNeighbourMyceliaSustain(): void***
  A függvény megnézi, hogy a szomszédos tektonok és velük gombafonállal összekötött tektonok még összeköttetésbe állnak-e gombatestel

```
procedure checkNiegbhourMyceliaSustain:

 // Inicializáció
 CLEAR notSustained

 FOR EACH neighbour IN this.neighbours DO BEGIN
      CALL neighbour.myceliaCheckSustain
 END

 FOR EACH tecton IN notSustained DO BEGIN
      FOR EACH m IN tecton.getMycelia DO BEGIN
            REMOVE m FROM tecton.mycelia
            CALL m.delete
      END FOR EACH
 END FOR EACH
END
```

- ***+getOccupants(): List<Insect>***
  A tektonon lévő rovarok listájának getterje
- ***+setOccupants(occupants: List<Insect>): void***
  A tektonon lévő rovarok listájának setterje

- ***+addOccupant(insect: Insect): void***
  Hozzáad egy rovart a tektonhoz
- ***+removeOccupant(insect: Insect): void***
  Levesz egy rovart a tektonról
- ***+hasMycelium(): boolean***
  true-t ad vissza, ha van-e legalább 1 gombafonál a tektonon ami nem növekszik, különben false
- ***+addMycelium(mycelium: Mycelium): void***
  Hozzáad egy gombafonalat a tektonhoz
- ***+addSpore(spore: Spore): void***
  Hozzáad egy spórát a tektonhoz
- ***+transferSpores(newSpores: List<Spore>): void***
  Hozzáad egyszerre több spórát a tektonhoz
- ***+addNeighbour(tecton: Tecton): void***
  Egy új szomszédot ad a tektonnak
- **accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void**
  Eldönti, hogy az adott gombafonál nőhet-e ezen a tektonon
- **accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void**
  Eldönti, hogy az adott gombatest nőhet-e ezen a tektonon
- ***+sustaining(): boolean***
  A sustaining tektonnál true, a többinél false, kivéve ha van rajtuk gombatest, akkor true
- ***killOccupants(): void***
  Megpróbál minden rajta lévő rovart eltávolítani
- **+eatSpore(insect: SporeEater):** Ha van spóra a tectonon, meghívja az első spórának az eatSpore() metódusát, a megkapott insect-el, mint argumentum
- **+cutMycelium():** Elvágódik az első spóra a tectonon
-

```
procedure cutMycelium:

      DEQUEUE mycelium FROM mycelia
      mycelium.cutWithDelay

END
```

- **+moveInsect(insect: InsectMover, insectLocation: Tecton):** Ha tud az insect a tectonra (amin meg volt hívva a metódus) mozogni, akkor megcsinálja ezt a műveletet

```
procedure moveInsect(insect, insectLocation):

 distance = insectLocation.distance(this)

      IF (distance IS EQUAL TO 1) AND (this.hasMycelium IS TRUE) THEN BEGIN
            insectLocation.removeOccupant(insect)
            this.addOccupant(insect)
            insect.setLocation(this)
            insect.setRemainingMoves(insect.getRemainingMoves() - 1)
      END
END
```

8.1.2 FertileTectonImpl

- **Felelősség**

A FertileTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl

- **Interfészek**

FertileTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+FertileTecton():***
    konstruktor, a gombafonál kapacitást beállítja 1-re és a BreakTimer-jét is beállítja
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
END
```

  - ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator , mushroomBody: MushroomBody): void***
    eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
```

```
 CALL mushroomBody.grow(sporeCount)
END
```

- ***+onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer - 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE

      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. FertileTectonnal ha van rajta gombatest akkor true-val tér vissza különben false

8.1.3 SemiFertileTectonImpl

- **Felelősség**

A SemiFertileTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → SemiFertileTectonImpl

- **Interfészek**

SemiFertileTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+SemiFertileTecton():***
    konstruktor, a gombafonál kapacitást beállítja 1-re és a BreakTimer-jét is beállítja
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)
```

```
 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)

END
```

- ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
  eldönti hogy a kapott gombatest nőhet-e az adott tektonon. Itt szimplán kitörli a kapott gombatestet, mivel SemiFertile tectonon sosem nőhet gombatest
- ***+onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer - 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE

      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. SemiFertileTectonnal mindig false-al tér vissza

8.1.4 MultiLayeredTectonImpl

- **Felelősség**

A MultiLayeredTecton típusu tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl → MultiLayeredTectonImpl

- **Interfészek**

MultiLayeredTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+MultiLayeredTecton():***
    konstruktor, a gombafonál kapacitást beállítja 3-ra
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
END
```

  - ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
    eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
 CALL mushroomBody.grow(sporeCount)
END
```

  - ***+onRoundBegin(): void***
    Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer - 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE
```

```
      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. MultiLayeredTectonnal ha van rajta gombatest akkor true-val tér vissza különben false

8.1.5 AridTectonImpl

- **Felelősség**

Az AridTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl → AridTectonImpl

- **Interfészek**

AridTecton, RoundBeginSubscriber, TurnBeginSubscriber

- **Attribútumok**
  - **-absorbCountdown**: **int**
    Azt mutatja, hogy hány kör múlva szívja fel a gombafonalat a tekton

- **Metódusok**
  - ***+AridTecton():***
    konstruktor, a gombafonál kapacitást beállítja 1-re
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
 absorbCountdown ← 5
END
```

- ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
  eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
 CALL mushroomBody.grow(sporeCount)
END
```

- ***+onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer - 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE


      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***+onTurnBegin(): void***
  Itt történik a tektonon lévő fonál elszáradása és így elpusztulása, ha az absorbCountdown eléri a 0-át

```
procedure onTurnBegin:

 IF absorbCountdown > 0 THEN BEGIN
      absorbCountdown ← absorbCountdown - 1
      IF absorbCountdown <= 0 THEN BEGIN
            mycelium DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.delete
            END
      END
 END
END
```

- ***sustaining(): boolean***

Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat. AridTectonnal ha van rajta gombatest akkor true-val tér vissza különben false

8.1.6 SustainingTectonImpl

- **Felelősség**

A SustainingTecton típusú tektonok növesztésének feltételét szabályozza. A többi felelősségét az absztrakt Tecton osztálytól örökli.

- **Ősosztályok**

TectonImpl → FertileTectonImpl → SustainingTectonImpl

- **Interfészek**

SustainingTecton, RoundBeginSubscriber

- **Metódusok**
  - ***+SustainingTecton():***
    konstruktor, a gombafonál kapacitást beállitja 1-re és a BreakTimer-jét is beállitja
  - ***+accept(myceliumGrowthEvaluator: MyceliumGrowthEvaluator, mycelium: Mycelium): void***
    eldönti hogy a kapott gombafonál nőhet-e az adott tektonon

```
procedure accept(myceliumGrowthEvaluator : MyceliumGrowthEvaluator, mycelium :
Mycelium)

 currentMyceliaCount ← GET this.getMycelia.size
 capacity ← GET this.getMyceliaCapacity

 IF currentMyceliaCount >= capacity THEN BEGIN
      CALL mycelium.delete
      RETURN
 END
 ADD mycelium TO this.getMycelia
 sporeCount ← GET this.getSpores.size
 CALL mycelium.grow(sporeCount)
END
```

  - ***+accept(mushroomBodyGrowthEvaluator: MushroomBodyGrowthEvaluator, mushroomBody: MushroomBody): void***
    eldönti hogy a kapott gombatest nőhet-e az adott tektonon

```
procedure accept(mushroomBodyGrowthEvaluator : MushroomBodyGrowthEvaluator,
mushroomBody : MushroomBody)

 sporeCount ← GET this.getSpores.size
 hasExistingMushroomBody ← (GET this.getMushroomBody IS NOT NULL)
 hasMycelia ← (GET this.getMycelia.isEmpty IS FALSE)

 IF (sporeCount < 3) OR (hasExistingMushroomBody IS TRUE) OR (hasMycelia IS
      FALSE)  THEN BEGIN
            CALL muhsroomBody.delete
            RETURN
 END

 CALL this.setMushroomBody(mushroomBody)
```

```
 CALL mushroomBody.grow(sporeCount)
END
```

- ***+onRoundBegin(): void***
  Itt történik a tektontörés és annak következményei

```
procedure onRoundBegin:

 currentBreakTimer ← GET this.getBreakTimer
 SET this.breakTimer ← currentBreakTimer – 1

 currentBreakTimer ← GET this.getBreakTimer
 IF currentBreakTimer <= 0 THEN BEGIN
      WHILE GET this.getMycelia.isEmpty IS FALSE DO BEGIN
            mycelium ← DEQUEUE FROM this.getMycelia
            IF mycelium IS NOT NULL THEN BEGIN
                  CALL mycelium.cutImmediate
            END
      END WHILE

      newFertileTecton ← CREATE new FertileTecton
      CALL newFertileTecton.addNeighbour(this)
      CALL this.addNeighbour(newFertileTecton)

 END

END
```

- ***sustaining(): boolean***
  Visszaadja, ha az adott tekton képes-e fenntartani gombafonalakat.
  SustainingTectonnal mindig true-val tér vissza

8.1.7 MushroomBodyImpl

- **Felelősség**

  A gombatestekért felelős osztály. A gombatest a spórák termeléséért és kilövéséért felelős. 3 spórakilövés után inaktívvá válik, amely abban nyilvánul meg, hogy a remaingEjects változó értéke 0 lesz. A gombatest az utolsó kilövése előtt csak valamely szomszédjára lőhet spórát. Fejlettnek az utolsó kilövése során minősül, amely abban nyilvánul meg, hogy ekkor a szomszédja szomszédjára is tud lőni.

- **Interfészek**

  Mushroom, MushroomBody, TurnBeginSubscriber

- **Attribútumok**

  - **-remainingEjects: int**

    A megmaradt spórakilövések számát tároló változó. Alapértelmezett értéke 3.

  - **-location: Tecton**

    A gombatest elhelyezkedése szerinti tektont tároló változó.

- **-mushroomSpores: List<Spore>**

  A gombatest spóráit tartalmazó lista. A lista alapértelmezetten üres.

- **Metódusok**

  - **+MushroomBody(location: FertileTecton, name: String)**

    Konstruktor, amely beállítja a létrehozandó gombatest nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó FertileTecton, továbbá a FertileTecton valamennyi leszármazottja, azaz AridTecton, MultiLayeredTecton és SustainingTecton esetén. A metódus pszeudokódja:

```
procedure MushroomBody(location: FertileTecton, name: String)
        // elmenti a céltektont
        SET this.location ← location

        // létrehoz egy MushroomBodyGrowthEvaluator példányt
        evaluator ← CREATE MushroomBodyGrowthEvaluator(this)

        // a feltételek fennállásának kiértékelése céljából meghívja a
        // a visit metódust
        CALL evaluator.visit(location, this)
END
```

  - **+MushroomBody(location: SemiFertileTecton, name: String)**

    Konstruktor, amely beállítja a létrehozandó gombatest nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó SemiFertileTecton esetén. A metódus pszeudokódja:

```
procedure MushroomBody(location: SemiFertileTecton, name: String)
        // elmenti a céltektont
        SET this.location ← location

        // létrehoz egy MushroomBodyGrowthEvaluator példányt
        evaluator ← CREATE MushroomBodyGrowthEvaluator(this)

        // a feltételek fennállásának kiértékelése céljából meghívja a
        // a visit metódust
        CALL evaluator.visit(location, this)
END
```

  - **+MushroomBody()**

    Paraméter nélküli (default) konstruktor.

- **+delete(): void**

  A növekedési feltételek hiánya esetében kerül meghívásra az előzetesen létrehozott gombatest törlése céljából.

- **+grow(sporeCount: int): void**

  A gombatest növekedési folyamatát lezáró metódus, amelyet a Mushroom interfész miatt szükséges a gombatestnél ilyen formában megvalósítani. A paramétert a céltektontól kapja. A tekton abban az esetben hívja meg ezt a metódust (és nem a delete()-et), ha a gombatest növesztési feltételeire vonatkozó vizsgálat pozitív eredményt hozott. Ezért ez a metódus a gombatest esetében nem, csak a gombafonálnál bír jelentőséggel.

- **+onTurnBegin(): void**

  A gombatest minden új körének kezdetekor – beleértve a játék első körét is – a gombatestben egy új spóra termelődik. A spóra típusa véletlenszerűen kerül kiválasztásra. A metódus pszeudokódja:

```
procedure onTurnBegin()
        // Egy spóratípust véletlenszerűen kiválasztásra kerül
        random ← RANDOM NUMBER BETWEEN 1 AND 5

        IF random == 1 THEN
                newSpore ← CREATE SpitSpore()
        ELSE IF random == 2 THEN
                newSpore ← CREATE StunSpore()
        ELSE IF random == 3 THEN
                newSpore ← CREATE PreventCutSpore()
        ELSE IF random == 4 THEN
                newSpore ← CREATE SpeedSpore()
        ELSE
                newSpore ← CREATE SlownessSpore()
        END IF

        // Hozzáadja az új spórát a gombatest spóralistájához
        CALL this.addSpore(newSpore)
END
```

- **+getRemainingEjects(): int**

  Visszaadja a gombatest megmaradt spórakilövéseinek számát.

- **+setRemainingEjects(remainingEjects: int): void**

  Beállítja a gombatest megmaradt spórakilövéseinek számát.

- **+getSpores(): List<Spore>**

  Visszaadja a gombatest spóráit tartalmazó listát.

- **+addSpore(newSpore: Spore): void**

  Hozzáad egy új spórát a gombatest spóráinak listájához.

- **+ejectSpores(target: Tecton): void**

  A gombatest spóráinak kilövéséért felelős metódus. A metódus pszeudokódja:

```
procedure ejectSpores(target: Tecton)
    // Ha már volt 3 spórakilövése, a gombatest inaktív, nem tud aktivitást
    // kifejteni, így spórát sem lőhet ki (nincs is már neki)
    IF remainingEjects == 0 THEN
        RETURN // A gombatest inaktív, nem tud aktivitást kifejteni!
    END IF

    // Ha ez az utolsó, azaz a 3. spórakilövése, a gombatest fejlett állapotú,
    // így a céltekton lehet szomszéd vagy a szomszéd szomszédja
    IF remainingEjects == 1 THEN
        reachable ← EMPTY SET

        FOR EACH primary IN this.neighbours DO
            ADD primary TO reachable
            FOR EACH secondary IN primary.neighbours DO
                ADD secondary TO reachable
            END FOR
        END FOR

        IF target IS IN reachable THEN
            IF this.mushroomSpores IS NOT EMPTY
                ejectSpores(target)
                remainingEjects ← remainingEjects – 1
            ELSE
                RETURN // A gombatestnek nincsen kilőhető
                       // spórája!
            END IF
        ELSE
            RETURN // A céltekton túl messze van!
        END IF

    ELSE
    // A gombatest még nem fejlett, ezért csak közvetlen szomszédjára lőhet
    // spórát
        IF target IS IN this.neighbours THEN
            IF this.mushroomSpores IS NOT EMPTY
                ejectSpores(target)
                remainingEjects ← remainingEjects – 1
            ELSE
                RETURN // A gombatestnek nincsen kilőhető
                       // spórája!
            END IF
        ELSE
            RETURN // A céltekton túl messze van!
        END IF
    END IF
END
```

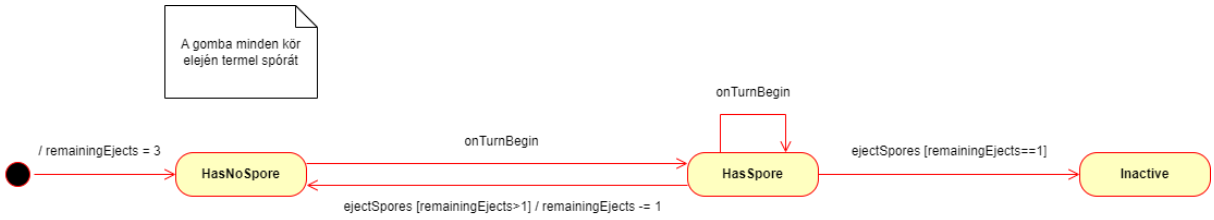
- **+getLocation(): Tecton**

  Visszaadja a gombatest elhelyezkedése szerinti tektont.

- **+setLocation(location: Tecton): void**

  Beállítja a gombatest elhelyezkedése szerinti tektont.

- **Állapot Diagramm**



8.1.8 MyceliumImpl

- **Felelősség**

A játékban a gombafonalakat reprezentáló osztály. Felelős a gombafonal növési folyamatban a növés gyorságáért, a gombafonalak elvágása esetén pedig a fonál kitörléséért és részben a többi fonál életben maradásának ellenőrzéséért is.

- **Interfészek**

Mushroom, Mycelium, TurnBeginSubscriber

- **Attribútumok**

- -growing: boolean
  Azt ábrázolja, hogy a gombafonál éppen növekedés alatt van-e. Alapértelmezett értéke hamis. Ha a tekton úgy dönt, hogy nőhet rajta a fonál akkor igaz lesz.
- -location: Tecton
  A tekton, ahol a gombafonál elhelyezkedik.
- -growTimer: int
  Az idő, ami alatt a gombafonál megnő. Alapértelmezett értéke 0. Növéskor, ha a céltektonon van spóra, akkor ez az érték 1, ha nincs akkor 2 lesz.
- -deathTimer:int
  Az idő, ami mulva elszakad a fonál. Alapértelmezett érték -1

- **Metódusok**

- +Mycelium(location: FertileTecton, name: String)
  Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó FertileTectonon, továbbá a FertileTecton valamennyi leszármazottjánál, azaz AridTecton, MultiLayeredTecton és SustainingTecton esetén.

```
procedure Mycelium(location: FertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location

    // létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
    myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

    //A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
    //függvényét, ami majd a location Tektonnal kommunikál
    CALL myceliumGrowthEvaluator.visit(location)
END
```

- +Mycelium(location: SemiFertileTecton, name: String)

Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó SemiFertileTectonon.

```
procedure Mycelium(location: SemiFertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location
```

```
    // létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
    myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

    //A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
    //függvényét, ami majd a location Tektonnal kommunikál
    CALL myceliumGrowthEvaluator.visit(location)
END
```

- +delete(): void
  Kitörli a gombafonalat
- +grow(sporeCount: int): void
  Ezt a jelzést a tekton fogja meghívni a gombafonálra, ha nőhet. A céltektonon lévő spóraszámtól függően módosítja a growtimert.
- +onTurnBegin(): void
  A growtimer visszaszámlálását végzi (minden körben eggyel csökken) és ha az lejárt, már nem lesz növésben a gombafonál.

  A deathTimer visszaszámlálását is végzi (minden körben eggyelc csökken), és ha az lejárt, meghivja a cutImmediate metódust
- +isGrowing(): boolean
  A growing attribútum getterje
- setGrowing(growing: boolean): void
  A growing attribútum setterje
- +cutImmediate(): void
  A gombafonál elvágódik, ezzel szól a többi gombafonálnak, hogy nézzék meg, hogy hozzá vannak-e csatlakoztatva gombatesthez

  Pszeudokód

```
Procedure cutImmediate:
      this.delete()
      location.checkNeighbourMyceliaSustain()
      if(location.getMycelia IS EMTPY) begin
            List<Insect> temp
            FOR EACH i IN location.getOccupants begin
                  ADD i TO temp
            end
            FOR EACH i IN temp begin
                  i.runAway()
            end
      end
end procedure
```

- +cutWithDelay(): void

  Beállítja a deathTimer-t 2-re
- +getLocation(): Tecton

A location attribútum getterje

- +setLocation(location: Tecton): void
  A location attribútum getterje

8.1.9 CarnivorousMycelium

- **Felelősség**

Az alapvető gombafonál funkciókon kívül speciális feltételek között a rovarak evését és fonál növesztését megvalósító osztály.

- **Interfészek**

Mushroom, Mycelium, TurnBeginSubscriber

- **Attribútumok**

- -growing: boolean
  Azt ábrázolja, hogy a gombafonál éppen növekedés alatt van-e. Alapértelmezett értéke hamis. Ha a tekton úgy dönt, hogy nőhet rajta a fonál akkor igaz lesz.
- -location: Tecton
  A tekton, ahol a gombafonál elhelyezkedik.
- -growTimer: int
  Az idő, ami alatt a gombafonál megnő. Alapértelmezett értéke 0. Növéskor, ha a céltektonon van spóra, akkor ez az érték 1, ha nincs akkor 2 lesz.
- -deathTimer:int
  Az idő, ami mulva elszakad a fonál. Alapértelmezett érték -1

- **Metódusok**

- +Mycelium(location: FertileTecton, name: String)
  Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó FertileTectonon, továbbá a FertileTecton valamennyi leszármazottjánál, azaz AridTecton, MultiLayeredTecton és SustainingTecton esetén.

```
procedure CarnivorousMycelium(location: FertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location

    // létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
    myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

    //A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
    //függvényét, ami majd a location Tektonnal kommunikál
    CALL myceliumGrowthEvaluator.visit(location)
END
```

- +CarnivorousMycelium(location: SemiFertileTecton, name: String)

Konstruktor, amely beállítja a létrehozandó gombafonál nevét és azt a tektont (céltekton), amelyen az elhelyezésre kerül. Ez a konstruktor használandó SemiFertileTectonon.

```
procedure Mycelium(location: SemiFertileTecton, name: String)
    // A fonál helyét beállítja a kapott helynek
    SET this.location ← location
```

// létrejön egy új MyceliumGrowthEvaluator aminek paraméterként beadjuk a Myceliumot
myceliumGrowthEvaluator ← CREATE MyceliumGrowthEvaluator(this)

//A fonálnövesztés feltételeinek ellenőrzésére meghívjuk a myceliumGrowthEvaluator visit
//függvényét, ami majd a location Tektonnal kommunikál
CALL myceliumGrowthEvaluator.visit(location)
END

- +delete(): void
  Kitörli a gombafonalat
- +grow(sporeCount: int): void
  Ezt a jelzést a tekton fogja meghívni a gombafonálra, ha nőhet. A céltektonon lévő spóraszámtól függően módosítja a growtimert.
- +onTurnBegin(): void
  A growtimer visszaszámlálását végzi (minden körben eggyel csökken) és ha az lejárt, már nem lesz növésben a gombafonál. Ha a tektonján levő rovarok Stunned állapotban vannak megöli a rovarokat és egy gombát növesztését kezdeményezi.

  A deathTimer visszaszámlálását is végzi (minden körben eggyelc csökken), és ha az lejárt, meghivja a cutImmediate metódust
- +isGrowing(): boolean
  A growing attribútum getterje
- setGrowing(growing: boolean): void
  A growing attribútum setterje
- +cutImmediate(): void
  A gombafonál elvágódik, ezzel szól a többi gombafonálnak, hogy nézzék meg, hogy hozzá vannak-e csatlakoztatva gombatesthez

  Pszeudokód

```
Procedure cutImmediate:
	this.delete()
	location.checkNeighbourMyceliaSustain()
	if(location.getMycelia IS EMTPY) begin
		List<Insect> temp
		FOR EACH i IN location.getOccupants begin
			ADD i TO temp
		end
		FOR EACH i IN temp begin
			i.runAway()
		end
	end
end procedure
```

- +cutWithDelay(): void

  Beállítja a deathTimer-t 2-re

- +getLocation(): Tecton
  A location attribútum getterje
- +setLocation(location: Tecton): void
  A location attribútum getterje