# Részletes tervek

## Osztályok és metódusok tervei.

### MushroomBodyImpl

#### Felelősség

#### A gombatestekért felelős osztály. A gombatest a spórák termeléséért és kilövéséért felelős. 3 spórakilövés után inaktívvá válik. 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.

#### -location: Tecton

A gombatest elhelyezkedése szerinti tektont táróló változó.

#### -mushroomSpores: List<Spore>

A gombatest spóráit tartalmazó lista.

#### Metódusok

#### +MushroomBody(location: FertileTecton, name: String)

Konstruktor, amely beállítja a létrehozandó gombatest nevét és azt a tektont, amelyen az elhelyezésre kerül.

#### +MushroomBody(location: SemiFertileTecton, name: String)

Konstruktor, …

#### +MushroomBody()

Paraméter nélküli (default) konstruktor.

#### +delete(): void

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

#### +grow(sporeCount: int): void

…

#### +onTurnBegin(): void

xxx

#### +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 spórák listájához.

#### +ejectSpores(target: Tecton): void

A gombatest spóráinak kilövéséért felelős metódus. A metódus pszeudokódja:

// Ha már volt 3 spórakilövése, a gombatest inaktív, nem tud aktivitást kifejteni // így spórát sem lőhet ki spórát (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

ejectSpores(target)

remainingEjects ← remainingEjects - 1

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

ejectSpores(target)

remainingEjects ← remainingEjects - 1

ELSE

RETURN // A céltekton túl messze van!

END IF

END IF

#### +getLocation(): Tecton

Visszaadja a gombatest elhelyezkedése szerinti tektont.

#### +setLocation(location: Tecton): void

Beállítja a gombatest elhelyezkedése szerinti tektont.