### ReactsToSunflare

#### Felelősség

Ez az interfészt azok az osztályok valósítják meg, amik reagálnak a napviharra

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

#### -

#### Metódusok

* **+getNotifiedAboutSunflare():** Egy absztrakt függvény amely elvégzi azokat a függvény hívásokat, amiket a napvihar esetén kell az osztálynak hívnia.

### Robot

#### Felelősség

Ez az osztály felel az robotértés függvényeiért.

#### Ősosztályok

Printable->Entity->NonPlayer->Robot

#### Interfészek

ReactsToSunflare, Moving, Mining

#### Attribútumok

* **-id: int:** Ez az adott Robot id-je, ezt arra használjuk fel, hogy az Robot-t azonosítani tudjuk.
* **-nextId: int:** Ez egy statikus tagváltozó mely azt tartja számon, hogy mi legyen a következő létrehozott id-t igénylő objektum id-je. Azért statikus, hogy az id-k egyediségét garantáljuk.
* -**asteroid:Asteroid**: Ezen az aszteroidán van a Robot.
* -**owner:Game**: Ez az osztály tárolja le a Robot-t
* **-isRandom: Boolean:** Egy boolean ami azt jelzi, hogy a Sun random módon, vagy determinisztikusan működik-e (kitörés esély számlálás közben), ez csak a teszteléshez kell.

#### Metódusok

* **+getUIString(): String:** Ez a függvény generálja az adott Robot fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az épület id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén.
* **+getNotifiedAboutSunflare():** Ez a függvény az osztály die függvényét hívja, amennyiben az asteroida-ja nem igaz értékkel tér vissza az isHollow függvényre
* **+move(destination: Asteroid):** egy függvény, aszteroidáról aszteroidára való mozgást végzi.

|  |
| --- |
| If(destination is reachable from NonPlayer.asteroid) {  destination.add(NonPlayer)  NonPlayer.asteroid.remove(NonPlayer)  NonPlayer.asteroid=destination } |

* **+move(destinationGate: StarGate):** ez a függvény, amely elvégzi azokat a mozgással kapcsolatos dolgokat ahol egy teleport kaput használunk, amiket a mozgásra képes osztályoknak kell.

|  |
| --- |
| If(destinationGate is active) {  destinationGate.neighbour.asteroid.add(NonPlayer)  NonPlayer.asteroid.remove(NonPlayer)  NonPlayer.asteroid= destinationGate.neighbour.asteroid } |

* **+die():** ez a függvény kiveszi a Robo-t az aszteroidáról, és jelzi a játéknak, hogy kivehető a létező elemekből.
* **+asteroidExploded():** ez a függvény biztosítja, hogy hogyan viselkedjen az osztály, amikor alatta egy aszteroida felrobban. A robot esetében ez move-t hív a robot aszteroidjának egyi szomszédjával mint paraméter. Amennyiben az nem lehetséges Robot die függvényét hívja.
* **+onTurn():** Egy függvény hajtja végre a Robot körön belüli viselkedését. 3 körig drill-t hívja, majd 4. körben az asteroida-jának véletlenszerűen kiválasztót szomszédjával a move függvényét meghívja. Majd ezt ismétli.
* **+drill():** Ez a függvény meghívja a Robot asteroid-n a drilling függvényt.

### Settler

#### Felelősség

Ez az osztály felel a játékos által irányítható telepesért.

#### Ősosztályok

Printable->Entity->Settler

#### Interfészek

ReactsToSunflare, Moving, Drilling, Mining

#### Attribútumok

* **-id: int:** Ez az adott Settler id-je, ezt arra használjuk fel, hogy az Settler-t azonosítani tudjuk.
* **-nextId: int:** Ez egy statikus tagváltozó mely azt tartja számon, hogy mi legyen a következő létrehozott id-t igénylő objektum id-je. Azért statikus, hogy az id-k egyediségét garantáljuk.
* -**asteroid:Asteroid**: Ezen az aszteroidán van a Settler.
* -**owner:Game**: Ez az osztály tárolja le a Robot-t
* **-inventory:Inventory:** Ebben tárolja a Settler általa össze gyűjtőt nyersanyagokat.
* **-hadActionThisTurn:Boolean**: Ez tárolja, hogy lehet e ezzel a telepessel lépni az adott körben.
* **-baseRecipe**: ebben van benne, hogy mi kell egy bázis megépítéséhez.
* **- robotRecipe**: ebben van benne, hogy mi kell egy robot megépítéséhez.
* **-stargateRecipe**: ebben van benne, hogy mi kell egy kapú megépítéséhez.

#### Metódusok

* **+getUIString(): String:** Ez a függvény generálja az adott Settler fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az épület id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén.
* **+getNotifiedAboutSunflare():** Ez a függvény az osztály die függvényét hívja, amennyiben az asteroida-ja nem igaz értékkel tér vissza az isHollow függvényre
* **+move(destination: Asteroid):** egy függvény, aszteroidáról aszteroidára való mozgást végzi.

|  |
| --- |
| If(destination is reachable from NonPlayer.asteroid) {  destination.add(NonPlayer)  NonPlayer.asteroid.remove(NonPlayer)  NonPlayer.asteroid=destination } |

* **+move(destinationGate: StarGate):** ez a függvény, amely elvégzi azokat a mozgással kapcsolatos dolgokat ahol egy teleport kaput használunk, amiket a mozgásra képes osztályoknak kell.

|  |
| --- |
| If(destinationGate is active) {  destinationGate.neighbour.asteroid.add(NonPlayer)  NonPlayer.asteroid.remove(NonPlayer)  NonPlayer.asteroid= destinationGate.neighbour.asteroid } |

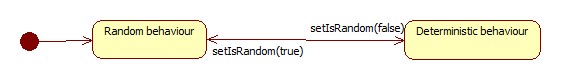
* **+die():** ez a függvény kiveszi a Settler-t az aszteroidáról, és jelzi a játéknak, hogy kivehető a létező elemekből.
* **+asteroidExploded():** ez a függvény meghívja a Settler die függvényét.
* **+drill():** Ez a függvény meghívja a Settler asteroid-n a drilling függvényt.
* **+mine()**: egy függvény, amely kivesz egy Material-t az aszteroidából, ha az aszteroida kérge 0.
* **+buildStarGate():** Amennyiben van kellő nyers anyag azokat kiveszi az Inventory-ból és egy kapu párthelyez az Inventory-ba
* **+buildRobot():** Amennyiben van kellő nyersanyag létrehoz egy robotot az aszteroidán, amin tartozkódik, és kiveszi a felhasznált nyersanyagokat az Inventory-ból
* **+buildBase():** Amennyiben van elegendő nyersanyag, létrehoz egy bázist az aszteroidán és a felhasznált nyersanyagokat kiveszi az Inventory-kból.
* **+dropMaterial(m:Material):** Amennyiben lehetséges (üres aszteroida átfúrt felszínnel), nyersanyagot helyez az aszteroidába az invetory-ból.
* **+placeStarGate():** Amennyiben van az Inventory-ben StarGate, lehelyezi azt az adott aszteroidára.

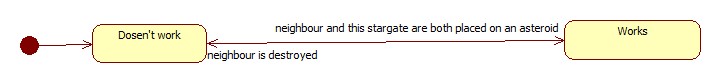
### StarGate

#### Felelősség

Ez az osztály felel a teleportkapukért s azok feladataiért, pl: hogy azok mindig jól össze legyenek párosítva, illetve ha a párból az egyik elpusztult akkor erről tudjon a másik (ha van rá szüksége).

#### Állapotdiagramok





(a felső feltétel a balról jobbra menő nyilakhoz tartozik, az alsó pedig a jobbról balra menő nyilakhoz)

#### Ősosztályok

Printable -> Building -> StarGate

#### Interfészek

reactsToSunFlare, Controllable. Moving

#### Attribútumok

* **-id: String:** Ez az adott StarGate id-je, ezt arra használjuk fel, hogy a játékos ki tudja választani a megfelelő objektumot, pl: amikor egy settlerrel StarGate-re lép, akkor a StarGate id-jét megadva választhatja ki melyikre kíván rálépni.
* **-nextId: int:** Ez egy statikus tagváltozó mely azt tartja számon, hogy mi legyen a következő létrehozott id-t igénylő objektum id-je. Azért statikus, hogy az id-k egyediségét garantáljuk.
* **-asteroid: Asteroid:** Az aszteroida amin a StarGate található.
* **-neighbour: StarGate:** A StarGate-tel összepárosított StarGate.
* **-works: Boolean:** Egy bool ami azt jelzi, hogy az adott StarGate működik-e.
* **-wasInSunFlare: Boolean:** Egy bool ami azt jelzi, hogy az adott StarGate-t érte-e már napvihar.
* **-isRandom: Boolean:** Egy boolean ami azt jelzi, hogy a StarGate random módon, vagy determinisztikusan működik-e (mozgás közben), ez csak a teszteléshez kell.
* **-owner: Game:** Egy referencia a Game osztályhoz.

#### Metódusok

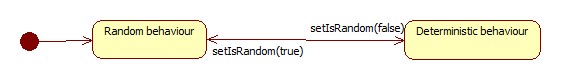
* **+genUIString(): String:** Ez a függvény generálja az adott StarGate fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az épület id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. A generált string a StarGate id-jét, isRandom értékét, works értékét, wasInSunFlare értékét, neighbour StarGate id-jét s az aszteroida id-jét tartalmazza.
* **+getNotifiedAboutSunflare():** Ez a függvény akkor hívódik meg a StarGaten amikor azt napvihar éri, ilyenkor a StarGateben true-ba billen a wasInSunFlare változó, innentől a StarGate elkezd mozogni.
* **+onTurn():** Egy függvény mely mozgatja a StarGatet minden körben, ha az már volt napviharban. Mozgás során a StarGate rálép az aszteroida neighbours listájában legelső helyen szereplő aszteroidára (ha van ilyen, ha nincs akkor nem mozog).
* **+move(destination: Asteroid):** Ebben a függvényben a StarGate átmozog a destination aszteroidára. Ez az átlépés nem történik meg ha a destination aszteroida nem szomszédos a stargate mostani aszteroidájával. Átlépés során lekerül az éppeni aszteroidájáról, s felkerül a cél aszteroidára.
* **+move(destination: StarGate):** Ebben afüggvényben a StarGate rálép a destination StarGate-re s azon keresztül annak párjának aszteroidájára. Ez az átlépés nem történik meg ha ez a StarGate s a destination StarGate nem azonos, illetve ha a destination StarGate működik. Átlépés során lekerül az éppeni aszteroidájáról, s felkerül a cél aszteroidára.

### Sun

#### Felelősség

Ez az osztály felel a játékban található napokért s azoknak feladataiért, pl: azért hogy azokhoz mely aszteroidák tartoznak, mikor törnek ki az egyes napok stb.

#### Állapotdiagramok



(a felső feltétel a balról jobbra menő nyilakhoz tartozik, az alsó pedig a jobbról balra menő nyilakhoz)

#### Ősosztályok

Printable -> Planet -> Sun

#### Interfészek

Controllable

#### Attribútumok

* **-id: String:** Ez az adott sun id-je, ezt arra használjuk fel, hogy a játékos ki tudja választani a megfelelő sunt. Ez most a sunban nincsen használva, de ha a továbbiakban bővül a specifikáció akkor használhatjuk.
* **-nextId: int:** Ez egy statikus tagváltozó mely azt tartja számon, hogy mi legyen a következő létrehozott id-t igénylő objektum id-je. Azért statikus, hogy az id-k egyediségét garantáljuk.
* **-owner: Game:** Egy referencia a Game osztályhoz.
* **-asteroids: Asteroid[0...\*]:** A naphoz tartozó aszteroidák listája, ez a játék elején állítódik be. Fontos hogy ez nem azokat az aszteroidákat tartalmazza melyek közel vannak ehhez a naphoz, hanem az összes naphoz tartozó aszteroidát, legyen az most közel vagy távol a naptól.
* **-isRandom: Boolean:** Egy boolean ami azt jelzi, hogy a Sun random módon, vagy determinisztikusan működik-e (kitörés esély számlálás közben), ez csak a teszteléshez kell.

#### Metódusok

* **+genUIString(): String:** Ez a függvény generálja az adott Sun fontos tulajdonságait tartalmazó stringet (ennek legelső eleme a Sun id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. A string a sun id-jét, isRandom értékét, illetve az összes aszteroidának a genUIString()-je által generált stringeket tartalmazza.
* **+onTurn():** Egy absztrakt függvény mely elvégzi az adott sun egy körben elvégzendő feladatait: kisorsolja hogy ebben a körben ez a nap kitör-e s ha igen akkor a sunFlare() függvényt meghívva elvégzi a napkitörést. Az onTurn működése az isRandom változótól függ. Random esetbel a sun minden körben x eséllyel törik ki, nem random esetben pedig a felhasználó hívja meg a sunFlare()-t amikor szeretné, így az onTurn nem csinál semmit.
* **+sunFlare():** Elvégzi a napkitörés feladatait: meghívja az összes napközelben lévő aszteroidáján a getNotifiedAboutSunflare() függvényt.

### UI

#### Felelősség

Ez az osztály felel a játékossal való kommnikációval, ebbe beletartozik a játék megjelenítése is.

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

* **-game: Game:** Az egyetlen game osztályra egy referencia, ez az osztály fogja végrehajtani az adott utasításokat amiket a játékos bead a UI-nak.

#### Metódusok

* **+displayMessage(s: String):** Ez a függvény kiír egy stringet a játékos számára a konzolra.
* **+execute(s: String):** Ez a függvény egy beolvasott utasítást végrehajttat a game osztállyal. A függvény sok egymásba ágyazott switch case blokkból áll. Az első switch case detektálja az utasítás típusát (a nyelv alapján), az ezután kövekető switch casek pedig az adott utasítás további argumentumait dolgozzák fel pl: melyik id-jű objektumot kell módosítani/létrehozni stb. Miután a parancsot meghatározzuk a game beli megfelelő függvényt kell meghívnunk.

### Uran

#### Felelősség

Ez az osztály az Uran viselkedésér-t felel

#### Ősosztályok

Printable->Material

#### Interfészek

-

#### Attribútumok

* **-id: int:** Ez az adott Material id-je, ezt arra használjuk fel, hogy az Material -t azonosítani tudjuk.
* **-nextId: int:** Ez egy statikus tagváltozó mely azt tartja számon, hogy mi legyen a következő létrehozott id-t igénylő objektum id-je. Azért statikus, hogy az id-k egyediségét garantáljuk.
* **#isRadio: boolean:** Ez a változó mondja meg, hogy radioaktív-e az adott nyersanyag.
* **#Name:String:** Ez tárolja az adott nyersanyag nevét.
* **#canEvaporate:boolean:** Ez a változó mondja meg, hogy elpárolog-e a nyersanyag ha napközelbe kerül.
* **-exposedCounter:int:** Ez tárolja számon, hogy milyen hosszan volt kitéve a napnak az adott nyersanyag.

#### Metódusok

* **+getUIString(): String:** Ez a függvény generálja az adott Material fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az épület id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén.
* **+exposedAndCloseToSun(a:Asteroid):** Ez a függvény hívodik, amikor napközelben felszínen van a nyersanyag. A paraméterül kapott Asteroid aminek a felszínén van. Növeli exposedCounter-t, ha 3 fölé emelkedik, akkor az asteroid amit paraméterül kapott explode függvényét meghívja.