UI

#### Felelősség

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

#### Állapotdiagramok

-

#### 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.

Game

#### Felelősség

Ez a függvény összefogja a játékot, felel a játék megnyeréséért s elvesztéséért, kontrollálja a játék által irányított objektumokat illetve végrehajtja az UI-tól kapott parancsokat.

#### Állapotdiagramok

-

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

* **ui: UI:** Az egyetlen UI amely a játékossal kommunikál, a game ennek visszaadhatja az egyes objektumok státuszát amit aztán az UI kiírhat a játékosnak.
* **-settlers: Settler[0...\*]:** A játékban létező összes Settlert tároló tömb.
* **-controllables: Controllable[0...\*]:** A játékban létező összes Controllablet tároló tömb.

#### Metódusok

* **+removeSettler(s: Settler):** Kiveszi a settlers listából az s Settlert.
* **+removeControllable(c: Controllable):** Kiveszi a controllables listából az s Controllablet.
* **+addSettler(s: Settler):** Hozzáadja a settlers listához az s Settlert.
* **+addControllable(c: Controllable):** Hozzáadja a controllables listához a c Controllablet.
* **+gameWon():** Elvégzi a játék megnyerésekor elvégzendő feladatokat: üzenet a játékosnak az UI-n keresztül, a game-ben tárolt listák kiürítése.
* **+isGameLost(): Boolean:** Leellenőrzi, hogy el van-e veszítve a játék s ennek megfelelő boollal tér vissza. A játék akkor van elveszítve amikor az összes settler meghalt, tahát ezt az ellenőrzést elvégezhetjük a settlers lista hosszának megnézésével.
* **+gameLost():** Elvégzi a játék elvesztésekor elvégzendő feladatokat: üzenet a játékosnak az UI-n keresztül, a game-ben tárolt listák kiürítése.
* **+startTurn():** Ez a függvény akkor hívódik meg amikor a játékos kiadja a lépés a következő körre parancsot, ezután minden controllable autonóm módon (az onTurn() függvényének segítségével) elvégzi az adott körben elvégzendő dolgát. Minden Controllable-re a game osztályban meghívjuk az onTurn függvényét.
* **+listAllSettlers(): String:** Ez a függvény visszaad egy stringet amiben az összes settler minden fontos tulajdonsága található, ezt a stringet a UI írja ki a list all settlers parancs hatására. A függvény végigloopol a settlers listán s meghívja mindegyik tagjára a genUIString függvényt, majd ezeket összefűzve visszaadja.
* **+initGame(isManual: Boolean):** A játék inicializálását végző függvény. Ha az isManual igaz akkor a felhasználó fogja a create s set függvényekkel a játékban található összes objektumot inicializálni. Ha az isManual false akkor ez a függvény autómatikusan létrehozza a játék objektumait, létrehozza a játékban található napokat, ezekhez tartozó aszteroidákat, majd létrehozza a telepeseket s űrélyneket s ezeket ráhelyezi random módon az aszteroidákra. Az aszteroidát ezek után még inicializálja is az initialize() függvényével, ez azért kell mert az aszteroidáknak pl. többféle nyersanyaqg lehet a belsejében, s ezek eldöntéséért felel az initialize() függvény. Mindezek után pedig random mennyiségű szomszédot hozzáadunk minden aszteroidához az összes aszteroida közül.
* **+loadGame(fileName: String):** Ez a függvény a fileName Stringgel meghatározott fileból betölti a játékot. A fileba a genUIString segítségével írtuk ki az objektuainkat, szóval a betöltés a genUIString viszafejtése alapján fog működni.
* **executeCommand(arguments: String):** Ez a függvény végrehajtat egy Contorllable objektummal egy parancsot. Az id-vel specifikált controllable objektumon végrehajtjuk a megfelelő parancsot.
* **startGame():** Ez a függvény elindítja a játékot. Ez jelzi az inicializálás befejeztét. Mivel a settlerek mindig a körben legutoljára kerülnek sorra ezért a játékot egy startTurn() hívással kezdjük.
* **saveGame(fileName: String):** Ez a függvény lementi a játék állását a fileName nevű fileba.
* **list(arguments: String):** Ez a függvény kilistázza az arguments-ben található id-vel sepcifikált settler információit a genUIString függvény segítségével.

Controllable

#### Felelősség

Ez egy olyan interface amit olyan osztályok valósítanak meg miknek van autonóm működése pl: a robotok melyek minden körben mozognak/fúrnak/nem csinálnak semmit.

#### Állapotdiagramok

-

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

-

#### Metódusok

* **+onTurn():** Egy absztrakt függvény mely elvégzi az adott objektum egy körben elvégzendő feladatait.

Printable

#### Felelősség

Ez az absztrakt ősosztály felel az UI-ra kiírandó objektumok releváns tulajdonságait tartalmazó stringek előállításáért, illetve azért, hogy minden objektum (amelyiknek kell) legyen egyedi id-je.

#### Állapotdiagramok

-

#### Ősosztályok

-

#### Interfészek

-

#### Attribútumok

* **-id: String:** Ez az adott objektum id-je, ezt arra használjuk fel, hogy a játékos ki tudja választani a megfelelő objektumot, pl: amikor egy settlerrel fúr, akkor a settler id-jét megadva választhatja ki melyikkel kíván fúrni.
* **-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.

#### Metódusok

* **+getUIString(): String:** Ez a függvény generálja az adott objektum fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az objektum id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. A Printable osztályban ez egy absztrakt függvény.

Building

#### Felelősség

Ez egy absztrakt osztály mely a játékban található épületekért felel. Elpusztítja őket, illetve elvégzi a dolgukat amikor rákerülnek egy aszteroidára.

#### Állapotdiagramok

-

#### Ősosztályok

Printable -> Building

#### Interfészek

ReactsToSunflare

#### Attribútumok

* **-id: String:** Ez az adott épület 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 az épület található.
* **-owner: Game:** Egy referencia a Game osztályhoz.

#### Metódusok

* **+getUIString(): String:** Ez a függvény generálja az adott épület fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az épület id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. A Building osztályban ez egy absztakt függvény.
* **+getNotifiedAboutSunflare():** Ez a függvény akkor hívódik meg az épületen amikor azt napvihar éri, ilyenkor az épület reagál arra ha kell neki.
* **+onPlace(a: Asteroid):** Ez a függvény végrehajtja az épület feladatait abban a pillanatban amikor ráhelyezzük az aszteroidára.
* **+destroy():** Ez a függvény végrehajtja az épület feladatait amikor az elpusztul.

Base

#### Felelősség

Ez az osztály felel a bázis épületekért s azoknak feladataiért pl: amikor ráhelyezzük őket egy aszeroidára.

#### Állapotdiagramok

-

#### Ősosztályok

Printable -> Building -> Base

#### Interfészek

ReactsToSunflare

#### Attribútumok

* **-id: String:** Ez az adott bázis id-je, ezt arra használjuk fel, hogy a játékos ki tudja választani a megfelelő bázist. Ez most a bázisban 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.
* **-asteroid: Asteroid:** Az aszteroida amin a bázis található.
* **-owner: Game:** Egy referencia a Game osztályhoz, ezen keresztül fog a bázis jelezni a játéknak, ha le lett helyezve s ezzel megnyerte a játékos a játékot.

#### Metódusok

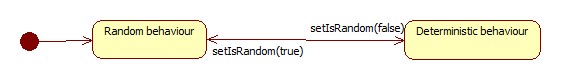
* **+getUIString(): String:** Ez a függvény generálja az adott bázis fontos tulajdonságait tartalmazó stringet (ennek legelső eleme a bázis id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. Ez most a bázisban nem lesz használva, de ha a továbbiakban bővül a specifikáció akkor használhatjuk.
* **+getNotifiedAboutSunflare():** Ez a függvény akkor hívódik meg a bázison amikor azt napvihar éri. Ez most a bázisban nem lesz használva, de ha a továbbiakban bővül a specifikáció akkor használhatjuk.
* **+onPlace(a: Asteroid):** Ez a függvény végrehajtja a bázis feladatait abban a pillanatban amikor ráhelyezzük az aszteroidára, ez az hogy szól a game osztálynak, hogy meg lett nyerve a játék. Mindezt úgy érjük el, hogy a game osztályban meghívjuk a gameWon() függvényt.
* **+destroy():** Ez a függvény végrehajtja a bázis feladatait amikor az elpusztul. Ez most a bázisban nem lesz használva, de ha a továbbiakban bővül a specifikáció akkor használhatjuk.

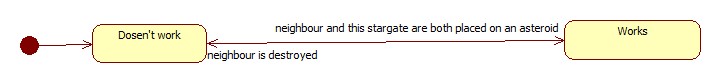
StarGate

#### 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

* **+getUIString(): 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.
* **+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.
* **+onPlace(a: Asteroid):** Ez a függvény végrehajtja a StarGate feladatait abban a pillanatban amikor ráhelyezzük az aszteroidára, csekkolja hogy a párja merre található, s ennek függvényében beállítja a works-öt true-ra vagy false-ra. Ha a párja már nem létezik akkor a works false lesz, ha a párja még az inventoryban van akkor is false lesz a works. Ha a párja egy aszteroidán van (ezt azzal ellenőrizzük, hogy a getNeighbourAsteroid() nem null-ot returnol) akkor ez a StarGate s a párja is works állapotba kerül (works = true).
* **+destroy():** Ez a függvény végrehajtja a StarGate feladatait amikor az elpusztul: ha a szomszédja le volt már helyezve akkor a szomszédjának a works változóját false-ba állítja.
* **+getNeighbourAsteroid(): Asteroid:** Ez a függvény visszaadja a StarGate párjának aszteroidáját (ha van neki, ha nincs null-al tér vissza).
* **+getWorks(): Boolean:** Ez a függvény visszaadja a works változó értékét.
* **+getAsteroid(): Asteroid:** Ez a függvény visszaadja az asteroid változó értékét.
* **+setAsteroid(a: Asteroid):** Ez a függvény beállítja az aszeroid tagváltozó értékét a-ra.
* **+setWorks(b: Boolean):** Ez a függvény beállítja a works tagváltozó értékét b-re.
* **+setNeighbour(sg: StarGate):** Ez a függvény beállítja a sg tagváltozó értékét sg-re.

Planet

#### Felelősség

Ez az absztrakt ősosztály felel a „bolygszerű” objektumokért a játékban, ebben a beadásban erre nem lenne szükség, de később grafikus megjelenítés esetén ezeket az objektumokat hasonlóképp kell majd megjeleníteni.

#### Állapotdiagramok

-

#### Ősosztályok

Printable -> Planet

#### Interfészek

Controllable

#### Attribútumok

* **-id: String:** Ez az adott planet id-je, ezt arra használjuk fel, hogy a játékos ki tudja választani a megfelelő planetet, pl: amikor egy settlerrel mozog, akkor az aszteroida 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.
* **-owner: Game:** Egy referencia a Game osztályhoz.

#### Metódusok

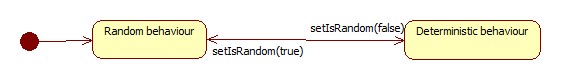
* **+getUIString(): String:** Ez a függvény generálja az adott planet fontos tulajdonságait tartalmazó stringet (ennek legelső eleme a planet id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. A Planet osztályban ez egy absztrakt függvény.
* **+onTurn():** Egy absztrakt függvény mely elvégzi az adott planet egy körben elvégzendő feladatait.

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

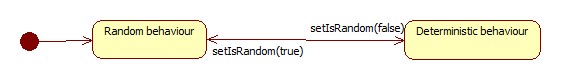
* **+getUIString(): String:** Ez a függvény generálja az adott Sun fontos tulajdonságait tartalmazó stringet (ennek legelső eleme a Sun id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén. Ez most a Sun-ban nem lesz használva, de ha a továbbiakban bővül a specifikáció akkor használhatjuk.
* **+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.

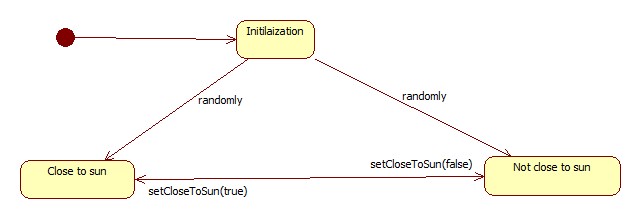
Asteroid

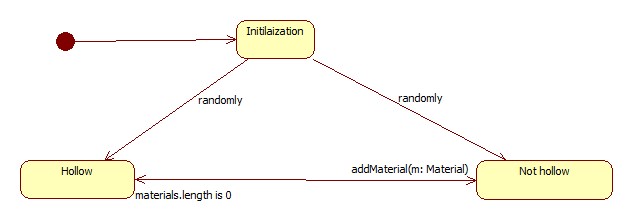
#### Felelősség

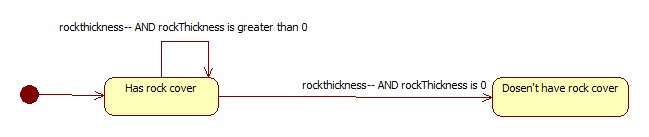
Ez az osztály felel a játékban található aszteroidák feladataiért, pl: amikor napkitörés van ez az osztály hívja meg a rajta található entitásokon s épületeken a getNotifiedAboutSunflare() függvényt, ezzel jelezve nekik hogy épp napkitörés van.

#### Állapotdiagramok









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

#### Ősosztályok

Printable -> Planet -> Asteroid.

#### Interfészek

Controllable, ReactsToSunflare

#### Attribútumok

* **-id: String:** Ez az adott aszteroida id-je, ezt arra használjuk fel, hogy a játékos ki tudja választani a megfelelő aszteroidát.
* **-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.
* **-neighbours: Asteroid[0...\*]:** Ez a lista tárolja az aszteroidával „fizikailag” (nem StarGate-n keresztül) szomszédos aszteroidákat.
* **-materials: Material:** Ez a lista tárolja az aszteroidában található materialokat, ha ez a lista null akkor az aszteroida üreges.
* **-settlers: Settler:** Ez a lista tárolja az aszteroidán található settlereket.
* **-nonPlayers: NonPlayer[0...\*]:** Ez a lista tárolja az aszteroidán található nonPlayereket.
* **-buildings: Building[0...\*]:** Ez a lista tárolja az aszteroidán található épületeket.
* **-rockThickness: int:** Ez az int tárolja az aszteroida kérgének vastagságát.
* **-closeToSun: Boolean:** Ez a bool azt jelzi, hogy az aszteroida éppen közel van e a napjához.
* **-isRandom: Boolean:** Ez a bool azt jelzi, hogy az aszteroida random módon, vagy determinisztikusan működik-e (napközelbe kerülés esély számlálás közben), ez csak a teszteléshez kell.
* **-capacity: int:** Ez az int azt jelzi, hogy az aszteroidába mennyi darab nyersanyag fér bele.

#### Metódusok

* **+getUIString(): String:** Ez a függvény generálja az adott Asteroid fontos tulajdonságait tartalmazó stringet (ennek legelső eleme az Asteroid id-je) amit a UI megkap s a játékosnak kiír megfelelő parancs esetén.
* **+onTurn():** Egy absztrakt függvény mely elvégzi az adott aszteroida egy körben elvégzendő feladatait: kisorsolja hogy ebben a körben ez az aszteroida változtatja-e a napközelségét (ha napközelben van akkor kikerül-e onnan s ha nincs napközelbe akkor bekerül-e oda) s ha igen akkor beállítja a closeToSun változót ennek megfelelően. Ezen kívül, ha egy aszteroida kérge 0 vastag akkor a benne található matrialon (ha van) meghívja az exposedAndCloseToSun() fgvt.
* **+getNotifiedAboutSunflare():** Ez a függvény akkor hívódik meg az aszteroidán amikor azt napvihar éri, ilyenkor az aszterida meghívja a getNotifiedAboutSunflare() fgvt az összes épületen, telepesen s nonPlayeren ami rajta van.
* **+addSettler(s: Settler):** Hozzáadja a settlers listához az s Settlert.
* **+addNonPlayer(np: NonPlayer):** Hozzáadja a nonPlayers listához az np NonPlayert.
* **+removeSettler(s: Settler):** Kivesszük a settlers listából az s Settlert.
* **+removeNonPlayer(np: NonPlayer):** Kivesszük a nonPlayers listából az np NonPlayert.
* **+explode():** Ez a függvény akkor hívódik meg amikor az aszteroida felrobban, ilyenkor meghívja a rajta található entityken az asteroidExploded() függvényt, illetve meghívja a rajta található épületeken a destroy() függvényt.
* **+drilling():** Ez a függvény akkor hívódik meg amikor valami fúr az aszteroidán, az aszteroida kérge csökken egyel (ha tud), illetve ha épp 0-ra csökken akkor meghívja az aszteroidában talákhat nyersanyagok egyikén az exposedAndCloseToSun() fgvt.
* **+removeMaterial(): Material:** Ez a függvény eltávolít egy nyersanyagot az aszteroida belsejéból (ha tud) s azt visszaadja.
* **+addMaterial(m: Material): Boolean:** Ez a függvény megpróbálja belehelyezni az aszteroida belsejébe az m nyersanyagot, a sikerességgel azonos boollal tér vissza. A nyersanyag visszahelyezése csak akkor sikertelen, ha az aszteroida kérge nagyobb mint nulla, vagy hogyha az aszteroida elérte a kapacitását (materials.length() == capacity)
* **+removeNeighbour(a: Asteroid):** Kivesszük a neighbours listából az a Asteroidot.
* **+addNeighbour(a: Asteroid):** Hozzáadja a neighbours listához az a Asteroidot.
* **+removeBuilding(b: Building):** Kivesszük a buildings listából a b Buildinget.
* **+addBuilding(b: Building):** Hozzáadjuk a buildings listához a b Buildinget.
* **+isHollow(): Boolean:** Ez a függvény leellenőrzi, hogy az aszteroida belseje üreges-e (üres-e a Materials lista) s ennek megfelelő boollal tér vissza.
* **+evaporateMaterial():** Ez a függvény elpárologtatja a belsejében található nyersanyagokat. Tehát kiveszi őket a materials listából.
* **+initialize():** Ez a függvény inicalizálja az aszteroidát. Kisorsolja, hogy az aszteroida belsejében milyen nyersanyag legyen (esetleg üres legyen-e), illetve hogy ebből a nyersanyagból mennyi legyen benne. Ezen kívül azt is kisorsolja, hogy az aszteroida épp napközelben van-e.