Reservoir:

* Make use of hitpoints from SWEntity, set hitpoints to 40 when constructing reservoir
* Override the takeDamage method from SWEntity, make it so that when taking damage and hp is below 20, change the descriptions and symbol. When hp is 0 or below, change the descriptions and symbol to meet the specifications
* Reservoirs can take damage but cannot be attacked (in our design). To implement this, make a new “Damageable” affordance and put it in reservoir
* Damageable extends SWAffordance implements SWActionInterface
* Damageable entities can only be damaged by items with special features (will relate to grenade)
* So it cannot be attacked by actors directly
* Change Attack affordance to extend Damageable instead of SWAffordance
* This change is made because not all Damageable entities can be attacked (reservoir)
* However, all Attackable entities are also Damageable
* So Attack will extend from Damageable
* Advantage: Good way to do area damage (will relate to grenade) since area damage will damage all entities that can be damaged/attacked.
* Disadvantage: Since actors cannot damage Damageable entities directly, the canDo method will return false always. This will take some processing time when checking the available actions for each actor.

Grenade:

* Create new entity Grenade
* Create new “Throwable” capability
* Create new Throw affordance and put it in Grenade
* Throw affordance will be an affordance that specifies if an entity can be thrown into a location and deal area damage
* The location for the throw action is the location of the actor, and then it will evaluate the locations around it and deal damage accordingly
* When initializing SWWorld, initialize multiple grenades at different locations
* Grenade will have a new attribute:
* int radius: the spread of the area damage
* Grenade will set its hitpoints to 20
* Grenade will have the Throwable capability
* This new capability will be checked by the canDo method of Throw, and canDo will return true only if the itemCarried of actor has the Throwable capability
* When act method of Throw is called, it will get hitpoints and radius of grenade, then gets the location of the actor calling the Throw.
* It will then start checking the locations around actor location in order of the radius
* So first it will start with radius 0 which means the location itself
* Then radius 1 will be the neighbor locations of actor location
* Then radius 2 will be the neighbor locations of those locations checked by radius 1
* And so on until it reaches the radius of the item
* For each of the locations checked, it will get all entities on the location
* For each entity in the location, it will check each entity if they have the Damageable affordance
* If they are Damageable, deal damage to the entity.
* Damage calculation will be item hitpoints/2r where r is the radius
* After every locations are checked, remove the item carried by actor for good by setting it to null
* The grenade example, it will check the actor location and will damage every entity except for actor in that location by 20/20 = 20
* Then it checks locations that can be reached in one step from the location, and will damage every entity except for actor in that location by 20/21 = 10
* And then it checks locations that can be reached in two steps from the location, and will damage every entity except for actor in that location by 20/22 = 5
* After that, setItemCarried by actor to null
* Since Droid is also Damageable, add Damageable affordance to Droid.
* Advantage: Making throw to consider radius will open up future developments for items that can be thrown/deal area damage
* Disadvantage: The name of the affordance “Throw” can be misleading. Its better to name this affordance “Explosion” because of the nature of dealing area damage, however we’re sticking to this name because making a new Explosion affordance and connecting it to Throw would be redundant.
* Future development: If there’s more entities that can be thrown, consider creating a new abstract class “Projectile” that will have the Throwable capability and radius attribute instead of adding those one by one to each Throwable entity.