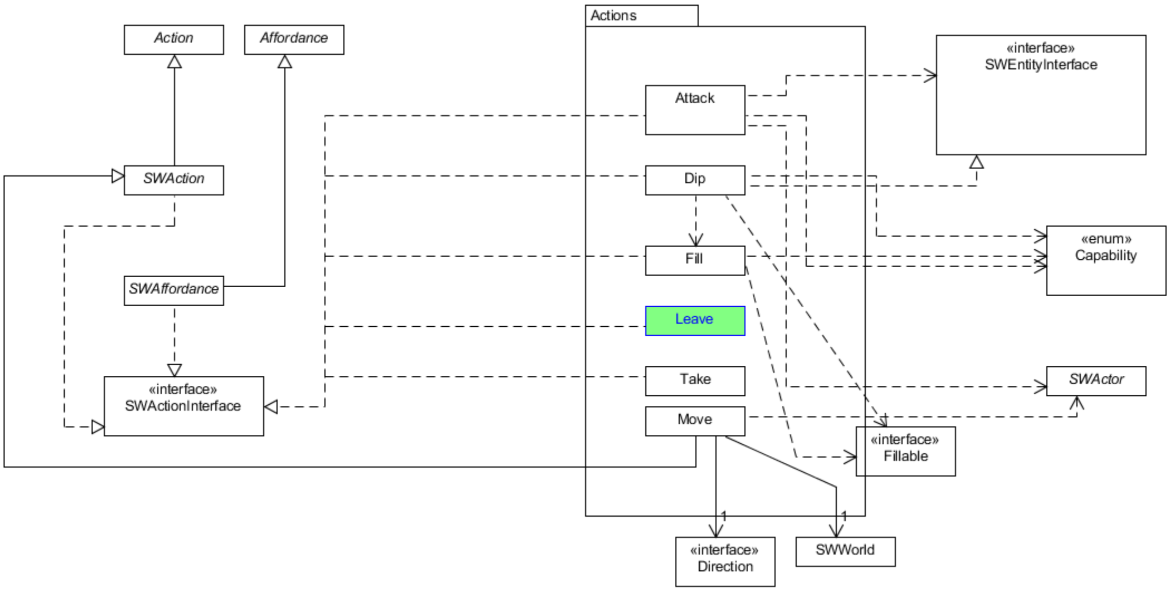
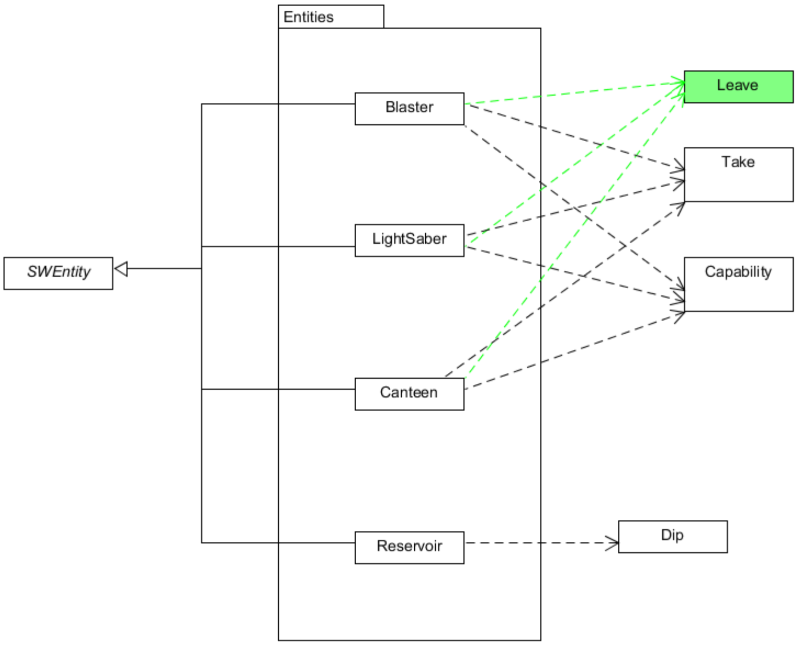
Leave Rationale

Main UMLs





What classes exist in your extended system?

The only class introduced in the extended system will be **Leave**.

*What is role and responsibility of **Leave**?*

The role of **Leave** is to allow "an actor to put down the object they are currently carrying".

Following the occurrence of **Leave**,

- "the actor should be holding nothing
- the item should be in the location of the actor when it was done
- the item should be able to be picked up again"

*How **Leave** relates to and interacts with the existing system.*

Leave will behave similarly to the **Take**. It will extend the SWAffordance class, so as to avoid repeated code, and be linked to the take affordance (or perhaps an affordance of its own TBA). Entities that can be "left" include the Blaster, LightSabre, and Canteen, just as they can be "taken".

Leave will use the EntityInterface methods to deliver the functionality of removing the entity from possession and placing the entity on the map.

How the (existing and new) classes will interact to deliver the required functionality.

-Allow an actor to put down the object they are currently carrying -After this is done, the actor should be holding nothing	This functionality will be completed within the **Leave** class. Roughly, it will use the `a.setItemCarried(x) (where x is none/null/whatever the java term is)` Premise of the **Take** class.
-the item should be in the location of the actor when it was done	Done using setLocation() on the item wherever the SWActor is. This Action will be achieved using the EntityInterface methods.
-and the item should be able to be picked up again	The take affordance will be re-instated after all the other functionality of **Leave** is completed.