**Further inspired by concepts utilized in Neo Scavenger.**

What functions does the Item class serve?

A lot of different functions. It is basically everything. From small debris on the floor to weapons of mass destruction.

* Miscellaneous items (Random items of some value).
* Clothing
* Usable objects which effect some sort of change (food, medicines, e.t.c.)
* Weapons
* Blunt instruments, firearms, projectile throwers, e.t.c.

Should all items have value? This seems the most straightforward way of allowing for things like trade and diplomacy.

**Point 2:** I’m also concerned with limiting the uses of items. For example, it should be possible to both eat a steak. As well as to wave said steak in a threatening manner in order to ward off predators. (An effective technique, I’m sure.)

**Technical Considerations:**

* Off hand, it seems like it may be desirable to have multiple subclasses of the Item superclass. I suppose I could have something like a “type” attribute in a single class, but that seems highly clumsy, especially given how varied the actual items are.
* In terms of **point 2** though, perhaps it would be better to simply have a single Item class. Clearly some functions would make absolutely zero sense (For example, a steak could not be reasonably worn by a person.)
* More generally though, I’m deeply concerned with creating a sensible network of items. I want …
* **Go with the Dwarf Fortress route. There’s a set of abstractized properties, and descriptions (and values) are generated/calculated based on those properties.**
* **This’ll remove the need for me to try to construct an uber-complex network of items (at least, not overwhelmingly complex), and allow individual entities some control over creating new objects.**
* What’s the best way to implement this? Are subclasses actually the way to go? I will actually differentiate between weapon items and consumable items. As such, I will subclass them. I think this will be the best way to conceptually and practically isolate/imagine the concepts.