Inventory

Nna, két irányból lehet megközelíteni ezt, xml és json.
(Az SQL azért nem jó mert (tudomásom szerint) nem képes 255 karakternél hosszabb string-et tárolni, legalábbis az Access nem tud)
A tiszta xml mentésekhez (mert az biztos, h nem lesz 1345156 tábla egyszerre nyitva) túl lassú az xml decoder.

Marad a json.

Ez azért fontos, mert így nem lehet metatable-t közvetlen betölteni, tehát ha oop-t akarunk akkor a mt-t újra és újra be kell állítani, ezt érdemes mérlegelni.
Ezenkivül van egy mélység aminél tovább nem lehet menni, bár ez valószínűleg nem lesz túllépve.

Tehát maga az inventory:

Az inventory-k eléggé nagy skálán mozognak, mivel nem mindnek van minden funkcióra szüksége.
Például gyorsíthatjuk az item keresést vagy ki is kapcsolhatjuk egy items mezővel, ezenkívül különböző event-eket lehetne még használni, ezeket a fejlseztés későbbi fázisára hagyjuk.

Egy inventory akármilyen element által birtokolható és használható, így inventory használattal működhetnek pl a járművek.

Egy inv. tartalmazhat 3 típusú item-et:

- big: formája van, nem stack-elhető, mátrixban tárolt
- stack: mint a big, de csak 1 mezőt foglal, stack-elhtő (De tényleg, durva mi?)
- med: nincs formája de van mérete, táblában van
- small: mint a med, de méret nélkül

még megjelenhetnek folyadékok is, amik gyakorlatilag dinamikus formák

Egy inventory (ha nem játékos használja) csak addig van nyitva amíg használatban van, utána mentjük egy json fájlba ami a nevével egyezik meg (alapértelmezésben).

Fontolóra lehet még venni lokális és globális cache-eket és külön garbagecollector-t a fájlokhoz.

Update-ek és fizika:

Egy mezei inv. nem fogja őket használni, de a játékosok számára fontos lesz a hátizsákjuk fizikája.
Mert nehogy már túl jó legyen nekik >:)

A fizika event-ekkel műxik, de ezek nem feltétlen kell h megfeleljenek az MTA event system-nek, bár arra lesznek alapozva.

A fizika egy furcsa naplózás alapú izé lesz, egyelőre ennyi tán elég is. Maradjunk annyiban hogy gyűlnek az event-ek amíg valami meg nem nyitja az inv-et (és nem kapcsolja be a fizikát, mert pölö ha a keresők is bekapcsolnák akkor aztán lenne fagyás) nem történik velük semmi, de aztán meg hajjaj.

Több az item-ekről

Egy inv-ben ugyebár minél kevesebb adatot kell minél gyorsabban tárolni, ezért megkülönböztetünk dinamikus és statikus item-eket.

A dinamikus table-ként jelenik meg és az első mezőjével hivatkozik a megfelelő globális adatra (tárolási mód kérdése még nyitott) és csak a saját adatait tárolja lokálisan.
Ilyen például egy telefon töltöttsége, de a darabszám nem, mert a stack jobban intézi azt.
A statikust nem lehet változtatni, tehát csak hivatkozik az eredetére.

Minden item action call (ja, lesznek action-ök is, tehát figyelembe kell venni az integrációt is) küld egy self értéket. (majdnem value-t írtam :))
A dinamikusok küldik még saját adataikat is.

Lehet még egy harmadik typus is: a full-dinamikus, ami tiszta table.

Funkciókat nem tárolhat item (legalábbis a lua default json modulja nem támogatja és szerintem az MTA toJSON-ja se, mert a Java nem tud semmit se kezdeni egy Lua funkcióval), de vonatkozhat rájuk ha van rá igény.