* Make classes like component extend from Serializable (a json class that enables jsonEncode) in order to make the whole system easily serializable to a file.

Also make entity manager serializable, engine serializable, etc..

* ~~@EntityManager: possible including method that returns entities that have n components. Used when a system needs for example all entities that have velocity component and also position component (Movement system).~~

~~def getEntitiesHavingComponents(self, \*args):~~

~~then do a set for every component of those entities that have that component,~~

~~finally do an intersection between those sets to get the entities that have them all, and return them.~~

* @EntityManager: frozen status? That would allow more than one entity manager.

Maybe frozen status in the engine? (implement freeze(), unfreeze() and add a Boolean is\_frozen + property)

* Think about metaEntities: [link](https://github.com/adamgit/Entity-System-RDBMS-Beta--Java-/blob/master/EntitySystemJava/src/com/wikidot/entitysystems/rdbmsbeta/MetaEntity.java)