General organization

src/main/java – holds all java files for mod

init –

items – holds classes describing each item, special items get their own class

ItemBase – extends Item and implements IHasModel

* Basic item class with constructor that sets UnlocalizedName, RegistryName, CreativeTab, and adds to ITEMS list
* registerModels registers the model with Minecraft: Main.proxy.registerItemRenderer(this, 0, “inventory”)

proxy – holds Client and Common Proxy classes with registerItemRenderer, only client proxy actually does anything this this method

util – utility package

handler – package to hold all classes in charge on handling, i.e. registrations

* onItemRegister registers all mod items with Minecraft
* onModelRegister calls registerModels of all items and blocks in ITEMS and BLOCKS

IHasModel – interface that declares registerModels method

Reference is a data holding class that defines MOD\_ID, NAME, VERSION, ACCEPTED\_VERSIONS, CLIENT\_PROXY\_CLASS, and COMMON\_PROXY\_CLASS

Main – main class of mod, not quite sure exactly what is supposed to be here

* creates a public instance and proxy variables
* also contains three event handlers: preInit, init, postInit

src/main/resources

assets.modid

lang – holds language files with format: item.<itemName>.name=<nameToDisplay>

models.item – holds model jsons for items added by mod. A new file for each item is needed including blocks

{

"parent": "item/generated",

"textures": {

"layer0": "<MOD\_ID>:items/<itemName>"

}

}

Textures.items – holds all png files for items in mod, must be named as <itemName>.png and be 16x16 pixels

Quick Instructionals

Adding an Item:

1. add item instantiation to ModItems in init
   1. public static final Item <ITEMNAME> = new <itemClass>(“<itemName>”);
      1. ITEMNAME: all caps itemName
      2. itemClass: class that inherits Item, ItemBase if no properties need to be changed
      3. itemName: same item name used elsewhere, needs to be unique among items
2. OPTIONAL: create class in items package to set specific properties
3. Add line to each language file
4. Add item json to models.item with itemName.json format as described above
5. Add item png to textures.items with itemName.png format as described above