To define a new bag, the mod expects a certain list of commands to define the different properties of a bag. The precise order of the sub commands doesn't matter, however the syntax and required commands must be followed exactly or the bag won't be created properly (or at all). The default file demonstrates every command usage (within some limitations).

**Basic Config Rules:**

All commands recognized by the config start with dollar sign ($), so $STARTBAG is the base command to start defining a new bag. Commands do not need to be all uppercase, but it makes the config a bit easier to read. Similarly, the spaces creating tabs are not required, but are added for readability.

Many commands have parameters to further define the command's properties. These are separated by a colon (:), so the bag start command above with all necessary parameters becomes $STARTBAG:<unlocalized bag name>:<bagID>

Some commands have parameters that are multiple strings (such as the bag text commands), for these, format the string portion as it should appear in game, except that any colons (:) should be prefaced by a dollar sign ($) to exclude it from the command separator.

**Specific Config Commands and Syntax:**

$STARTBAG:<unlocalized bag name>:<bagID> - This command is used to begin defining a new loot bag. The unlocalized bag name is the name used internally to determine the shown name. This will need to be localized in the .lang file or the bag will appear as it's unlocalized name in game, which contains elements that aren't in the parameter. The bagID is what is used to ensure that the bags stay consistent through saving and loading. The bagID needs to be distinct and unused, which shouldn't be too difficult as the config is the only place those IDs can be assigned. If a bag is attempted to be opened before the previous bag is properly closed, all the previous bag information is discarded and an error is thrown.

$BAGCOLOR:<main color>:<secondary color> - This command will set the bag texture to a default two-toned bag instead of a pre-defined one, colorizing the body to the main color and the neck to the secondary color. This has the benefit of bag customization without the need of resource pack creation. The colors are RGB colors separated by vertical bars (R|G|B). The default Worn Out bag shows an example of this command.

$ISSECRET:<boolean> - This command is used to indicate if the bag shows up in creative inventory lists. Does nothing if the general config option ShowSecretBags is enabled.

$BAGTEXTCOLOR:<Color> - This command determines the color of the bag text in its name. It supports all the standard minecraft text formatting (listed later).

$BAGTEXTUNOPENED:<Color>:<Text> - This command determines the text shown in the tooltip when the bag has not been opened. This command supports multiple uses, each one adds a new line. Neglecting the color parameter uses the default text color. Using $DROPCHANCES in the text parameter uses that bag's drop chances instead of a set text.

$BAGTEXTOPENED:<Color>:<Text> - This command determines the text shown in the tooltip when the bag has been opened. This command supports multiple uses, each one adds a new line. Neglecting the color parameter uses the default text color. Using $DROPCHANCES in the text parameter uses that bag's drop chances instead of a set text.

$BAGTEXTSHIFT:<Color>:<Text> - This command determines the text shown in the tooltip when left shift is pressed. This command supports multiple uses, each one adds a new line. Neglecting the color parameter uses the default text color. Using $DROPCHANCES in the text parameter uses that bag's drop chances instead of a set text.

$WEIGHT:<Integer> - This command sets the weight of the bag, which is used to determine crafting ratios and how much loot value is needed to fabricate the bag in the recycler. If the bag is not crafted from or into anything and not used in the recycler the value does not matter.

$CRAFTEDFROM:<Bag Name> - This command is used to indicate which bag this bag is crafted from. The bag name is the name used in the bag start command. Using $NULL will disable crafting into this bag.

$PASSIVESPAWNWEIGHT:<Integer> - This is the weight for the bag to spawn from a "passive" mob, i.e. one that implements EntityAnimal or IAnimals. A value of 0 means this bag cannot spawn from this type of mob.

$PLAYERSPAWNWEIGHT:<Integer> - This is the weight for the bag to spawn from a player, i.e. one that implements EntityPlayer. A value of 0 means this bag cannot spawn from players.

$BOSSSPAWNWEIGHT:<Integer> - This is the weight for the bag to spawn from a boss, i.e. ones that implements IBossDisplayData. A value of 0 means this bag cannot spawn from such mobs.

$MOBSPAWNWEIGHT:<Integer> - This is the weight for the bag to spawn from enemy mobs, i.e. ones that implement EntityMob or IMob. A value of 0 means this bag cannot spawn from such mobs.

$USEGENERALLOOTSOURCES:<Boolean> - This command determines if the bag uses the general loot table as an initial loot source.

$MAXIMUMITEMS:<Integer> - This command determines the maximum number of items the bag can hold. This value must be at minimum 1, at maximum 5, and greater than or equal to the minimum item count.

$MINIMUMITEMS:<Integer> - This command determines the minimum number of items the bag can hold. This value must be at minimum 1, at maximum 5, and less than or equal to the maximum item count.

$MAXIMUMGENERALLOOTWEIGHT:<Integer> - This command determines the maximum weight of general loot that can show up in the bag. A value of -1 leaves this field unbounded, any positive value must be greater than or equal to the minimum general loot weight value.

$MINIMUMGENERALLOOTWEIGHT:<Integer> - This command determines the minimum weight of general loot that can show up in the bag. A value of -1 leaves this field unbounded, any positive value must be less than or equal to the maximum general loot weight value.

$PREVENTITEMREPEATS:<Text Code> - This command determines if items are permitted to repeat in a single bag. The text code must be one of these options:

* none - Repeats are permitted without limit.
* damage - Repeats are blocked for same items with different damage values
* item - Repeats are blocked for the same item class

$STARTWHITELIST - This command begins the whitelist subgroup. It must have a list of whitelist entries followed by the $ENDWHITELIST command. The whitelist entries need the following format: <mod id>:<item name>:<damage range>:<minimum stack size>:<maximum stack size>:<weight>:<NBT bytes>. The mod id and item name can be obtained using NEI's item dump function. The damage range can be a single integer, for a single damage value, or a collection of integers, &s, and #s. A # indicates a skip between two numbers, and an & indicates inclusion between two numbers. For example, 0&5#8&10 uses the damage values 0, 1, 2, 3, 4, 5, 8, 9, 10. The minimum and maximum stack sizes must be within the limits of the item, typically 1 and 64. The weight is independent of the general minimum and maximum weights detailed earlier and is not scaled to those values. The NBT bytes is an optional (leave the last colon off as well if so) set of data to set the NBT data of the itemstack. This byte list can be obtained through the lootbags\_pullnbt command in game, which will create a dump file of the information of the currently held item. This command can be skipped, it will be treated as an empty whitelist.

$STARTBLACKLIST - This command begins the blacklist subgroup. It must have a list of blacklist entries followed by the $ENDBLACKLIST command. The blacklist entries can be of two forms, <mod id> or <mod id>:<item name>:<damage range>. The mod id only version will blacklist that entire mod from the bag. This command can be skipped, it will be treated as an empty blacklist.

$EXCLUDEENTITIES:<Boolean> - This command determines if the entity list (if present) acts as a whitelist or as a blacklist, with true acting as a whitelist and false as a blacklist.

$STARTENTITYLIST - This command begins the entity list subgroup. It must have a list of entity entries followed by a $ENDENTITYLIST command. An entity entry is comprised of the command $VISIBLENAME:<name as it appears in game> OR $INTERNALNAME:<name as it appears in the internal registry>. For visible name entries, this is used in cases where a specific name is desired, for instance a player name, and will trigger on any entity who is visibly named that name. For internal name entries, every mob registered in the game has an internal name. This is usually the same as the entity's English localized name, though it is not guaranteed to be the case for modded mobs. This command can be skipped, it will be treated as an empty entity list.