

Custom Tools

Posted on June 24, 2018

Now that we've added basic blocks and items, let's move on to something more interesting: custom tools. You'll need five new classes in your item package, one for each of the tools. Name them `ItemTutorialAxe`, `ItemTutorialPickaxe`, `ItemTutorialHoe`, `ItemTutorialShovel`, and `ItemTutorialSword`. Each one will extend the vanilla class of its type (e.g. `ItemAxe`). Note that the vanilla shovel class is called `ItemSpade`, not `ItemShovel`.

```
package com.cubicoder.tutorial.item;

import com.cubicoder.tutorial.TutorialMod;

import net.minecraft.item.ItemPickaxe;

public class ItemTutorialPickaxe extends ItemPickaxe {

    public ItemTutorialPickaxe(ToolMaterial material) {
        super(material);
        setCreativeTab(TutorialMod.TUTORIAL_TAB);
    }

}
```

This is the code for the `ItemTutorialPickaxe`. Your other classes will be the same except in name (excluding `ItemTutorialAxe`). Everything in the constructor should be familiar from the basic item, other than the `ToolMaterial`. This material defines things like durability and harvest speed, and we'll create one for our tools soon. But first, we need to address the `ItemTutorialAxe` class.

Your `ItemTutorialAxe` class is not going to look the same as the others, because axes define their attack speed and damage within the class, rather than from the tool material. We'll make two constructors for the `ItemTutorialAxe`, one that allows us to set our own attack speed and damage and another that sets these values to the same as iron (for convenience).

```
package com.cubicoder.tutorial.item;

import com.cubicoder.tutorial.TutorialMod;

import net.minecraft.item.ItemAxe;

public class ItemTutorialAxe extends ItemAxe {

    public ItemTutorialAxe(ToolMaterial material, float damage, float speed) {
        super(material, damage, speed);
        setCreativeTab(TutorialMod.TUTORIAL_TAB);
    }

    public ItemTutorialAxe(ToolMaterial material, String unlocalizedName, float damage, float speed) {
        this(material, 8.0F, -3.1F);
    }

}
```

Now create a class named `TutorialMaterials` in the package `com.cubicoder.tutorial.material`. In this class, we use Forge's `EnumHelper` to create a new tool material.

```
package com.cubicoder.tutorial.material;

import com.cubicoder.tutorial.TutorialMod;

import net.minecraft.item.Item.ToolMaterial;
import net.minecraftforge.common.util.EnumHelper;

public class TutorialMaterials {

    public static final ToolMaterial TUTORIAL_TOOL = EnumHelper.addToolMaterial(TutorialMod.MOD_ID, "TutorialTool", 3, 1000, 1000);
}
```

Name : The name of the enum value that Forge will be adding. **Make sure to prefix this with your mod id!** Harvest level : The harvest level determines which blocks your tools can harvest. 0 is wood and gold level, 1 is stone, 2 is iron, and 3 is diamond. You can set this level higher than three, and if you or another mod adds a block with a higher harvest level than three, you'll be able to mine it with your tools. As long as your tools' harvest level is greater than or equal to the harvest level of the block you're

mining, it can be harvested. Max uses : The durability of your tools. Efficiency : How fast your tools mine/dig/whatever (not attack speed, though). Damage : The base amount of damage your tools deal. Different tool types increment this value certain amounts; for example, a sword will have a damage value three points higher than the value you set here. Enchantability : How well your item can be enchanted.

Let's fill in these values so that our material is somewhere between iron and diamond.



```
public static final ToolMaterial TUTORIAL_TOOL = EnumHelper.addToolMaterial
```

Now you just need to register the items like you would any other item, and add entries for them in your lang file. In your models, however, the `parent` field should be `item/handheld`, not `item/generated`:

```
{
  "parent": "item/handheld",
  "textures": {
    "layer0": "tutorialmod:items/tutorial_axe"
  }
}
```

Run the game to see your wonderful tools!



 (</feed.xml>)  (<https://github.com/cubicoder>)

cubicoder • 2019 • Home (<https://cubicoder.github.io>)

Theme by beautiful-jekyll (<https://deanattali.com/beautiful-jekyll/>)