First Block

Posted on June 20, 2018

Now that we've added a basic item, let's add a basic block. The process for this is very similar to adding an item. The only difference is that we actually have to add a block and an item. This is because a block in Minecraft exists in two places: in the world, as a `Block`, and in the inventory as an `ItemBlock`, which is an item that corresponds to a `Block`. Let's get started! First, create a new class named `BlockFirstBlock` in the package `com.cubicoder.tutorial.block`.

```
package com.cubicoder.tutorial.block;

import com.cubicoder.tutorial.TutorialMod;

import net.minecraft.block.Block;
import net.minecraft.block.SoundType;
import net.minecraft.block.material.Material;
import net.minecraft.creativetab.CreativeTabs;

public class BlockFirstBlock extends Block {

        public BlockFirstBlock() {
                super(Material.STONE);
                setCreativeTab(CreativeTabs.MISC);
                setSoundType(SoundType.STONE);
        }

}
```

A `Block` contains a `Material`, which defines things like whether the block is flammable, or if can be seen through. Minecraft already defines a number of materials which are good enough for most purposes, but you could make a custom material if you wanted to. Right now, we'll just stick to the vanilla ones.

The `BlockFirstBlock` constructor is just like the one in `ItemFirstItem`.

In your `init` package, create a new class called `ModBlocks`. This will hold references to our mod's blocks.

```
package com.cubicoder.tutorial.init;

import com.cubicoder.tutorial.TutorialMod;

import com.cubicoder.tutorial.block.BlockFirstBlock;
import net.minecraftforge.fml.common.registry.GameRegistry.ObjectHolder;

@ObjectHolder(TutorialMod.MODID)
public class ModBlocks {

        public static final BlockFirstBlock FIRST_BLOCK = null;

}
```

Next, we need to register the `Block`, as well as the `ItemBlock` corresponding to that `Block`. We do this in our `EventSubscriber`.

```
package com.cubicoder.tutorial;

import com.cubicoder.tutorial.TutorialMod;
import com.cubicoder.tutorial.block.BlockFirstBlock;
import com.cubicoder.tutorial.init.ModBlocks;
import com.cubicoder.tutorial.item.ItemFirstItem;

import net.minecraft.block.Block;
import net.minecraft.block.material.Material;
import net.minecraft.item.Item;
import net.minecraft.item.ItemBlock;

import net.minecraftforge.event.RegistryEvent.Register;
import net.minecraftforge.fml.common.Mod.EventBusSubscriber;
import net.minecraftforge.fml.common.eventhandler.SubscribeEvent;

@EventBusSubscriber(modid = TutorialMod.MODID)
public final class EventSubscriber {

	@SubscribeEvent
	public static void registerBlocks(Register<Block> event) {
		final Block[] blocks = {
				new BlockFirstBlock().setRegistryName("fi
		};

		event.getRegistry().registerAll(blocks);
	}

	@SubscribeEvent
	public static void registerItems(Register<Item> event) {
		final Item[] items = {
				new ItemFirstItem().setRegistryName("firs
		};

		final Item[] itemBlocks = {
		new ItemBlock(ModBlocks.FIRST_BLOCK).setRegistryName(ModB
	};

		event.getRegistry().registerAll(items);
		event.getRegistry().registerAll(itemBlocks);
	}

}
```

Here, we add a new method that registers the blocks. We also register an `ItemBlock` for each block in the `registerItems()` method. Make sure to set the registry name of the `ItemBlock` to be the same as the block's.

In the `ClientEventSubscriber` class, add a line in `registerModels()` to register your block's model.

```
registerModel(Item.getItemFromBlock(ModBlocks.FIRST_BLOCK));
```

Now, we need to create the block models. Each block needs two models: a blockstate JSON in the `assets/tutorialmod/blockstates` folder, and an block model JSON in the `assets/tutorialmod/models/block` folder.

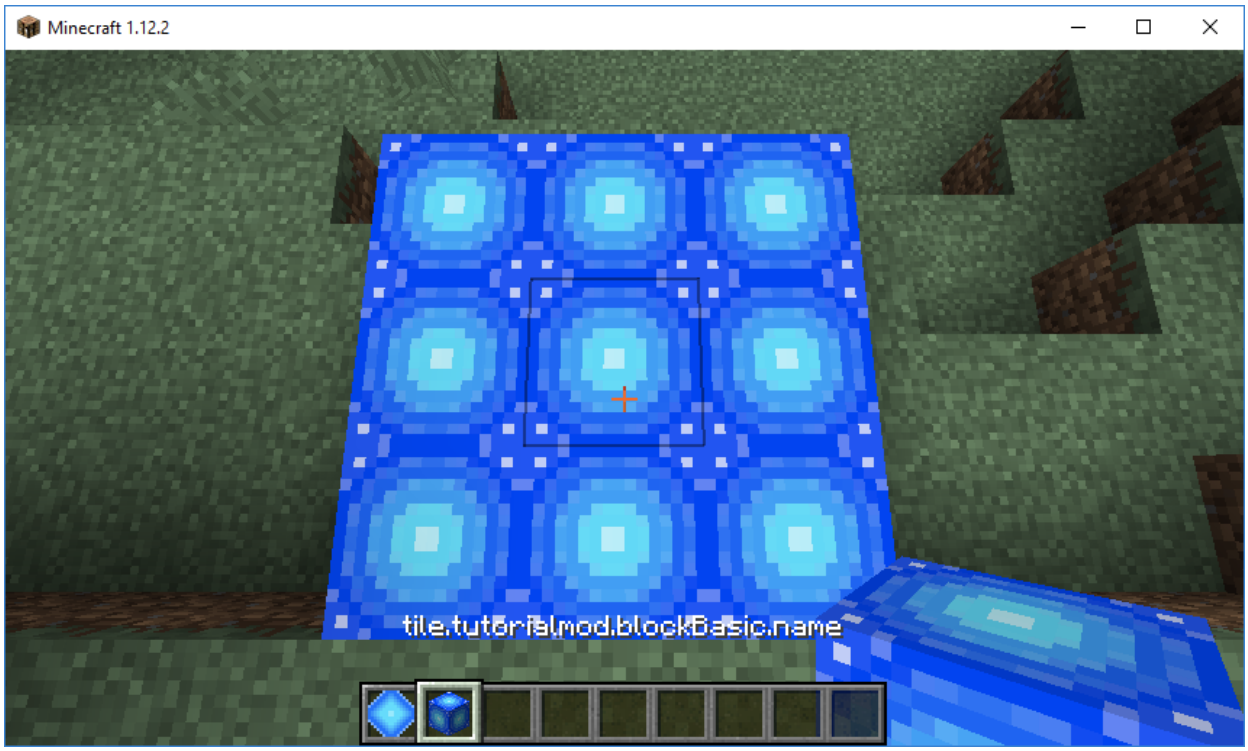
Blockstate JSON (named `first_block.json`):

```
{
    "forge_marker": 1,
    "defaults": {
        "model": "tutorialmod:first_block"
    },
    "variants": {
        "normal": [{}],
        "inventory": [{}]
    }
}
```

Block Model JSON (also named `first_block.json`):

```
{
    "parent": "block/cube_all",
    "textures": {
        "all": "tutorialmod:blocks/first_block"
    }
}
```

Finally, put your texture in the `assets/tutorialmod/textures/blocks` folder. Run the game to see your custom block!



(ht... (ht... (ht...

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

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

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