

FABRIC 1.20 × 我的世界 MOD

第 7 课

会升级的四态方块

右键加一，整数决定模型与能力



本课成就 用 IntProperty 在 0-3 循环，并按等级改变模型、亮度和粒子。

建议 120-140 分钟 | [simon-minecraft-template-1.20](#) | Java 21

01 本课任务：只增加一个新能力

把第 6 课的 true/false 升级为有限整数状态机

布尔开关只能保存两种状态。成长核心需要四个档位：原木、红木、燃烧和焦黑。每次右键加一，到 3 后回到 0；同一个属性还能驱动模型、亮度与粒子。

本课三颗知识钻石

- 使用 `IntProperty.of("level", 0, 3)` 限定合法值。
- 理解 $(\text{current} + 1) \% 4$ 的循环规律。
- 区分同一方块切状态与替换成另一个方块。

1

定义范围

2设置 `level=0`**3**读取 `current`**4**

取模循环

5

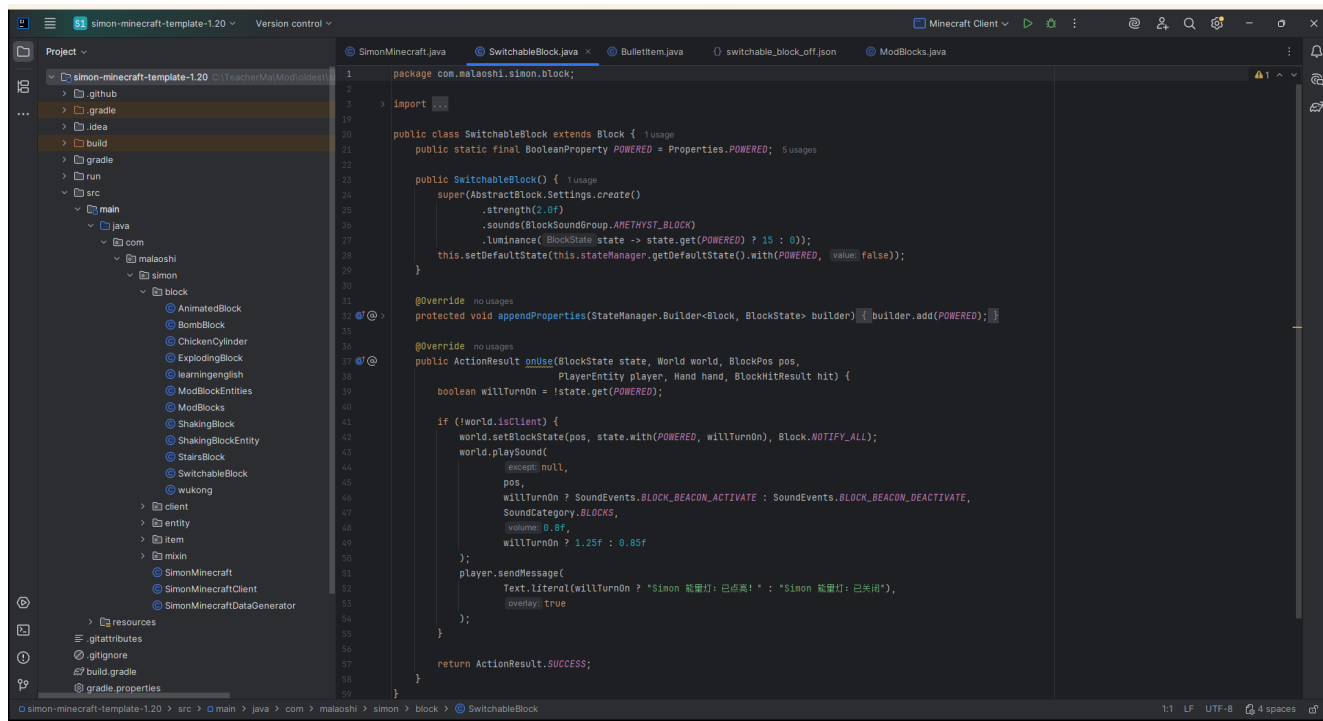
驱动表现

顺序感

第 7 课复用第 6 课的服务端同步思路，只替换属性类型和模型数量，再逐步加入粒子与替换方块实验。

02 在真实 IntelliJ 工程中定位文件

同一 simon-minecraft-template-1.20 工程界面，不是生成图



真实 IntelliJ 截图 · 左侧项目树 + Java 编辑器 + Gradle 工程

本课只打开这些位置

- src/main/java/com/malaoshi/simon/block/FourSwitchableBlock.java
- src/main/java/com/malaoshi/simon/block/ModBlocks.java
- assets/simon-minecraft/blockstates/four_switchable_block.json
- assets/simon-minecraft/models/block/b0.json ... b3.json
- assets/simon-minecraft/textures/block/b0.png ... b3.png

版本锁定

Minecraft 1.20 · Yarn 1.20+build.1 · Fabric API 0.83.0+1.20。当前模板 fabric.mod.json 要求 Java 21。

03 IntProperty 限定 0-3 四个合法值

先读懂，再逐行输入；不要整段复制后跳过理解

```
FourSwitchableBlock.java

public static final IntProperty LEVEL =
    IntProperty.of("level", 0, 3);

public FourSwitchableBlock() {
    super(AbstractBlock.Settings.create()
        .luminance(state ->
            state.get(LEVEL) == 2 ? 15 : 0)
        .ticksRandomly());

    setDefaultState(getStateManager()
        .getDefaultState()
        .with(LEVEL, 0));
}

@Override
protected void appendProperties(
    StateManager.Builder<Block, BlockState> builder) {
    builder.add(LEVEL);
}
```

程序怎么读

IntProperty 不是任意整数变量。范围写成 0-3 后，所有 BlockState 都只能使用这四个合法值。

04 右键加一并安全绕回 0

Java 与资源文件必须使用同一个 MOD_ID 和注册名

```
FourSwitchableBlock.java · onUse

@Override
public ActionResult onUse(
    BlockState state, World world, BlockPos pos,
    PlayerEntity player, Hand hand, BlockHitResult hit) {

    int current = state.get(LEVEL);
    int next = (current + 1) % 4;

    if (!world.isClient) {
        world.setBlockState(
            pos,
            state.with(LEVEL, next),
            Block.NOTIFY_ALL
        );
    }

    return ActionResult.SUCCESS;
}
```

连接点

3 + 1 得到 4，再对 4 取余得到 0。这样结果永远处于 0-3，不会把非法值写进 BlockState。

05 整数状态可以驱动哪些效果？

保持状态数量有限，才能让模型和测试可控

模型

level=0...3 映射 b0...b3。

亮度

仅 level=2 返回 15，其它状态返回 0。

粒子

randomDisplayTick 按等级显示火焰或烟雾。

红石

可把 level 转换为 0、5、10、15 信号。

真正替换

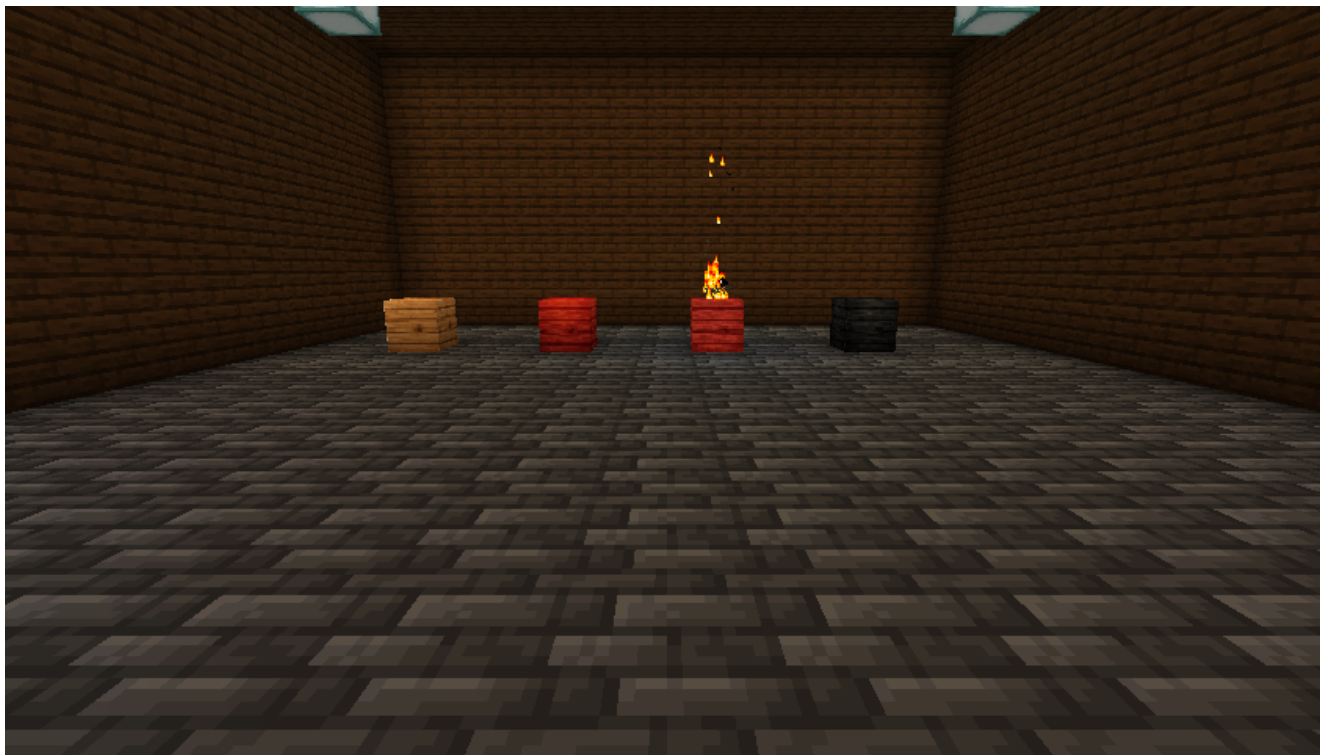
若硬度、掉落物和方块类都要变，改用另一个 BlockState。

课堂检查

大量连续数据、物品栏或自定义 NBT 不适合塞进 Property，应使用 BlockEntity。替换带数据方块时还要迁移原数据。

06 进入游戏验证结果

真实 Fabric 客户端截图，由 Minecraft ScreenshotRecorder 保存



游戏实拍 • level=0、1、2、3 四态并排，level=2 正在发光并产生火焰

检查 1

连续右键顺序始终是 0→1→2→3→0。

检查 2

四个状态显示四个不同模型。

检查 3

level=2 发光并显示火焰粒子。

检查 4

重进世界后保持最后整数状态。

实机证据

截图来自本地 runClient 的复制测试世界；课程生成后临时截图代码已撤销，模板重新执行 Gradle build 成功。

07 故障地图与通关挑战

到 3 后崩溃

不要直接 +1; 使用 $(current + 1) \% 4$ 。

所有状态同一模型

检查 blockstates 中 level=0...3 的键。

粒子太多卡顿

每次只生成少量粒子，不要循环上千次。

替换后数据丢失

BlockEntity 数据需先读取并迁移到新方块。

Boss 挑战

增加 HORIZONTAL_FACING，先计算 4 个等级 × 4 个方向 = 16 种组合，再决定使用 variants 还是 multipart，并完成测试表。

下一课

完成第 1-7 课后，可以进入 BlockEntity、GUI、配方与网络同步专题。