

FABRIC 1.20 × 我的世界 MOD

第 9 课

让红石控制能量灯

监听邻居更新，让方块自动开关



本课成就 红石信号到来时自动点亮，信号消失时自动熄灭，并同步模型与亮度。

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

从第 6-8 课的玩家交互，升级为世界中的邻居更新事件

玩家不一定要亲手右键方块。拉动拉杆、推动活塞或改变红石线，Minecraft 会通知相邻方块。我们在 `neighborUpdate` 中读取信号，把 LIT 自动同步为 `true` 或 `false`。

本课三颗知识钻石

- 理解 `neighborUpdate` 何时由世界调用。
- 用 `world.isReceivingRedstonePower` 读取当前位置的红石信号。
- 只在状态变化时写回世界，避免无意义循环更新。

1

定义 LIT

2

监听邻居

3

读取信号

4

比较旧状态

5

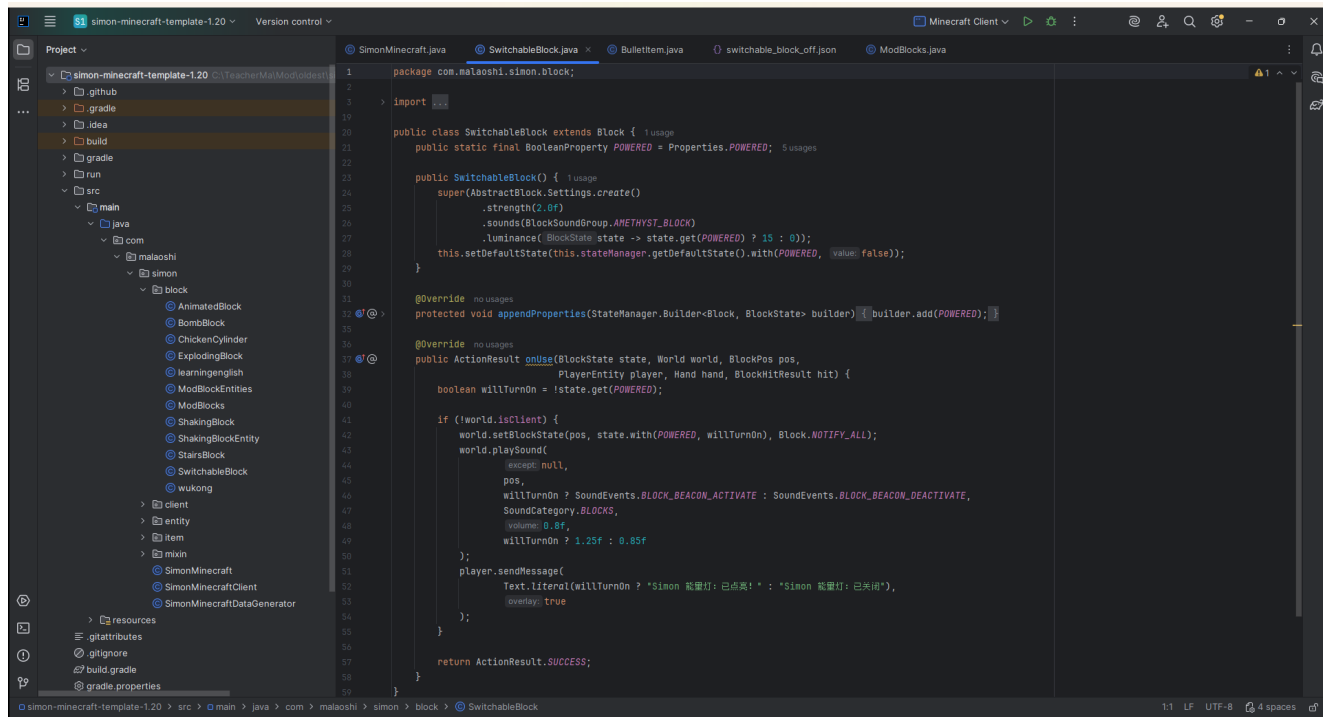
同步亮度

顺序感

第 6 课由玩家右键切换 `BooleanProperty`；本课保留相同属性结构，只把触发源替换成红石邻居事件。

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/RedstoneEnergyBlock.java
- src/main/java/com/malaoshi/simon/block/ModBlocks.java
- assets/simon-minecraft/blockstates/redstone_energy_block.json
- assets/simon-minecraft/models/block/redstone_energy_off.json
- assets/simon-minecraft/models/block/redstone_energy_on.json

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 定义 LIT 状态和动态亮度

先读懂，再逐行输入；不要整段复制后跳过理解

```
RedstoneEnergyBlock.java

public class RedstoneEnergyBlock extends Block {
    public static final BooleanProperty LIT = Properties.LIT;

    public RedstoneEnergyBlock(Settings settings) {
        super(settings.luminance(state ->
            state.get(LIT) ? 15 : 0));

        setDefaultState(getStateManager()
            .getDefaultState()
            .with(LIT, false));
    }

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

程序怎么读

LIT 与第 6 课的布尔属性完全相同。属性仍负责保存结果，改变的是谁来触发它。

04 邻居变化时读取红石并按需更新

Java 与资源文件必须使用同一个 MOD_ID 和注册名

```
RedstoneEnergyBlock.java · neighborUpdate

@Override
public void neighborUpdate(
    BlockState state, World world, BlockPos pos,
    Block sourceBlock, BlockPos sourcePos,
    boolean notify) {

    if (!world.isClient) {
        boolean receiving =
            world.isReceivingRedstonePower(pos);

        if (state.get(LIT) != receiving) {
            world.setBlockState(
                pos,
                state.with(LIT, receiving),
                Block.NOTIFY_ALL);
        }
    }
}
```

连接点

先比较旧值与信号，只在二者不同的时候 setBlockState。这样不会因为每次通知都重复写入而形成更新风暴。

05 一次红石更新的因果链

拉杆改变 → 邻居收到通知 → 读取信号 → 更新 LIT → 模型发光

信号源

拉杆、红石火把或红石块提供 0-15 信号。

邻居通知

Minecraft 调用相邻方块的 `neighborUpdate`。

信号判断

`isReceivingRedstonePower` 返回当前位置是否被供能。

状态同步

LIT 写回 `BlockState` 并通知客户端。

表现层

亮度函数与 `blockstates` 同时读取 LIT。

课堂检查

不要在 `neighborUpdate` 中无条件 `setBlockState`。更新方块会继续通知邻居，条件缺失时可能造成大量重复更新。

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 · 左侧红石线路通电并点亮灯，右侧无信号保持熄灭

检查 1

拉下拉杆后方块立即点亮。

检查 2

关闭信号后方块立即熄灭。

检查 3

模型和亮度始终使用同一个 LIT。

检查 4

相邻红石快速变化时游戏不卡顿。

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

灯一直不亮

确认读取的是方块 pos，而不是 sourcePos。

灯一直亮

关闭信号后也要把 LIT 写回 false。

游戏明显卡顿

只在 state.get(LIT) 与 receiving 不同的时候更新。

模型不变

blockstates 的键必须是 lit=false 与 lit=true。

Boss 挑战

增加 IntProperty POWER_LEVEL=0-15，并输出与红石强度相同的亮度或比较器信号；说明为什么会产生 16 个状态。

下一课

第 10 课进入 BlockEntity：让方块拥有每刻变化的数据和自定义动态渲染。