

第33课

方块事件侦探

读懂事件，建立边界清楚的开采规则



事件是一次动作的线索包。先认清世界、位置和模式，再决定是否取消；用同一份数字规则画清出生点保护范围，区分破坏许可与真正拾取。

前置背包：监听器、条件判断、方法与权限。新项目BlockAcademyL33继续使用Java21、Paper1.21.11和第31课构建脚本。

阅读方式：先猜结果，再运行核对；每次只改一个条件。

环境：JDK 21、普通 Paper 1.21.11、同版本客户端。服务器插件由 Paper 加载；只有明确标注的规则测试使用 main。

开始前：先认清运行位置

前30课打好了普通 Java 程序的基础。这一阶段把代码接到 Paper 服务器：Minecraft 客户端负责进入世界，服务器加载插件，事件和命令把玩家动作交给你的方法。

本书固定使用 JDK 21 与 Paper 1.21.11，客户端版本也保持一致。下载页面上的最新版本可能已经变了，按版本号选择。本书采用普通 plugin.yml 与 Bukkit API；本阶段没有声称支持 Folia。

图框中的短例是局部片段，先读本节的位置说明，不要各自保存成插件。完整文件会明确给出文件名；同一文件分成多段时，按编号连接，包名、import 与外层类只出现一次。

- 预测：先写成功条件和预期消息。
- 运行：构建成功后，在专用本机测试服安装并重启。
- 核对：普通玩家、管理身份与控制台分别走对应路径。

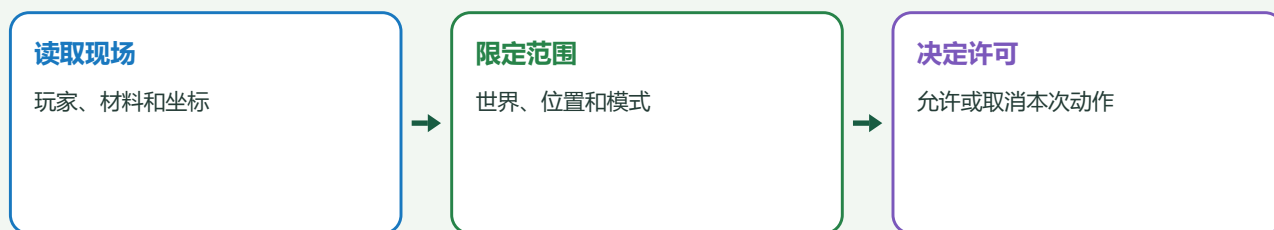
场景插画帮助理解故事；图解解释真实规则，不作为 IDE 或游戏实测截图。做 Java 修改后重新构建、停服替换新 JAR，再启动；不要把旧 JAR 的表现当作新代码结果。

01 案发现场藏在 event 里

侦探拿到“方块被挖”的消息，差点给整个服务器都拉上警戒线。先问在哪个世界、什么模式、哪种方块，再决定是否拦住，这才是会办案的监听器。

event 是这一次动作的信息包。先过滤与规则无关的动作，余下的才判断权限和位置。取消事件改变本次动作是否发生；仅发送警告文字不会自动取消。

案发现场藏在 event 里



事件时机：BlockBreakEvent 用于处理即将进行的方块破坏。本课观察提示写成“获准开采”，不把事件通知当成最终拾取到钻石的证明。

读取现场 • 位置见本节说明

```

var player = event.getPlayer();
var block = event.getBlock();
Material type = block.getType();
String worldName = block.getWorld().getName();
    
```

临时诊断日志 • 位置见本节说明

```

getLogger().info(player.getName() + " -> " + type
    + " @ " + block.getX() + "," + block.getY() + "," + block.getZ());
    
```

02 先筛选世界和游戏模式

练习规则只管名为 world 的世界。出生点保护对该世界所有模式生效；钻石执照只约束生存模式。把范围写清，管理员就知道为什么创造模式有不同表现。



名字不是魔法：如果练习世界实际叫 classroom，就同步替换代码中的 world。通过 block.getWorld 读取事件真正所属的世界。

只处理练习世界 • 位置见本节说明

```
if (!event.getBlock().getWorld().getName().equals("world")) return;
```

执照只检查生存模式 • 位置见本节说明

```
// 本句放在出生点保护之后
if (event.getPlayer().getGameMode() != GameMode.SURVIVAL) return;
```

核对表

场景	出生点保护	钻石执照
world 生存	检查	检查
world 创造	检查	跳过
其他世界	跳过	跳过

03 钻石也有深板岩外套

只判断 DIAMOND_ORE 会漏掉 DEEPSLATE_DIAMOND_ORE。把“是不是钻石矿”的规则写成方法，所有监听方法都调用同一份判断。

钻石也有深板岩外套

DIAMOND_ORE

普通钻石矿

DEEPSLATE_DIAMOND_ORE

深板岩钻石矿

其他材料

执照规则不干预

先说明统计口径：挖到矿石、物品掉落、玩家拾取，是不同事情。精准采集和时运都会影响最终掉落；本课只管破坏许可。

统一判断矿石 · 位置见本节说明

```
private boolean isDiamond(Material type) {
    return type == Material.DIAMOND_ORE
        || type == Material.DEEPSLATE_DIAMOND_ORE;
}
```

快速退出无关方块 · 位置见本节说明

```
if (!isDiamond(event.getBlock().getType())) return;
// 下面只处理两种钻石矿
```

- 把 STONE 输入这个规则，答案应该是 false。
- 把两种钻石矿分别输入，答案都应该是 true。
- 增加金矿时先列出对应材料名称，再修改方法和测试表。

04 取消动作时告诉玩家原因

权限节点像执照编号。没有钻石开采资格，就取消本次破坏并解释原因；它不会自动发一本纸质执照，也不会让玩家获得管理员的其他能力。

取消动作时告诉玩家原因

无执照

取消并给出原因

有执照

仍须通过出生点规则

已被取消

不恢复别人的决定

怎样测试身份：练习默认 op 有执照及出生点绕过权限。在测试服可以用 op 与 deop 切换测试身份；正式服用权限插件分别授予节点，不用管理员权限代替执照。

拒绝无执照开采 • 位置见本节说明

```
if (!player.hasPermission("blockacademy.mine.diamond")) {  
    event.setCancelled(true);  
    player.sendMessage(Component.text("钻石矿需要矿工执照"));  
    return;  
}
```

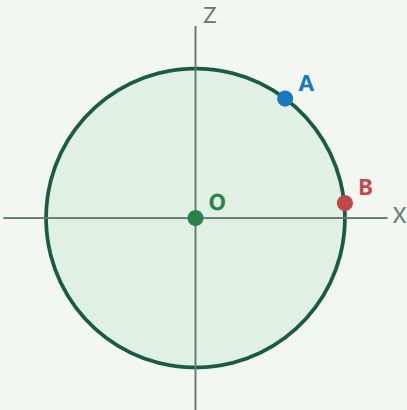
本课权限声明 • 位置见本节说明

```
permissions:  
  blockacademy.mine.diamond:  
    default: op  
  blockacademy.bypass:  
    default: op
```

05 出生点保护是一根圆柱

把出生点的方块坐标当圆心，只比较 X 和 Z。半径10格、边界包含在内，因此上方和地下也受到保护。先确认同一世界，才有比较坐标的意义。

只看X/Z的保护圆：半径10，包含边界



圆内 O (0,0)

$0^2 + 0^2 = 0$ ，保护

边界 A (6,8)

$6^2 + 8^2 = 100$ ，仍保护

圈外 B (10,1)

$10^2 + 1^2 = 101$ ，不保护

形状由公式决定：加入dy平方会变成球形规则。本课故意忽略Y，不能把它说成球。也不要对不同世界的位置直接调用distanceSquared。

不用开平方也能比较 • 位置见本节说明

```
double dx = (double) block.getX() - spawn.getBlockX();
double dz = (double) block.getZ() - spawn.getBlockZ();
boolean protectedArea = dx * dx + dz * dz <= 100.0;
```

核对表

相对坐标 dx dz	平方和	受保护
0 0	0	是
6 8	100	是，边界
10 1	101	否
11 0	121	否

06 规则处理和观察分开

优先级决定先后顺序，不代表谁更厉害。普通监听器可以修改事件；MONITOR 用来观察处理结果，按照约定不再改变事件。ignoreCancelled=true 会跳过已经取消的事件。

规则处理和观察分开

规则监听

可以按合同取消

MONITOR

只观察，不修改事件

ignoreCancelled

跳过已经取消的动作

不要抢救别人的取消：setCancelled(false) 可能撤销其他保护插件的决定。本课不恢复已取消事件，取消的开采也不发送成功广播。

规则方法 • 位置见本节说明

```
@EventHandler(ignoreCancelled = true)
public void protect(BlockBreakEvent event) {
    // 检查条件，必要时 setCancelled(true)
}
```

观察方法 • 位置见本节说明

```
@EventHandler(priority = EventPriority.MONITOR, ignoreCancelled = true)
public void observe(BlockBreakEvent event) {
    // 只观察，不取消，不恢复，不修改方块
}
```

07 侦探不能堵住服务器走廊

方块事件运行频繁。每挖一块都下载网页或保存整个文件，玩家就会在原地表演幻灯片。先做廉价过滤，必要时只更新内存，保存策略留到数据管理章节。

停下来想一想：代码只发出“这里不能挖”的提示，方块却仍被挖走。是聊天故障吗？还缺什么？

侦探不能堵住服务器走廊

先做廉价过滤

无关世界尽早退出

只更新必要信息

每次事件保持短

慢工作另安排

不逐块下载或重写大文件

想通这一步：提示正常，但没有取消本次事件。满足拦截条件时调用 `setCancelled(true)`，再说明原因。

把无关事件尽早放行 • 位置见本节说明

```
if (!inLessonWorld(event)) return;
if (event.getPlayer().getGameMode() != GameMode.SURVIVAL) return;
if (!isDiamond(event.getBlock().getType())) return;
```

故障实验：下面故意保留错误，仅在练习副本中观察。不要把它合并进可用插件。

错误的事件做法 • 故意错误，先预测再修复

```
// 不要在每一次挖矿事件里调用耗时操作
// 下载网页、扫描整个世界、保存巨大文件
```

- 测试时可暂时打印坐标；确认规则后去掉每块都打印的日志。
- 频繁拒绝提示会刷屏，可在第34课学会按玩家计时后限制提示频率。
- 不把“异步”当万能加速按钮；世界API的线程限制在第35课讨论。

08 把几何规则拿出来测试

新建与主类同包的 MiningRules.java。这个类只处理数字，不需要启动Minecraft。先验证边界，再把方法接入事件，定位问题会容易得多。

把几何规则拿出来测试

圆内

距离平方小于100

边界

$6 \times 6 + 8 \times 8 = 100$

圈外

$10 \times 10 + 1 \times 1 = 101$

同一公式三次出场：第三个例子把圆心整体平移，结果仍在边界上。这样能发现忘了减出生点坐标的错误。

完整 MiningRules 类 · 完整文件

```
package de.teacherma.lesson33;

public final class MiningRules {
    private MiningRules() { }
    public static boolean isProtected(int x, int z, int sx, int sz) {
        double dx = (double) x - sx;
        double dz = (double) z - sz;
        return dx * dx + dz * dz <= 100.0;
    }
}
```

在普通 main 中验算 · 位置见本节说明

```
System.out.println(MiningRules.isProtected(6, 8, 0, 0));
System.out.println(MiningRules.isProtected(10, 1, 0, 0));
System.out.println(MiningRules.isProtected(106, 208, 100, 200));
```

运行核对

```
true
false
true
```

09 矿工执照程序上半部

建立 MinerLicensePlugin.java。本页与下一页按顺序合并，另外保留上一页的 MiningRules.java。主类登记监听器，并统一世界和矿石的判断。

矿工执照程序上半部

统一入口

启用时登记

统一判断

世界与两种钻石矿

保护优先

圆内先检查绕过权限

当前这次事件：spawn 从 block.getWorld 获取，因此与当前方块属于同一世界；MiningRules只收到数字，不负责世界判断。

主类第1部分 • 本段1/2，顺序连接

```
package de.teacherma.lesson33;

import net.kyori.adventure.text.Component;
import org.bukkit.GameMode;
import org.bukkit.Material;
import org.bukkit.event.EventHandler;
import org.bukkit.event.EventPriority;
import org.bukkit.event.Listener;
import org.bukkit.event.block.BlockBreakEvent;
import org.bukkit.plugin.java.JavaPlugin;

public final class MinerLicensePlugin extends JavaPlugin implements Listener {
    @Override
    public void onEnable() {
        getServer().getPluginManager().registerEvents(this, this);
    }
    private boolean inLessonWorld(BlockBreakEvent event) {
        return event.getBlock().getWorld().getName().equals("world");
    }
    private boolean isDiamond(Material type) {
        return type == Material.DIAMOND_ORE
            || type == Material.DEEPSLATE_DIAMOND_ORE;
    }
    @EventHandler(ignoreCancelled = true)
    public void protect(BlockBreakEvent event) {
```

09 矿工执照程序上半部（续）

主类第1部分 • 本段2/2, 顺序连接

```
if (!inLessonWorld(event)) return;
var player = event.getPlayer();
var block = event.getBlock();
var spawn = block.getWorld().getSpawnLocation();
boolean nearSpawn = MiningRules.isProtected(block.getX(), block.getZ(),
    spawn.getBlockX(), spawn.getBlockZ());
if (nearSpawn && !player.hasPermission("blockacademy.bypass")) {
```

10 矿工执照程序下半部

继续上一页，完成取消规则和观察消息。src/main/resources/plugin.yml 的基础六行参照第31课，将name改为BlockAcademyL33、main改为de.teacherma.lesson33.MinerLicensePlugin；再加入本课权限声明。

矿工执照程序下半部

取消规则

拒绝时说明原因

模式分流

执照仅管生存

观察结果

未取消时报告获准开采

构建与部署：settings 中项目名改为BlockAcademyL33。使用第31课build.gradle.kts，构建后只部署本课JAR到练习测试服并重启。

主类第2部分 • 顺序接在同一源文件

```

        event.setCancelled(true);
        player.sendMessage(Component.text("出生点十格以内请保留原貌"));
        return;
    }
    if (player.getGameMode() != GameMode.SURVIVAL) return;
    if (isDiamond(block.getType())
        && !player.hasPermission("blockacademy.mine.diamond")) {
        event.setCancelled(true);
        player.sendMessage(Component.text("钻石矿需要矿工执照"));
    }
}

@EventHandler(priority = EventPriority.MONITOR, ignoreCancelled = true)
public void observe(BlockBreakEvent event) {
    if (!inLessonWorld(event)) return;
    if (event.getPlayer().getGameMode() != GameMode.SURVIVAL) return;
    if (!isDiamond(event.getBlock().getType())) return;
    getServer().broadcast(Component.text(
        event.getPlayer().getName() + " 获准开采钻石矿! "));
}
}

```

10 矿工执照程序下半部（续）

plugin.yml：把入口与权限放在同一个完整文件

src/main/resources/plugin.yml • 完整文件

```
name: BlockAcademyL33
version: 1.0.0
main: de.teacherma.lesson33.MinerLicensePlugin
api-version: '1.21.11'
author: TeacherMa
website: https://www.teacher-ma.de
permissions:
  blockacademy.mine.diamond:
    default: op
  blockacademy.bypass:
    default: op
```

11 验收规则要分清谁先拦住

出生点和执照可能同时阻止一次开采。测试执照时先走出保护圈，否则你看到的始终是出生点提示。测试表每一行只改变一个条件。

验收规则要分清谁先拦住

保护测试

圈内、边界与圈外

执照测试

走出圈再改权限

观察测试

已取消时不得发成功消息

广播不等于钻石入袋：观察到未取消的破坏意图，并不保证最终拾取数量。未来做经济奖励时，还需要定义更严格的事件和统计规则。

核对表

条件	预期
world圈内，无绕过权限	所有模式的破坏被取消
world边界6 8，无绕过	取消；边界也保护
圈外生存，无执照，钻石矿	取消并提示执照
圈外生存，无执照，普通石头	允许
圈外生存，有执照，两种钻石矿	允许并广播获准开采
其他世界	本插件不干预
事件先被其他插件取消	不广播成功消息



马老师教编程
www.teacher-ma.de

15 / 18

12 拓展挑战与自己的验收记录

留下源代码、实际运行结果和一条问题记录。先用自己的话解释，再运行验证；结果与预测不同的地方，往往最值得讨论。

拓展任务与通关复盘

木镐

解释圆柱形状

铁镐

增加矿石并验收

钻石镐

限制拒绝提示频率

下一站：第34课给每位玩家建立独立数据和冷却。
官方延伸阅读：docs.papermc.io/paper/dev/event-listeners/

- 木镐：挖两种金矿时发提示，普通石头保持安静。
- 铁镐：把保护半径改为8格，增加半径内、边界和半径外测试。
- 钻石镐：结合第34课Map，为每位玩家记录获准开采次数；说明取消事件是否计数。
- 验收要求：分别测试普通钻石矿和深板岩钻石矿；在保护半径内、边界上、边界外记录是否允许及原因。

核对表

我能独立完成	已掌握	再练一次
能按世界和材料过滤	☐	☐
能解释圆柱保护边界	☐	☐
能区分取消与观察	☐	☐



马老师教编程
www.teacher-ma.de

16 / 18

可独立运行的规则验证

在项目内新建 checks/L33RuleCheck.java，完整保存下面代码。新建 checks/sources.txt，写入下面两条路径，然后从项目根目录执行命令。这是普通 Java 测试，不是插件主类。

L33RuleCheck.java • 完整文件

```
import de.teacherma.lesson33.MiningRules;

public class L33RuleCheck {
    public static void main(String[] args) {
        System.out.println(MiningRules.isProtected(0, 0, 0, 0));
        System.out.println(MiningRules.isProtected(6, 8, 0, 0));
        System.out.println(MiningRules.isProtected(10, 1, 0, 0));
        System.out.println(MiningRules.isProtected(106, 108, 100, 100));
    }
}
```

运行核对

```
true
true
false
true
```

checks/sources.txt • 完整内容

```
src/main/java/de/teacherma/lesson33/MiningRules.java
checks/L33RuleCheck.java
```

项目根目录终端命令 • 分别执行两行

```
javac -encoding UTF-8 -d checks/out @checks/sources.txt
java -cp checks/out L33RuleCheck
```

纯规则通过只证明数字合同，权限、玩家动作、事件登记和服务加载仍按前面的游戏验收表检查。

把证据留给下一次冒险

把“能编译”“数字规则通过”和“游戏实际表现正确”分成三栏。没有运行的场景写待测；发现 Bug 时记录触发条件、预期、实际、最小修复与复测结果。

你的验证记录

类别	你选择的操作	留下的证据
正常	本课的一次成功路径	版本与实际输出
边界	结束前一刻、最大值或满背包	与合同逐项比较
拒绝	错误参数、无权限或不相关事件	反馈清楚，状态未误改

- 解释本课每条箭头为什么从这里指向那里；并列卡片没有先后含义。
- 先改一个规则，再同时修改说明和预期，用同一输入复测。

技术核对：本课以 Paper 1.21.11 API 和 Java SE 21 为准。参考文档可能切换到较新版本，查阅时注意版本号。

<https://jd.papermc.io/paper/1.21.11/>
<https://docs.papermc.io/paper/dev/>
<https://docs.oracle.com/en/java/javase/21/docs/api/>