

JAVA / LESSON 33 OF 40

The Block Event Detective

Read the events and establish clear rules of exploitation at the border.

Environment: JDK 21, ordinary Paper 1.21.11, client of the same version. Server plugins are loaded by Paper; only explicitly labeled rule tests use main. | 100-120 minutes



Spawn point protection and miner license

An event is a clue package for an action. Recognize the world, location and mode first, and then decide whether to cancel; use the same number rules to draw the protection range of the birth point, and distinguish between permission to destroy and actual pickup.

Front-end backpack: listener, conditional judgment, method and authority. The new project BlockAcademyL33 continues to use Java21, Paper1.21.11 and the Lesson 31 build script.

Before you start: It is recommended to complete Lessons 31-32 first

Read the full lesson and copy the code:

<https://teacher-ma.de/minecraft-java-lesson.php?lesson=33&lang=en>

Long code lines may wrap at the > margin marker. The marker is not part of the code.

In this lesson

01. Before starting: first recognize the running position
02. The crime scene is hidden in event
03. First screen the world and game mode
04. Diamond also has a deep slate jacket
05. Tell the player the reason when canceling the action
06. The birth point protection is a cylinder
07. Rule processing and observation are separated
08. Detectives cannot block the server corridor
09. Take out the geometric rules and test them
10. The first half of the miner license process
11. Miner License Procedure Second Half
12. The acceptance rules should distinguish who stops first
13. Expansion challenges and own acceptance records
14. Rule verification that can run independently
15. Leave the evidence for the next adventure

01 / 15

Before starting: first recognize the running position

The first 30 lessons lay the foundation for ordinary Java programs. At this stage, the code is connected to the Paper server: the Minecraft client is responsible for entering the world, the server loads the plug-in, and the events and commands hand over player actions to your methods.

This book uses JDK 21 and Paper 1.21.11, and the client version is also consistent. The latest version on the download page may have changed, select it by version number. This book uses normal plugin.yml and the Bukkit API; no claims are made to support Folia at this stage.

The short examples in the picture frame are partial fragments. Read the location description of this section first, and do not save each as a plug-in. The file name will be clearly given for the complete file; when the same file is divided into multiple segments, they will be connected by number, and the package name, import and outer class will only appear once.

- Prediction: Write success conditions and expected messages first.
- Run: After successful construction, install and restart on the dedicated local test server.
- Verification: Ordinary players, management identities and consoles take corresponding paths respectively.

The scene illustrations help to understand the story; the illustrations explain the real rules and are not used as IDE or game actual measurement screenshots. After making Java modifications, rebuild, stop the server, replace it with the new JAR, and start it again; do not treat the performance of the old JAR as the result of the new code.

02 / 15

The crime scene is hidden in event

The detective got the news that "the cube was dug" and almost put the entire server on cordon. First ask which world, which mode, and which block, and then decide whether to block it. This is the listener that can handle the case.

event is the information packet of this action. First filter the actions that have nothing to do with the rules, and then determine the permissions and locations of the rest. The cancellation event changes whether this action occurs; just sending a warning text will not automatically cancel it.

The crime scene is hidden in event

Step	Explanation
1. Read the scene	Players, Materials and Coordinates
2. Limited scope	World, location and mode
3. Decide permission	Allow or cancel this action

Event timing: BlockBreakEvent is used to handle upcoming block destruction. The observation prompt in this lesson is written as "permission to mine", and the event notification is not regarded as proof that the diamond was finally picked up.

Read the scene · See the description in this section for the location

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

Temporary diagnostic log · See the description in this section for the location

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

03 / 15

First screen the world and game mode

The training rules only apply to the world named world. Spawn point protection takes effect in all modes of the world; diamond licenses only apply to survival mode. By writing the scope clearly, administrators will know why creative mode behaves differently.

First screen the world and game mode

Step	Explanation
1. world world	Return directly to other worlds
2. Birth point circle	All modes are checked
3. Survival license	Check diamond permissions outside the circle

Before entering the next step along the arrow, the conditions of the previous step must be met; the rejected branch returns early.

Names are not magic: If the practice world is actually called classroom, replace world in the code synchronously. Read the world to which the event actually belongs via `block.getWorld`.

Only handles the practice world · See the description in this section for the location

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

The license only checks survival mode. See the description in this section for the location.

```
// This sentence is placed after the birth point protection  
if (event.getPlayer().getGameMode() != GameMode.SURVIVAL) return;
```

Checklist

Scene	Spawn point protection	Diamond License
world survival	Check	Check
world creation	Check	Skip
Other worlds	Skip	Skip

04 / 15

Diamond also has a deep slate jacket

Only judging `DIAMOND_ORE` will miss `DEEPSLATE_DIAMOND_ORE`. Write the rule "Is it a Diamond Ore" as a method, and all listening methods call the same judgment.

Diamonds also have deep slate coatings

Element	Explanation
<code>DIAMOND_ORE</code>	Ordinary Diamond Ore
<code>DEEPSLATE_DIAMOND_ORE</code>	Deep Slate Diamond Ore
Other materials	License rules do not interfere

Let me first explain the statistical caliber: digging out ore, dropping items, and picking up items are all different things. Accurate collection and luck will affect the final drop; this lesson only deals with breaking permissions.

Unified judgment of ore·See the description in this section for its location

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

Quickly exit irrelevant blocks · See the description in this section for the location

```
if (!isDiamond(event.getBlock().getType())) return;  
// Only two kinds of diamond ores are processed below
```

- Enter STONE into this rule and the answer should be false.
- Enter the two Diamond Ores separately, and the answer should be true.
- When adding gold mines, first list the corresponding material names, and then modify the method and test table.

05 / 15

Tell the player the reason when canceling the action

The authority node is like a license number. If you are not qualified to mine diamonds, cancel this destruction and explain the reason; it will not automatically issue a paper license, nor will it allow the player to obtain other abilities of the administrator.

Tell the player the reason when canceling the action

Element	Explanation
No license	Cancel and give reasons
Licensed	Still must pass spawn point rules
Canceled	Do not restore other people's decisions

How to test identity: Practice the default op with license and spawn bypass permissions. In the test server, you can use op and deop to switch the test identity; the official permission plug-in is granted to the nodes separately, and administrator permissions are not used instead of licenses.

Reject mining without a license · See the description in this section for the location

```
if (!player.hasPermission("blockacademy.mine.diamond")) {  
    event.setCancelled(true);  
    player.sendMessage(Component.text("Diamond Ores require a miner's license"));  
    return;  
}
```

Permission statement for this lesson · See the description in this section for the location

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

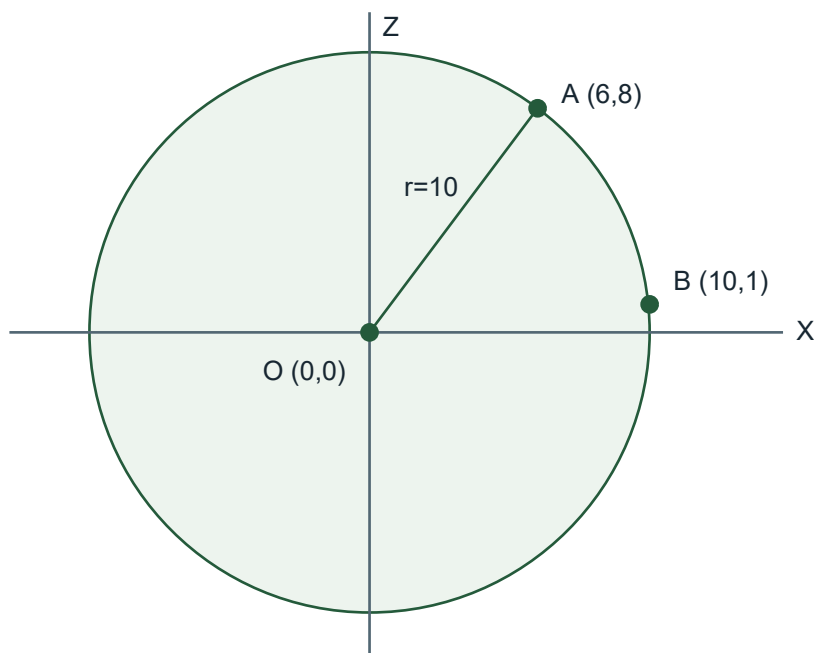
06 / 15

The birth point protection is a cylinder

Use the square coordinates of the birth point as the center of the circle, and only compare X and Z. The radius is 10 tiles, and the borders are included, so the top and below are also protected. Only after confirming the same world can it make sense to compare coordinates.

Only look at the guard circle of X/Z: radius 10, including the boundary

$$dx^2 + dz^2 \leq 10^2$$



X / Z	Distance squared	Protected?
O (0, 0)	0	Yes
A (6, 8)	100	Yes, on the boundary
B (10, 1)	101	No

Measure distance in the X/Z plane; ignore Y height.

The shape is determined by the formula: adding dy squared will become a spherical rule. Y is deliberately ignored in this lesson because it cannot be described as a ball. Also don't call distanceSquared directly on locations in different worlds.

Can be compared without taking the square root · See the description in this section for the position

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

Checklist

Relative coordinates dx dz	Sum of squares	protected
0 0	0	Yes
6 8	100	Yes, boundary
10 1	101	No
11 0	121	No

07 / 15

Rule processing and observation are separated

Priority determines the order and does not mean who is more powerful. Ordinary listeners can modify events; MONITOR is used to observe the processing results and will not change events according to the agreement. ignoreCancelled=true will skip canceled events.

Rule processing and observation are separated

Element	Explanation
Rule monitoring	Can be canceled according to contract
MONITOR	Only observe, do not modify events
ignoreCancelled	Skip canceled actions

Don't rescue other people's cancellations: `setCancelled(false)` may undo the decisions of other protection plugins. This class does not restore canceled events, nor does canceled mining send a success broadcast.

Rule method · See the description in this section for the location

```
@EventHandler(ignoreCancelled = true)
public void protect(BlockBreakEvent event) {
    // Check conditions, if necessary setCancelled(true)
}
```

Observation method · See the description in this section for the location

```
@EventHandler(priority = EventPriority.MONITOR, ignoreCancelled = true)
public void observe(BlockBreakEvent event) {
    // Only observe, do not cancel, do not restore, do not modify the block
}
```

08 / 15

Detectives cannot block the server corridor

Block events run frequently. Download the web page or save the entire file with each piece dug, and the player will perform the slide show in place. Do cheap filtering first, only update the memory when necessary, and save the strategy for the data management chapter.

Stop and think about it: the code only sends out the prompt "cannot dig here", but the block is still dug away. Is it a chat glitch? What's missing?

Detectives cannot block server corridors

Element	Explanation
Do cheap filtering first	Exit the irrelevant world as soon as possible
Only update necessary information	Each event is kept short
Slow work will be arranged separately	Do not download or rewrite large files chunk by chunk

Figure out this step: the prompt is normal, but this event is not canceled. Call `setCancelled(true)` when the interception conditions are met, and then explain the reason.

Release irrelevant events as soon as possible · See the description in this section for the location

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

Fault experiment: The errors are deliberately kept below and are only observed in the practice copy. Do not merge it into available plugins.

Wrong event approach · Intentional mistakes, predict first and then fix

```
// Do not call time-consuming operations in every mining event
// Download web pages, scan the entire world, save huge files
```

- The coordinates can be temporarily printed during the test; after confirming the rules, remove the log printed for each block.
- Frequently rejecting prompts will cause the screen to swipe. You can limit the frequency of prompts after learning to time the player in Lesson 34.
- Do not regard "asynchronous" as a universal acceleration button; the thread limit of the world API is discussed in Lesson 35.

09 / 15

Take out the geometric rules and test them

Create a new MiningRules.java in the same package as the main class. This class only handles numbers and does not require starting Minecraft. Verify the boundary first, and then connect the method to the event. It will be much easier to locate the problem.

Take out the geometric rules and test them

Element	Explanation
Inside the circle	Distance squared is less than 100
Border	$6 \times 6 + 8 \times 8 = 100$
Outside the circle	$10 \times 10 + 1 \times 1 = 101$

The same formula appears three times: the third example translates the center of the circle as a whole, and the result is still on the boundary. This way you can find the mistake of forgetting to subtract the coordinates of the birth point.

Complete MiningRules class · Complete file

```
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;
    }
}
```

Check calculation in ordinary main · See the description in this section for the location

```
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));
```

Output

```
true
false
true
```

10 / 15

The first half of the miner license process

Create MinerLicensePlugin.java. This page is merged with the next page in order, and the MiningRules.java of the previous page is retained. The main class registers the listener and unifies the judgment of the world and ores.

Miner License Process Part 1

Element	Explanation
Unified entrance	Register when enabled
Unified judgment	The world and two Diamond Ores
Protection takes priority	First check the bypass permissions within the circle

The current event: spawn is obtained from block.getWorld, so it belongs to the same world as the current block; MiningRules only receives numbers and is not responsible for world judgment.

Main class part 1 · This paragraph 1/2, sequential connection

```
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) {
```

Main Class Part 1·This paragraph 2/2, sequential connection

```
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")) {
```

11 / 15

Miner License Procedure Second Half

Continue to the previous page to complete the cancellation rules and observation messages. For the basic six lines of `src/main/resources/plugin.yml`, refer to Lesson 31. Change the name to `BlockAcademyL33` and main to `de.teacherma.lesson33.MinerLicensePlugin`; then add the permission statement of this lesson.

Miner Licensing Process Second Half

Element	Explanation
Cancel rules	Explain the reason when rejecting
Mode diversion	License is just for survival
Observation results	Report is allowed to mine when not canceled

Construction and deployment: The project name in settings is changed to `BlockAcademyL33`. Use Lesson 31 `build.gradle.kts`. After building, only deploy the JAR of this lesson to the practice test server and restart.

Main class part 2·sequentially connected to the same source file

```

        event.setCancelled(true);
        player.sendMessage(Component.text("Please keep the original appearance within ten blocks
> of the birth point"));
        return;
    }
    if (player.getGameMode() != GameMode.SURVIVAL) return;
    if (isDiamond(block.getType())
        && !player.hasPermission("blockacademy.mine.diamond")) {
        event.setCancelled(true);
        player.sendMessage(Component.text("Diamond Ores require a miner's license"));
    }
}

@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() + " Permission to mine Diamond Ores!"));
}
}

```

plugin.yml: Put the entry and permissions in the same complete file

src/main/resources/plugin.yml · Complete file

```

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

```

12 / 15

The acceptance rules should distinguish who stops first

A spawn point and a license may prevent a mining at the same time. When testing the license, get out of the protective circle first, otherwise you will always see the spawn point prompt. Only one condition is changed for each row of the test table.

The acceptance rules must distinguish who stops first

Element	Explanation
Protection test	Inside the circle, boundary and outside the circle
License Test	Step out of the circle and then change the permissions
Observation test	No success message shall be sent when canceled

Broadcasting does not equal diamonds in the bag: Uncanceled destructive intent is observed and the final number of picks is not guaranteed. When making financial rewards in the future, more stringent event and statistical rules will need to be defined.

Checklist

Conditions	Expected
In the world circle, there is no bypass permission	Destruction of all modes is canceled
world boundary 6 8, no bypass	Cancel; Boundaries are also protected
Survival outside the circle, no license, Diamond Ore	Cancel and prompt license
Survival outside the circle, no license, ordinary stones	Allow
Survival outside the circle, with license, two kinds of Diamond Ores	Allow and broadcast permission to mine
Other worlds	This plug-in does not intervene
The event was first canceled by other plug-ins	Do not broadcast success message

13 / 15

Expansion challenges and own acceptance records

Leave the source code, actual running results and a problem record. Explain it in your own words first, and then run the verification; where the results are different from the predictions, they are often the most worthy of discussion.

Expansion tasks and clearance review

Element	Explanation
Wooden Pickaxe	Explain cylindrical shape
Iron Pickaxe	Add ore and check for acceptance
Diamond Pickaxe	Limit rejection prompt frequency

Next step: Lesson 34 establishes independent data and cooldown for each player.

Official extended reading: docs.papermc.io/paper/dev/event-listeners/

- Wooden pickaxe: prompts when digging two kinds of gold mines, while ordinary stones remain quiet.
- Iron Pickaxe: Change the protection radius to 8 blocks, and add tests within the radius, border and outside the radius.
- Diamond Pickaxe: Combined with the 34th Lesson Map, record the number of times allowed to be mined for each player; indicate whether the cancellation event counts.
- Acceptance requirements: Test ordinary Diamond Ores and deep slate Diamond Ores respectively; record whether it is allowed and the reasons within the protection radius, on the boundary, and outside the boundary.

Checklist

I can complete it independently	Mastered	Practice again
Ability to filter by world and material	☐	☐
Can explain cylinder protection boundary	☐	☐
Can distinguish between cancellation and observation	☐	☐

14 / 15

Rule verification that can run independently

Create new checks/L33RuleCheck.java in the project and save the following code completely. Create a new checks/sources.txt, write the following two paths, and then execute the command from the project root directory. This is a normal Java test, not the main plugin class.

L33RuleCheck.java · Complete file

```
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));
    }
}
```

Output

```
true
true
false
true
```

checks/sources.txt · Full content

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

Project root directory terminal command · Execute two lines respectively

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

Pure rules only prove the digital contract, and permissions, player actions, event registration and server loading are still checked according to the previous game acceptance form.

15 / 15

Leave the evidence for the next adventure

Divide "can compile", "number rules pass" and "actual performance of the game is correct" into three columns. Scenarios that are not running are written to be tested; when bugs are discovered, trigger conditions, expected, actual, minimum repairs, and retest results are recorded.

Your verification record

Category	The operation you selected	Evidence left behind
Normal	A successful path for this lesson	Version and actual output
Border	The moment before the end, the maximum value or the full backpack	Compare item by item with contract
Reject	Wrong parameters, no permissions or irrelevant events	The feedback is clear and the status has not been changed by mistake.

- Explain why each arrow in this lesson points from here to there; the parallel cards have no sequential meaning.
- First change a rule, then modify the description and expectations at the same time, and retest with the same input.

Technical check: This lesson is based on Paper 1.21.11 API and Java SE 21. The reference document may be switched to a newer version, so pay attention to the version number when checking it.

<https://jd.papermc.io/paper/1.21.11/>

<https://docs.papermc.io/paper/dev/>

<https://docs.oracle.com/en/java/javase/21/docs/api/>