

JAVA / LESSON 37 OF 40

Create Custom Items

Write a reliable identity tag on the miner's compass

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



Miner's compass identified with PDC

The name is shown to the player, and the identity data is given to the program. Make an original compass with instructions, identify it with four checks of material, namespace, type and value, and then let the main hand right-click to only display the task once.

Prerequisites: command, event, null value check and List. The new project BlockAcademyL37 follows the construction script of Lesson 31 and continues to lock Java21 and ordinary Paper1.21.11.

Before you start: It is recommended to complete Lessons 31-36 first

Read the full lesson and copy the code:

<https://teacher-ma.de/minecraft-java-lesson.php?lesson=37&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 article body and instructions are modified separately
03. The anvil can be renamed but cannot prove identity.
04. Put production and recognition into the same class
05. Check material type and value
06. Don't rush to report with both hands
07. Reading tasks does not mean approval of interaction
08. Register command for compass project
09. The first half of the compass main class
10. lower half of compass main class
11. real and fake compasses take the exam together
12. Expansion challenges and own acceptance records
13. Leave the evidence for the next adventure

01 / 13

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 / 13

The article body and instructions are modified separately

The ordinary compass is renamed "Super Miner Compass" in the anvil, so come and apply for special abilities. The guard looked at the name and almost let me go. The display name can be changed, and reliable identification requires viewing the identity tag written by the plug-in.

ItemStack is the item and ItemMeta is the description and additional data. Write the PDC tag corresponding to NamespacedKey when making; check the material, data type and value when identifying. The name is responsible for allowing players to understand it, and the label is responsible for allowing the program to recognize it correctly.

The article body and instructions are modified separately

Element	Explanation
ItemStack	Item body
ItemMeta	Name description and additional data
setItemMeta	Write modifications back to ontology

This lesson does not change the texture: this is the original compass with custom metadata, and the appearance still uses the game's compass texture; the texture pack will not be automatically added, and the pointer target will not be changed.

First make a readable compass. See the instructions in this section for the position.

```
ItemStack item = new ItemStack(Material.COMPASS);
var meta = item.getItemMeta();
meta.displayName(Component.text("Miner's Compass"));
meta.lore(List.of(Component.text("Right-click the main hand to view classroom tasks")));
item.setItemMeta(meta);
```

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

Deliberately missing the last step · Intentional mistake, predict first and then fix

```
// If you forget setItemMeta, Check if the name and description are actually saved.
```

03 / 13

The anvil can be renamed but cannot prove identity.

Rename the ordinary compass to "Miner's Compass", it looks real, but the name is just a label. The program checks for PDC tags hidden in the item's data. The display name is responsible for the player to recognize it, and the data key is responsible for the program to recognize it.

The anvil can be renamed but cannot prove identity.

Element	Explanation
Display name	Convenient for players to understand
NamespacedKey	Stable data location
BYTE value 1	Identity mark of this chapter

Not an anti-counterfeiting safe: PDC is not a secret, nor is it an anti-cheating encryption. It is still possible for administrators or other plugins to copy and modify item data; this addresses reliable identification, not all security issues.

Create a key for this plug-in · See the instructions in this section for the location

```
compassKey = new NamespacedKey(this, "miner_compass");
```

Write a mark with a clear type. See the description in this section for the location.

```
meta.getPersistentDataContainer().set(  
    compassKey, PersistentDataType.BYTE, (byte) 1);
```

Checklist

Information	Purpose
Display name and lore	Let players understand
NamespacedKey	Namespace plus key name to reduce conflicts
BYTE type and value 1	The compass identity agreed in this lesson

04 / 13

Put production and recognition into the same class

Create a new `CompassItems.java` and put it into the `de.teacherma.lesson37` package. Use the same key and the same type for production and identification to avoid writing `miner_compass` on one side and looking for `mining_compass` on the other, which is like sending the key to the wrong door.

Put production and recognition into the same class

Step	Explanation
1. Create compass	Select original material
2. Write metadata	Name, lore and PDC
3. Write back items	setItemMeta and then return

The name can be changed: After the anvil is renamed, the program will still recognize it as long as the material and markings still meet the conditions of this lesson. An ordinary compass simply changed to the same name will not get the PDC mark.

Complete CompassItems.java · Complete file

```
package de.teacherma.lesson37;

import java.util.List;
import net.kyori.adventure.text.Component;
import org.bukkit.Material;
import org.bukkit.NamespacedKey;
import org.bukkit.inventory.ItemStack;
import org.bukkit.persistence.PersistentDataType;

public final class CompassItems {
    private CompassItems() { }
    public static ItemStack create(NamespacedKey key) {
        ItemStack item = new ItemStack(Material.COMPASS);
        var meta = item.getItemMeta();
        meta.displayName(Component.text("Miner's Compass"));
        meta.lore(List.of(Component.text("Right-click the main hand to view classroom tasks")));
        meta.getPersistentDataContainer().set(key, PersistentDataType.BYTE, (byte) 1);
        item.setItemMeta(meta);
        return item;
    }
    public static boolean isOurs(ItemStack item, NamespacedKey key) {
        if (item == null || item.getType() != Material.COMPASS || !item.hasItemMeta())
            return false;
        Byte marker = item.getItemMeta().getPersistentDataContainer()
            .get(key, PersistentDataType.BYTE);
        return Byte.valueOf((byte) 1).equals(marker);
    }
}
```

05 / 13

Check material type and value

item may be null when empty-handed. Ordinary compasses may not have meta; if they do have meta, they may not have our marks. The identification function eliminates mismatches in order, and finally compares whether the BYTE value is 1. Only checking "key exists" will include other tag values.

Check material type and value

Element	Explanation
Material COMPASS	Exclude irrelevant items first
Key and type	Read corresponding BYTE mark
Value equals 1	It is recognized as the compass of this chapter

The key must also be stable: the plug-in name participates in the default namespace. If you change the name of plugin.yml at will after publishing, the old items may no longer match; you need to design data migration when changing the name.

Safety comparison may be empty values · See the description in this section for the location

```
Byte marker = meta.getPersistentDataContainer()
    .get(compassKey, PersistentDataType.BYTE);
boolean ours = Byte.valueOf((byte) 1).equals(marker);
```

Delete the fragment of the custom identity · See the description in this section for the location

```
meta.getPersistentDataContainer().remove(compassKey);
item.setItemMeta(meta);
```

Checklist

Item data	Recognition results
Ordinary compass or unmarked	false
Same key STRING text "1"	false
Same key BYTE value 2	false
COMPASS plus the same key BYTE value 1	true

06 / 13

Don't rush to report with both hands

Interaction events may be triggered by two hands respectively. If the off-hand is also holding a compass, two mission prompts may appear for the same action. We only accept `EquipmentSlot.HAND`, which is the main hand; we do not rely on "delete a message after sending it" to remedy it.

Stop and think about it: Should the True Compass still be recognized after it has been renamed by the player? Should the ordinary compass gain abilities after being renamed with the same name?

Don't rush to report with both hands

Step	Explanation
1. Main hand	<code>EquipmentSlot.HAND</code>
2. Right click action	Both air and blocks are accepted
3. Item matching	Prompt once after passing three levels

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

Think about this step: As long as the materials and PDC markings still comply with the rules, the true compass can still be recognized; the ordinary compass with only the same name and no markings cannot gain the ability.

First screen the main hand. See the description in this section for the position.

```
if (event.getHand() != EquipmentSlot.HAND) return;
```

Both right-clicks are allowed. See the description in this section for the position.

```
Action action = event.getAction();  
if (action != Action.RIGHT_CLICK_AIR  
    && action != Action.RIGHT_CLICK_BLOCK) return;
```

- The master holds the compass in this lesson and right-clicks the air: a task is displayed.
- Right-click the block with the main hand: the task is displayed, and the original interaction with the block can still be continued.
- Holding the compass in the second hand or waving the left button: No tasks will be displayed in this lesson.

07 / 13

Reading tasks does not mean approval of interaction

When right-clicking on the air, the original version may predict "no action" and mark the event as cancelled. Therefore this read-only prompt listener does not have `ignoreCancelled=true` set. It only looks at message concurrency prompts, does not restore cancellation, and does not authorize block operations.

Reading the task does not represent approval of the interaction

Element	Explanation
Read only observation	Only send task text
Original interaction	Not canceled or reinstated by this chapter
There are side effects in the future	Licensing strategy must be redesigned

Make the boundary clear: even if other plug-ins prohibit interaction, this lesson may display read-only task text. If the right click will transfer, charge or change blocks in the future, permissions, protection and corresponding interaction results need to be checked separately, and this strategy cannot be copied directly.

The entrance and location of the read-only observation method is explained in this section.

```
@EventHandler(priority = EventPriority.MONITOR)
public void onUse(PlayerInteractEvent event) {
    // After the main hand, actions, and items are all matched, only the task text is sent..
}
```

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

Do not unblock protection for prompts · Intentional mistakes, predict first and then repair

```
event.setCancelled(false); // Don't do this in this lesson
```

08 / 13

Register command for compass project

The project name is set to BlockAcademyL37. The resources directory only requires plugin.yml, and there is no config.yml in this lesson. The main class and item tool class are in the lesson37 package, and the build script follows Lesson 31.

Register command for compass project

Element	Explanation
CompassItems	Making and identifying shared rules
CompassPlugin	Registration command and listener
plugin.yml	Statement to receive orders and permissions

Receiving and using are separated: by default, only OP can receive it, but players who get the compass can view the read-only tasks. If you need to bind the owner, you can save the UUID and verify it in the advanced task.

Complete plugin.yml · Complete file

```
name: BlockAcademyL37
version: 1.0.0
main: de.teacherma.lesson37.CompassPlugin
api-version: '1.21.11'
author: TeacherMa
website: https://www.teacher-ma.de
commands:
  minercompass:
    description: Get the Classroom Miner's Compass
    usage: /minercompass
permissions:
  blockacademy.compass.give:
    default: op
```

Java files in the complete project · Complete files

```
src/main/java/de/teacherma/lesson37/
  CompassItems.java
  CompassPlugin.java
```

09 / 13

The first half of the compass main class

Create new CompassPlugin.java. This page first declares dependencies, creates a key, and registers commands and listeners; the next page continues with commands and interaction logic. Don't add a new class between two pages.

The first half of the compass main class

Element	Explanation
When enabled	Create stable keys
Command registration	The collection entrance can be found
Monitoring registration	You can receive notifications by right clicking on the main hand

Less duplication of work: NamespacedKey is created and reused when enabled. When the real interaction occurs, it only needs to be read and compared, without having to reinvent a set of identity rules.

Main class part 1·sequentially connected to the same source file

```
package de.teacherma.lesson37;

import java.util.Objects;
import net.kyori.adventure.text.Component;
import org.bukkit.NamespacedKey;
import org.bukkit.command.Command;
import org.bukkit.command.CommandSender;
import org.bukkit.entity.Player;
import org.bukkit.event.EventHandler;
import org.bukkit.event.EventPriority;
import org.bukkit.event.Listener;
import org.bukkit.event.block.Action;
import org.bukkit.event.player.PlayerInteractEvent;
import org.bukkit.inventory.EquipmentSlot;
import org.bukkit.plugin.java.JavaPlugin;

public final class CompassPlugin extends JavaPlugin implements Listener {
    private NamespacedKey compassKey;
    @Override
    public void onEnable() {
        compassKey = new NamespacedKey(this, "miner_compass");
        Objects.requireNonNull(getCommand("minercompass")).setExecutor(this);
        getServer().getPluginManager().registerEvents(this, this);
    }
}
```

10 / 13

lower half of compass main class

Immediately following the previous page. Check the sender, permissions and parameters before receiving it; put the compass that cannot fit in your backpack at your feet. The interactive method only does three filters and one prompt, and is short enough for you to explain line by line.

The lower half of the compass main class

Element	Explanation
Receiving	First check identity, permissions and parameters
Remaining	Full backpack is handled truthfully
Observation	Do not change the block, only send prompts

Installation verification: Run `.\gradlew.bat clean build`. After stopping the server, put the JAR of this lesson into plugins and restart the test server; execute `/minercompass` with a player authorized by the administrator.

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

```

@Override
public boolean onCommand(CommandSender sender, Command command,
                        String label, String[] args) {
    if (!(sender instanceof Player p)) {
        sender.sendMessage("Please get the compass in the game"); return true;
    }
    if (!p.hasPermission("blockacademy.compass.give")) {
        p.sendMessage("Please collect it after authorization from the teacher"); return true;
    }
    if (args.length != 0) { p.sendMessage("Usage: /minercompass"); return true; }
    var left = p.getInventory().addItem(CompassItems.create(compassKey));
    left.values().forEach(item -> p.getWorld().dropItemNaturally(p.getLocation(), item));
    p.sendMessage(left.isEmpty() ? "The compass has been included in the package, try
> right-clicking on the main hand" : "Compass at feet");
    return true;
}

@EventHandler(priority = EventPriority.MONITOR)
public void onUse(PlayerInteractEvent event) {
    if (event.getHand() != EquipmentSlot.HAND) return;
    Action action = event.getAction();
    if (action != Action.RIGHT_CLICK_AIR && action != Action.RIGHT_CLICK_BLOCK) return;
    if (!CompassItems.isOurs(event.getItem(), compassKey)) return;
    event.getPlayer().sendMessage(Component.text("Classroom tasks: Prepare tools and participate
> in the ore sprint competition!"));
}
}

```

11 / 13

real and fake compasses take the exam together

Only testing the true compass will miss most of the problems. Prepare an ordinary compass, a renamed compass and a compass for this lesson, and let the "fake students" also take the exam. Document expectations and actualities, and don't declare graduation just by seeing a sentence in the chat box.

Take the exam with both true and false compasses

Element	Explanation
Genuine items	Still identifiable after name change
Fake items with the same name	No recognition without PDC
Action Boundary	No prompts for empty hand, off-hand and left button

What is saved: The label is saved with the item data, which does not mean that the plug-in has saved the task progress. This lesson displays fixed task text and does not automatically remember how much ore you have mined.

Checklist

Test	Expected
True compass main hand right click air	Display a task
True Compass main hand right-click box	Prompt appears, box interaction is not changed by this lesson
Ordinary compass with the same name	Do not display tasks
Empty hand or left button	Do not display tasks
The true compass is only placed in the deputy hand	Do not display tasks
Receive full backpack	Place the remaining compass at your feet and prompt
Restart normally after saving items	Plug-ins with the same name and key still identify tags

12 / 13

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	Add two lines of lore
Iron Pickaxe	Test keys, types and values
Diamond Pickaxe	Design three tools with different labels

Next step: Lesson 38 combines commands, events, player data, and timers into an Ore Sprint.

Official extended reading: docs.papermc.io/paper/dev/pdc/

- Wooden pickaxe: Add two lines of lore to the compass and explain why `setItemMeta` is needed.
- Iron pickaxe: four identification tests of production material, key, type and value. Ordinary items cannot be accidentally triggered.
- Diamond Pickaxe: Design a three-piece miner set, with different labels to distinguish functions; indicate the behavior after receiving permissions and restarting.
- Acceptance requirements: Test real items, fake items with the same name, empty hands, off-hands, and renamed real items; one operation cannot be repeated with two hands.

Checklist

I can complete it independently	Mastered	Practice again
Ability to create and save ItemMeta	☐	☐
Can read and write PDC identity tags	☐	☐
Can handle main hand and right click actions	☐	☐

13 / 13

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/>