

第08课

生物群系导航器

一个选择，对照一张清楚的路线表



认识传统 switch 的刹车，读懂箭头式 switch 与返回值；清理输入、接住未知选项，最后制作八选一的控制台交通站。

阅读方式：先猜结果，再运行核对；每次只改一个条件。

本课使用 JDK 21，无需开启预览功能。故事与插画借用方块世界，练习运行在普通 Java 控制台。

01 路牌多到长成了丛林



史蒂夫在村口钉路牌：去沙漠向左，去丛林向右，去雪原再绕一下……写到第八块，藤蔓已经把他捆成“人形书签”。Alex 把目的地排成菜单：先看同一个选择值，再找对应标签。

switch 擅长整理固定名单：一个编号或词语，对照几种明确选项。生命是不是低于8，需要范围判断，仍适合 if；目的地编号等于哪一项，则适合 switch。

- 为什么传统冒号式 switch 常需要 break？
- 箭头式 switch 怎样保证不会落入下一条路线？
- 空白、大写与未知编号怎样得到清楚的提示？

第1到7课使用的 Java 11 环境能运行传统 switch。本课现代写法统一按 JDK 21 运行，编辑器的项目 JDK 也要设为21。目的地与物资是故事练习菜单，打印文字不会启动游戏传送。

02 传统 switch 的四个零件

switch 拿一个值来选，case 写明确标签，default 接住未列出的值。冒号式从命中的标签进入，break 负责离开 switch，之后继续执行外面的代码。

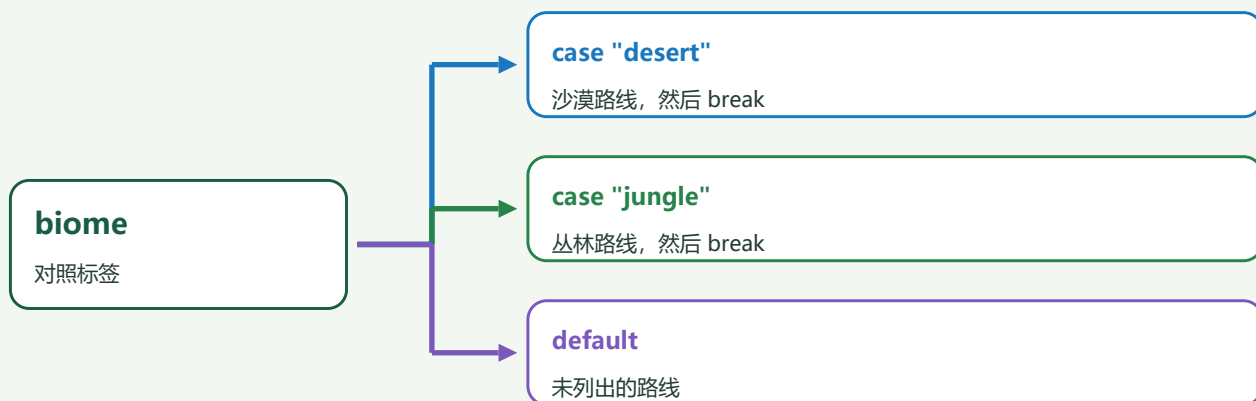
片段：放在 main 中，单独运行

```
String biome = "desert";
switch (biome) {
    case "desert":
        System.out.println("沙漠路线");
        break;
    case "jungle":
        System.out.println("丛林路线");
        break;
    default:
        System.out.println("未列出的路线");
}
System.out.println("路线检查结束");
```

运行核对

沙漠路线
路线检查结束

完整值与标签匹配



String 的 case 比较文字内容，但区分大小写。把 biome 改为 DESERT，不会命中 desert。后面的短例子分别放进 main 运行。

03 漏了 break，矿车就开过站



故意漏刹车：观察贯穿执行

```
int station = 2;
switch (station) {
    case 1: System.out.println("平原站");
    case 2: System.out.println("沙漠站");
    case 3: System.out.println("雪原站");
    default: System.out.println("维修站");
}
```

运行核对

沙漠站
雪原站
维修站

它只匹配了 case 2，却从那里一路执行下面的语句，这叫贯穿。后面的 case 标签并不会重新当作条件检查；连 default 的语句也可能被执行。

片段：放在 main 中，单独运行

```
int station = 2;
switch (station) {
    case 1: System.out.println("平原站"); break;
    case 2: System.out.println("沙漠站"); break;
    case 3: System.out.println("雪原站"); break;
    default: System.out.println("维修站");
}
```

运行核对

沙漠站

测试1、2、9，比较两版输出。贯穿是合法语法，能用于共享动作；需要“一票一站”的菜单时，漏 break 通常就是逻辑错误。

04 箭头式：选中一格就执行一格

case 后改用 `->`，这条规则完成后不会贯穿到下一条规则，不需要加 `break`。要多件事，就把它放进一对花括号。

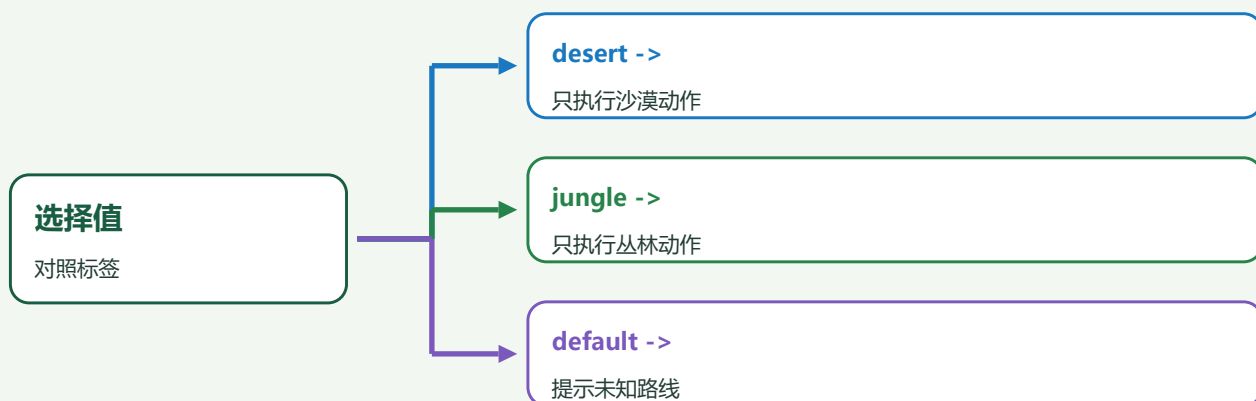
片段：放在 `main` 中，单独运行

```
String biome = "jungle";
switch (biome) {
    case "desert" -> System.out.println("沙漠路线");
    case "jungle" -> System.out.println("丛林路线");
    case "snow" -> System.out.println("雪原路线");
    default -> System.out.println("未知路线");
}
```

运行核对

丛林路线

箭头规则：执行后离开本次 `switch`



片段：放在 `main` 中，单独运行

```
int station = 4;
switch (station) {
    case 4 -> {
        System.out.println("雪原站开门");
        System.out.println("查看练习物资单");
    }
    default -> System.out.println("其他站台");
}
```

运行核对

雪原站开门
查看练习物资单

05 switch 表达式：交回一个结果

switch 还可以产生一个值，装进变量。每条路线都必须产生合适的结果，不能让某种输入拿到空宝箱。这里 default 保证菜单总有一个 String 结果。

片段：放在 main 中，单独运行

```
String biome = "desert";
String advice = switch (biome) {
    case "desert" -> "检查沙漠路线";
    case "jungle" -> "检查丛林路线";
    case "snow" -> "检查雪原路线";
    default -> "未知路线";
};
System.out.println(advice);
```

运行核对

检查沙漠路线

把选择结果装进变量



片段：放在 main 中，单独运行

```
String zone = "far";
int ticket = switch (zone) {
    case "near" -> 2;
    case "far" -> 5;
    default -> 0;
};
System.out.println("练习票价: " + ticket);
```

运行核对

练习票价：5

表达式末尾是 `;`，因为整句在声明并赋值。前一页只是 switch 语句，末尾右花括号后不需要这个分号。票价为练习积分，不代表真实游戏交易。

06 多句规则，用 yield 交回值

表达式的箭头后若有多句，用花括号包起来，再用 yield 交回结果。break 的意思是离开，yield 的意思是把这个结果交给外层 switch 表达式。

片段：放在 main 中，单独运行

```
String zone = "far";
int ticket = switch (zone) {
    case "near" -> 2;
    case "far" -> {
        System.out.println("正在准备远途路线");
        yield 5;
    }
    default -> 0;
};
System.out.println("票价=" + ticket);
```

运行核对

正在准备远途路线
票价=5

语句与表达式的任务不同

switch 语句

执行动作，可以打印多句。任务结束后往下走。

switch 表达式

产生一个值。每种输入都要有结果，结果类型必须兼容。

本课不用 switch 模式匹配或预览功能。先把固定数字与字符串菜单走熟：看选择值、看标签、看动作或返回值、看未知选项。

07 先清理车票，再找站台



玩家可能输入 JuNgLe，也可能多打空格。先 strip，再按 Locale.ROOT 转为小写，才能让大小写和两端空白不影响程序内部命令。

片段：放在 main 中，单独运行

```
String raw = " JuNgLe ";
String biome = raw.strip()
    .toLowerCase(java.util.Locale.ROOT);
String advice = switch (biome) {
    case "desert" -> "沙漠路线";
    case "jungle" -> "丛林路线";
    case "snow" -> "雪原路线";
    case "" -> "还没有输入";
    default -> "未知路线: " + biome;
};
System.out.println(advice);
```

运行核对

丛林路线

default 要把问题说清楚，别把拼错的 jungel 偷偷解释成平原。输入错误与普通默认建议，是两种不同任务；设计时先说明默认行为。

本课没有 case null 的 String switch，选择值为 null 时会抛出异常，default 不会代替 null 检查。nextLine 正常读取到的是字符串；没有可读行则是另一种输入异常。

08 几张标签，共用一条规则

箭头式可以用逗号把多个固定标签并列。传统冒号式则可以连续写几个 case 标签，让它们共用后面的语句。不要把逗号写法随手搬进旧版语法。

片段：放在 main 中，单独运行

```
String dimension = "nether";
String level = switch (dimension) {
    case "nether", "end" -> "练习高风险路线";
    case "overworld" -> "练习普通路线";
    default -> "未知维度";
};
System.out.println(level);
```

运行核对

练习高风险路线

八选一菜单：编号就是固定标签

编号	目的地	编号	目的地
1	平原	5	海洋
2	沙漠	6	沼泽
3	丛林	7	恶地
4	雪原	8	蘑菇岛

片段：放在 main 中，单独运行

```
int day = 7;
String plan = switch (day) {
    case 1, 2, 3, 4, 5 -> "课后编程";
    case 6, 7 -> "周末建造";
    default -> "日期错误";
};
System.out.println(plan);
```

运行核对

周末建造

09 站牌急诊室



标签要明确，不能抢位置

- case 2 重复出现会编译失败。每张相同标签只出现一次。
- 本课的固定值菜单中，case 标签必须是常量；普通可变的局部变量不是常量标签。
- 判断 `health <= 8` 这样的范围，用 `if`；别把比较条件塞进固定值 case。
- String switch 表达式若只列两个词、没有 `default`，不能保证每种输入都有结果。

片段：放在 `main` 中，单独运行

```
final int DESERT = 2;
int station = 2;
switch (station) {
    case DESERT -> System.out.println("沙漠站");
    default -> System.out.println("其他站");
}
```

运行核对

沙漠站

这里 `final int DESERT` 用常量表达式2初始化，可以作为标签。不能把所有 `final` 变量都当编译期常量。调试先读第一条相关报错，再查标签、`break` 和结果覆盖。

10 先预测，再运行，再解释

给选择留下证据

先预测

写出清理后的值与命中标签。



再运行

对照每一行实际输出。



能解释

说明为什么其他规则没执行。

片段：放在 main 中，单独运行

```
String biome = " SNOW ".strip()
    .toLowerCase(java.util.Locale.ROOT);
int boots = switch (biome) {
    case "snow" -> 1;
    default -> 0;
};
System.out.println(boots);
```

片段：放在 main 中，单独运行

```
int day = 6;
String plan = switch (day) {
    case 6, 7 -> "周末建造";
    case 1, 2, 3, 4, 5 -> "课后采集";
    default -> "日期错误";
};
System.out.println(plan);
```

核对：第一题输出1，清理后是 snow；第二题输出“周末建造”，6命中第一组。把 day 改成0、1、5、7、8，预测结果后再跑。

如果结果和预期不同，先确认选择值究竟是什么。输入清理错了，后面再漂亮的 switch 也只能选择错误路线。

11 项目蓝图：八选一交通站

一次只办一张票：显示完整菜单，读入编号，解析、选路、输出练习物资单。没有循环，要买第二张票就重新运行。

输入、选择、输出各管一件事



- 合法编号输出一个目的地，不得出现其他目的地。
- 0、9与其他越界整数：报告编号范围错误。
- abc、小数或装不进 int 的大数：报告整数输入错误。
- 输入 “ 3 ” 仍然选择丛林；0002与+5可以解析成2与5。

第一个 switch 的 default 交回空字符串作为内部无效记号，随后用 isEmpty 报错；它不会显示成目的地。第二个 switch 的 default 给合法路线提供通用练习物资，这次默认值不是输入错误。

12 交通站代码：菜单与目的地

新建 BiomeTransit.java，使用 JDK 21。将本页和下一页代码按顺序放进同一文件；本页最后的 switch 还没有结束。

BiomeTransit.java • 上半部分，下一页继续

```
import java.util.Scanner;

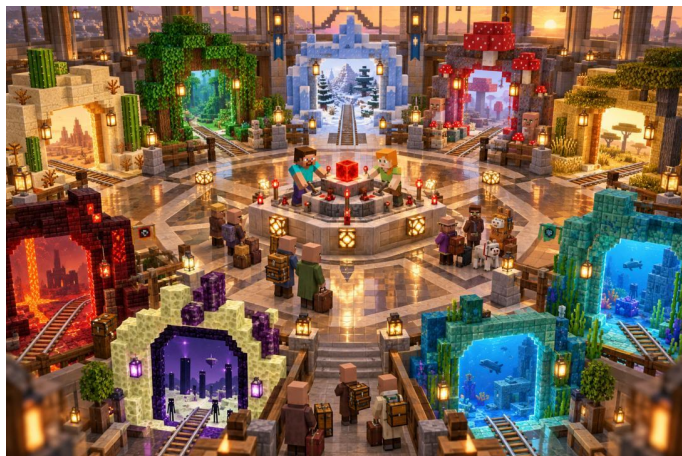
public class BiomeTransit {
    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        System.out.println("=== 生物群系交通站 ===");
        System.out.println("1 平原  2 沙漠  3 丛林  4 雪原");
        System.out.println("5 海洋  6 沼泽  7 恶地  8 蘑菇岛");
        System.out.println("目的地编号1到8: ");
        String text = in.nextLine().strip();
        try {
            int choice = Integer.parseInt(text);
            String destination = switch (choice) {
                case 1 -> "平原";
                case 2 -> "沙漠";
                case 3 -> "丛林";
                case 4 -> "雪原";
                case 5 -> "海洋";
                case 6 -> "沼泽";
                case 7 -> "恶地";
                case 8 -> "蘑菇岛";
                default -> "";
            };
        }
```

手推三张票：“3”清理成3并命中丛林；0走 default 并得到空字符串；abc 在 parseInt 处失败并跳到 catch。

13 车站代码：验证与物资

BiomeTransit.java • 紧接上一页

```
if (destination.isEmpty()) {
    System.out.println("编号必须在1到8之间");
} else {
    String supply = switch (choice) {
        case 2 -> "水和食物";
        case 3 -> "剪刀与火把";
        case 4 -> "皮靴";
        case 5 -> "船";
        default -> "床和食物";
    };
    System.out.println("目的地: " + destination);
    System.out.println("练习物资单: " + supply);
}
} catch (NumberFormatException e) {
    System.out.println("请输入整数编号, 例如3");
}
}
```



第二个 switch 只处理已经合法的编号。输入3应在完整菜单之后输出“目的地：丛林”和“练习物资单：剪刀与火把”。

物资单是故事任务，不是完整的生存攻略。“路线已准备”也只是程序文字；要把代码接进游戏，需要以后学习的 MOD 开发接口。

14 正常、边界、错误都验收

交通站验收表

输入	预期目的地或错误	练习物资单
1	平原	床和食物
2	沙漠	水和食物
3	丛林	剪刀与火把
4 / 5	雪原 / 海洋	皮靴 / 船
6 / 7 / 8	沼泽 / 恶地 / 蘑菇岛	床和食物
0 / 9 / -1	编号必须在1到8之间	不输出物资单
abc / 4.5 / 过大整数	请输入整数编号，例如3	不输出物资单

用输入证明 “一票一站”

- 八个合法编号各测一次，核对目的地与物资。
- 再测试空字符串、两端空白、0002、+5。
- 重复运行同一编号，输出应一致。
- 故意重复一张 case 标签，观察编译报错，再恢复。

不要只记录“成功”。写下输入、预测、实际输出；若多出了下一站，回头查传统写法的 break。若全部走 default，先检查值与标签是否一致。

15 给你的菜单加一颗钻石



木镐任务：四件工具说明书

接受 pickaxe、axe、shovel、hoe，分别输出工具用途；未知词报错。先清理空白和大小写，测试四个合法词、空字符串与 sword。

铁镐任务：一周任务表

读取1到7，1到5共用课后计划，6和7共用周末计划。0、8与 abc 分别检查范围或解析错误。解释多个标签为什么可以共用一个结果。

钻石镐任务：三维菜单加确认

选择 overworld、nether、end 后再读 yes/no。未知维度或确认词都报错，只有 yes 才显示准备完成。至少测试8组，覆盖空白、大小写、每个维度与未知词。

背包复盘：固定名单用 switch，范围条件用 if。冒号式留意 break，箭头式不贯穿；表达式必须交回结果。下一课用 for 把重复施工交给计数器。

技术核对：Oracle Java 21 语言指南，switch 表达式与语句。

<https://docs.oracle.com/en/java/javase/21/language/switch-expressions-and-statements.html>