

[Java SE](#) > [Java SE 规范](#) > [Java 语言规范](#)

Java ®语言规范

[下一个](#)

# Java ®语言规范

*Java SE 21 版*

詹姆斯·高斯林

比尔·乔伊

盖·斯蒂尔

吉拉德·布拉查

亚历克斯·巴克利

丹尼尔·史密斯

加文·比尔曼

2023-08-23

[法律声明](#)

## 目录

[二、简介](#)

- [1.1. 规范的组织](#)
- [1.2. 示例程序](#)
- [1.3. 符号](#)
- [1.4. 与预定义类和接口的关系](#)
- [1.5. 预览功能](#)
- [1.6. 反馈](#)
- [1.7. 参考](#)

## [2. 语法](#)

- [2.1. 上下文无关语法](#)
- [2.2. 词汇语法](#)
- [2.3. 句法语法](#)
- [2.4. 语法符号](#)

## [3. 词汇结构](#)

- [3.1. 统一码](#)
- [3.2. 词汇翻译](#)
- [3.3. Unicode 转义](#)
- [3.4. 线路终止符](#)
- [3.5. 输入元素和标记](#)
- [3.6. 空白](#)
- [3.7. 评论](#)
- [3.8. 身份标识](#)
- [3.9. 关键词](#)
- [3.10. 文字](#)
  - [3.10.1. 整数文字](#)
  - [3.10.2. 浮点文字](#)
  - [3.10.3. 布尔文字](#)
  - [3.10.4. 字符文字](#)
  - [3.10.5. 字符串文字](#)
  - [3.10.6. 文本块](#)
  - [3.10.7. 转义序列](#)
  - [3.10.8. 空文字](#)
- [3.11. 分离器](#)
- [3.12. 运营商](#)

## [4. 类型、值和变量](#)

- [4.1. 类型和值的种类](#)
- [4.2. 原始类型和值](#)

[4.2.1. 整数类型和值](#)[4.2.2. 整数运算](#)[4.2.3. 浮点类型和值](#)[4.2.4. 浮点运算](#)[4.2.5. 类型boolean和布尔值](#)[4.3. 参考类型和值](#)[4.3.1. 对象](#)[4.3.2. 班上Object](#)[4.3.3. 班上String](#)[4.3.4. 当引用类型相同时](#)[4.4. 类型变量](#)[4.5. 参数化类型](#)[4.5.1. 参数化类型的类型参数](#)[4.5.2. 参数化类型的成员和构造函数](#)[4.6. 类型擦除](#)[4.7. 可具体化的类型](#)[4.8. 原始类型](#)[4.9. 交叉口类型](#)[4.10. 子类型化](#)[4.10.1. 原始类型之间的子类型化](#)[4.10.2. 类和接口类型之间的子类型化](#)[4.10.3. 数组类型之间的子类型化](#)[4.10.4. 最小上界](#)[4.10.5. 类型投影](#)[4.11. 使用类型的地方](#)[4.12. 变量](#)[4.12.1. 原始类型变量](#)[4.12.2. 引用类型变量](#)[4.12.3. 变量的种类](#)[4.12.4. final变量](#)[4.12.5. 变量的初始值](#)[4.12.6. 类型、类和接口](#)[5. 转换和上下文](#)[5.1. 转换种类](#)[5.1.1. 身份转换](#)

[5.1.2. 扩大原语转换](#)[5.1.3. 缩小原始转换](#)[5.1.4. 扩大和缩小原始转换](#)[5.1.5. 扩大参考转换](#)[5.1.6. 缩小参考转换范围](#)[5.1.6.1. 允许缩小参考转换](#)[5.1.6.2. 选中和未选中的缩小参考转换](#)[5.1.6.3. 在运行时缩小引用转换范围](#)[5.1.7. 拳击转换](#)[5.1.8. 拆箱转换](#)[5.1.9. 未经检查的转换](#)[5.1.10. 捕捉转换](#)[5.1.11. 字符串转换](#)[5.1.12. 禁止转换](#)[5.2. 作业上下文](#)[5.3. 调用上下文](#)[5.4. 字符串上下文](#)[5.5. 选角环境](#)[5.6. 数字上下文](#)[6. 姓名](#)[6.1. 声明](#)[6.2. 名称和标识符](#)[6.3. 声明的范围](#)[6.3.1. 表达式中模式变量的范围](#)[6.3.1.1. 条件与运算符&&](#)[6.3.1.2. 条件或运算符||](#)[6.3.1.3. 逻辑补码运算符!](#)[6.3.1.4. 条件运算符?:](#)[6.3.1.5. 模式匹配运算符instanceof](#)[6.3.1.6. switch表达式](#)[6.3.1.7. 带括号的表达式](#)[6.3.2. 语句中模式变量的范围](#)[6.3.2.1. 积木](#)[6.3.2.2. if声明](#)[6.3.2.3. while声明](#)[6.3.2.4. do声明](#)[6.3.2.5. for声明](#)

[6.3.2.6. switch声明](#)

[6.3.2.7. 标签语句](#)

[6.3.3. case标签中模式变量的范围](#)

## [6.4. 阴影和模糊](#)

[6.4.1. 影子](#)

[6.4.2. 遮蔽](#)

## [6.5. 确定名字的含义](#)

[6.5.1. 根据上下文对名称进行句法分类](#)

[6.5.2. 上下文模糊名称的重新分类](#)

[6.5.3. 模块名称和包名称的含义](#)

[6.5.3.1. 简单的包名称](#)

[6.5.3.2. 合格的包名称](#)

[6.5.4. PackageOrTypeNames的含义](#)

[6.5.4.1. 简单包或类型名称](#)

[6.5.4.2. 限定的包或类型名称](#)

[6.5.5. 类型名称的含义](#)

[6.5.5.1. 简单类型名称](#)

[6.5.5.2. 限定类型名称](#)

[6.5.6. 表达式名称的含义](#)

[6.5.6.1. 简单的表达式名称](#)

[6.5.6.2. 限定表达式名称](#)

[6.5.7. 方法名称的含义](#)

[6.5.7.1. 简单的方法名称](#)

## [6.6. 访问控制](#)

[6.6.1. 确定可访问性](#)

[6.6.2. protected访问详情](#)

[6.6.2.1. 访问protected会员](#)

[6.6.2.2. 访问protected构造函数](#)

## [6.7. 完全限定名称和规范名称](#)

## [7. 包和模块](#)

[7.1. 套餐会员](#)[7.2. 主机对模块和包的支持](#)[7.3. 编译单位](#)[7.4. 包裹声明](#)[7.4.1. 命名包](#)[7.4.2. 未命名的包](#)[7.4.3. 包的可观察性和可见性](#)[7.5. 进口报关](#)[7.5.1. 单一类型导入声明](#)[7.5.2. 类型按需导入声明](#)[7.5.3. 单一静态导入声明](#)[7.5.4. 静态按需导入声明](#)[7.6. 顶级类和接口声明](#)[7.7. 模块声明](#)[7.7.1. 依赖关系](#)[7.7.2. 导出和打开的包裹](#)[7.7.3. 服务消费](#)[7.7.4. 提供服务](#)[7.7.5. 未命名模块](#)[7.7.6. 模块的可观察性](#)[8. 课程](#)[8.1. 类声明](#)[8.1.1. 类修饰符](#)[8.1.1.1. abstract课程](#)[8.1.1.2. sealed、non-sealed、和final类](#)[8.1.1.3. strictfp课程](#)[8.1.1.4. static课程](#)[8.1.2. 通用类和类型参数](#)[8.1.3. 内部类和封闭实例](#)[8.1.4. 超类和子类](#)[8.1.5. 超级接口](#)[8.1.6. 允许的直接子类](#)[8.1.7. 班级团体和成员声明](#)[8.2. 班级成员](#)

## [8.3. 字段声明](#)

### [8.3.1. 字段修改器](#)

#### [8.3.1.1. static领域](#)

#### [8.3.1.2. final领域](#)

#### [8.3.1.3. transient领域](#)

#### [8.3.1.4. volatile领域](#)

### [8.3.2. 字段初始化](#)

### [8.3.3. 初始化程序中字段引用的限制](#)

## [8.4. 方法声明](#)

### [8.4.1. 形式参数](#)

### [8.4.2. 方法签名](#)

### [8.4.3. 方法修饰符](#)

#### [8.4.3.1. abstract方法](#)

#### [8.4.3.2. static方法](#)

#### [8.4.3.3. final方法](#)

#### [8.4.3.4. native方法](#)

#### [8.4.3.5. strictfp方法](#)

#### [8.4.3.6. synchronized方法](#)

### [8.4.4. 通用方法](#)

### [8.4.5. 方法 结果](#)

### [8.4.6. 方法抛出](#)

### [8.4.7. 方法体](#)

### [8.4.8. 继承、覆盖和隐藏](#)

#### [8.4.8.1. 重写（通过实例方法）](#)

#### [8.4.8.2. 隐藏（通过类方法）](#)

#### [8.4.8.3. 覆盖和隐藏的要求](#)

#### [8.4.8.4. 继承具有重写等效签名的方法](#)

### [8.4.9. 超载](#)

## [8.5. 成员类和接口声明](#)

## [8.6. 实例初始化器](#)

## [8.7. 静态初始化器](#)

## [8.8. 构造函数声明](#)

### [8.8.1. 形式参数](#)

### [8.8.2. 构造函数签名](#)

[8.8.3. 构造函数修饰符](#)

[8.8.4. 泛型构造函数](#)

[8.8.5. 构造函数抛出](#)

[8.8.6. 构造函数的类型](#)

[8.8.7. 构造函数体](#)

[8.8.7.1. 显式构造函数调用](#)

[8.8.8. 构造函数重载](#)

[8.8.9. 默认构造函数](#)

[8.8.10. 防止类的实例化](#)

[8.9. 枚举类](#)

[8.9.1. 枚举常量](#)

[8.9.2. 枚举体声明](#)

[8.9.3. 枚举成员](#)

[8.10. 记录课程](#)

[8.10.1. 记录组件](#)

[8.10.2. 记录机构声明](#)

[8.10.3. 记录会员](#)

[8.10.4. 记录构造函数声明](#)

[8.10.4.1. 普通规范构造函数](#)

[8.10.4.2. 紧凑规范构造函数](#)

[9. 接口](#)

[9.1. 接口声明](#)

[9.1.1. 接口修饰符](#)

[9.1.1.1. abstract接口](#)

[9.1.1.2. strictfp接口](#)

[9.1.1.3. static接口](#)

[9.1.1.4. sealed和non-sealed接口](#)

[9.1.2. 通用接口和类型参数](#)

[9.1.3. 超级接口和子接口](#)

[9.1.4. 允许的直接子类和子接口](#)

[9.1.5. 接口主体和成员声明](#)

[9.2. 接口成员](#)

[9.3. 字段（常量）声明](#)

[9.3.1. 接口字段的初始化](#)



## [9.4. 方法声明](#)

### [9.4.1. 继承和重写](#)

#### [9.4.1.1. 重写 \(通过实例方法\)](#)

#### [9.4.1.2. 覆盖中的要求](#)

#### [9.4.1.3. 继承具有重写等效签名的方法](#)

### [9.4.2. 超载](#)

### [9.4.3. 接口方法体](#)

## [9.5. 成员类和接口声明](#)

## [9.6. 注释接口](#)

### [9.6.1. 注释界面元素](#)

### [9.6.2. 注释界面元素的默认值](#)

### [9.6.3. 可重复的注释接口](#)

### [9.6.4. 预定义注释接口](#)

#### [9.6.4.1. @Target](#)

#### [9.6.4.2. @Retention](#)

#### [9.6.4.3. @Inherited](#)

#### [9.6.4.4. @Override](#)

#### [9.6.4.5. @SuppressWarnings](#)

#### [9.6.4.6. @Deprecated](#)

#### [9.6.4.7. @SafeVarargs](#)

#### [9.6.4.8. @Repeatable](#)

#### [9.6.4.9. @FunctionalInterface](#)

## [9.7. 注释](#)

### [9.7.1. 普通注释](#)

### [9.7.2. 标记注释](#)

### [9.7.3. 单元素注释](#)

### [9.7.4. 注释可能出现的位置](#)

### [9.7.5. 同一接口的多个注解](#)

## [9.8. 功能接口](#)

## [9.9. 功能类型](#)

## [10. 数组](#)

### [10.1. 数组类型](#)

### [10.2. 数组变量](#)

### [10.3. 数组创建](#)

### [10.4. 数组访问](#)

- [10.5. 数组存储异常](#)
- [10.6. 数组初始化器](#)
- [10.7. 数组成员](#)
- [10.8. `Class`数组对象](#)
- [10.9. 字符数组不是`String`](#)

## [11. 例外情况](#)

### [11.1. 异常的种类和原因](#)

- [11.1.1. 例外的种类](#)
- [11.1.2. 异常的原因](#)
- [11.1.3. 异步异常](#)

### [11.2. 编译时异常检查](#)

- [11.2.1. 表达式异常分析](#)
- [11.2.2. 语句异常分析](#)
- [11.2.3. 异常检查](#)

### [11.3. 运行时异常处理](#)

## [12. 执行](#)

### [12.1. Java虚拟机启动](#)

- [12.1.1. 加载类`Test`](#)
- [12.1.2. 链接`Test`: 验证、准备、\(可选\) 解决](#)
- [12.1.3. 初始化测试: 执行初始化程序](#)
- [12.1.4. 调用`Test.main`](#)

### [12.2. 类和接口的加载](#)

- [12.2.1. 加载过程](#)
- [12.2.2. 类加载器一致性](#)

### [12.3. 类和接口的链接](#)

- [12.3.1. 二进制表示的验证](#)
- [12.3.2. 类或接口的准备](#)
- [12.3.3. 符号引用的解析](#)

### [12.4. 类和接口的初始化](#)

- [12.4.1. 初始化发生时](#)
- [12.4.2. 详细的初始化过程](#)

### [12.5. 创建新的类实例](#)

## [12.6. 类实例的最终确定](#)

### [12.6.1. 实施最终确定](#)

### [12.6.2. 与内存模型的交互](#)

## [12.7. 类和接口的卸载](#)

## [12.8. 程序退出](#)

# [13. 二进制兼容性](#)

## [13.1. 二进制的形式](#)

## [13.2. 二进制兼容性是什么和不是什么](#)

## [13.3. 包和模块的演变](#)

## [13.4. 阶级的演变](#)

### [13.4.1. abstract课程](#)

### [13.4.2. sealed、non-sealed、和final类](#)

#### [13.4.2.1. sealed课程](#)

#### [13.4.2.2. non-sealed课程](#)

#### [13.4.2.3. final课程](#)

### [13.4.3. public课程](#)

### [13.4.4. 超类和超级接口](#)

### [13.4.5. 类类型参数](#)

### [13.4.6. 班级团体和成员声明](#)

### [13.4.7. 访问成员和构建者](#)

### [13.4.8. 字段声明](#)

### [13.4.9. final字段和static常量变量](#)

### [13.4.10. static领域](#)

### [13.4.11. transient领域](#)

### [13.4.12. 方法和构造函数声明](#)

### [13.4.13. 方法和构造函数类型参数](#)

### [13.4.14. 方法和构造函数形式参数](#)

### [13.4.15. 方法结果类型](#)

### [13.4.16. abstract方法](#)

### [13.4.17. final方法](#)

### [13.4.18. native方法](#)

### [13.4.19. static方法](#)

### [13.4.20. synchronized方法](#)

### [13.4.21. 方法和构造函数抛出](#)

### [13.4.22. 方法和构造函数体](#)

### [13.4.23. 方法和构造函数重载](#)

### [13.4.24. 方法重写](#)

### [13.4.25. 静态初始化器](#)

[13.4.26. 枚举类的演变](#)

[13.4.27. 记录类的演变](#)

[13.5. 接口的演变](#)

[13.5.1. public接口](#)

[13.5.2. sealed和non-sealed接口](#)

[13.5.3. 超级接口](#)

[13.5.4. 接口成员](#)

[13.5.5. 接口类型参数](#)

[13.5.6. 字段声明](#)

[13.5.7. 接口方法声明](#)

[13.5.8. 注释接口](#)

[14. 块、语句和模式](#)

[14.1. 报表的正常和突然完成](#)

[14.2. 积木](#)

[14.3. 本地类和接口声明](#)

[14.4. 局部变量声明](#)

[14.4.1. 局部变量声明符和类型](#)

[14.4.2. 局部变量声明语句](#)

[14.5. 声明](#)

[14.6. 空洞的陈述](#)

[14.7. 标签语句](#)

[14.8. 表达式语句](#)

[14.9. 声明if](#)

[14.9.1. if-声明then](#)

[14.9.2. if-then-声明else](#)

[14.10. 声明assert](#)

[14.11. 声明switch](#)

[14.11.1. 开关块](#)

[14.11.1.1. 详尽的开关块](#)

[14.11.1.2. 确定运行时应用哪个开关标签](#)

[14.11.2. switch语句的 Switch 块](#)

[14.11.3. 执行switch声明](#)

[14.12. 声明while](#)

[14.12.1. 突然完成while声明](#)

## [14.13. 声明do](#)

### [14.13.1. 突然完成do声明](#)

## [14.14. 声明for](#)

### [14.14.1. 基本for声明](#)

#### [14.14.1.1. for语句初始化](#)

#### [14.14.1.2. for语句的迭代](#)

#### [14.14.1.3. 突然完成for声明](#)

### [14.14.2. 增强的for声明](#)

## [14.15. 声明break](#)

## [14.16. 声明continue](#)

## [14.17. 声明return](#)

## [14.18. 声明throw](#)

## [14.19. 声明synchronized](#)

## [14.20. 声明try](#)

### [14.20.1. 执行try-catch](#)

### [14.20.2. 执行try-finally和try-catch-finally](#)

### [14.20.3. try-有资源](#)

#### [14.20.3.1. 基础try资源](#)

#### [14.20.3.2. 扩展try资源](#)

## [14.21. 声明yield](#)

## [14.22. 无法访问的语句](#)

## [14.30. 图案](#)

### [14.30.1. 图案种类](#)

### [14.30.2. 模式匹配](#)

### [14.30.3. 模式的属性](#)

## [15. 表达式](#)

### [15.1. 评估、含义和结果](#)

### [15.2. 表达形式](#)

### [15.3. 表达式的类型](#)

### [15.4. 浮点表达式](#)

### [15.5. 表达式和运行时检查](#)

### [15.6. 正常和突然完成评估](#)

### [15.7. 评估顺序](#)

#### [15.7.1. 首先计算左手操作数](#)

[15.7.2. 运算前评估操作数](#)

[15.7.3. 求值尊重括号和优先级](#)

[15.7.4. 参数列表是从左到右计算的](#)

[15.7.5. 其他表达式的求值顺序](#)

## [15.8. 主要表达](#)

[15.8.1. 词汇文字](#)

[15.8.2. 类文字](#)

[15.8.3. `this`](#)

[15.8.4. 合格的`this`](#)

[15.8.5. 带括号的表达式](#)

## [15.9. 类实例创建表达式](#)

[15.9.1. 确定正在实例化的类](#)

[15.9.2. 确定封闭实例](#)

[15.9.3. 选择构造函数及其参数](#)

[15.9.4. 类实例创建表达式的运行时评估](#)

[15.9.5. 匿名类声明](#)

### [15.9.5.1. 匿名构造函数](#)

## [15.10. 数组创建和访问表达式](#)

[15.10.1. 数组创建表达式](#)

[15.10.2. 数组创建表达式的运行时求值](#)

[15.10.3. 数组访问表达式](#)

[15.10.4. 数组访问表达式的运行时求值](#)

## [15.11. 字段访问表达式](#)

[15.11.1. 使用主设备进行现场访问](#)

[15.11.2. 使用访问超类成员`super`](#)

## [15.12. 方法调用表达式](#)

[15.12.1. 编译时步骤 1: 确定要搜索的类型](#)

[15.12.2. 编译时步骤 2: 确定方法签名](#)

[15.12.2.1. 确定潜在适用的方法](#)

[15.12.2.2. 第 1 阶段: 通过严格调用识别适用的匹配参数方法](#)

[15.12.2.3. 第 2 阶段: 识别可通过松散调用应用的匹配数量方法](#)

[15.12.2.4. 第 3 阶段: 确定可变参数调用适用的方法](#)

[15.12.2.5. 选择最具体的方法](#)[15.12.2.6. 方法调用类型](#)[15.12.3. 编译时第 3 步：选择的方法是否合适？](#)[15.12.4. 方法调用的运行时评估](#)[15.12.4.1. 计算目标参考（如果需要）](#)[15.12.4.2. 评估论点](#)[15.12.4.3. 检查类型和方法的可访问性](#)[15.12.4.4. 找到要调用的方法](#)[15.12.4.5. 创建帧、同步、传输控制](#)[15.13. 方法参考表达式](#)[15.13.1. 方法引用的编译时声明](#)[15.13.2. 方法引用的类型](#)[15.13.3. 方法引用的运行时评估](#)[15.14. 后缀表达式](#)[15.14.1. 表达式名称](#)[15.14.2. 后缀增量运算符++](#)[15.14.3. 后缀自减运算符--](#)[15.15. 一元运算符](#)[15.15.1. 前缀增量运算符++](#)[15.15.2. 前缀减运算符--](#)[15.15.3. 一元加运算符+](#)[15.15.4. 一元减运算符-](#)[15.15.5. 按位补码运算符~](#)[15.15.6. 逻辑补码运算符!](#)[15.16. 强制转换表达式](#)[15.17. 乘法运算符](#)[15.17.1. 乘法运算符\\*](#)[15.17.2. 分部操作符/](#)[15.17.3. 余数运算符%](#)[15.18. 加法算子](#)[15.18.1. 字符串连接运算符+](#)[15.18.2. 数字类型的加法运算符 \(+和\)-](#)[15.19. 班次操作符](#)[15.20. 关系运算符](#)

[15.20.1. 数值比较运算符<, <=, >, 和>=](#)

[15.20.2. instanceof 操作员](#)

## [15.21. 等式运算符](#)

[15.21.1. 数值相等运算符==和!=](#)

[15.21.2. 布尔相等运算符==和!=](#)

[15.21.3. 参考相等运算符==和!=](#)

## [15.22. 按位和逻辑运算符](#)

[15.22.1. 整数按位运算符&, ^, 和|](#)

[15.22.2. 布尔逻辑运算符&, ^, 和|](#)

## [15.23. 条件与运算符&&](#)

## [15.24. 条件或运算符||](#)

## [15.25. 条件运算符?:](#)

[15.25.1. 布尔条件表达式](#)

[15.25.2. 数字条件表达式](#)

[15.25.3. 参考条件表达式](#)

## [15.26. 赋值运算符](#)

[15.26.1. 简单赋值运算符=](#)

[15.26.2. 复合赋值运算符](#)

## [15.27. 拉姆达表达式](#)

[15.27.1. 拉姆达参数](#)

[15.27.2. 拉姆达身体](#)

[15.27.3. Lambda 表达式的类型](#)

[15.27.4. Lambda 表达式的运行时求值](#)

## [15.28. switch表达式](#)

[15.28.1. switch表达式的 Switch 块](#)

[15.28.2. switch表达式的运行时求值](#)

## [15.29. 常量表达式](#)

# [16. 明确分配](#)

## [16.1. 明确的赋值和表达式](#)

[16.1.1. 布尔常量表达式](#)

[16.1.2. 条件与运算符&&](#)

[16.1.3. 条件或运算符||](#)



[16.1.4. 逻辑补码运算符!](#)

[16.1.5. 条件运算符?:](#)

[16.1.6. switch表达式](#)

[16.1.7. 其他类型的表达式boolean](#)

[16.1.8. 赋值表达式](#)

[16.1.9. 运营商++和--](#)

[16.1.10. 其他表达方式](#)

## [16.2. 明确的赋值和声明](#)

[16.2.1. 空语句](#)

[16.2.2. 积木](#)

[16.2.3. 本地类和接口声明](#)

[16.2.4. 局部变量声明语句](#)

[16.2.5. 标签语句](#)

[16.2.6. 表达式语句](#)

[16.2.7. if声明](#)

[16.2.8. assert声明](#)

[16.2.9. switch声明](#)

[16.2.10. while声明](#)

[16.2.11. do声明](#)

[16.2.12. for声明](#)

[16.2.12.1. for语句的初始化部分](#)

[16.2.12.2. for语句的增量部分](#)

[16.2.13. break、yield、continue、return和throw语句](#)

[16.2.14. synchronized声明](#)

[16.2.15. try声明](#)

## [16.3. 明确的赋值和参数](#)

## [16.4. 明确赋值和数组初始值设定项](#)

## [16.5. 明确赋值和枚举常量](#)

## [16.6. 明确的分配和匿名类](#)

## [16.7. 明确的分配以及成员类和接口](#)

## [16.8. 明确赋值和静态初始化器](#)

## [16.9. 明确赋值、构造函数和实例初始值设定项](#)

## [17. 线程和锁](#)

### [17.1. 同步](#)

### [17.2. 等待设置和通知](#)

[17.2.1. 等待](#)

[17.2.2. 通知](#)

[17.2.3. 干扰](#)

[17.2.4. 等待、通知和中断的交互](#)[17.3. 睡眠和产量](#)[17.4. 内存模型](#)[17.4.1. 共享变量](#)[17.4.2. 行动](#)[17.4.3. 节目和节目顺序](#)[17.4.4. 同步顺序](#)[17.4.5. 发生在订单之前](#)[17.4.6. 处决](#)[17.4.7. 格式良好的执行](#)[17.4.8. 执行和因果关系要求](#)[17.4.9. 可观察的行为和不间断的执行](#)[17.5. final 字段语义](#)[17.5.1. final 字段语义](#)[17.5.2. 施工期间的阅读final领域](#)[17.5.3. final 字段的后续修改](#)[17.5.4. 写保护字段](#)[17.6. 文字撕裂](#)[17.7. double 和的非原子处理long](#)[18. 类型推断](#)[18.1. 概念和符号](#)[18.1.1. 推理变量](#)[18.1.2. 约束公式](#)[18.1.3. 界限](#)[18.2. 减少](#)[18.2.1. 表达式兼容性约束](#)[18.2.2. 类型兼容性约束](#)[18.2.3. 子类型化约束](#)[18.2.4. 类型相等约束](#)[18.2.5. 检查异常约束](#)[18.3. 注册成立](#)[18.3.1. 互补的界限对](#)[18.3.2. 涉及捕获转换的界限](#)[18.4. 解决](#)

## [18.5. 推理的用途](#)

### [18.5.1. 调用适用性推断](#)

### [18.5.2. 调用类型推断](#)

#### [18.5.2.1. Poly 方法调用兼容性](#)

#### [18.5.2.2. 附加参数约束](#)

### [18.5.3. 函数式接口参数化推理](#)

### [18.5.4. 更具体的方法推断](#)

### [18.5.5. 记录模式类型推断](#)

## [19. 语法](#)

### [A. 有限许可授予](#)

# 示例列表

## [3.10.5-1. 字符串文字](#)

## [3.10.6-1. 文本块](#)

## [3.10.6-2. 文本块中的转义序列](#)

## [3.10.6-3. 文本块内容的转换顺序](#)

## [3.10.6-4. 文本块的计算结果为String](#)

## [4.2.2-1. 整数运算](#)

## [4.2.4-1. 浮点运算](#)

## [4.3.1-1. 对象创建](#)

## [4.3.1-2. 原始和参考身份](#)

## [4.4-1. 类型变量的成员](#)

## [4.5.1-1. 无界通配符](#)

## [4.5.1-2. 有界通配符](#)

## [4.8-1. 原始类型](#)

## [4.8-2. 原始类型和继承](#)

## [4.11-1. 类型的用法](#)

## [4.12.3-1. 不同类型的变量](#)

## [4.12.4-1. 最终变量](#)

## [4.12.5-1. 变量的初始值](#)

## [4.12.6-1. 变量的类型与对象的类](#)

## [5.0-1. 编译时和运行时的转换](#)

## [5.0-2. 不同环境下的转换](#)

## [5.1.2-1. 扩大原语转换](#)

## [5.1.3-1. 缩小原始转换](#)

## [5.1.3-2. 缩小丢失信息的原始转换](#)

## [5.2-1. 基本类型的赋值](#)

## [5.2-2. 引用类型的分配](#)

- 5.2-3. [数组类型的赋值](#)
- 5.5-1. [引用类型的转换](#)
- 5.5-2. [数组类型的转换](#)
- 5.5-3. [在运行时转换不兼容的类型](#)
- 5.6-1. [一元数字提升](#)
- 5.6-2. [二进制数字提升](#)
- 6.1-1. [独特的包名称](#)
- 6.1-2. [独特的模块名称](#)
- 6.1-3. [描述性类名](#)
- 6.1-4. [常规类型变量名称](#)
- 6.3-1. [类声明的范围](#)
- 6.3-2. [局部变量声明的范围](#)
- 6.4-1. [尝试隐藏局部变量](#)
- 6.4.1-1. [局部变量声明对字段声明的遮蔽](#)
- 6.4.1-2. [一个类型声明被另一个类型声明遮蔽](#)
- 6.5.2-1. [上下文模糊名称的重新分类](#)
- 6.5.5.1-1. [对类型参数的引用](#)
- 6.5.5.2-1. [限定类型名称](#)
- 6.5.6.1-1. [简单的表达式名称](#)
- 6.5.6.1-2. [对实例变量的引用](#)
- 6.5.6.1-3. [对局部变量和形式参数的引用](#)
- 6.5.6.2-1. [限定表达式名称](#)
- 6.5.6.2-2. [使用类型名称限定表达式](#)
- 6.5.7.1-1. [简单的方法名称](#)
- 6.6-1. [访问控制](#)
- 6.6-2. [访问public字段、方法和构造函数](#)
- 6.6-3. [进入课堂public和非public课堂](#)
- 6.6-4. [通过包访问来访问字段、方法和构造函数](#)
- 6.6-5. [访问private字段、方法和构造函数](#)
- 6.6.2-1. [访问protected字段、方法和构造函数](#)
- 6.7-1. [完全限定名称](#)
- 6.7-2. [完全限定名称与规范名称](#)
- 7.4.2-1. [未命名包](#)
- 7.5.1-1. [单一类型导入](#)
- 7.5.1-2. [重复的类声明](#)
- 7.5.1-3. [不导入子包](#)
- 7.5.1-4. [导入同时也是包名称的类型名称](#)
- 7.5.2-1. [按需类型导入](#)
- 7.6-1. [顶级类和接口声明冲突](#)
- 7.6-2. [顶级类和接口的范围](#)
- 7.6-3. [完全限定名称](#)
- 7.1.1-1. [指令的决议requires transitive](#)

- 8.1.1.1-1. [抽象类声明](#)
- 8.1.1.1-2. [禁止子类的抽象类声明](#)
- 8.1.2-1. [相互递归类型变量界限](#)
- 8.1.2-2. [嵌套泛型类](#)
- 8.1.3-1. [内部类声明和静态成员](#)
- 8.1.3-2. [内部类声明](#)
- 8.1.4-1. [直接超类和子类](#)
- 8.1.4-2. [超类和子类](#)
- 8.1.4-3. [阶级取决于自身](#)
- 8.1.5-1. [非法超级接口](#)
- 8.1.5-2. [超级接口](#)
- 8.1.5-3. [接口的非法多重继承](#)
- 8.1.5-4. [超级接口的实现方法](#)
- 8.2-1. [班级成员的使用](#)
- 8.2-2. [具有包访问权限的类成员的继承](#)
- 8.2-3. [类成员的继承](#)`public``protected`
- 8.2-4. [类成员的继承](#)`private`
- 8.2-5. [访问不可访问类的成员](#)
- 8.3-1. [多重继承字段](#)
- 8.3-2. [字段的重新继承](#)
- 8.3.1.1-1. [static领域](#)
- 8.3.1.1-2. [隐藏类变量](#)
- 8.3.1.1-3. [隐藏实例变量](#)
- 8.3.1.3-1. [字段的持久性](#)`transient`
- 8.3.1.4-1. [volatile领域](#)
- 8.3.2-1. [字段初始化](#)
- 8.3.2-2. [对类变量的前向引用](#)
- 8.3.3-1. [字段引用的限制](#)
- 8.4.2-1. [覆盖等效签名](#)
- 8.4.3.1-1. [抽象/抽象方法重写](#)
- 8.4.3.1-2. [抽象/非抽象覆盖](#)
- 8.4.3.6-1. [synchronized显示器](#)
- 8.4.3.6-2. [synchronized方法](#)
- 8.4.6-1. [将变量类型作为引发的异常类型](#)
- 8.4.8-1. [遗产](#)
- 8.4.8.1-1. [压倒一切](#)
- 8.4.8.1-2. [压倒一切](#)
- 8.4.8.2-1. [隐藏类方法的调用](#)
- 8.4.8.3-1. [协变返回类型](#)
- 8.4.8.3-2. [来自返回类型的未检查警告](#)
- 8.4.8.3-3. [不正确的覆盖，因为throws](#)
- 8.4.8.3-4. [擦除影响覆盖](#)

- 8.4.8.4-1. [重写等效方法的继承](#)
- 8.4.9-1. [超载](#)
- 8.4.9-2. [重载、覆盖和隐藏](#)
- 8.8-1. [构造函数声明](#)
- 8.8.7-1. [构造函数体](#)
- 8.8.7.1-1. [对显式构造函数调用语句的限制](#)
- 8.8.7.1-2. [合格的超类构造函数调用](#)
- 8.8.9-1. [默认构造函数](#)
- 8.8.9-2. [构造函数与类的可访问性](#)
- 8.8.10-1. [通过构造函数可访问性防止实例化](#)
- 8.9.2-1. [枚举体声明](#)
- 8.9.2-2. [对枚举常量自引用的限制](#)
- 8.9.3-1. [使用增强循环迭代枚举for常量](#)
- 8.9.3-2. [切换枚举常量](#)
- 8.9.3-3. [具有类体的枚举常量](#)
- 8.9.3-4. [多个枚举类](#)
- 9.3-1. [不明确的继承字段](#)
- 9.3-2. [多重继承字段](#)
- 9.3.1-1. [对字段的前向引用](#)
- 9.4.2-1. [重载abstract方法声明](#)
- 9.6.1-1. [注解接口声明](#)
- 9.6.1-2. [标记注解接口声明](#)
- 9.6.1-3. [单元素注解接口声明](#)
- 9.6.2-1. [带默认值的注解接口声明](#)
- 9.6.3-1. [格式错误的包含注解接口](#)
- 9.6.3-2. [限制注解可能重复的位置](#)
- 9.6.3-3. [可重复的包含注解接口](#)
- 9.7.1-1. [普通注解](#)
- 9.7.2-1. [标记注解](#)
- 9.7.3-1. [单元素注解](#)
- 9.8-1. [功能接口](#)
- 9.8-2. [功能接口和擦除](#)
- 9.8-3. [通用功能接口](#)
- 9.9-1. [功能类型](#)
- 9.9-2. [通用函数类型](#)
- 10.2-1. [数组变量的声明](#)
- 10.2-2. [数组变量和数组类型](#)
- 10.4-1. [数组访问](#)
- 10.5-1. [ArrayStoreException](#)
- 10.6-1. [数组初始化器](#)
- 10.7-1. [数组是可克隆的](#)
- 10.7-2. [克隆后的共享子数组](#)

- 10.8-1. [Class数组对象](#)
- 10.8-2. [数组Class对象是共享的](#)
- 11.2.3-1. [捕获检查异常](#)
- 11.3-1. [抛出和捕获异常](#)
- 12.4.1-1. [超类在子类之前初始化](#)
- 12.4.1-2. [仅声明字段的类static被初始化](#)
- 12.4.1-3. [接口初始化不会初始化超级接口](#)
- 12.5-1. [实例创建评估](#)
- 12.5-2. [实例创建期间动态调度](#)
- 13.4.4-1. [改变超类](#)
- 13.4.4-2. [引入超类](#)
- 13.4.6-1. [改变班级主体](#)
- 13.4.6-2. [改变超类](#)
- 13.4.7-1. [改变可访问性](#)
- 13.4.8-1. [添加字段声明](#)
- 13.4.9-1. [将变量更改为final](#)
- 13.4.16-1. [改变方法abstract](#)
- 13.4.17-1. [改变方法final](#)
- 13.4.23-1. [添加重载方法](#)
- 13.5.4-1. [删除接口成员](#)
- 13.5.7-1. [添加默认方法](#)
- 14.3-1. [本地类声明](#)
- 14.4-1. [声明的局部变量var](#)
- 14.4.1-1. [声明的局部变量的类型var](#)
- 14.7-1. [标签和标识符](#)
- 14.11.3-1. [声明中的失败switch](#)
- 14.13-1. [声明do](#)
- 14.14-1. [增强型for和阵列](#)
- 14.14-2. [增强for和拆箱转换](#)
- 14.15-1. [声明break](#)
- 14.16-1. [声明continue](#)
- 14.19-1. [声明synchronized](#)
- 14.20.1-1. [捕获异常](#)
- 14.20.2-1. [处理未捕获的异常finally](#)
- 14.21-1. [声明yield](#)
- 14.22-1. [条件编译](#)
- 15.7.1-1. [首先计算左侧操作数](#)
- 15.7.1-2. [复合赋值运算符中的隐式左手操作数](#)
- 15.7.1-3. [突然完成左侧操作数的求值](#)
- 15.7.2-1. [运算前操作数的评估](#)
- 15.7.4-1. [方法调用时的评估顺序](#)
- 15.7.4-2. [参数表达式的突然完成](#)

- 15.8.3-1. [表达式this](#)
- 15.9.4-1. [评估顺序和内存不足检测](#)
- 15.10.2-1. [数组创建评估](#)
- 15.10.2-2. [多维数组创建](#)
- 15.10.2-3. [OutOfMemoryError和维度表达评估](#)
- 15.10.4-1. [首先评估数组引用](#)
- 15.10.4-2. [数组参考评估的突然完成](#)
- 15.10.4-3. [null数组参考](#)
- 15.11.1-1. [字段访问的静态绑定](#)
- 15.11.1-2. [接收器变量与现场访问无关static](#)
- 15.11.2-1. [表达式super](#)
- 15.12.2-1. [方法适用性](#)
- 15.12.2-2. [选择方法时不考虑返回类型](#)
- 15.12.2-3. [选择最具体的方法](#)
- 15.12.4.1-1. [目标参考文献和static方法](#)
- 15.12.4.1-2. [方法调用期间的评估顺序](#)
- 15.12.4.4-1. [重写和方法调用](#)
- 15.12.4.4-2. [方法调用使用super](#)
- 15.12.4.5-1. [调用的方法签名与编译时方法签名的擦除方式不同](#)
- 15.17.3-1. [整数余数运算符](#)
- 15.17.3-2. [浮点余数运算符](#)
- 15.18.1-1. [字符串连接](#)
- 15.18.1-2. [字符串连接和条件](#)
- 15.20.2-1. [类型比较运算符](#)
- 15.26.1-1. [对数组组件的简单赋值](#)
- 15.26.2-1. [对数组组件的复合赋值](#)
- 15.26.2-2. [复合赋值左侧的值在评估右侧之前保存](#)
- 15.29-1. [常量表达式](#)
- 16-1. [明确赋值考虑语句和表达式的结构](#)
- 16-2. [明确赋值不考虑表达式的值](#)
- 16-3. [明确取消分配](#)
- 17.4-1. [不正确同步的程序可能会表现出令人惊讶的行为](#)
- 17.4.5-1. [发生在一致性之前](#)
- 17.4.8-1. [发生在一致性之前是不够的](#)
- 17.5-1. [finalJava 内存模型中的字段](#)
- 17.5-2. [final安全领域](#)
- 17.5.3-1. [积极优化final领域](#)
- 17.6-1. [单词撕裂检测](#)
- 18.5.5-1. [记录模式类型推断](#)



---

[法律声明](#)