1. 方法调用
2. 概述

方法调用并不等同于方法中的代码被执行，方法调用阶段的唯一任务就是确定被调用方法的版本（即调用哪一个版本）。由于Class文件的编译过程中不包含连接步骤，因此方法调用在Class文件中存储的只是符号引用，而不是方法在实际运行时内存布局中的入口地址（直接引用）。这就让Java有了更强的动态扩展能力，某些调用需要在类加载甚至运行时才能确定直接引用。

1. 解析
2. 概述

在类加载的解析阶段，能够将部分符号引用转换为直接引用，这种解析能够成立的前提是：方法在编译时就已经确定了调用的版本，且该版本在运行期间是不可变的，这种方法的调用称为解析。

1. 适用方法

对于静态方法，私有方法这种不会通过继承或者别的方式重写出其他版本的方法，适合在类加载的解析阶段进行解析。同时只要能够被invokestatic和invokespecial指令调用的方法，都可以进行解析，如静态方法，私有方法，实例构造器，父类方法；还有被final修饰的方法，被invokevirtual指令调用。这五种方法统称为非虚方法。

1. 分派
2. 概述

分派是另一种方法调用形式，可能是静态的也可能是动态的，按照分派依据的宗量数分为单分派和多分派，因此组合成了静态单分派，静态多分派，动态单分派，动态多分派。

1. 静态分派

静态分派与方法的重载是相关的，我们设置两个类，一个类是父类Human，一个是子类Man，那么当我们用以下的方式初始化一个对象时，Human表示该变量的静态类型，Man表示该变量的时机类型：

Human man = new Man();

静态类型和实际类型在程序中都可能发生变化，区别是静态类型的变化仅仅在使用时发生，且变量本身的静态类型不会被改变，且静态类型在编译器是可知的；而实际类型变化的结果在运行期才能够确定。

在方法重载时，使用哪个重载版本取决于传入参数的数量和数据类型，编译器在重载时是通过参数的静态类型而不是实际类型来决定重载方法的，静态类型在编译期是可知的。所有依赖于静态类型来决定方法执行版本的分派动作，都称为静态分派。

方法重载时编译器会不断地寻找合适的方法，如果静态类型所对应的方法没有找到，就会从该静态类型能够转换的类型开始寻找，如果这种方法还没有找到，就会进行自动装箱找它的封装类型，如果还没有找到，就会寻找该装箱类所实现的接口类型或从下往上地寻找父类。

1. 动态分派

动态分派与重写是相关的，当两个子类都重写了父类的某一个方法时，编译器如何分辨这两个方法呢？从Class字节码的指令中可以看出，两个子类对象首先被压入栈顶，他们是将要执行重写方法的所有者，称为接受者。然后执行invokespecial指令，参数是父类方法的符号引用，并不是子类方法的符号引用，也就是说是invokespecial指令搞的鬼。

invokespecial指令首先找到操作数栈顶的第一个元素所指向的对象的实际类型，如果在该类型中找到与指令参数中的父类符号引用中的描述符和简单名称都相符的方法，则进行访问权限校验，如果通过则返回这个方法的直接引用，否则返回java.lang.IllegalAccessError异常。如果没有找到该方法，则按照继承关系从下往上对该类型的各个父类进行搜索和验证。如果最终还是没有找到合适的方法，则抛出java.lang.AbstractMethodError异常。

这种在运行期根据实际类型确定方法执行版本的分派过程称为动态分派。动态分派是由虚方法调用指令invokespecial实现的，因此只对方法有效，对字段是无效的，因为字段无法使用这条指令，字段是不参与多态的。当子类声明了父类同名的字段时，子类的字段会覆盖父类的同名字段。

1. 单分派和多分派

方法的接受者和方法的参数统称为方法的宗量，根据宗量的数量，将分派分为单分派和多分派两种，单分派是根据一个宗量对目标方法进行选择，多分派是根据多个宗量选择。Java语言的静态分派是根据静态类型和方法参数来选择的，属于多分派类型；动态分派是根据接受者的实际类型选择的，因此属于单分派类型。

1. 虚拟机多分派的实现

动态分派执行地非常频繁，而且动态分派的方法版本选择过程需要运行时在接受者类型的方法元数据中搜索合适的目标方法，因此基于性能的考虑，在方法区中建立一个虚方法表（vtable），接口对应建立接口方法表（itable），使用方法表索引来代替元数据查找以提高性能。

虚方法表中存放着各个方法的实际入口地址，如果某个方法在子类中没有被重写，则子类的虚方法表中的地址入口和父类相同方法的地址入口是一致的，都指向父类的实现入口。为了实现方便，具有相同签名的方法，在父类，子类的虚方法表中都应当具有同样的索引序号，这样在类型转换时仅需要变更查找的虚方法表，就可以从不同的虚方法表中按照索引转换出需要的入口地址。虚方法表一般在类加载的连接阶段初始化， 在初始化了类的变量后，初始化虚方法表，同时Java中的方法默认是虚方法。