一个表达式计算案例的设计和实现 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/144/2021_2022__E4_B8_80_E4_B8_AA_E8_A1_A8_E8_c104_144464.htm 问题由来 在我做过的一个针对网络设备和主机的数据采集系统中，某些采集到的数据需要经过一定的计算后才保存入库，而不是仅仅保存其原始值。为了提供给用户最大的灵活性，我设想提供一个用户界面，允许用户输入计算表达式（或者称为计算公式）。这样，除了需要遵从少量的规则，用户可以得到最大的灵活性。 这样的表达式具有什么特点呢？它一般不是纯的可立即计算的表达式（简单的如：1 2*3-4）。它含有我称为变量的元素。变量一般具有特殊的内定的语法，例如可能用"@totalmemory"表示设备或主机（下面简称为设备）的物理内存总数，那么表达式"(@totalmemory-@freememory)/@totalmemory*100"就表示设备当前内存使用率百分比。如果与告警系统联系起来，监测此值超过80系统就发出Warning，那么这就成为一件有意义的事情。不同种类的采集数据入库前可能需要经过复杂度不同的计算。但显然，最后求值的时候，必须将那些特定的变量用具体数值（即采集到的具体数值）替换，否则表达式是不可计算的。这个过程是在运行时发生的。问题的一般性 我认为表达式计算是个一般性的话题，并且也不是一个新的话题。我们可能在多处碰到它。我在读书的时候编写过一个表达式的转换和计算程序，当时作为课余作业。我看到过一些报表系统，不管它是单独的，还是包含在MIS系统、财务软件中，很多都支持计算公式。我认为这些系统中的计算公式和

我所遇到的问题是大致相同的。对我来说，我在数据采集项目中遇到这个问题，下次可能还会在其他项目中遇到它。既然已经不止一次了，我希望找到一个一般性的解决方案。一些已有的设计和实现不能满足要求 在设计和实现出第一个版本之后，我自己感觉不很满意。随后我花点时间上网搜索了一下（关键字：表达式 中缀 后缀 or Expression Infix Postfix）。令人稍感失望的是，所找到的一些关于表达式的转换、计算的程序不能满足我的要求。不少这样的程序仅仅支持纯的可立即计算的表达式，不支持变量。而且，表达式解析和计算是耦合在一起的，很难扩展。增加新的运算符（或新的变量语法）几乎必定要修改源代码。在我看来，这是最大的缺陷了（实际上，当年我编写的表达式转换和计算的程序，虽然当时引以自豪，但是现在看来也具有同样的缺陷）。但是，表达式的转换和计算本身有成熟的、基于栈的的经典算法，许多计算机书籍或教材上都有论述。人们以自然方式书写的表达式是中缀形式的，先要把中缀表达式转换为后缀表达式，然后计算后缀表达式的值。我打算仍然采用这个经典的过程和算法。我的设计想法和目标 既然表达式的转换和计算的核心算法是成熟的，我渴望把它们提取出来，去除（与解析相关的）耦合性！试想，如果事物具有相对完整的内涵和独立性，会产生这个需要，并且也能够通过形式上的分离和提取来把内涵给表现出来，这个过程离不开抽象。我不久意识到自己实际上在设计一个小规模的关于表达式计算的框架。 表达式要支持加减乘除运算符，这是基本的、立即想到的。或许还应该支持平方，开方（sqrt），三角运算符如sin，cos等。那么如果还有其它怎么办，包括自定义的运算符？

你能确定考虑完备了吗？像自定义的运算符，是完全存在的、合理的需求。在数据采集系统中，我一度考虑引入一个diff运算符，表明同一个累加型的采集量，在相距最近的两次（即采集周期）采集的差值。以上的思考促使我决定，运算符的设计必须是开放的。用户（这里指的是用户程序员，下同）可以扩展，增加新的运算符。 表达式中允许含有变量。对于变量的支持贯穿到表达式解析，转换，计算的全过程。在解析阶段，应该允许用户使用适合他/她自己的变量语法，我不应该事先实现基于某种特定语法的变量识别。 由于支持可扩展的运算符，未知的变量语法，甚至连基本的数值（象123，12.3456，1.21E17）理论上也有多种类型和精度（Integer/Long/Float/Double/BigInteger/BigDecimal），这决定了无法提供一个固化的表达式解析方法。表达式解析也是需要可扩展的。最好的结果是提供一个容易使用和扩展的解析框架。对于新的运算符，新的变量语法，用户在这个框架上扩展，以提供增强的解析能力。从抽象的角度来看，我打算支持的表达式仅由四种元素组成：括号（包括左右括号），运算符，数值和变量。一个最终用户给出的表达式字符串，解析通过后，可能生成了一个内部表示的、便于后续处理的表达式，组成这个表达式的每个元素只能是以上四种之一。 数值 一开始我写了一个表达数值的类，叫做Numeral。我为Numeral具体代表的是整数、浮点数还是双精度数而操心。从比较模糊的意义上，我希望它能表达以上任何一种类型和精度的数值。但是我也希望，它能够明确表达出代表的具体是哪种类型和精度的数值，如果需要的话。甚至我想到Numeral最好也能表达BigInteger和BigDecimal（设想恰巧在

某种场合下，我们需要解析和计算一个这样的表达式，它允许的数值的精度和范围很大，以至于Long或Double容纳不下），否则在特别的场合下我们将遇到麻烦。在可扩展性上，数值类不大像运算符类，它应该是成熟的，因而几乎是不需要扩展的。 经过反复尝试的混乱（Numeral后来又经过修改甚至重写），我找到了一个明智的办法。直接用java.lang.Number作为数值类（实际上这是一个接口）。我庆幸地看到，在Java中，Integer，Long，Float，Double甚至BigInteger，BigDecimal等数值类都实现了java.lang.Number（下面简称Number）接口，使用者把Number作为何种类型和精度来看待和使用，权利掌握在他/她的手中，我不应该提前确定数值的类型和精度。选择由Number类来表达数值，这看来是最好的、代价最小的选择了，并且保持了相当的灵活性。作为一个顺带的结果，Numeral类被废弃了。括号 在表达式中，括号扮演的角色是不可忽视的。它能改变运算的自然优先级，按照用户所希望的顺序进行计算。我用Bracket类来表示括号，这个类可以看作是final，因为它不需要扩展。括号分作括号和右括号，我把它们作为Bracket类的两个静态实例变量（并且是Bracket类仅有的两个实例变量）。public class Bracket{ private String name. private Bracket(String name) { this.name = name. } public static final Bracket LEFT_BRACKET = new Bracket("("), RIGHT_BRACKET = new Bracket(")"). public String toString() { return name. }} 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com