Java表达式计算工具-Aviator的使用

添加依赖:

```
<!-- https://mvnrepository.com/artifact/com.googlecode.aviator/aviator -->
<dependency>
    <groupId>com.googlecode.aviator</groupId>
    <artifactId>aviator</artifactId>
    <version>5.3.0</version>
</dependency>
```

Aviator 有两种常用的使用方式：

- 直接在Java代码中使用(表达式运算)
- 在.av脚本文件中使用

直接在Java代码中

基本使用: 引入依赖直接在Java代码中传入需要计算的表达式和变量的值

```
@Test
public void test1() {
    Map<String, Object> paramMap = new HashMap<>();
    paramMap.put("x", 12);
    paramMap.put("y", 10);
    String expression = "x + y";
    Long value = (Long) AviatorEvaluator.compile(expression).execute(paramMap);
    Assert.isTrue(NumberUtil.equals(value, 22L));
    log.info("计算成功： {} = {}", expression, value);
}
```

如上面代码所示：基本使用非常简单。直接使用`AviatorEvaluator`类提供的方法即可， 使用Map<String, Object>
来传递表达式的参数。需要注意的是，传递参数的Map的Value类型必须是Object，
且计算返回的值默认是Object类型，需要根据代码上下文转换为所需要的类型。

av脚本中使用

先写一个.av的脚本， 语法和JS有点相似

脚本目录：resources/av/test1.av

```
let a = 1;
let c = -2;

println(a + b);
```

再在Java代码中调用执行

```
@Test
public void testSimpleAvScript() {
    Expression expression =
AviatorEvaluator.getInstance().compileScript("av/test1.av");
    expression.execute();
}
```

Aviator计算数字类型转换

- 传入整数：无论Java代码中传入的整数是byte/stort/int/long 都会被转换成long，对于比long大的数会被转换成BigInteger, 若表达式中有数字以N结尾，Aviator自动识别为BigInteger。
- 传入浮点数：无论Java代码中传入float/double都会被转换成double。
- 精确计算：Aviator中的BigDecimal对应BigDecimal，当在Java中传入BigDecimal类型的数或表达式中有以M结尾的数都会被解析成BigDecimal。

- 单一类型参与的运算，结果仍然为该类型，比如整数和整数相除仍然是整数，double 和 double 运算结果还是 double。
- 多种类型参与的运算，按照下列顺序： `long -> bigint -> decimal -> double` 自动提升，比如 long 和 bigint 运算结果为 bigint， long 和 decimal 运算结果为 decimal，任何类型和 double 一起运算结果为 double

参考文档：

AviatorScript相关语法详情请: https://www.yuque.com/boyan-avfmj/aviatorscript/lvabnw