Kepler - Bug #6439

Double rounding fails in some cases while evaluating Expressions

03/04/2014 04:24 AM - Owsiak Michal

Status:	Closed	**Start date:**	03/04/2014
Priority:	Urgent	**Due date:**	
Assignee:	Christopher Brooks	**% Done:**	100%
Category:	actors	**Estimated time:**	0.00 hour
Target version:	2.3.0	**Spent time:**	0.20 hour
Bugzilla-Id:			

Description

It seems that addition of doubles can produce values slightly different than they should to be.

Please take a look at attached workflow (simple_error.xml).

Condition that should be satisfied to escape the loop is: 1.7 > 1.5 + 0.1

However, loop is interrupted sooner, because of incorrect calculation of doubles. Value of "p" is set to: 1.6000000000000003

This, of course, makes it impossible to use doubles as check points for the loops.

However, it seems that casting to string and back works fine (take a look at second workflow - simple.xml)

Cheers

Michal

History

#1 - 03/04/2014 08:34 AM - Christopher Brooks

- Status changed from New to Closed

- Assignee changed from Derik Barseghian to Christopher Brooks

- % Done changed from 0 to 100

It is almost never a good idea to compare doubles without using an epsilon factor. Because of rounding, doubles are unlikely to precisely represent a value.

One workaround in the expression language is to use

close(value1, value2)

which is defined as:

Return true if the first argument
is close to the second (within
EPSILON, where EPSILON is
a static public variable of this
class)

Tokens also have an isCloseTo() method:

```
/** Test whether the value of this Token is close to the argument                              *  Token.  The argument and this token are converted to
*  equivalent types, and then compared.  Generally, this is the                               *  higher of the type of this token and the argument type.
*  Subclasses should implement the protected _isCloseTo() method                               *  to perform the correct type-specific operation.
*  @see #isEqualTo                                                           *  @param rightArgument The token to test closeness of this token with.
*  @param epsilon The value that we use to determine whether two                               *   tokens are close.
*  @return A boolean token that contains the value true if the                              *   units of this token and the argument token are the same,
and their                         *   values are close.                                            *  @exception IllegalActionException If the
argument token is not                             *   of a type that can be compared with this token, or the units                          *   are not
the same.
*/
public final BooleanToken isCloseTo(Token rightArgument, double epsilon)
```

See 13.4.1 Invoking Methods in http://ptolemy.eecs.berkeley.edu/books/Systems/chapters/Expressions.pdf

Files

simple_error.xml	83.1 KB	03/04/2014	Owsiak Michal
simple.xml	107 KB	03/04/2014	Owsiak Michal