Tech Info Library

HyperCard: How It Does Math (7/92)

Revised: 12/21/93
Security: Everyone

HyperCard: How It Does Math (7/92)

==========

Article Created: 24 March 1992
Article Reviewed/Updated: 23 July 1992

TOPIC --

This article has four sections:
- HyperCard Uses SANE
- Floating-point Numbers
- NaNs and INFs
- Mixing Reals and Integers

(A related Tech Info Library article deals with HyperCard's round, trunc, and random functions.)

DISCUSSION --

HyperCard Uses SANE

HyperCard uses SANE (Standard Apple Numerics Environment) routines for doing math in HyperTalk. See Apple Numerics Manual, Second Edition, for any details.

Floating-point Numbers

Any floating-point (or real) number used by HyperCard is stored as an extended type. Each number is represented in 80 bits (10 bytes) and has a precision of roughly 19-20 decimal places. Because an extended type is allotted a fixed amount of memory, only a finite number of values can be represented exactly.

Example: type "1000000000000000000+1" into the message box. HyperCard evaluates this expression to 10000000000000000000.

NaNs and INFs

Calculations via SANE can produce infinities and NaNs (Not-a-Number). Such

results may be passed back in HyperTalk. Consult the Apple Numerics Manual for the specifics.

Examples in HyperCard:
sqrt(-1)-> "NAN(001)"
ln(0)-> "INF"
ln(-1)-> "NAN(036)"
-1/0-> "-INF"
(-1)^(-.5)-> "NAN(037)"
1/ln(0)-> 0

Mixing Reals and Integers

HyperTalk, being an untyped language, lets scripters mix real and integer values. Sometimes this can produce unexpected results.

Examples:
(.3 * 10) DIV 1-> 2
(.3 * 10) MOD 1-> 1
trunc(.59*100) < 59-> true

The reason behind the unexpected result is that, internally, HyperCard is applying the math operations to extended type numbers. For example, ".3*10" is not equal to 3 (because .3 cannot be represented exactly by an extended type). The solution to such problems is to have HyperCard evaluate expressions first. This can be accomplished by putting the value into a field or by taking the value of the expression. Evaluating an expression converts it to a "practical" precision (using the numberFormat property).

Example:
value(.3*10) DIV 1-> 3

This behavior is exactly what one would expect if doing math in any high-level language. Both of the following, the first in Pascal and the second in HyperTalk, produce the same results:

```
program Math;

var
    x:extended;
    factor, truncX, K: integer;

begin
    ShowText;{ for THINK Pascal }
     factor := 1000;
     x := 0.0;
     for K := 1 to factor do
    begin
              x := K / factor;
             truncX := trunc( factor * x );
             if (truncX <> K) then
           writeln(K);
     end;
```

```
end.

on mouseUp
   set cursor to watch
   put 1000 into factor
   put EMPTY into cd field "Table"
   put EMPTY into theTable
    repeat with K = 1 to 1000
       put K/factor into x
        if (trunc(factor*x) <> K) then
             put K & return after theTable
       end if
    end repeat
     put theTable into cd field "Table"
end mouseUp
```

Both of these produce the list:

```
1
2
4
8
16
32
63
64
126
128
252
256
504
507
512
```

This article is adapted from the Claris Tech Info database.

Tech Info Library Article Number:14252