CALC SCRIPT 2

A simple programmable scripting tool.

EXPRESSION RUNTIME

- Feature Set.
- Add expressions at runtime.
- Supports If then else, Loop , Case expressions separate by semi colons.
- Supports multiple expression statements per calculation.
- Bind to Local Variables in your favourite language.
- Bind to Classes in your favourite language.
- Pass result of an expression to next statement separated semi colons.
- Conditionally bind vairables and classes.
- Platform independent expressions.
- Simple Binding interface to classes.
- Mix and Match multiple language Dlls for seemless expression results.

SUPPORTED INTERFACE

- Tokenise (Expression), returns a numeric ID to a parsed expression.
- Compute(ID), compute an expression that has been tokenised.
- ComputeStrFromStr(Expression), return a string result from an expression.
- Bind Int, Real Str(Variable), bind a variables of Int, real or string.
- Bind Int IFC, Real IFC, String IFC(variable,IGetSetBindable), bind a class that supports get set. BindInt(context, address), BindReal(context ,address),Bindstrref(context name cstring,address,length)
- **getParamStr**(number), get a parameter result of an expression.
- getparamcount() , get the number of parameters in an expression.
- getboundname(), get the bound name of the function executing.
- getboundcontext() , get the bound context of the function executing.
- Bind classes and access the math results directly.
- Bind local functions in any language that supports C or pascal calling.

CALC EXAMPLE IN CLARION

- Calc2 &Icalcscript ! Declare a calc script interface to calc2
- Expression cstring(255) ! expression
- Expressionhandle long ! Expression tokenised ID
- QTYYTD long ! Local qty long
- QTYTotal long ! Local qty total long
  - **Code**
- Calc2 &= **Calccreatescript** () ! Create calc script instance
- Calc2.Bindint('QTYYTD ',QTYYTD) ! Bind a long variable to calc to use in expressions
- QTYYTD = 45000
- Expression = '45*QTYYTD' ! Create an runtime expression
- expressionhandle = Calc2.Tokenise(Expression) ! Parse an expression
- QTYTotal = Calc2.Computeint(expressionhandle) ! Compute the result