Collabora

LibreOffice
The Document Foundation

SUSE

LibreOffice BASIC
Hidden Gems

MySQL

LINUX MAGAZINE

datto

ADMIN Network & Security

fedora

TUXEDO COMPUTERS

.riessapplications

Required Magic
advanced technology

LibreOffice Basic Hidden Gems

- LibreOffice Basic
  - Star/Open/Libre legacy
- VBA (not so) partial support
  - ca. 2000

Hi

- Alain Romedenne
- alain.romedenne@libreoffice.org
- LibreOfficiant
- LOfficiant

Agenda

- Compiler/Runtime statements
- Rules of thumbs
- Enumerations
- OOP
- Accessers & variable scope
- ParamArray
- Keyword parameters
- Throwing errors
- Logging

Compiler/Runtime Statements

Option Base 0

Option Explicit

- Option Compatible
- Option ClassModule
- Option VBASupport 1

✘ CompatibilityMode()

Rule of Thumbs

- ✔ Stick to LibreOffice Basic + API
- ✔ Extend with Option Compatible
  - compulsory Set statement
  - Keyword parameters
  - Property Get | Let | Set
- ✔ Add Option Compatible for OOP
  - Accessers as class constructors
  - Private | Public support
  - Method overloading

- ✗ Caution: Option VBASupport 1
  - Affects Basic Runtime behavior
  - Limit to doc. conversion scenarii
- ✗ Avoid CompatibilityMode(bool)
  - Bad coding practice

Enum Statement

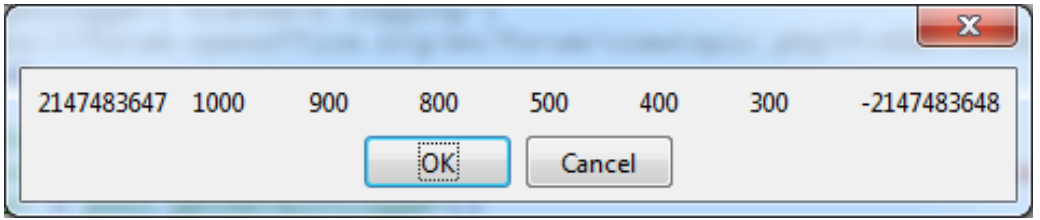
- API examples
- WindowManager example

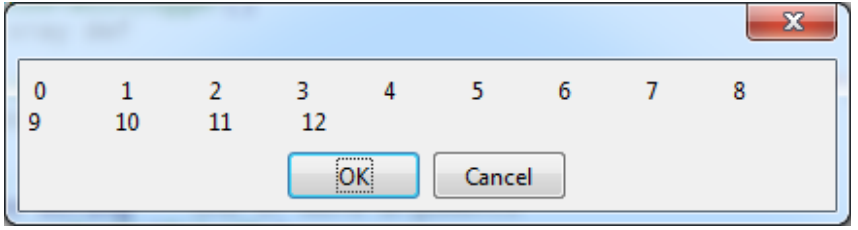
- … overcoming the Long datatype ?

- Type Def alternative

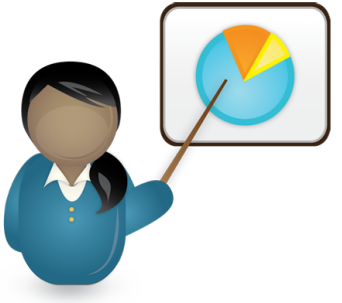
Enum – API

```
' Create an enumeration from API constant group
Dim LogLevel
LogLevel = com.sun.star.logging.LogLevel
With LogLevel
  Print .OFF, .SEVERE, .WARNING, .INFO, .FINE, .FINER, .FINEST, .ALL
End With
```






```
GeneralFunction = com.sun.star.sheet.GeneralFunction
With GeneralFunction
Print .NONE, .AUTO, .SUM, .COUNT, .AVERAGE, .MAX, .MIN, .PRODUCT, _
  .COUNTNUMS, .STDEV, .STDEVP, .VAR, .VARP
End With
```



Enum WinMgr

Parameters:

Within a given enumeration, fit together values that logically relate to one another.

Example:

```
Option VBASupport 1
Private Enum _WindowManager
    W1ND0WS = 1 ' Windows
    OS2PM = 2 ' OS/2 Presentation Manager
    MACINTOSH = 3 ' Macintosh
    MOTIF = 4 ' Motif Window Manager / Unix-like
    OPENLOOK = 5 ' Open Look / Unix-like
End Enum
Public Function WindowManager() As Object
    WindowManager = _WindowManager
End Function ' <library>.<module>.WindowManager.XXX
```

Enumerated values are rendered to **Long** datatype. Basic functions are public accessors to enumerations. Enumeration names and value names must be unique within a library and across modules.

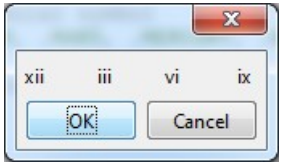
Usage:

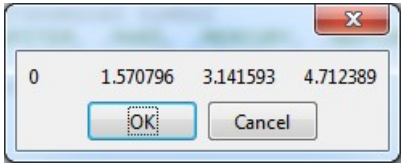
Display WindowManager grouped constant values:

```
Dim winMgr As Object : winMgr = <library>.<module>.WindowManager
With winMgr
    Print .MACINTOSH, .MOTIF, .OPENLOOK, .OS2PM, .W1ND0WS
End With
```

Enum – Values are Int32 !

```
Public Enum Cardinal ' Clockwise
  North = 12 ' XII
  East = 3   ' III
  South = 6  '  VI
  West = 9   '  IX
End Enum 'Cardinal

Public Enum Azimut ' ° degrees
  N = 0
  NE = 45
  E = 90
  SE = 135
  S = 180
  SW = 225
  W = 270
NW = 315
End Enum ' Azimut ° degrees
```






```
Public Enum Radian
  North = 0
  East = 2   ' CLng(Round(Pi/2,0))
  South = 3  ' CLng(Round(Pi,0))
  West = 5   ' CLng(Round(3 * Pi/2,0))
End Enum

Enum Rainbow ' As com.sun.star.script.NativeObjectWrapper
' Rainbow colors as of Isaac Newton's arbitrary choice
  RED = 3
  ORANGE = 6.1
  YELLOW = -6.2
  GREEN = 5
  BLUE = 4
  INDIGO = -.006E3
  VIOLET = -.0006E4
' Rainbow colors arbitrary values
End Enum ' <library>.<module>.Rainbow
```

Enum – Type Def alternative

- Isaac Newton’s rainbow colors
  - RGB expressiveness

Solar System Planets

Mass: floating # for kilograms

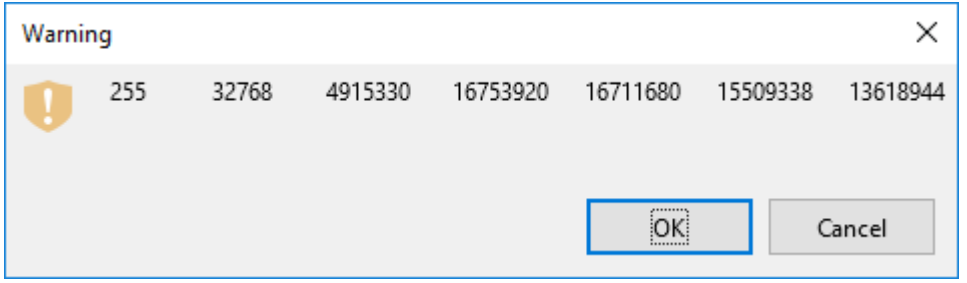
MeanDistanceFromSun: BIG # for kilometers

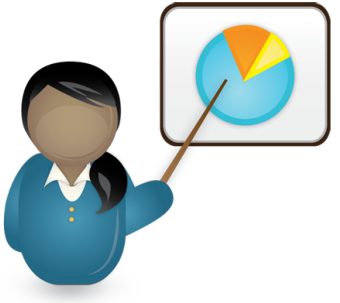
Symbols: as in astrology

Enum – Isaac Newton's rainbow

```
Private Type _RAINBOW
  BLUE As Long
  GREEN As Long
  INDIGO As Long
  ORANGE As Long
  RED As Long
  VIOLET As Long
  YELLOW As Long
End Type

Public Function Rainbow() As _RAINBOW
  Static it As _RAINBOW : Rainbow = it
  If it.BLUE<>0 Then Exit Function
  With it
    .BLUE = 255
    .GREEN = 256 * 128
    .INDIGO = RGB(75,0,130)
    .ORANGE = &hffa500
    .RED = 16711680 ' RGB(255,0,0) aka &hFF0000
    .VIOLET = 238*255^2 + 130*255 + 238
    .YELLOW = &hcFcf00
  End With
End Function ' Standard.Newton.Rainbow Enum.eration
```

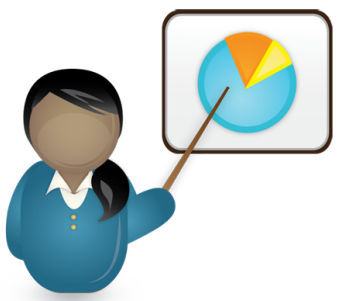
```
Dim e : e = Rainbow()
With e
  Print .BLUE, .GREEN, .INDIGO, .ORANGE, .RED, .VIOLET, .YELLOW
End With
```

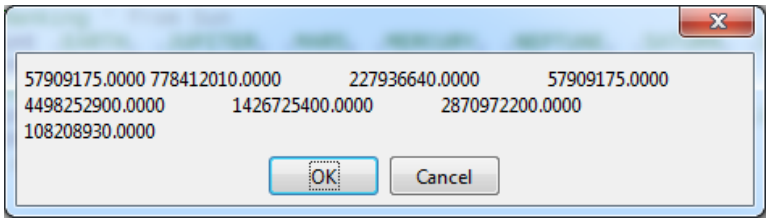


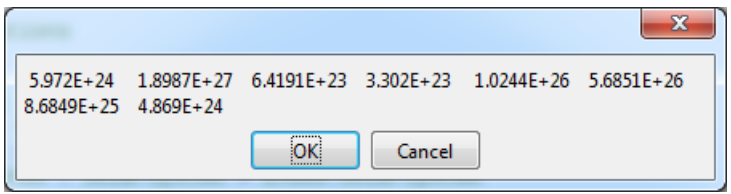


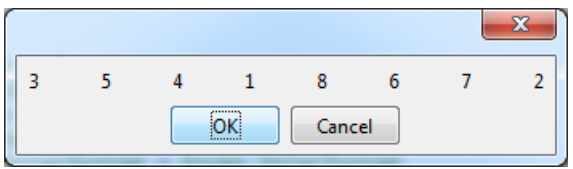
Enum – SolarSystem module

```
Sub demo_Astronomy
  With Astronomy.SolarSystem
    With .MeanDistanceFromSun ' in kilometers
      Print .EARTH, .JUPITER, .MARS, .MERCURY, .NEPTUNE, .SATURN, .URANUS, .VENUS
    End With
    With .Mass ' in kilograms
      Print .EARTH, .JUPITER, .MARS, .MERCURY, .NEPTUNE, .SATURN, .URANUS, .VENUS
    End With
    With .Ranking ' from Sun
      Print .EARTH, .JUPITER, .MARS, .MERCURY, .NEPTUNE, .SATURN, .URANUS, .VENUS
   End With
   With .Symbol ' as astronomical symbol
     Print .EARTH, .JUPITER, .MARS, .MERCURY, .NEPTUNE, .SATURN, .URANUS, .VENUS
   End With
  End With ' Astronomy.SolarSystem
End Sub
```









cf. Using enumerations in VBA

OOP, sort of ...

- Session class
- Platform class
- Access2Base library examples

OOP – Object-Oriented Programming

- LibreOffice Basic Collection objects
  - Dictionary / Map class extension
- Access2Base Trace console

- Exception Basic object
  - custom error codes

Accessers & variable scope

Extend scope using accessers

<library>.<module>.accesser()

- Computer.Desktop.WindowManager
- Basic.lang.Exception() | Map()
- Isaac.Newton.RainbowColors
- Astronomy.SolarSystem.Distance | Planet | Symbol

variable	*scope*
enumeration	global
object	library
User datatype	module

ParamArray argument modifier

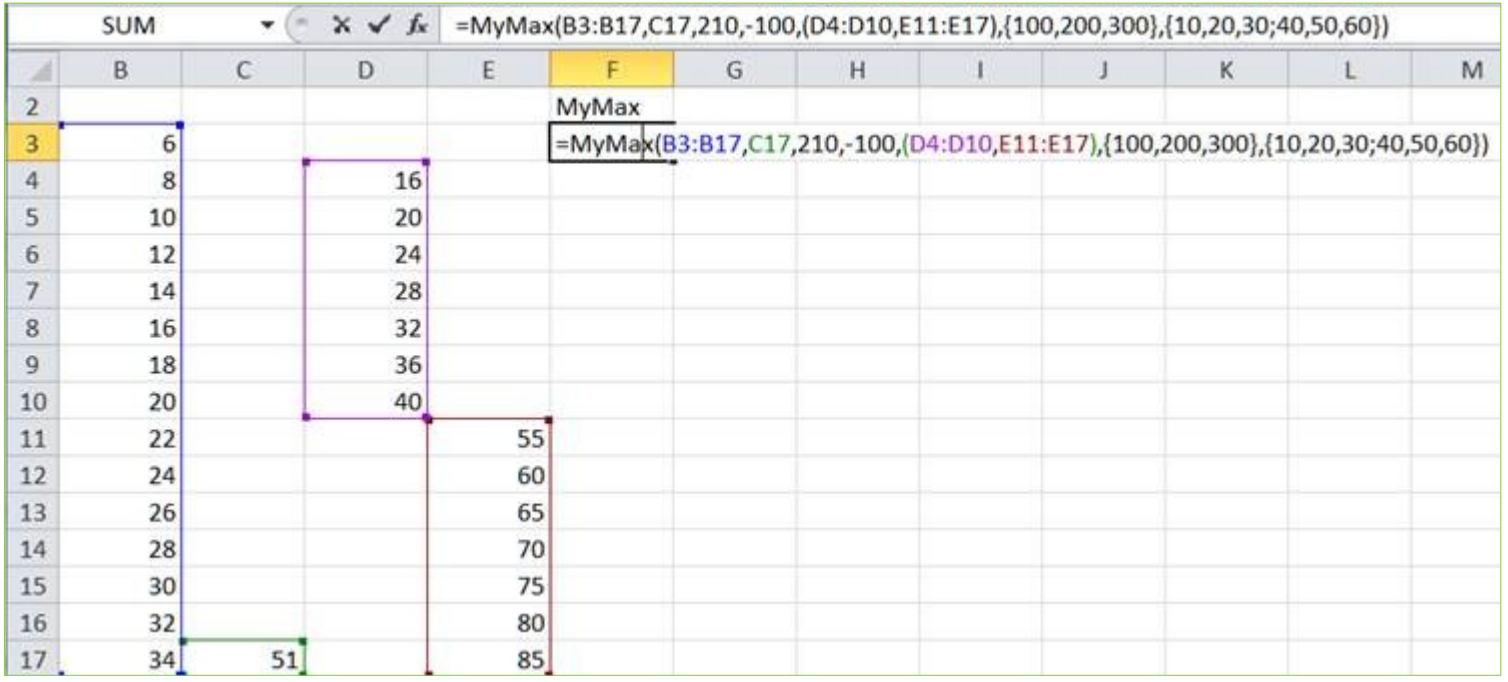
main purpose

- Calc User Defined Function (UDF)
  receiving multiple selections

other use cases

- Access2Base.Compatible.DebugPrint method
- Access2Base Trace console & Python *args idiom

ParamArray – Calc UDF



Keyword Parameters

requires Option Compatible

Collection class

- Count: Long
- Add(item, opt key: str, opt before:= position|key)
- Add(item, opt key: str, opt after:= position|key)
- Item(index|key)
- Remove(position|key)

cf. Call statement

Fun with Keyword Parameters

Mixing positional / keyword arguments

- InputBox(prompt:= str, opt title:=”LibreOffice *version*”, default:=””)
- MsgBox(prompt:= str, opt buttons:=MB_OK, opt title:=”LibreOffice *version*”)
  - bonus: rich text MsgBox w/ API

Kw parms + Accessers = Method overloading

- Point
  - x: value
  - y: value
  + Point(x:value, y: value) # 2 arguments
- Circle
  - center: Point
  - radius: num
  + Circle(center: Point, radius: value)
  + Circle(points[0 to 1]: Point) # center + 1 perimetric point
  + Circle(points[0 to 2]: Point) # 3 perimetric points

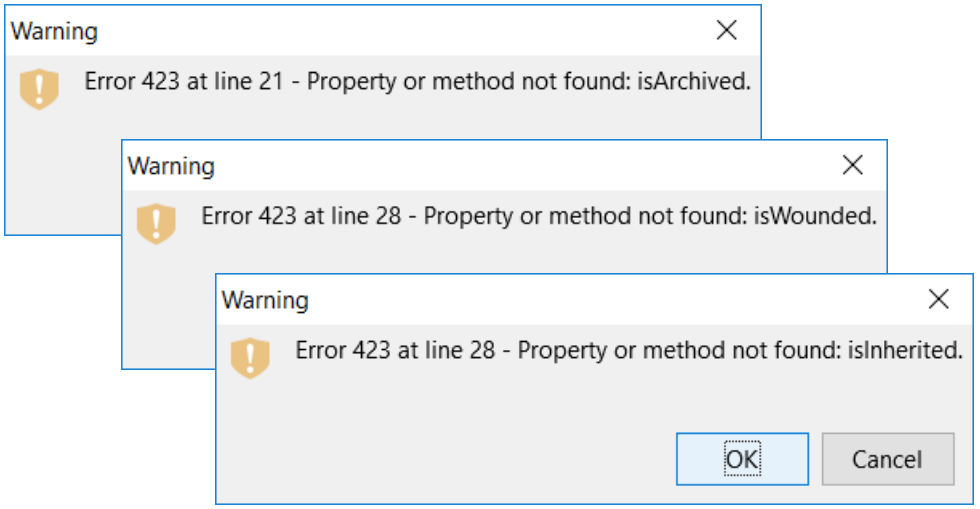
requires Option Compatible

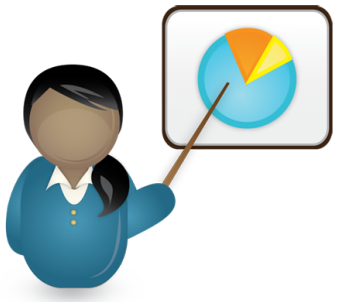
Raising errors

- Typ Def trick
  - # 423 – Property or method not found: <routine>
- VBA Err object wrapper
  - custom error code, description & <method>

Raising errors – Type Def trick

```
Option Explicit
Option Compatible

Private Type _ERROR
  anything as Variant
End Type
Private Function isArchived As Boolean
  ' Not available in this particular context
  Dim  e as _ERROR : e.isArchived
End Function ' <library>.<module>.isArchived
Private Property Get isBroken As Boolean
  Dim  e as _ERROR : e.isWounded
End Property ' <library>.<module>.isBroken

Sub Main
  Dim e As _Error
try: On Local Error Goto catch
  MsgBox isArchived
  MsgBox isBroken
  MsgBox e.isInherited
finally: Exit Sub
catch: Print "Error "&Err; " at line "&Erl; " - "; Error$
  Resume Next
End Sub
```

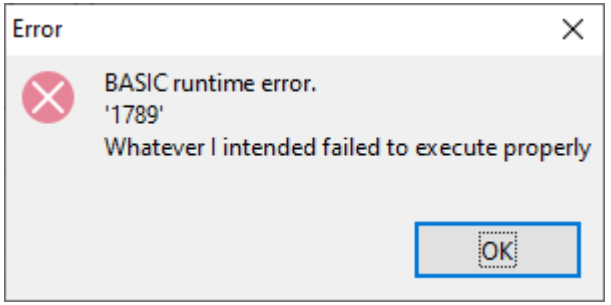


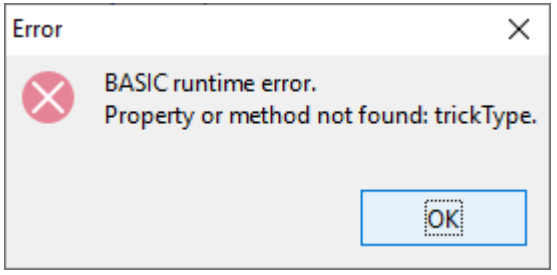


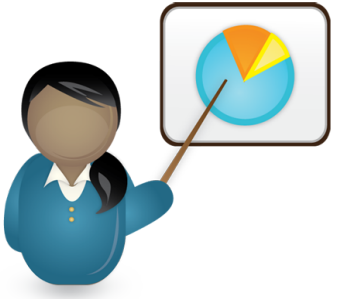
Raising errors – VBA Err object

Exception class

- Description: String
- Number : Long
- Source: String
- Clear()
- Raise(errCode As Long,
  source As String,
  desc As String)



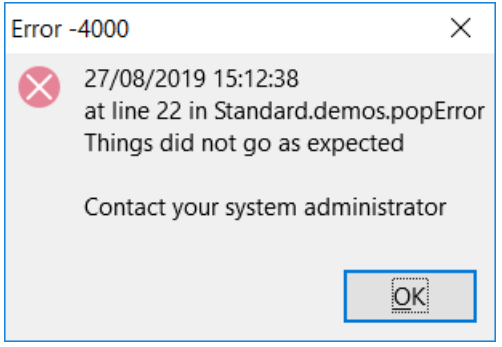




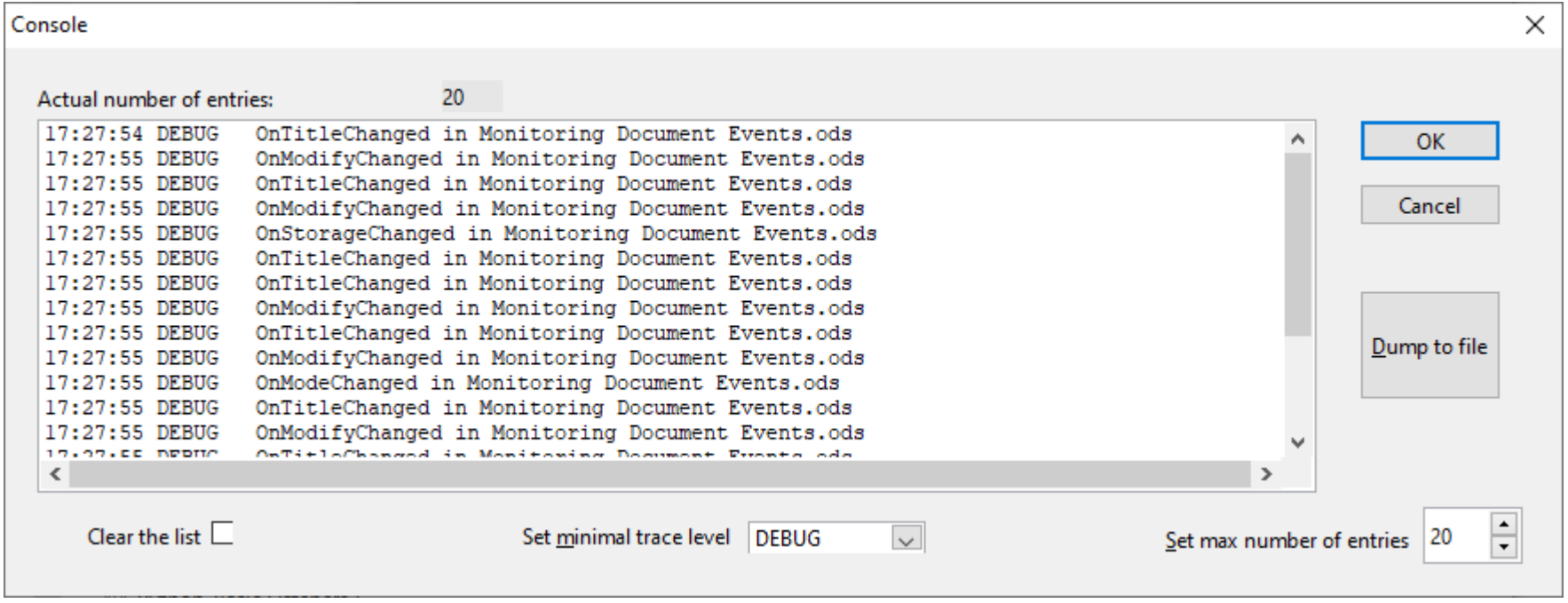
Handling custom errors

```
''' LibreBasic.lang module

Option Explicit

Option Compatible


Public Property Get Exception As Object

''' Return a unique instance of _Exception class '''

  Static obj As Object ' VBA Err object wrapper

  If IsNull(obj) Then obj = New _Exception

  Exception = obj ' wrapper of VBA Err object singleton

End Property ' LibreBasic.lang.Exception
```

```
''' <library>.<module>
Sub popError
  try: Dim e As Object : e = LibreBasic.lang.Exception
  On Local Error Goto catch
  e.raise( -4000, "Standard.demos.popError", _
    "Things did not go as expected" _
    &Chr(13)&Chr(13)& _
    "Contact your system administrator")
finally: ' Any cleanup process goes here
  Exit Sub
catch:
  MsgBox Now &" in "& e.Source &Chr(10)& _
    Error,MB_ICONSTOP, "Error "& e.Number
  Resume finally
End Sub
```



Logging w/ Acces2Base Trace module



Oldies but Goodies

- Option Compatible
  - Keyword parameters when multiple optional args
  - Property Get | Let | Set – defensive coding
  - ParamArray for Calc UDF/ varying argument lists / Python to Basic calls
  - Method overloading, class constructors
  - Encapsulation w/ Private | Public – defensive coding
- Option VBA Support 1
  - Enumerations w/ Enum statement
  - Raising custom errors w/ Err object
  - Round(), StrReverse() & alter

My 2 cents

- Option Explicit
- Fully qualified methods
  - <library>.<module>.<method>
- CamelCasing
  - FileList, bottle_of_wine, isMacOsX
  - GetKey, openDoor, lock_Windows
- “_” prefix for Private variable
- Explicit UNO objects as shown here
- Collection class
- Code Completion
- Enum/Type enumerators
- Throwing errors

Don’t code !

- Calc formulas
  - com.sun.star.Sheet.FunctionAccess
- X-scripting between Basic / JavaScript / Python
  - Python standard modules:
    cmath, datetime, json, math, net, re, unittest, …
  - JavaScript features:
    JSON, …
- python.os.filelen(fn: String)

TDD – Test Driven Dev.

- Build Basic test cases
  with Python unittest module

E-References

- LibreOffice Basic help
- LibreOffice API documentation
- Office VBA Reference
- Basic primitives by J-F Nifenecker

Python scripts made simple
w/ IDE_utils

Thank You