Novell® VBA Interoperability

Noel Power

15 September 2006

Novell®

Introduction

About Me

Working/hacking Openoffice.org ~ 4 years

working on VBA interoperability

A Bit of History

Started in response to customer demand

Various people have stuck there hand in here and there

Releases

- SUSE 10.1 (first appearance)
- SUSE SLED 10 (first supported version)

Started VBA Interoperability incubator project

- http://vba.openoffice.org or
- http://wiki.services.openoffice.org/wiki/VBA

Why provide VBA Interoperability

Argghh!!

- "We already have all of these spreadsheets what are we going to do with them"
- "we don't event understand how they work anymore"

There is already a proprietary solution

VBA Interoperability is the "sugar" to sweeten the migration path

Why bother, you will never get 100% compatibility?

How does it work

Changes to import filters to

- import the VBA macros (uncommented)
- stitch additional event bindings

Changes to the basic engine

- compatibility options
- support additional “exotic” vba syntax

```
Option VBASupport 1
Sub SelectToFromCells()
Range("FromCell", "ToCell").Select
End Sub
```

Provides a compatibility model

Supporting changes to calc core

Upstreaming

Not enough hacks upstreamed yet

Option VBASupport

- provides a basic indicator that interoperability is 'on'
- used a hook internally in the basic engine

css::com::ArrayWrapper

- multi-dimensional array support for Objects
- influence the base of the Array regardless of the Module 'Option Base' setting

Array(...) function & Option Base

Support a 'default method' concept

VBA exotics

```
set r1 = Range(“a1”)
r1 = “foo” ' value of cell A1 is set to “foo”
dim aVar as String ' or variant
'aVar wil contain “foo”
aVar = r1
'Collections
dSheets = Worksheets ' global
wrb = dSheets(1)
wrb = dSheets.Item(1)
wrb = dSheets.("Sheet1")
wrb = dSheets.Item("Sheet1")
```

What needs to be upstreamed

Dim r1 as Range

Set and assignment for vba objects

Most vba constants are available (maho)

Default properties { LHS & RHS }

Global objects

- Application, Sheets...

Compatibility API

- Range, Workbook, Worksheet...

Better event support

- Excel toolbox controls, shapes, images, workbook, worksheet

Larger basic module support

Whats Next

Basic Support for Userforms & Controls

- tie Userform module to Dialog
- allow contols to be accessed from Worksheet
  - `Sheet(1).CommandButton1`
  - `Sheet(1).MyComboxBox.AddItem( ListItem )`

Compatibile Array handling

- array of array syntax `myarray(2)(3)`
- copy by value

Import internal Collection class

- handle string key in Item & add methods

Generally Improve API coverage

More document analysis

What can you do to help

Get involved

irc, #go-oo { me = noelp }, mail noel.power@novell.com

http://wiki.services.openoffice.org/wiki/VBA

Identify missing api & test existing api

some basic tests at ooo-build/tests/macros

Donate representative documents

If you can't send documents, send code

– use test-msvba from the libgsf (see wiki)

Code some compatibility api

its easier than your think

its a good introduction into openoffice hacking