

Telelogic Americas 2002 User Group Conference Las Vegas Oct 20-23, 2002 *Plan to Win!*

Telelogic Americas 2002 User Group Conference

DOORS DXL - Adventures in Microsoft OLE Automation

Michael Sutherland
Galactic Solutions Group LLC
michael@galactic-solutions.com





Excel Exporter

- Telelogic provides an Excel Exporter for DOORS
 - `$DOORSHOME/lib/dxl/standard/export/office/excel.dxl`
- Functional, but does not export:
 - OLE Objects (graphics)
 - Rich Text
 - Outlining/Hierarchy
 - Color Columns and Object Heading row Color
 - Page Layouts (Headers, Footers, Paper Size, Paper Orientation, Margins)
 - Column Widths and Fixed Header Row
- To create an Enhanced Export to Excel for DOORS, knowledge of the DOORS API and Microsoft OLE Automation is required

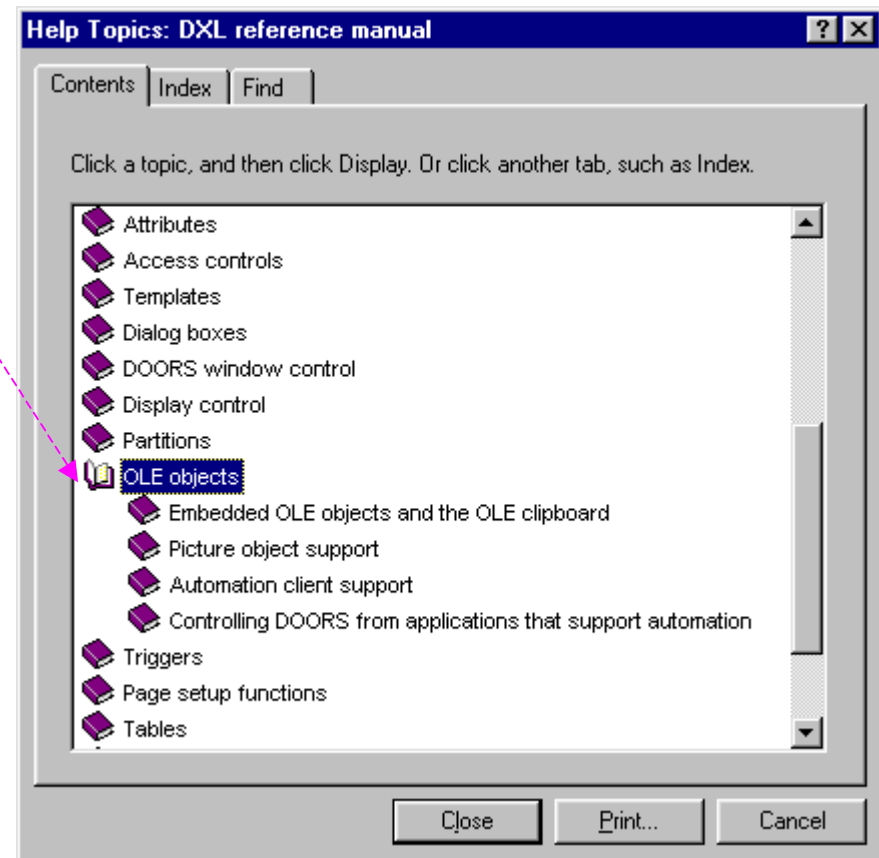
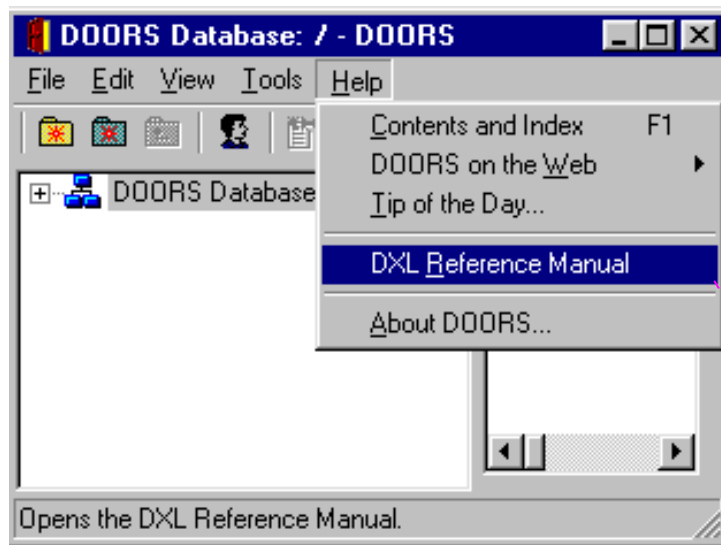


DOORS DXL (DOORS eXtension Language)

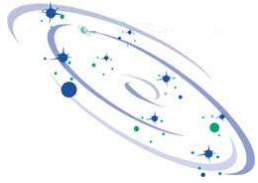
- DXL is the Application Program Interface (API) for DOORS
- DXL is a “macro” language to:
 - Automate repetitive tasks
 - Manipulate Database information
 - Create new user interactions
 - (dialogues, forms, events, etc.)



DOORS DXL Reference



DOORS DXL - Adventures in Microsoft OLE Automation
© 2002 Galactic Solutions Group LLC - Michael Sutherland - michael@galactic-solutions.com

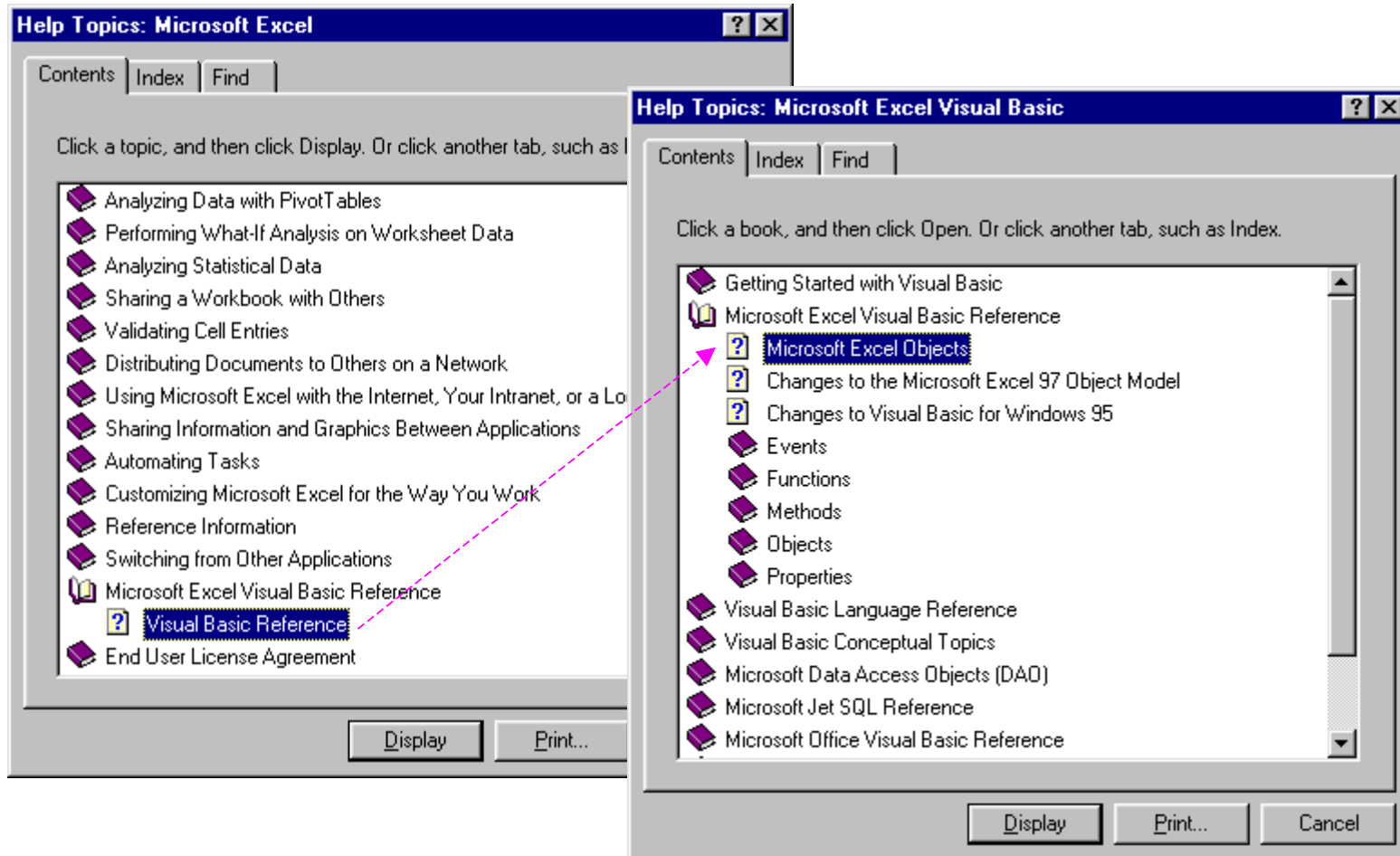


Microsoft OLE Automation

- Uses Component Object Model (COM)
- Method used in Microsoft Windows Operating System to communicate with Windows Applications
- Allows referencing of another Microsoft Windows Application's Objects, Properties and Methods
- Note: Microsoft Office 97 and above have dropped the "OLE" (Object Linking and Embedding) and called this "*Microsoft Automation*"
 - OLE still used to create Compound Documents



Microsoft Visual Basic Reference





OLE Application Object References

- DOORS uses `OleAutoObj` to declare Object variables that reference Application Objects

- Create an reference to the Server Application Object using the Application's "OLE Programmatic Identifier"

```
OleAutoObj objExcel =  
    oleCreateAutoObject( "Excel.Application" )
```

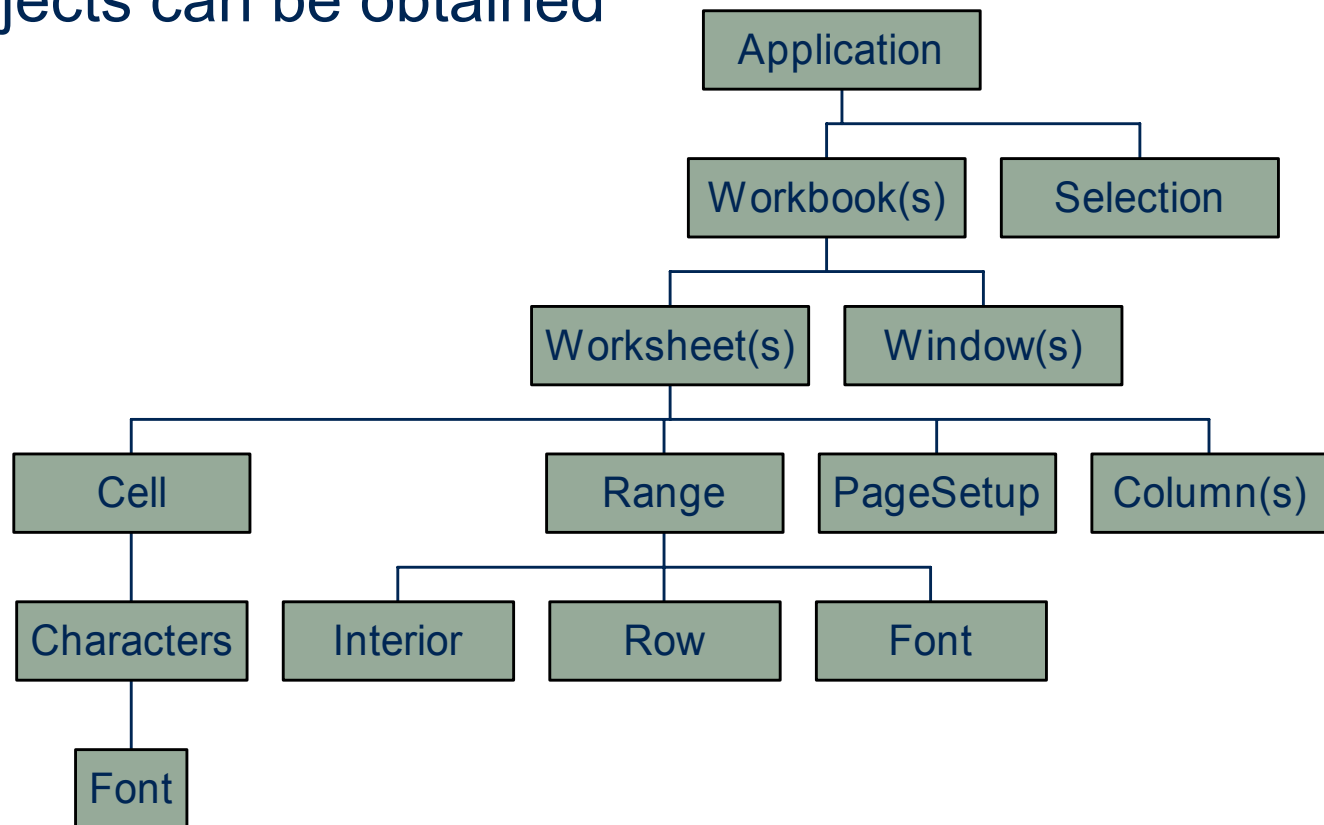
- The Objects of the Application are now accessible, and the Properties and Methods of the Objects can be applied
- When finished, close the Application

```
oleCloseAutoObject( objExcel )
```



Microsoft Excel Object Hierarchy

- Starting with the Excel Application, references to other Excel Objects can be obtained





Object Properties

- OLE Objects have Properties (Attributes)
- DOORS uses `oleGet` and `olePut` to access the properties of Objects
- Property can be of type:

`string | int | bool | char | OleAutoObj`

Note: A few OLE properties are of type real, which DOORS cannot currently handle

- Getting a property uses a “return value” variable
- Example: Boolean property of the Excel Application

```
bool isVisible
```

Note (“**isVisible**” is the return value)

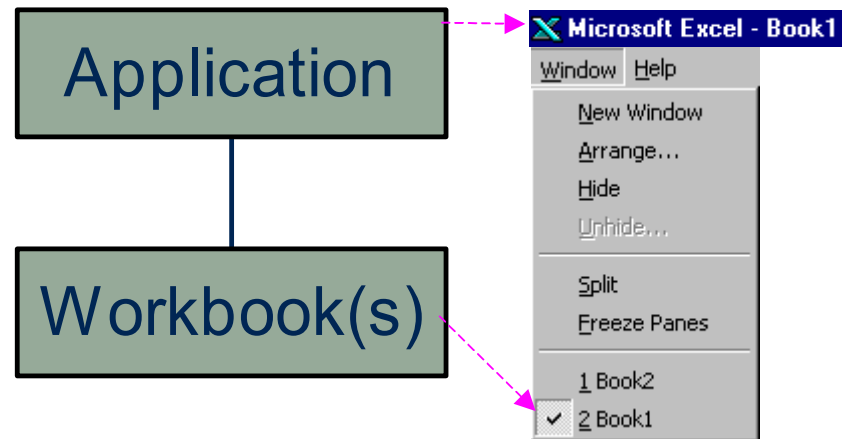
```
oleGet( objExcel, "Visible", isVisible )
```

```
olePut( objExcel, "Visible", true )
```



Obtaining Object Reference

- One important Property of an Object is the Objects it contains



- Example: The Excel Application (v97) can have 0-255 Workbook(s) open. The Workbooks Collection Object can be obtained from parent Excel Application Object:

```
OleAutoObj objWorkbooks
```

```
oleGet( objExcel, "Workbooks", objWorkbooks )
```



Object Methods

- Object have Methods, which are procedures or functions that act on the Object or transform data
- DOORS uses `oleMethod` to access Methods

- Example:

```
oleMethod( objWorkbooks, "Add" )
```

- Example: Sheet Activation

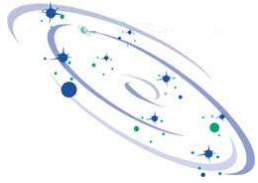
```
oleMethod( objSheet, "Activate" )
```



OleAutoArgs

- Properties and Methods sometimes requires arguments to be passed
- Declare and define OleAutoArgs variable
- Example: get first Sheet from Workbook Collection

```
OleAutoObj objSheet = null  
OleAutoArgs objArgBlock = create  
clear( objArgBlock )  
put( objArgBlock, 1 )  
oleGet( objWorkbook, "Sheets",  
        objArgBlock, objSheet  
delete objArgBlock
```



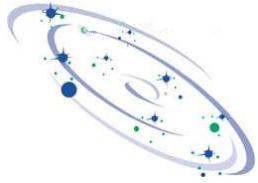
Object Methods with Arguments

- Example: Inches to Points

```
clear( objArgBlock )  
put( objArgBlock, inches )  
int points  
oleMethod( objExcel, "InchesToPoints", objArgBlock, points )
```

- Example: Saving Changes to Workbook

```
clear( objArgBlock )  
put( objArgBlock, "SaveChanges", true )  
oleMethod( objWorkbook, "Close", objArgBlock )
```



DOORS Provided OLE Library

- DOORS MS Office Exporters uses

```
#include <utils/ole.inc>
```

- Contains constants for the names Properties, Methods, Parameters and values of Symbolic References

```
Objects:    const string cObjExcelApplication = "Excel.Application"
```

```
Properties: const string cPropertyRange      = "Range"
```

```
Methods:    const string cMethodSelect      = "Select"
```

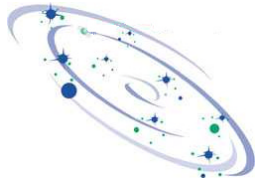
```
Parameter:  const string cParamSaveChanges  = "SaveChanges"
```

```
Symbolic:   const int    xlActiveWindow     = 1
```

- Contains common functions to aid in communication with Microsoft OLE Applications

```
checkPlatform, closeIfNonNull, checkRes
```

```
makeVisible, connectToApp, disconnectFromApp
```



Experienced VBA Programmers

- Although DOORS allows access to all of an OLE Application's Objects, Properties, and Methods, the actual programming is done in the DOORS API (DXL) and not in Microsoft *Visual Basic for Applications* (VBA)
- Veteran VBA programmers will miss constructs which loop through collections of Objects such as:
 - **For Each...Next:** Loop through each Object in a Collection, and allows a group of statements to be executed for each Object in the Collection
 - Use "Count" Property and use "Item" Method to index into the Collection
 - **With:** Runs a series of statements on the same Object
- It is possible to execute VBA macros stored in an Application Library, although this was not necessary for the implementation of the Enhanced Excel Exporter



Running VBA Macros from DOORS

- Procedure to execute a Macro stored in an Excel workbook:
 - (1) Connect to Excel Application
 - (2) Get Workbooks Collection
 - (3) Add Workbook (file) containing Macro(s) to Workbooks Collection
 - Note: This does not have to be the same Workbook file that data will be exported to
 - (4) Run Macro, passing arguments if necessary

```
mySum = Application.Run("MYCUSTOM.XLM!My_Func_Sum", 1, 5)
```

- If this method is used, the Excel Workbook containing the Macro(s) must be distributed to all DOORS users and placed in the proper directory

Enhanced Export to Excel





Copying OLE Objects to Excel

(1) Copy OLE Object to Windows clipboard

(2) Choose Target Sheet and Cell

(3) Set "Range" to single Cell

(4) Select Range

(5) Paste OLE Object

```
if ( oleCopy( o ) ) {  
    put( objArgBlock, "C12" )  
    OleAutoObj objRange = null  
    oleGet( objSheet, "Range", objArgBlock, objRange ) )  
    oleMethod( objRange, "Select" )  
    put( objArgBlock, "Link", false )  
    put( objArgBlock, "DisplayAsIcon", false )  
    oleMethod( objSheet, "PasteSpecial", objArgBlock )  
}
```



OLE Objects on Sheet

- OLE Objects reside on Worksheet, **not in** a cell
- OLE Objects are associated with a Cell (upper-left corner of OLE Object)
- OLE Objects are not strongly attached to cell, and can be easily moved around the sheet
- Sheet is formatted so that OLE Object initially fits within cell

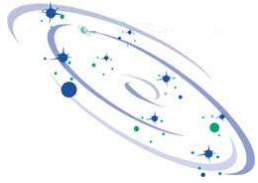


DOORS and Excel Columns

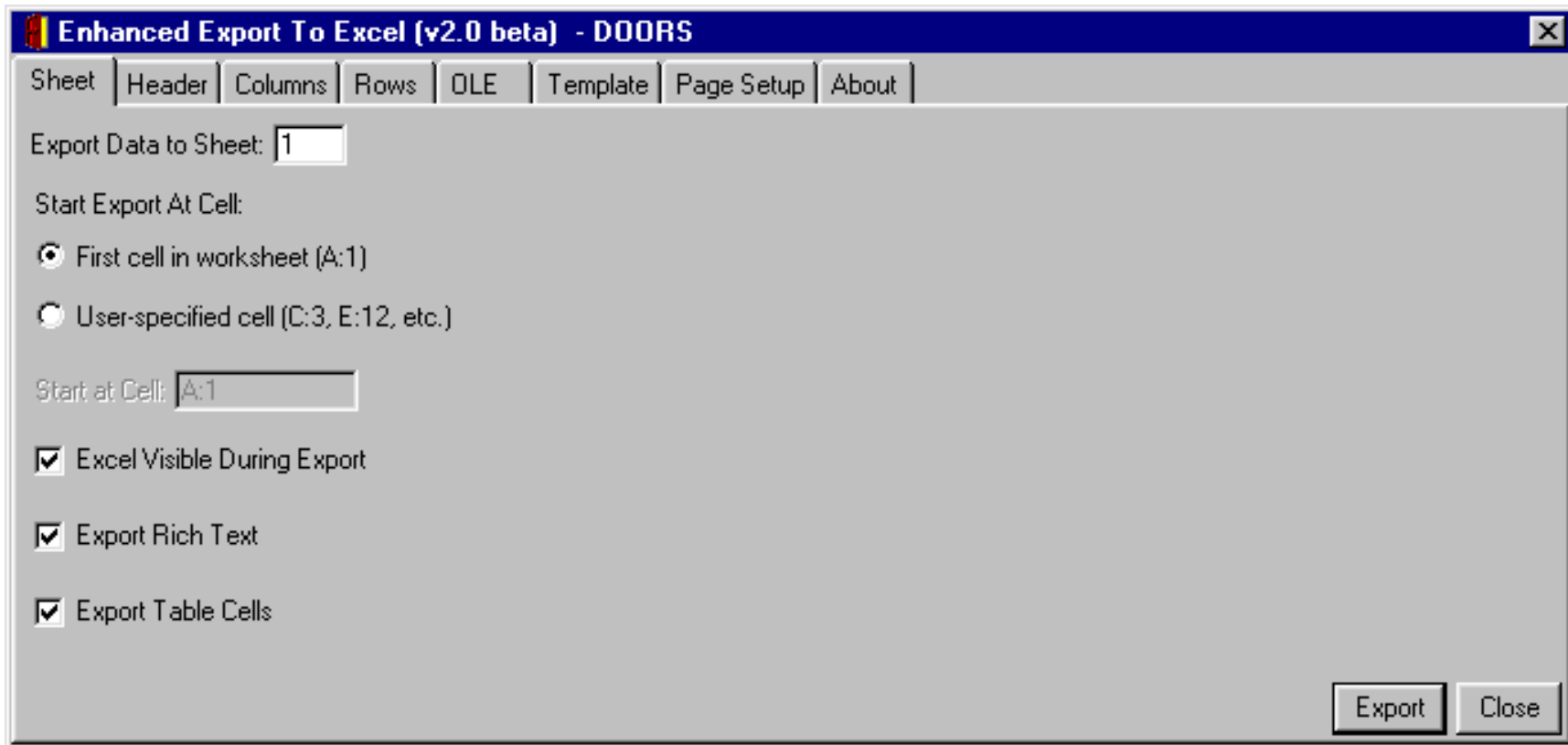
- Excel uses letters to enumerate Columns

	A	B	C	D	E
1					
2					
3					
4					
5					

- Existing DOORS Excel Exporter will not export more than 26 Columns (“A” - “Z”)
 - A DOORS Module allows 32 Columns
 - Excel 97 allows 256 Columns (“A” - “IV”)
- Routines have been enhanced to allow for 32 Column Export with offset
 - Export need not start at cell “A1”, start cell is user defined

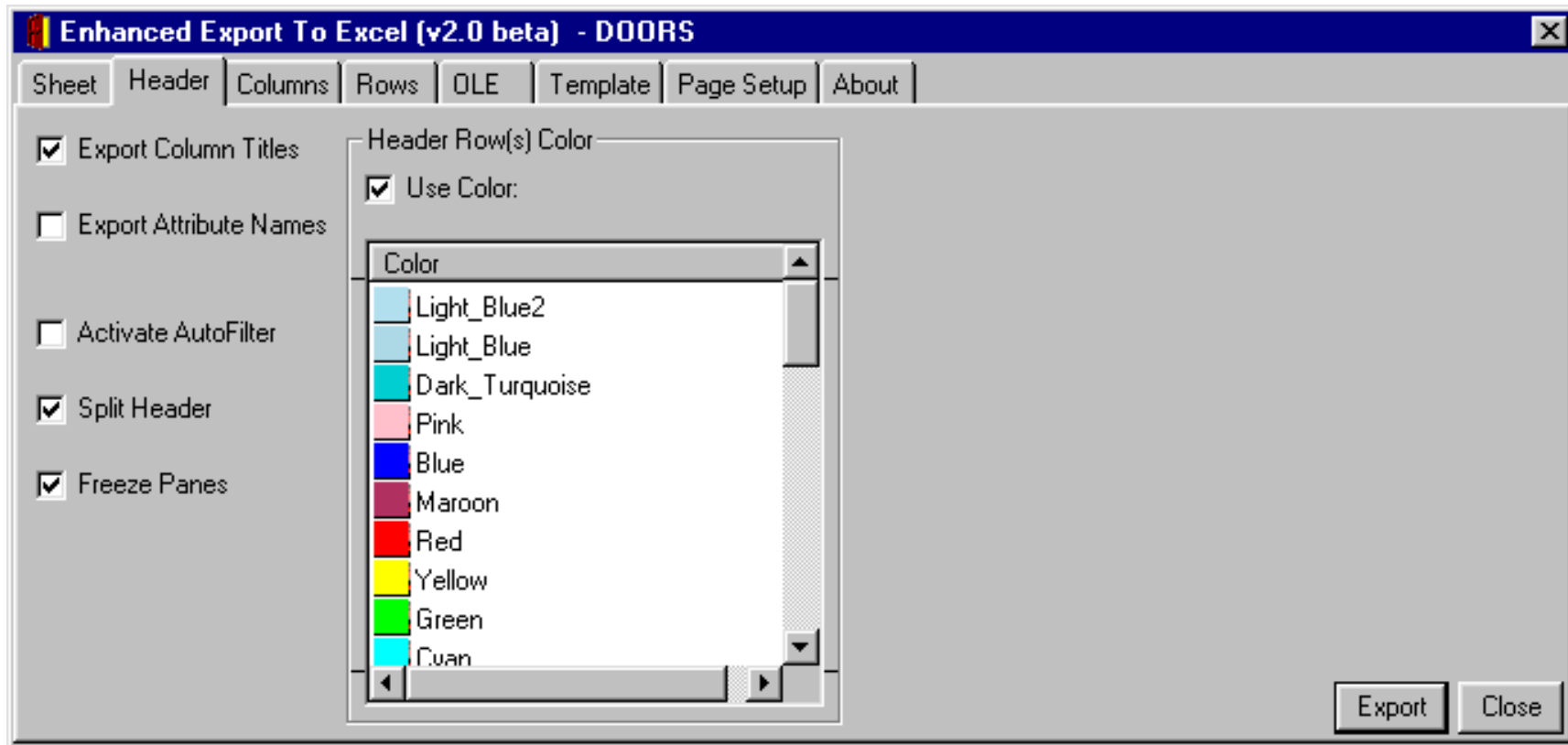


Enhanced Export to Excel - Sheet





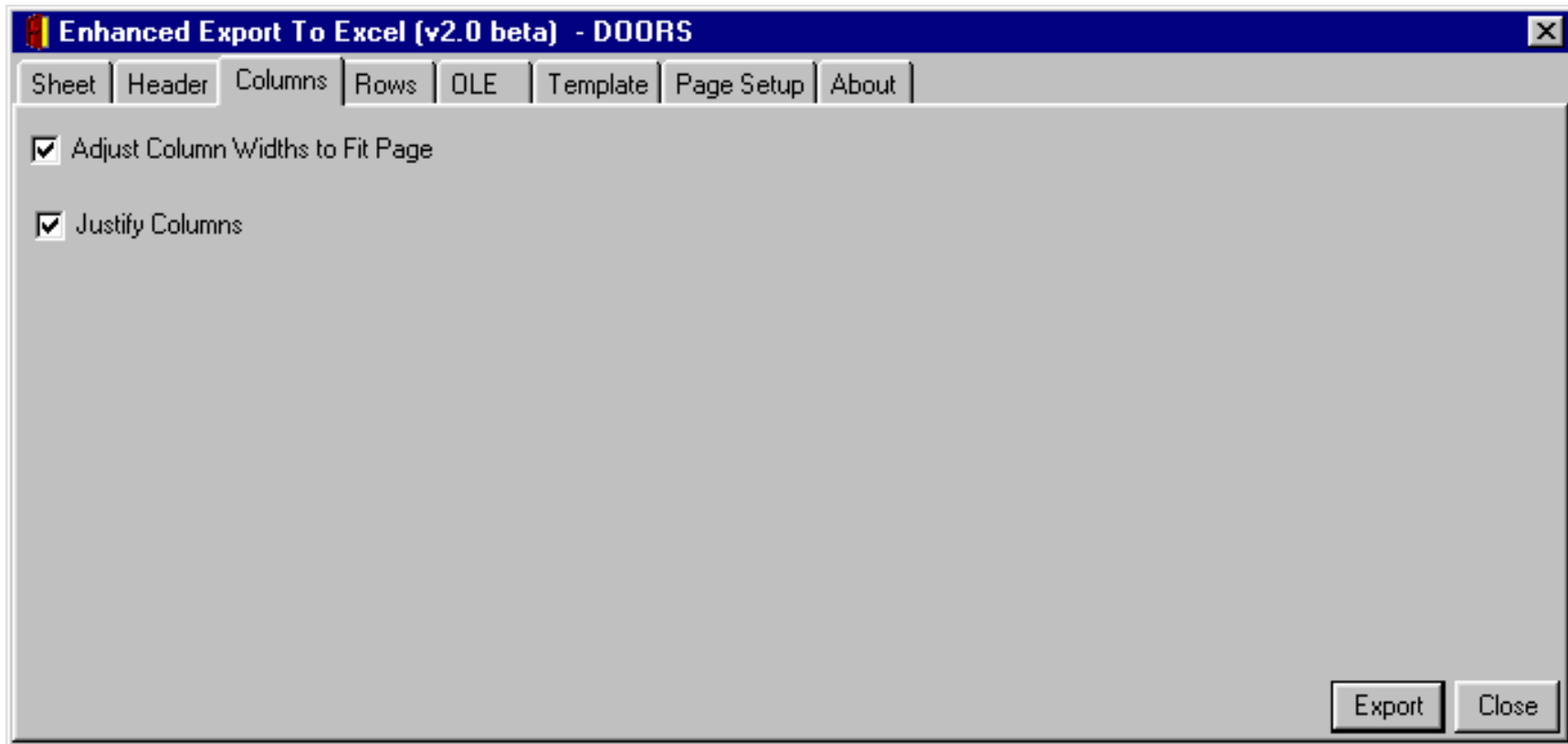
Enhanced Export to Excel - Header

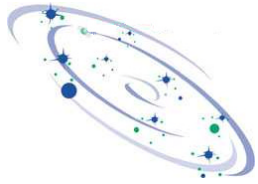


DOORS DXL - Adventures in Microsoft OLE Automation
© 2002 Galactic Solutions Group LLC - Michael Sutherland - michael@galactic-solutions.com

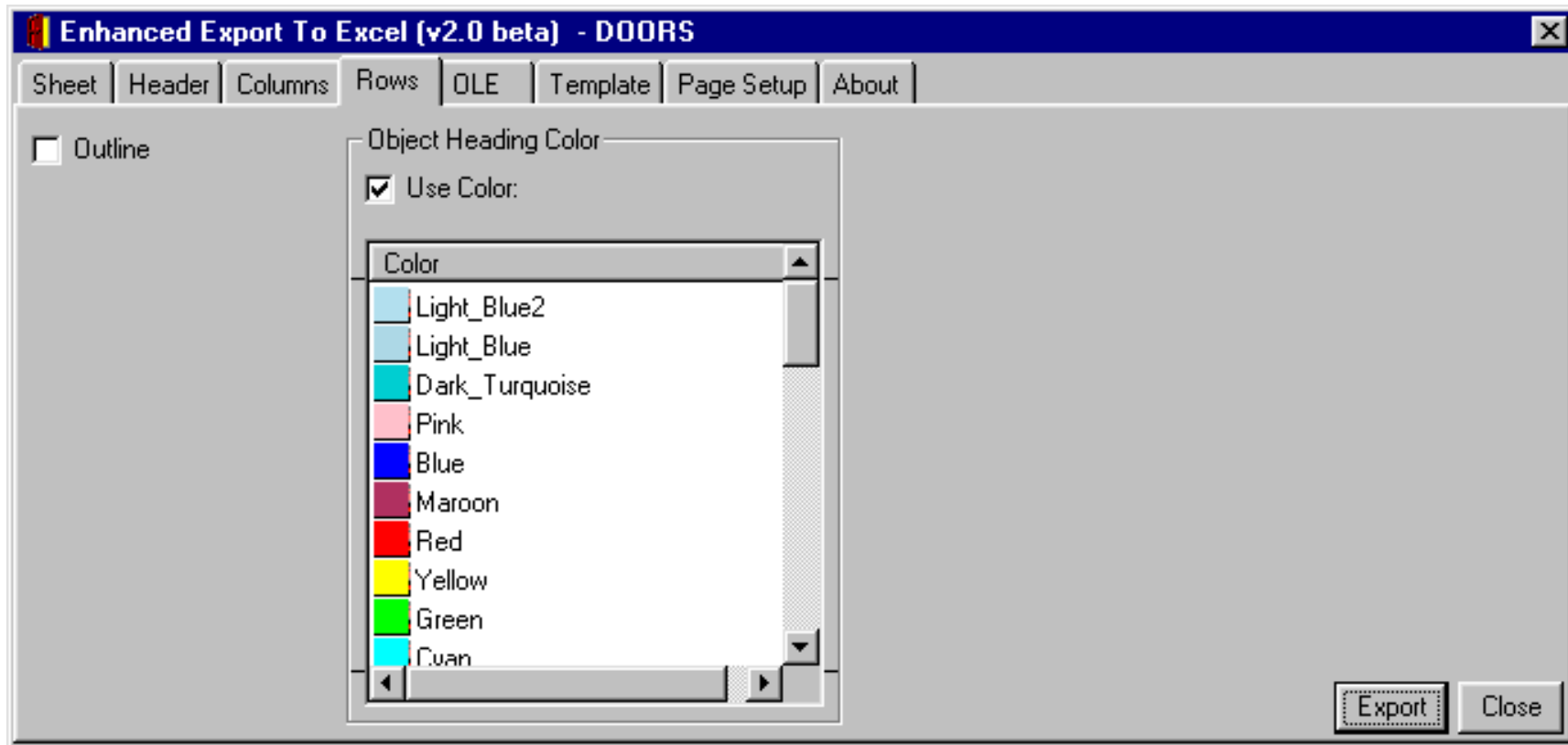


Enhanced Export to Excel - Columns



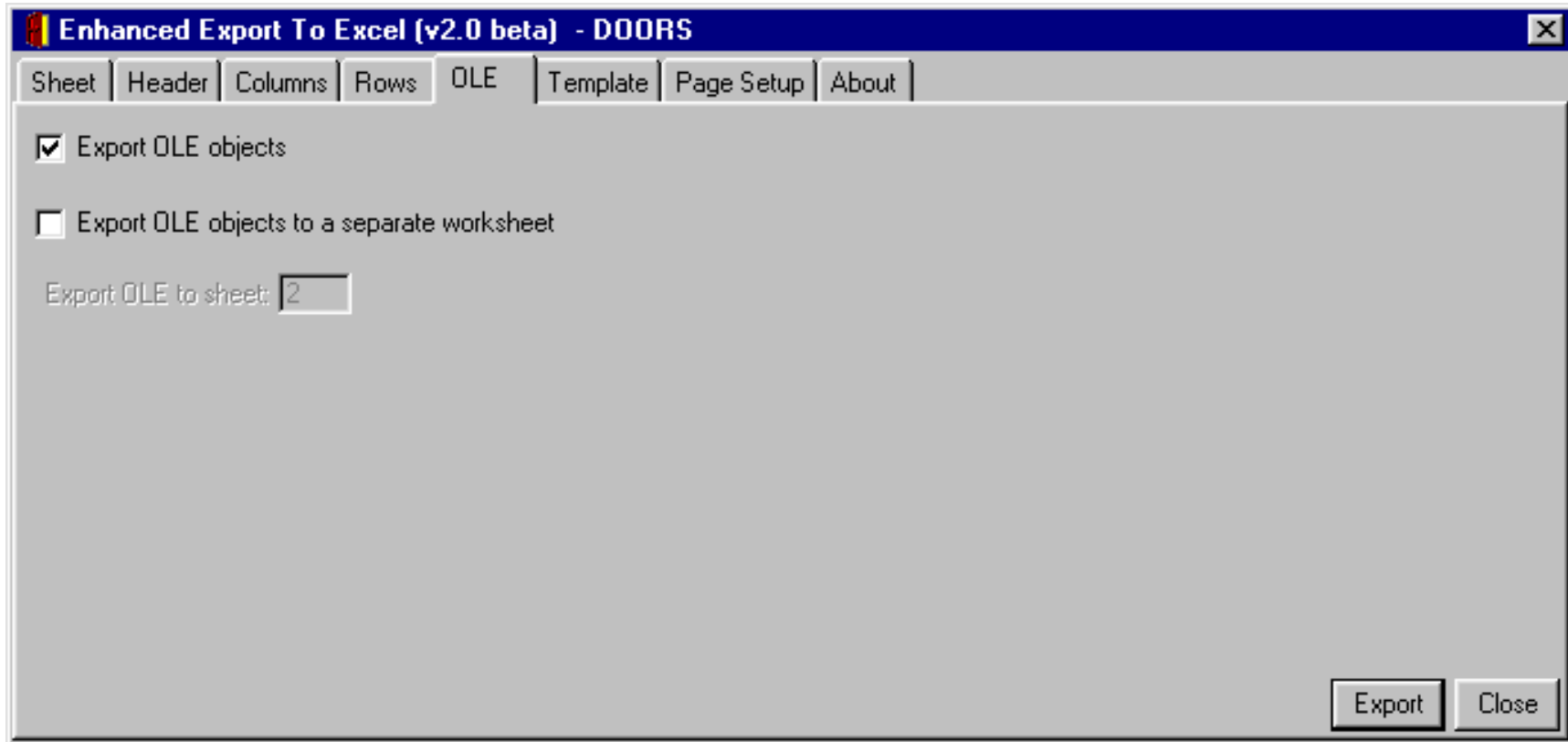


Enhanced Export to Excel - Rows



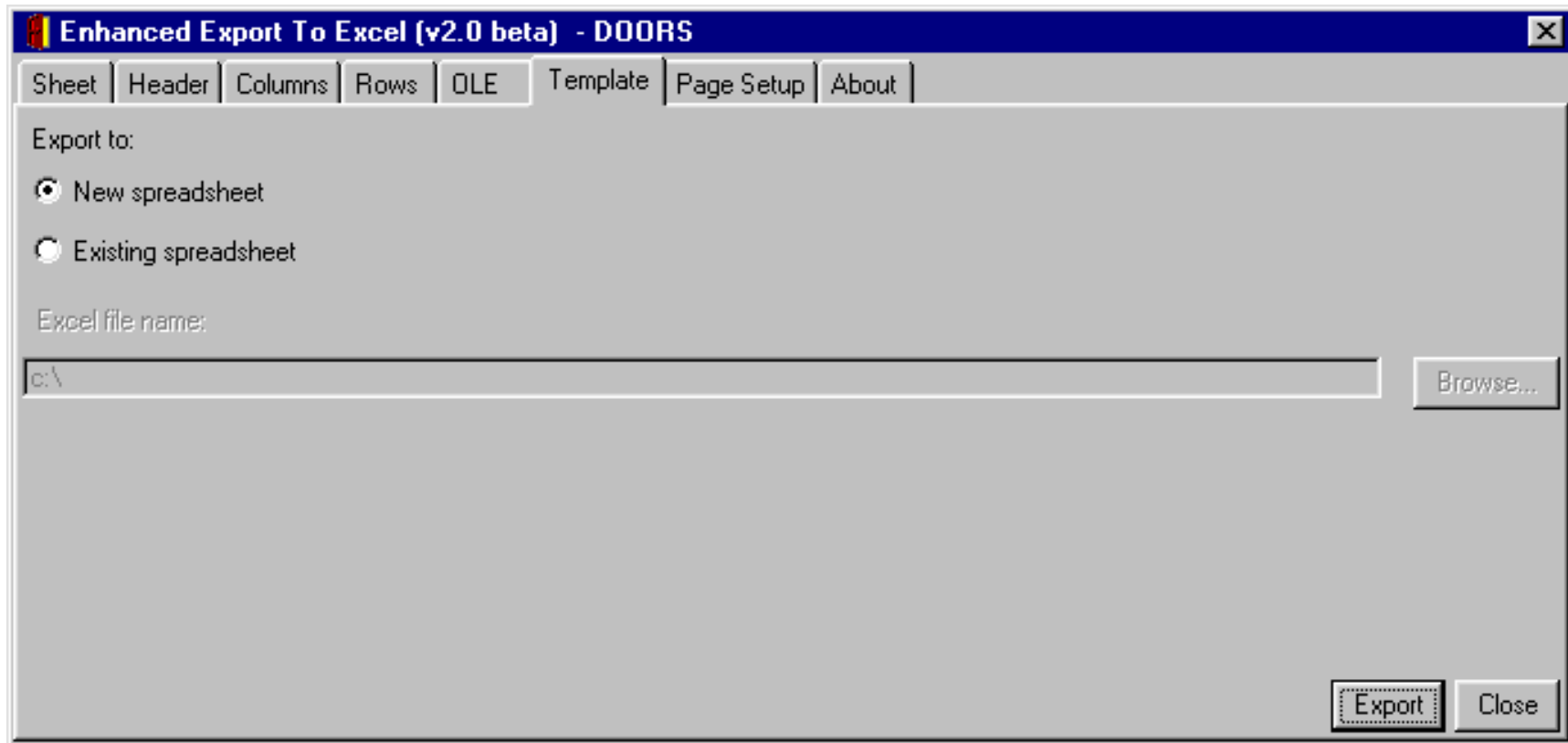


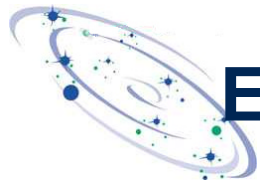
Enhanced Export to Excel - OLE





Enhanced Export to Excel - Template





Enhanced Export to Excel - Page Setup

The screenshot shows the 'Page Setup' dialog box for 'Enhanced Export To Excel (v2.0 beta) - DOORS'. The dialog has several tabs: Sheet, Header, Columns, Rows, OLE, Template, Page Setup (selected), and About. The 'Page Setups' list on the left contains 'Standard layout' and 'Test'. The main area is divided into several sections:

- Paper size:** A4 (dropdown), Width: 296, Height: 210
- Margins:** Left: 30, Right: 20, Top: 20, Bottom: 25
- Orientation:** Portrait (radio button), Landscape (radio button, selected)
- Column titles:** Every page (radio button, selected), First page only (radio button)
- Header and footer print options:** &N - page number, &C - page count, &M - module name, &P - project name, &V - version, &U - user name, &D - print date, &T - print time
- Header:** Three empty text boxes.
- Footer:** &M, Page &N of &C, Printed &D

Buttons for 'Export' and 'Close' are located at the bottom right.



DOORS Module to Export

Formal module 'EasyStart/Sub-systems/Car Use Scenario' current 0.0 - DOORS

File Edit View Insert Link Analysis Table Tools User BOP BIMS Help

Excel Export All levels

Car Use Scenario

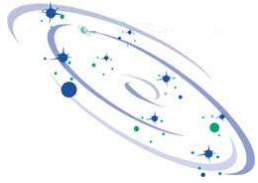
- 1 Plan journey
 - The user shall be able to
 - The user shall be able to
 - The user shall be able to
 - The user shall be able to
 - The user shall be able to
- 2 Prepare car
 - 2.1 Check car status
 - The user shall be able to
 - The user shall be able to
 - 2.2 Prepare basics
 - The user shall be able to
 - The user shall be able to
- 3 Undertake journey
 - 3.1 Outward journey
 - 3.1.1 Enter car
 - The driver shall be able to
 - 3.1.2 Load car
 - 3.1.2.1 Load luggage
 - The user shall be able to
 - The user shall be able to
 - 3.1.2.2 Load passengers
 - The driver shall be able to
 - The passenger shall be able to
 - 3.1.3 Drive car
 - 3.1.3.1 Control car
 - The driver shall be able to
 - The driver shall be able to
 - The driver shall be able to
 - 3.1.3.2 Condition
 - The driver shall be able to
 - The driver shall be able to
 - 3.1.3.3 Status
 - The driver shall be able to
 - The driver shall be able to
 - The driver shall be able to
 - 3.1.3.4 Journey status

Scenario for the use of the passenger car

Requirement	Priority
1 Plan journey	
The user shall be able to decide date of journey.	Important
The user shall be able to decide route of journey.	Minor importance
The user shall be able to estimate distance of journey.	Luxury
The user shall be able to estimate fuel required for journey.	Important
The user shall be able to estimate duration time for journey.	Important
2 Prepare car	
2.1 Check car status	
The user shall be able to ascertain fuel state.	Important
The user shall be able to ascertain maintenance state.	Luxury
2.2 Prepare basics	
The user shall be able to fuel car.	Important
The user shall be able to top up fluid levels.	Important
3 Undertake journey	
	Important

Category	Jan	Feb	Mar	Apr	May	Jun
Food	12	17	22	14	12	19
Gas	17	11	29	10	17	15
Motel	10	21	14	17	10	20

Username: Eric McCall Exclusive edit mode



DOORS Module Exported to Excel

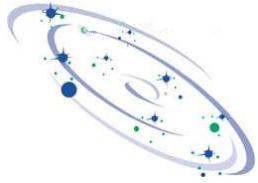
The screenshot shows an Excel spreadsheet with the following content and annotations:

- Outlining:** A callout box points to the collapse/expand icons in the left margin.
- Split and Frozen Header:** A callout box points to the top row of the spreadsheet.
- Heading Row Color:** A callout box points to the first row of the data table.
- Attribute Color:** A callout box points to the 'Important' and 'Luxury' cells in the data table.
- OLE Object "in" Resized Cell:** A callout box points to a bar chart embedded in a cell.

Scenario for the use of the passenger car	
1 Plan journey	
The user shall be able to decide date of journey.	Important
The user shall be able to decide route of journey.	Minor importance
The user shall be able to estimate distance of journey.	Luxury
The user shall be able to estimate fuel required for journey.	Important
The user shall be able to estimate duration time for journey.	Important
2 Prepare car	
2.1 Check car status	
The user shall be able to ascertain fuel state.	Important
The user shall be able to ascertain maintenance state.	Luxury
2.2 Prepare basics	
The user shall be able to fuel car.	Important
The user shall be able to top up fluid levels.	Important
3 Undertake journey	
3.1 Outward journey	
3.1.1 Enter car	Important

Bar Chart Data:

Category	Jan	Feb	Mar	Apr	May	Jun
Food	12	17	22	14	12	19
Gas	17	11	29	10	17	15
Motel	10	21	14	17	10	20



Issue - DOORS Tables

- What to do with DOORS Tables?
 - Current DOORS Excel Exporter “linearizes” them (Table Cells are exported in row order as equal children)

1 Heading One

Cell 1	Cell 2
Cell 3	Cell 4



1 2		A
1		1 Heading One
2	•	Cell 1
3	•	Cell 2
4	•	Cell 3
5	•	Cell 4

- Proposed solution:
 - Export as Word Table, embed into Excel as OLE Object



References

- Microsoft Visual Basic for Applications
 - VBA for Dummies®, 3rd Edition
http://www.dummies.com/extras/vba_fd_3e/
 - Introduction to Office Automation - Sheffield Hallam University
<http://maths.sci.shu.ac.uk/units/ioa/>
 - Microsoft Developer Network <http://msdn.microsoft.com>
- Microsoft Excel Automation
 - Jwalk & Associates, The Spreadsheet Page <http://j-walk.com/ss>
 - Pearson Software Consulting, LLC <http://www.cpearson.com/excel.htm>



Obtaining a copy of the Software

- DOORS Users are encouraged to obtain, use, share, and improve upon the software mentioned in this presentation.
- For a free copy:

Contact: michael@galactic-solutions.com

or download from

<http://galactic-solutions.com>