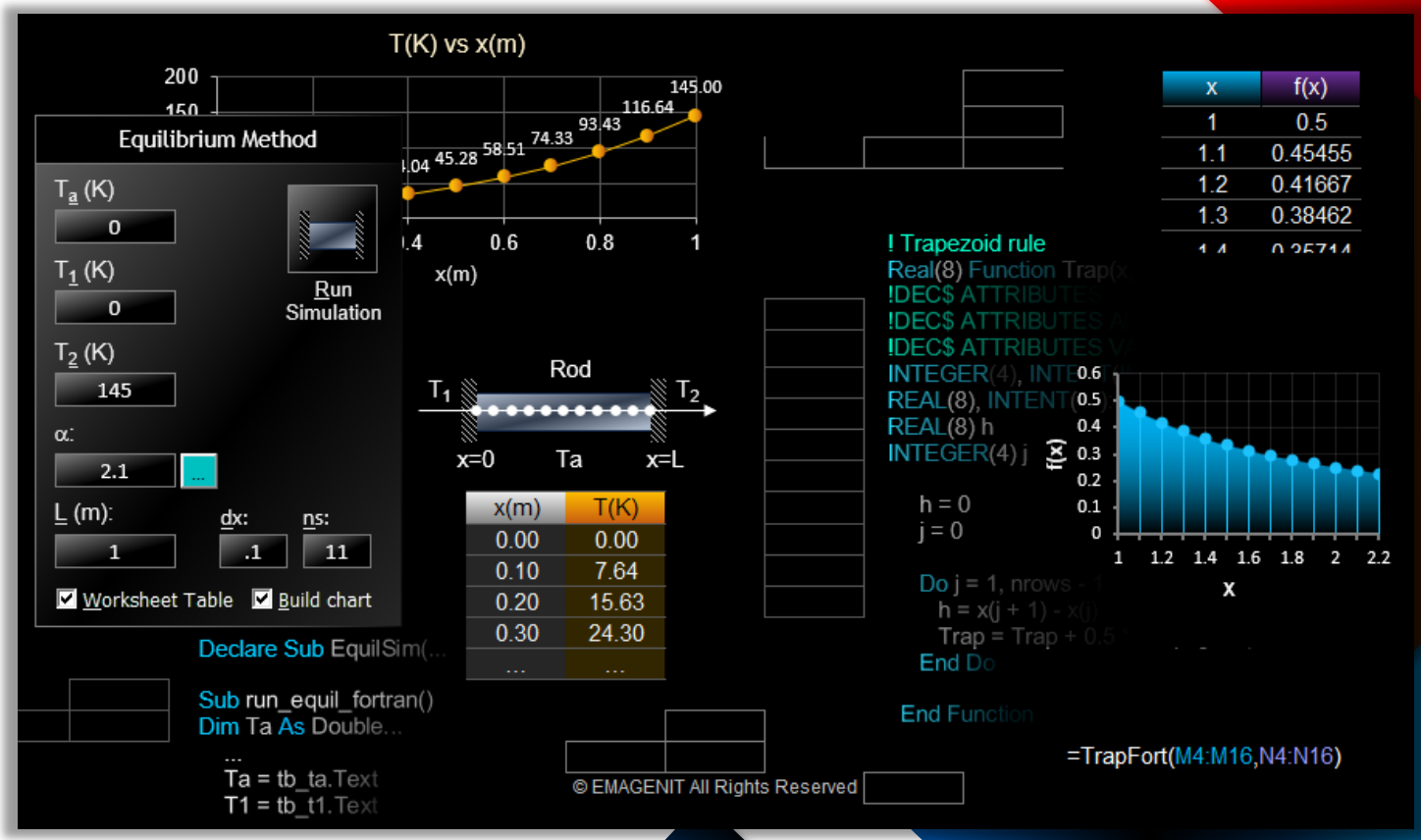
EMAGENIT
State of the art Excel training.

Running Fortran DLLs

From Excel VBA



Think Fortran is a dead language? What about all that verified legacy code? Learn to combine the speed of Fortran with the GUI, charting, modeling, and data processing capabilities of Excel VBA to create lightning-fast engineering and science tools.

www.emagenit.com >

How our class can help you.

Our 3-day class shows you “hands-on how” to create powerful tools by combining the speed of Fortran with the GUI, data, and modeling abilities of Excel VBA. It shows how to use the key Excel VBA and text file elements necessary to process, pass, and return Fortran DLL data.

Examples of what you’ll learn.

Also covered is how to design and compile Fortran DLLs or upgrade existing Fortran code to run from VBA. Our class also discusses how to automate Excel's data tools and charts; build models; and create GUIs from userforms, shapes, worksheets, and ActiveX to run Fortran DLLs.

Who should attend the class?

Engineers, scientists, and technicians. Class examples will be determined by those in attendance.

Minimum Excel skills needed for the class.

Select this Excel training if you or your group have:

- Operated Excel's data processing tools like Filter, Text to Columns, Remove Duplicates, Sort…
- Used worksheet functions before and built formulas
- Created charts, Excel tables, and used drawing shapes
- Have used Excel VBA before in a basic capacity
- Been exposed to the concepts of basic programming like loops, logic, functions, and variables
- Have used Fortran before in a basic capacity

How we run the class.

We focus our training on what our customers need. When training begins, we analyze those needs and shift our outline appropriately. We will stress or add topics that our customers want.

Class formats and signup.

In-Person, Virtually, and Onsite. Our live hands-on classes can be attended virtually or in-person. Please visit our public signup page for class times and pricing >. Contact EMAGENIT directly at 805.498.7162 for more information about our onsites.

Key Excel VBA topics covered in class.

- Full review of the VBA language, Excel objects, VBA Editor, and debugging
- How to track worksheet ranges, headers, and data in your VBA code
- Running Excel and VBA functions in your code to analyze and lookup data
- Using loops and logic to process and assemble Fortran DLL data and arrays
- Looping through workbooks, worksheets, and folders to gather data for Fortran
- Various ways to rapidly output Fortran data to the worksheet and format it
- Using Excel VBA to read / write Fortran DLL text files
- Basic coding strategies for designing and retrofitting Fortran code to run in DLLs
- Complete Fortran and VBA procedure calling and argument data typing rules
- Rules for compiling 32- and 64-bit Fortran DLLs
- Creating worksheet GUIs and events to run your Fortran DLLs
- Using userform, ActiveX controls, and events run Fortran DLLs
- Designing worksheet models and custom worksheet functions (UDFs) to run Fortran DLLs
- Using VBA to create charts, shapes, and PivotTables to display Fortran DLL output

Detailed class syllabus.

Day-1

VBA Language, Module, and Editor Review for Fortran DLL Projects

*** This section is discussed when needed in the class.**

- How to use the Visual Basic Editor, its debugging tools, and module types for Fortran DLL projects
- Review of the VBA language; variables; data types; logic operators; and Excel objects, properties, and methods
- Review of technical equation construction in VBA using the various math operators
- Review of VBA procedure design, scope, argument lists, and passing rules
- Working with VBA arrays; dimensioning arrays; clearing arrays; and re-dimensioning arrays

Controlling Ranges, Formats, Functions, and Table Lookups with VBA

- Using VBA to track worksheet data ranges, header positions, and data subsets
- Calculating math, trig, and statistical values in your code using Excel and VBA functions
- Parsing and merging date / time values using the Today, DateAdd ... functions in your code
- Performing table lookups in your code using the MATCH, VLOOKUP, XLOOKUP... functions
- Using Excel VBA to format worksheet tables and insert, delete... rows, columns, and cells

- Using VBA to control Excel's data features like AutoFilter, Sort, Remove Duplicates...

Using VBA to Control Workbooks, Worksheets, Folders, and the Windows Registry

- Tracking workbooks and worksheets in Excel VBA using the Set statement
- Opening, saving, adding, and closing workbooks with Excel VBA
- Using Excel VBA to add, move, delete, and rename worksheets
- Creating, deleting, moving, and copying folders and files with VBA
- Using SaveSetting, Getsetting... to store program settings in the Windows Registry

Using VBA to Assemble, Process, and Output Fortran Data

- Using the Range, Cells, Offset... properties in your code to access worksheet data
- Processing complex worksheet data patterns for Fortran using loops and logic
- Building arrays for Fortran from worksheet data using loops, logic, and ranges
- Scanning folders and workbooks for data using loops, logic, and the FileSystemObject
- Scanning open workbooks using loops and logic to find data and model information
- Outputting Fortran data to the worksheet using loops, logic, arrays, and range commands

Controlling and Processing Text Data with VBA

- Analyzing text data in your macro code using the Replace, Mid, Split, Format... functions

- Automating Text Wizard and Text to Columns to load and parse worksheet text data
- Using the TextStream object to open, close, read, and write to text files
- Writing Excel data to a text file using VBA loops, logic, and string concatenation
- Using loops, logic, and functions to analyze text file data and output it to the worksheet

Visualizing Fortran Output Using VBA, Charts, Shapes, and PivotTables

- Using VBA to create PivotTables and output Fortran data to them
- Using Excel VBA to create and format charts (line, xy scatter, column, area...)
- Reassigning Excel chart data sources with Excel VBA
- How to position charts on a worksheet using Excel VBA
- Using VBA to color chart data points, create limit lines, move data points...
- Using Excel VBA to create shapes and load Fortran data in them
- Combining shapes with charts to form visual displays

Day-2

Calling Fortran DLLs from Excel VBA Procedures

- Creating a VBA Declare statement for a Fortran DLL Subroutine or Function routine
- Calling a Fortran DLL Subroutine or Function routine from Excel VBA
- Using argument lists to pass data between Excel VBA and Fortran
- Function design rules for calling Fortran DLL routines
- Sub procedure design rules for calling Fortran DLL routines

- Property procedure design rules for calling Fortran DLL routines

Designing a Fortran Routine to be Called from VBA

- Using the STDCALL, ALIAS, and DLLEXPORT attributes to define your DLL entry points
- Using the REFERENCE and VALUE attributes to specify argument passing convention
- Designing Fortran arguments to accept VBA numbers, strings, Boolean, logic... values
- Laying out Fortran arguments to accept string arrays
- Creating Fortran arguments to accept various array types (explicit, adjustable…)

Passing Data Types Between Fortran and VBA

- Passing numbers, dates, and Booleans between VBA and Fortran
- Passing numeric arrays between VBA and Fortran
- Passing strings between VBA and Fortran
- Passing string arrays to Fortran from VBA
- Passing a user defined type to Fortran from VBA
- Passing a complex number between VBA and Fortran
- Passing a Fortran derived data type back to Excel VBA

Using Text Files to Pass Data to a Fortran DLL

*** General text file control discussed on Day-1**

- Rules for assembling VBA text files for Fortran DLLs
- Designing Fortran routines that quickly read text files into variables and arrays

- Using VBA, ADO, SQL, and Power Query to query large Fortran output text files

Compiling and Debugging a Fortran DLL

- Compiling a Fortran DLL in Visual Studio using Intel Visual Fortran
- Debugging a Fortran DLL from Fortran
- Debugging a Fortran DLL from Excel VBA
- Different ways VBA reacts when it encounters an error in a Fortran DLL

Day-3

Creating Worksheet GUIs to Control Your Fortran DLLs

- Designing interfaces from worksheet cells, Data Validation, ActiveX controls, and shapes
- Tracking worksheet information in VBA using cell and table names
- Using Conditional Formatting to auto format DLL inputs and outputs on the worksheet
- Using VBA to control and read ActiveX controls on the worksheet (load list boxes, set scroll bars...)
- How to design ActiveX control and worksheet events to run your DLLs
- Using shapes and VBA to display program information, status....
- Designing VBA procedures to run your code at specific times or by pressing a key

Creating Userforms and Folder / File Pickers to Run Your DLLs

- When to design a modal vs. a non-modal userform; userform execution strategies
- How to create ActiveX controls, text, and pictures on userforms

- Designing a VBA procedure to display a userform and load its controls
- Using VBA to command ActiveX controls (check boxes, text boxes, scroll bars...)
- Using VBA to load combo boxes and list boxes
- Designing userform and ActiveX control events to run Fortran DLLs
- Designing toolbars, shortcut keys, and Ribbon buttons to run your Fortran DLLs
- Creating folder / file pickers to track files and folders

Designing Excel Models to Run Fortran DLLs

- Laying out model input and output ranges on the Excel worksheet
- Using ActiveX controls and Data Validation to control user input
- Using names and Excel tables in VBA to rapidly access model data
- Creating VBA Sub procedures to calculate and assemble model data for DLLs
- Using VBA to discretize calculations for use in Fortran numerical methods
- Designing custom worksheet functions (UDFs) to execute DLLs from formulas
- Using VBA to create worksheet formulas on the fly and output results
- Using VBA to create model upload, download, and trade study procedures