marxer
engineering & computing

Mathematica and Excel

Data Exchange and Control

Author: Norbert Marxer

Version V01R02 (31 October 2005)

Abstract

Both Mathematica and Excel are excellent and widely used programs. Quite often people wish to transfer data between these two applications or write some code so that these two applications can communicate with each other.

This documentation package (.pdf and .nb file, a palette with clipboard functions, a style sheet, approximatley 40 .xls / .csv / .txt / .xml ... example files) covers three main areas:

- **Data exchange via clipboard**
  How can you exchange data between *Mathematica* and Excel using the clipboard? Single cells, tables, charts etc. A palette is derived and implemented which automatically transforms between *Mathematica* tables (list of lists) and Excel tables in the clipboard (Tab and Newline separated strings).
- **Data exchange via files**
  How can you exchange data between *Mathematica* and Excel using files? Using different formats like CSV, XLS, XML etc.
- **Data reading and writing using .NETLink**
  How can you write directly into Excel work sheets using *Mathematica* and .NETLink? Using interop assemblies. Writing values and formulas, doing some formatting, exchanging charts etc.

Table of Contents

Introduction

Both *Mathematica* and Excel are excellent and widely used programs. Quite often people wish to transfer data between these two applications or write code so that these two applications can use the functions of the other application or can communicate with each other.

There are many methods to facilitate the data transfer, the usage of these external functions or to code such a communication.

Data exchange via clipboard

One method for data transfer is to exchange data via clipboard. This can be done interactively using shortcut keys (e.g. Ctrl+C), menu items or programmatically using *Mathematica* FrontEnd programming with the functions FrontEndExecute and FrontEndToken with the appropriate Tokens. Using the clipboard both data (single cells, multiple cells, tables, text, numbers, macro code, formulas) and charts can be exchanged.

The following methods using the clipboard will be discussed in this documentation:

- a palette which makes it possible to write a selected *Mathematica* table (list, or list of lists) to the clipboard with a button click in such a format that it can be pasted in Excel using Ctrl+V;
- a palette which makes it possible to read a selected Excel table (Range object) by using Ctrl+C in Excel and pressing a button in *Mathematica* which will insert it in the *Mathematica* format (with curly brackets);
- the construction of this standalone palette and all its functions are discussed thouroughly; it will not only work with Excel but also with other spreadsheet programs like "Calc" from OpenOffice.org;
- the behaviour ot the ClipboardNotebook[] and the different Copy As and Paste As commands are investigated using FrontEnd programming;
- you will also use JLink and Java commands to inspect the clipboard content and see the many different formats in which the copied data are available; you can also acces the HTML or Rich Text Format (and other formats) if you know how to interpret it.

Data exchange via files

Another method is data exchange via files. Both *Mathematica* and Excel can read and write text files, e.g. data which are separated by tabs, commas or spaces. Starting with Version 5.1 *Mathematica* also supports the proprietary (Microsoft Excel format) XLS. It can be used for Import and Export. But note that it is not possible to read formulas, charts, Visual Basic for application code etc. Another important format is XML, which is supported by both applications.

The following methods using file reading / writing will be discussed in this documentation:

- in Excel: we will discuss which file format (.xls, .csv, .txt, .prn, .xml ...) you should choose to import / export tables, text, numbers, formulas, charts, ...
- in Excel: a short Excel macro is given to export a chart into a file which can be imported in *Mathematica*;
- in *Mathematica*: we will discuss which file format for the function Export (CSV, Table, Text, ...) you should choose to import / export tables, text, numbers, formulas, charts, ...

- in *Mathematica*: you should not (as is usually recommended in the MathForum) read Excel tables using the Import function with the format "Table" because empty rows are lost (which makes all the formula cell references obsolete); the solution to this is given;
- in *Mathematica*: we give the format you have to choose to write more than one Worksheet into a file;
- in *Mathematica*: we explain how to write an .xml file in the XMLSS specification;
- String matching and Expression matching techniques show how to extract the information from imported data and convert to *Mathematica* expressions;

Write into and Read from Excel using .NETLink

It is also possible to (directly) write tables and create charts in Excel and read from Excel in *Mathematica* using .NETLink and the Excel libraries. This means that you can use *Mathematica* as a development tool for Excel files. You can programmatically create Excel files, charts, unique formatting etc. Or you can test the calculations done in Excel with the calculations in *Mathematica,* find out whether the Excel functions and algorithms work as they should, whether the machine precision is good enough, whether (when optimizing) the global minimum is found etc...

The following methods using .NETLink to work with Excel will be discussed in this documentation:

- a short introduction to .NETLink programming and the use of (Primary) Interop Assemblies is given;
- you will find many links for downloading the PIA's, the (Excel) language references and more, and how to get information on the available Excel functions;
- *Mathematica* code is supplied which starts Excel, adds Workbooks and Worksheets, sets cell content, sets tabular data, makes charts ... and also quits Excel;
- you can call these functions interactively or programmatically;
- in this way you can use *Mathematica* as a development environment and manage your Excel applications (formulas, formatting) completely from *Mathematica*;

The package "*Mathematica* Link for Excel"

In the methods mentioned above the main focus was the *Mathematica* side. In contrast to the above you could also call *Mathematica* functions and .m files from Excel using the package "*Mathematica* Link for Excel". It is also possible (within Excel) to use the *Mathematica* DatabaseLink and have access to databases. This means that you can evaluate the data, produce nice graphical output (both in *Mathematica*) and embed it in Excel. Because the focus of the package "*Mathematica* Link for Excel" (which costs $189 at the moment) is the Excel side, it is not so relevant to what is described in this documentation.

This document

This document gives a systematic overview on the methods and tools mentioned above and gives answers to the following questions:

- How can you exchange data between *Mathematica* and Excel using the clipboard? Single cells, tables, charts etc.
- How can you exchange data between *Mathematica* and Excel using files? Using different formats like CSV, XLS, XML etc.
- How can you write directly into Excel work sheets using *Mathematica* and .NETLink? Using interop assemblies. Writing values and formulas, doing some formatting, exchanging charts etc.

You can find quite a lot of information distributed all over the WEB. From time to time you can find questions in the MathForum at the Wolfram Inc. WEB pages related to the questions above. This document gives a detailed overview and many links to important and related areas.

I am convinced that - by using this document - you can save many hours (probably days). Instead of searching the WEB, the *Mathematica* HelpBrowser, the .NETLink, JLink, Visual Basic for Applications, Excel etc. documentation you will get a quick start and can study and learn from many examples.

Software Used for this investigation

Note that this document was developed using the following configuration:

- *Mathematica* Version 5.2.0.0 (English version)
- Microsoft Excel 2002 (German version)
- Microsoft Windows XP

It is possible that you may get different results if you are working on a different operating system and / or are using different versions of the software (*Mathematica*, Excel, COM or interop libraries).

But because most of the time the single steps are described in great detail, you should be able to reproduce the results and - if not - make the (minor) adjustements to the code which are appropriate for your system.

Style Sheet Used

This notebook uses the Style Sheet HelpBrowserNM.nb which has been (slightly) adapted from the Standard *Mathematica* HelpBrowser.nb style sheet. The main differences are:

- all the input cells are in blue;
- all the output cells are in green;
- a BulletedList2 style has been added which indents more than the standard BulletedList;
- InputSmall and OutputSmall styles have been defined to display (long) input and output with smaller font size and line/paragraph spacing;

Disclaimer

mec (marxer engineering & computing) does not warrant or assume any legal liability or responsibility for the accuracy, completeness, reliability, suitability or usefulness of any information in this document.

Note also that the code of this documentation has been run and tested on one special system and configuration and it can not be assumed that it will also run on other systems and configurations in exactly the same way.

Copyright Notice

Data Exchange via Clipboard

Introduction

This chapter on "Data Exchange via Clipboard" discusses various clipboard methods for transfering data between *Mathematica* and Excel. Some of the methods mentioned here can also be used for other applications.

Data transfer via clipboard is basically a very simple method and involves the following steps:

- you select an item in application A (mostly using the mouse);
- you copy this selection to the clipboard (mostly using the shortcut key combination Ctrl+C);
- you position the cursor in application B where you want to insert the clipboard content (mostly using the mouse);
- you paste the clipboard content in application B (mostly using Ctrl+V);

It is easy to select and transfer single items like ...

- a text string, a number or a graphics (in *Mathematica*)
- a cell value, formula, VBA Code, chart etc. (in Excel)

... using Ctrl+C and Ctrl+V (or the corresponding menu commands), because both *Mathematica* and Excel know how to handle the other application's string, number or graphics.

The task is not so easy if you want to keep the formatting information.

Furthermore it is also not so easy (but very much desired) to transfer tabular data, the main objects in spreadsheet programs. In *Mathematica* a table is represented as a list of lists. In Excel a table is a Range object. Copying (e.g.) a 2x2 table {{a,b,},{c,d}} in *Mathematica* and Pasting it in Excel fills one cell with the curly brackets and values (we would prefer the individual table elements to be put in separate cells). Vice versa copying a 2x2 table in Excel and pasting it in *Mathematica* leads to ...

```
a b
c d
```

... or ...

```
abcd
```

... depending on which Paste (Paste, PasteSpecial ...) you select.

In other words: you can transfer the data, but they won't have the form you would like them to have. The solution to this problem is to convert (within *Mathematica*) the table before copying it to the clipboard and after reading it from the clipboard. To make the whole process as user friendly as possible, this conversion is done by clicking a palette button. Then the whole process of transfering a table using the clipboard consists of the usual steps with the exception that Ctrl+C and Ctrl+V are replaced by clicking the corresponding palette buttons.

Step by Step it will be shown how you develop such a palette with all the needed functions, which will allow you to transfer tables between *Mathematica* and Excel. The same palette will also work for other spreadsheet programs like e.g. Calc from OpenOffice.org.

The table transfer mentioned above is not the only topic which will be discussed in this chapter. There are more questions like: What will happen if you copy several cells to the clipboard? What happens if you use different Copy As parameters? What happens to the formatting. How can you use JLink to extract the clipboard content in the format you wish (note that the clipboard content is usually present in many different formats).

Table of Contents

The section titled "*Mathematica* Clipboard Commands" discusses the different methods (Cut, Copy, Copy As, Paste, Paste As) you can use to put various content (cell, formula, number, tables etc.) into the clipboard or extract it from there. Also discussed are different ways (interactive, programmed) you can use to accomplish this task.

The section "*Mathematica* (Ctrl+C) → Clipboard → *Mathematica* (Paste)" serves as a preparatory step for the sections which follow and looks closely at how you can use FrontEnd commands and programming to copy/paste whithin *Mathematica*. We will find that the built-in ClipboardNotebook[] does not always behave as we would like it to behave. Nevertheless we will find a solution to this.

The section "*Mathematica* (Ctrl+C) → Clipboard → Excel (Paste)" discusses how you can best transfer *Mathematica* objects to Excel.

The section "Excel (Ctrl+C) → Clipboard → *Mathematica* (Ctrl+V)" discusses how you can best transfer Excel objects to *Mathematica*.

The section "Conversion of Tables" discusses (the very important) method how lists and tables can be exchanged between *Mathematica* and Excel.

The section "Clipboard Palette" derives and implements a palette, which performs important clipboard functions using functions which were defined and explained in earlier sections.

The section "Transfer using JLink" discusses methods using JLink to get access to the clipboard content. You will see that simple content is (sometimes) stored in an enormous number of formats, ranging from plain/text over HTML to RTF ...

After reading this section you will have a thourough understanding of the *Mathematica* clipboard functions and know how to write FrontEnd programs, how to transfer tables between *Mathematica* and Excel, how to build a standalone palette, how to use JLink to inspect the clipboard content and much more ...

Mathematica Clipboard Commands

Mathematica (Ctrl+C) → Clipboard → *Mathematica* (Paste)

Mathematica (Ctrl+C) → Clipboard → Excel (Paste)

Excel (Ctrl+C) → Clipboard → *Mathematica* (Ctrl+V)

Conversion of Tables

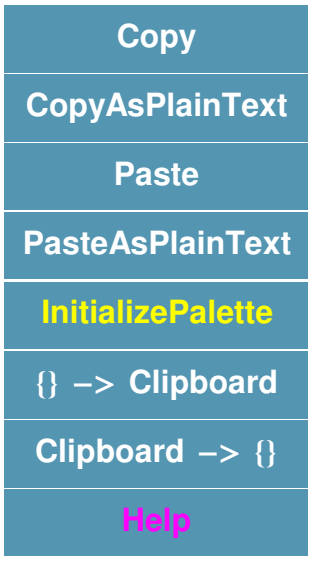
Clipboard Palette

Introduction

In the previous section we developed the functions which are needed to transfer *Mathematica* tables to Tab/Newline separated strings in the clipboard.

The tables had to be entered manually into the functions. Now we want to make life even easier. We want to use the mouse to select the table which should be copied to the clipboard and click a palette button to transfer it to the clipboard. And we want to use the mouse to select the position (in *Mathematica*) where to place the table from the clipboard and click a palette button to put the table there.

We will use the following Palette, which will be explained and derived below.



Help Button

The clipboard palette has eight Buttons. The first four are just the standard menu items. The next four are used in conjunction with table conversions. The last button is a Help button whose (exact) content is given in the following subsubsection. The information given below explains both here and also interactively in the palette the meaning of the different buttons. After this subsubsection it will be explained how this Help information can be embedded easily when building the palette.

Help for the ClipboardPalette.nb

You have to use the "Initialize Palette" button to define the functions which convert and copy/paste tables between *Mathematica* and Excel.

The "Initialize Palette" button will define the following global symbols: convertToExcelFormat, convertToMathematicaFormat, myClipboardNotebook, myClipboardExists, myCBNB, getTableFromClipboard and putTableToClipboard. Do not change these definitions.

The first time you use the button which performs a table conversion, the notebook myClipboard might appear. You can minimize the window, but you may not close it. Otherwise the conversion won't work anymore.

If you want the conversion to work again you have to click the "Initialize Palette" button again.

The palette has the following buttons

- "Copy", "CopyAsPlainText", "Paste", "PasteAsPlainText"
  These are just the standard menu items.
- "Initialize Palette"
  This will define all the needed functions and open the notebook myClipboard[]. Do not close this if it will become visible, minimize it instead.
- "{} -> Clipboard"
  This converts a (selected) *Mathematica* table (or list) into a Tab and Newline separated string and puts it to the clipboard, from which it can be pasted in Excel using Ctrl+V.
- "Clipboard -> {}"
  This converts an Excel table (or list), which has been copied to the Clipboard (using Ctrl+C) to a *Mathematica* table (or list) and pastes it at the cursor position.
- "Help"
  This opens a new notebook which gives you some Help information. You can close the notebook by clicking the close button (X) at the top right corner of the window.

Disclaimer: the palette can be used as is. It is not guaranteed that it works in all environments and under any circumstances.

Paste the Help into the Palette

To Build the Clipboard Palette

To Make a Standalone Palette

Clipboard and JLink

Data Exchange via File

Introduction

Another method for data exchange between two applications is via file writing / reading. The first application writes to a file, the second application reads from this file. One of the advantages of this method is, that Excel does not have to be running, that you don't even have to have Excel installed on your system. For reading you just need a file which contains the data, formulas etc.

It is important that *Mathematica* and Excel support a common file format, a format which is understood by both programs: more specifically *Mathematica* should be able to write files, which can be read by Excel, and *Mathematica* should be able to read files, which have been written by Excel.

To complicate matters there are different kinds of (Excel) data objects which can be exchanged: strings, numbers, formulas in tabular form, charts and other objects. Different methods have to be used to transfer each kind.

It will turn out that the *Mathematica* Import and Export functions (with more than 80 *Mathematica* file formats) with the "Table", "CSV", "Text", "XLS" and "XLM" format are the most useful. Some options can be set to finetune (e.g. specify row separators) the transfer.

The following table gives a short overview:

Table, CSV, Text	Excel and *Mathematica* can both read and write text files. For the exchange to work Excel and *Mathematica* have to use the same separators for the columns and rows of the tabular data.
XLS	This is the Microsoft proprietary format of Excel. These files contain the tables, formulas, charts, macro code etc; Mathematica can only read and write the tabular values;
XLM	Excel can store files in an XML format. Only tabular values, formulas, but no charts and macro code are stored in theses files.

Similarly in Excel - when using the Save As command - the file formats "CSV", "TXT", "XLS" and "XLM" are the most useful.

Table of Contents

In this section we will investigate which file formats are most useful to transfer data and how the files are written and read both by *Mathematica* and Excel.

The section "Excel file formats" will discuss the different Excel file formats which can be used.

The section "*Mathematica* file formats" will discuss the available file formats using the *Mathematica* Import and Export functions. First the different formats for textual and tabular data are analyzed.

The section "*Mathematica* file formats for textual and numerical data" investigates the text data formats thouroughly.

After these preparatory steps the section "Excel tabular data via .txt, .csv, .tsv, .dat" will discuss the exchange of tabular data between *Mathematica* and Excel by reading from and writing to text files. We will discuss which format and file extension should be chosen when exporting tables and which format and what kind of further processing when importing tables.

The section "Excel Formulas via .csv, .txt" explains how you can transfer Excel formulas from *Mathematica* to Excel and back.

The section "Excel tabular data via .xls" shows how tabular data can be exchanged using the proprietary Microsoft format, which is supported by *Mathematica* since Version 5.1. It also discusses what kind of data *Mathematica* can import from an .xls file and what kind of data it can write to an .xls file.

The section "Excel Workbooks via .xml" looks more closely at the features of .xml files. We will present examples showing the structure of Excel .xml files, will discuss the XMLSS specification, show how to read .xml files and extract the desired information using string matching and expression matching techniques and finally export an .xml file (in the XMLSS specification) which can be opened by Excel.

The section "Excel Charts via .jpg,.gif" will explain how charts can be exchanged via files (e.g. the graphics formats .gif, .jpg). We will use Export in *Mathematica* and a short macro in Excel to create such graphics files, which can be easily imported both in *Mathematica* and Excel.

The section "Excel Macros via .bas" discusses methods how Macro code can be imported from Excel or how Macro code can be written in *Mathematica* and transfered to Excel.

After reading this chapter you will know how to exchange tables, formulas, charts and macro code between *Mathematica* and Excel via files, which Import and Export format is best in each case, what kind of data processing is necessary besides Import and Export, what you have to do on the Excel side, how you can use sophisticated string and expression matching techniques, and more ...

Excel file formats

Mathematica file formats

Mathematica file formats for textual and numerical data

Excel tabular data via .txt, .csv, .tsv, .dat

Excel Formulas via .csv, .txt

Excel tabular data via .xls

Excel Workbooks via .xml

Excel Charts via .jpg,.gif

Excel Macros via .bas

Data Exchange and Control via NETLink

Introduction

So far we have discussed data exchange between *Mathematica* and Excel: either via files or via clipboard. One application puts an object there and the other program extracts it from there.

In this section we will talk about doing everything in *Mathematica.* We will read data from an Excel Workbook and we will write into an Excel Workbook. We do not have to enter any input or modify something in Excel, but we can do so.

One requisite is - of course - that Excel is installed on the computer. Then - in order to read / write into Excel - requires (in *Mathematica*) to launch the .NET runtime and to load the Excel libraries.

Launching the .NET runtime

If you have Version 1.1 (at least) of the .NET Framework installed on your computer, you can launch the .NET runtime with two *Mathematica* commands. If not, you have to download the .NET Framework from the web first. The software is free of charge.

Loading the Excel libraries

The Excel classes, methods, ... are available in a socalled COM library. To get acces to these libraries you can choose between two methods: late binding and early binding. With late binding you do not have to perform any preparatory steps and you can create an Excel application object just by calling the function CreateCOMObject. There are some disadvantages though using this method and therefore the second method with early binding is used in this document. To use early binding the COM library has to be converted to an interop assembly first. You can do this

- using a freely available conversion software (i.e. tlbimp.exe); or
- using the *Mathematica* function LoadCOMTypeLibrary; or
- downloading the interop assembly from the web;

After these preparatory steps all you need is the command LoadNETAssembly[...] to get access to all the Excel functions.

Available Excel functions

There are several methods to find out which Excel functions are available for you to use:

- The .NETLink command NETTypeInfo[...] will give you all the classes, interfaces, structures, delegates and enumerations of NETAssembly[] expressions and it will give you all the type, constructors, properties, methods and events of NETObject[] expressions.
- The Excel documentation. Many Links are given below.
- The object catalog in the Excel Visual Basic Editor.
- Sample C#, Visual Basic ... Code.

In the following three sections the steps mentioned above are explained in great detail. It is also highly recommende to read the .NETLink documentation in the Help Browser and the Excel documentation (e.g. on the web), but it is not necessary to do that in order to understand the content of this section.

After these preparatory steps, in the fourth and later sections we will show how you can control Excel from *Mathematica* interactively and programmatically. We will explain how to start Excel, to quit Excel, to add a Workbook, to add a Worksheet, to read and write tabular data, create charts ... all from *Mathematica*.

The last section will summarize the important commands which have been explained and derived in this document.

After reading this section you will be able to manipulate Excel from *Mathematica,* read / write data, create charts, know how to use .NETLink, how and where to get relevant information on Excel functions and the interop assembly, and much more ...

Launching the .NET runtime

Calling COM from *Mathematica*

Excel functions overview

Function Prototype

Interactive Control

Start Excel

Quit Excel (and garbage collection)

Add a Workbook and Worksheets

Add a Worksheet

Read and Write the content of an individual Cell or Range

Add a Chart

Further commands

Summary of Important Commands

Appendix

Other Methods

Initialization and System Dependent Code