

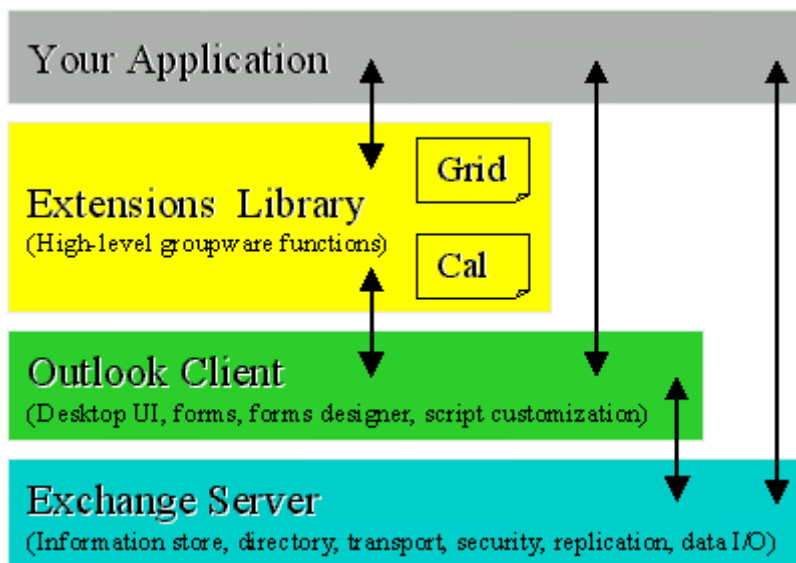
Extensions Library *for Outlook*

Microsoft Outlook provides an attractive form-based environment for developing collaborative end-user applications. For the simplest projects, Outlook's base functionality can sometimes be sufficient. In order to deliver sophisticated, real-world solutions, however, it is usually necessary to add customized programming.

The Extensions Library *for Outlook* is an object-based toolkit that provides you, the developer, with a robust set of objects and methods to support you in delivering complex applications to your users. The Library turns Outlook into a true productivity tool, allowing you to provide real business solutions without having to 'reinvent the wheel'.

The Library is packaged as an ActiveX DLL, along with several ActiveX controls. Its objects and methods are called directly from your Outlook forms, using the standard VBScript language contained within the form. Using the Extensions Library in your forms means:

- the applications you provide to your users are greatly enhanced in functionality
- the amount of program code needed in your Outlook forms is dramatically reduced
- the reliability of the solutions you deliver is vastly improved because you are taking advantage of proven, time-tested components developed by experts in the field



The following sections provide you with more details about the capabilities of the Extensions Library.

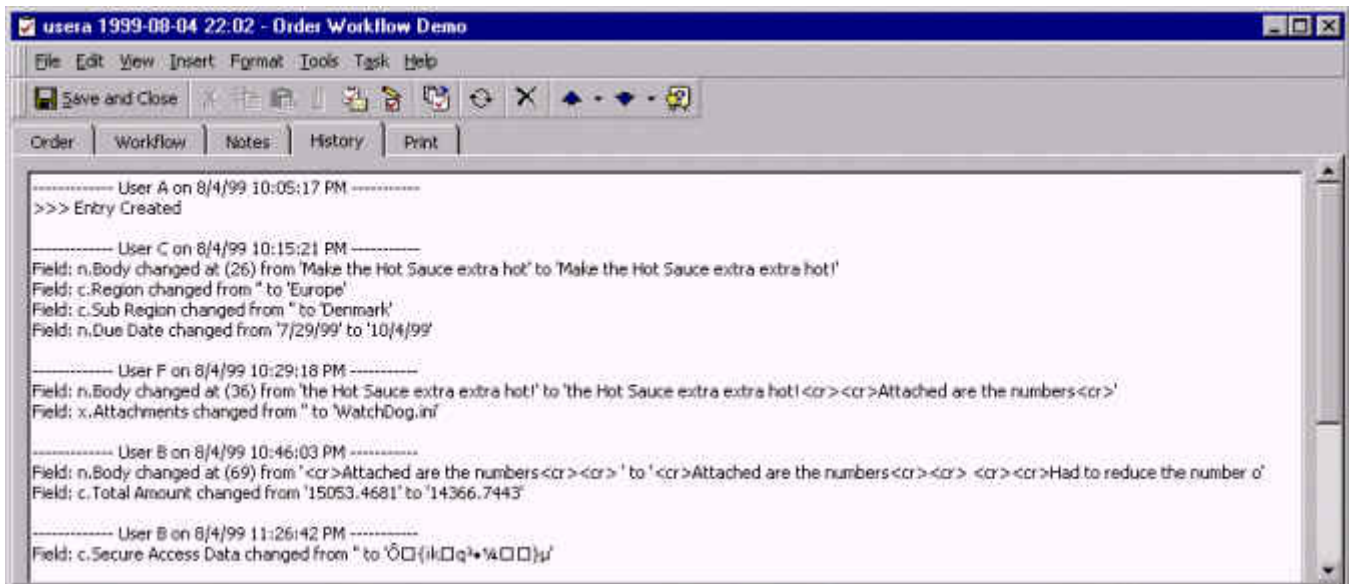
History Tracking

This set of object methods provides for easy-to-use tracking of changes to fields on your Outlook form.

With two simple Library calls, one during the form's *Item_Open* event and one during the *Item_Write* event, your form will have a complete audit trail capability, detailing which users changed what data, and when the changes occurred. This information is maintained in a custom field on the form itself.

You also have the ability to specify *which fields* on your form should be included in history tracking. Automatically handles complex Memo and Keyword fields intelligently.

Below is a sample of a history field:



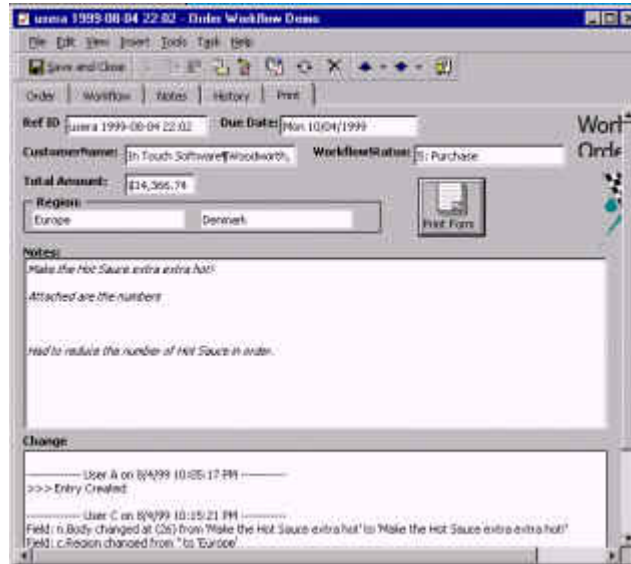
Item Locking

Outlook uses "optimistic locking" for its items: if two people open and change the same item at the same time, the first one to save will 'win'; the other person's edits will either be lost or will generate a conflict message.

The Extensions Library *for Outlook* includes the capability for you to implement "pessimistic" locking in your VBScript form code. One function call to the Library locks an item when it is opened. If another user attempts to open the same item, information will be returned that the item is locked, by whom, and when. You can then use this information to display a message to the user, disable certain controls on the form, etc. Our locking method does not "dirty" your form on read only operations like simply updating a field would. This non-dirty locking is important to make sure to not cause false modifications and unnecessary update traffic for off-line or low bandwidth users.

Graphical (WYSIWYG) Form Printing

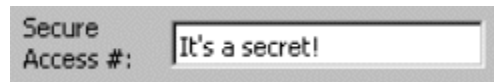
Outlook lacks the capability to print a form in its true on-screen graphical format. The Extensions Library provides methods for you to accomplish this. You can also choose between bitmapped or full form layout (vector) rendering. Simply layout your controls as you want them to appear and adjust their properties as desired.



Field-level Data Encryption

Often your Outlook forms contain certain fields of information that should be accessible to only a subset of that form's users. The Extensions Library gives a form's developer the ability to encrypt, or encode, any selected fields on an Outlook form.

Using the Exchange Global Address List (GAL) for authentication, only those designated users will see the decoded data. All other users will see a string of encrypted characters. Plus, the list of authorized users is maintained in a separate public folder which is accessible only to the form's designer or manager.



Secure
Access #: It's a secret!

Authorized users see the actual data



Secure
Access #: *****

All others see a string of asterisks



Secure Access Data 01k1q7P4llμ

The data is stored within Exchange as an encrypted string

Read/write of Table Data

Many times when building an Outlook form-based application, you have a need to store and retrieve a set of values from a database table. Common examples of this need are configuration and table lookup information.

One solution to this need is to use Access or other ODBC/ADO data sources, called directly from your VBScript form code. While this approach can work, it increases your application's complexity, and deviates from the built-in information store capabilities of Exchange. It is often "overkill" for relatively simple data access needs.

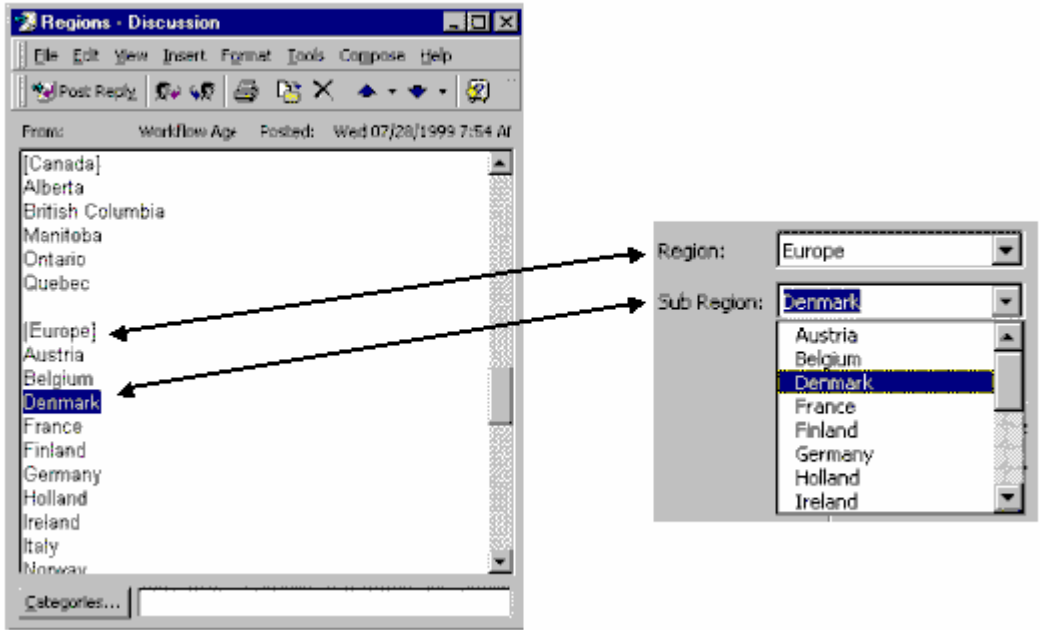
An alternative is to store the data within items in the Exchange store itself as individual message records. The drawback to this approach is that, even though messages in an Exchange Public Folder behave much like records in a database table, they have far too much overhead associated for simple data.

This is where the Extensions Library *for Outlook* comes into play. We provide a set of methods for reading and writing data to/from items in a designated folder, using the standard "Section-Key-Value" similar to those found in Windows INI files. This approach creates a data-access mechanism which is:

- simple to code and use
- operates off-line
- high speed (cached) operation, low overhead
- uses only the native Exchange data store
- provides an easy way to maintain/update the data

These Extensions Library tools provide a perfect way to manage lookup tables and configuration information, without having to either (a) use an external data source (which adds complexity to the application), or (b) hard-code the data into the Outlook form itself (which adds form size and also constrains your ability to quickly make data changes). This approach has proven highly effective for thousands of data points per message file.

An example shown below is a pair of combo boxes (on the right) allowing the selection of a Region and associated Sub-Region. Both controls are being populated from the message item shown at the left.



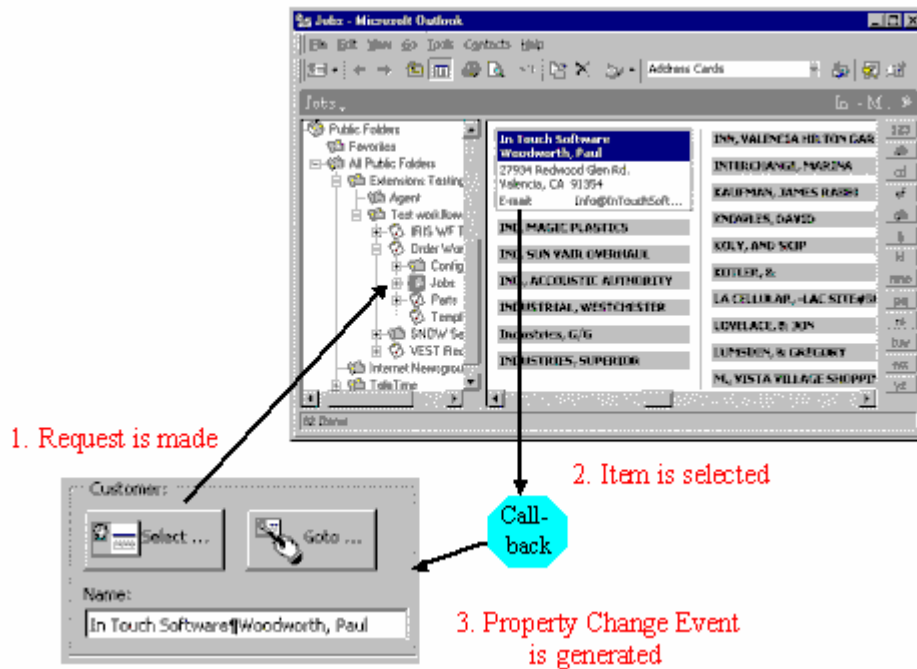
Additionally, in order to support the use of these INI-like data messages, we make available a publishing tool which automatically pulls data out of any standard database table and writes it to data items, properly formatted. This utility can be set to run at regular intervals, thereby insuring that your data items are always kept up-to-date.

Item Linking to Other Folders

In Outlook, as on the Web, linking from one piece of information to another is a powerful feature and key to delivering application that are integrated with Exchange. Currently, Outlook users wanting to insert a link reference to another Outlook item are limited to its 'Insert Item' interface.

The Extensions Library provides a set of methods which handles the linking of items from other folders. This capability allows your Outlook forms to display a list of items in any other folder, capture the user's selection(s) from that list, and return links to the selected item(s) on the form.

For example, you may have an Order form that requires the selection of a Customer, where your customers are kept in a Contacts folder. Your form calls the Extensions Library's *Get Item* method when the "Select..." button is clicked. This displays the folder you define as the "Customers" folder. When the user double-clicks on the desired Contact item, the window is automatically closed and the selected Contact (EntryID) is returned to the calling Order form.



ActiveX Grid Control

The Extensions Library *for Outlook* includes OIGrid, a 32-bit ActiveX custom control that provides a full-featured editable grid designed specifically for use on Microsoft Outlook forms.

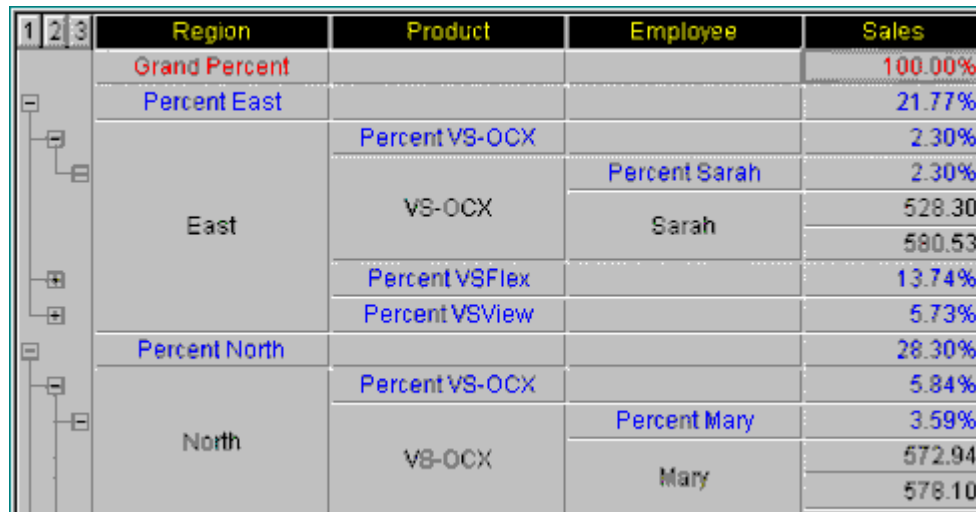
OIGrid is based on, and encapsulates, Videosoft's 32-bit VSFlex3.OCX grid control. OIGrid implements most of VSFlex's properties, methods, and events.

While theoretically a VSFlex grid can be placed directly on an Outlook form, one quickly runs into limitations: first, Outlook forms do not support any events other than the Click event for controls contained on them. In addition, no mechanism exists for binding Outlook data to the rows and columns of a grid, as is possible using standard data binding in Visual Basic. OIGrid provides a solution for both of these issues.

OIGrid also includes several methods - ArrayToGrid and GridToArray - which allow for easy loading and saving of grid data to and from an Outlook field or a user-constructed data array. This provides a convenient way of "binding" MAPI (Outlook field) data to the grid.

And because our Outlook Grid encapsulates the VSFlex Grid, you can take advantage of all of its features, such as:

- in-cell editing; in-cell combo box
- merge cells, subtotals, and outlines
- graphics support
- and much more...



| 1 | 2 | 3 | Region | Product | Employee | Sales | |
|---|---|---|---------------|----------------|----------------|---------|--------|
| | | | Grand Percent | | | 100.00% | |
| | | | Percent East | | | 21.77% | |
| | | | East | Percent VS-OCX | Percent Sarah | 2.30% | |
| | | | | VS-OCX | Sarah | 528.30 | |
| | | | | | | 580.53 | |
| | | | | | Percent VSFlex | | 13.74% |
| | | | | | Percent VSView | | 5.73% |
| | | | Percent North | | | 28.30% | |
| | | | North | Percent VS-OCX | Percent Mary | 3.59% | |
| | | | | VS-OCX | Mary | 572.94 | |
| | | | | | | 578.10 | |

ActiveX Calendar Control

The Extensions Library *for Outlook* includes *OICalndr*, a 32-bit ActiveX custom control that matches Outlook's built-in date field in both look and operation.

Standard Microsoft Outlook forms which include date-type fields (such as the Task and Appointment forms) include a drop-down monthly calendar as part of that date field. While various companies (including Microsoft itself) provide monthly calendar controls that can be used on an Outlook form, none of these provide the same design or functionality as Outlook's built-in control. *OICalndr* fills this gap.

Looks and feels like Outlook

OICalndr reproduces very closely the look of the built-in Outlook drop-down calendar. Your users will probably not notice the difference!



Outlook's built-in Calendar



OICalndr

Includes textbox

The control includes not only the drop-down monthly calendar, but also the textbox portion of the date field. So it acts like a text control, into which you can type a date; or you can click the down-arrow next to the textbox and expose the monthly calendar grid to select a date by clicking on it.

Includes AutoDate capability

OICalndr's textbox allows you to enter English-language date phrases, just like Outlook's built-in date control, and it will interpret them and convert them to a proper date format. For example, you can type "in 5 days", or "next Thur", or "3rd Sat in July".

Additional Features

In addition to the features detailed above, the Extensions Library *for Outlook* provides developers with the following capabilities:

Folder Navigation

Provides the ability to return a MAPIFolder object by traversing up and/or down relative to the current folder location, or absolutely from the information store down.

Text/Array Conversion

A set of methods allowing you to easily convert between delimited text strings and array variables, in both directions. Arrays can be multi-dimensional (i.e. include columns). These functions allow you to fully utilize the built-in MAPI keyword data array fields.

Generating Reference IDs

Creates a unique ID string that can be assigned to an item. You can define the elements and formatting that make up the ID.

Event Tracing

Provides Outlook form developers with an easy way to trace through the operation of a form and the events it generates. For many cases this is preferable to resorting to the overhead of the script debugger.

Error Handling

Using the visual error mode, library errors are displayed in dialog and do not pass the error back which would cause the script to fail and stop. This mode provides much needed feedback to help in developing forms and keeps non-critical errors from "crashing" your form.

Generating Outlook URLs

Returns an Outlook text URL, including full path information, to any folder or message. Useful for generating text URLs to place in messages (text bodies) to allow for easy one-click linking to other messages or folders.

Getting Property Values

Provides an easy way to retrieve the value from an Outlook field. In an Outlook form's Item_CustomPropertyChange or Item_PropertyChange events, the field name is returned but not the value. Getting the value from the name may seem fairly easy using the Item.UserProperties property, however, complexities arise when the value is an array or is empty, and therein lies the utility of this method. This routine also optionally provides automatic event and data tracing to a log file. (This tracing/logging capability is often a favorable alternative to using a simple MsgBox function or even the script debugger, because of the number of spurious events and the fact that the display of any dialog often disrupts the normal event pattern.)

Controlling Outlook Form Positioning

Under normal Outlook conditions, an Outlook form remembers the size of the previous form. This can be less than optimal for custom forms that are designed to a specific size. This method provides the ability to easily control the position and size of your Outlook form.

Providing missing Visual Basic functionality

Microsoft's VBScript is missing some of the functionality that is available in the full Visual Basic language implementation and via calls to the Windows API. Our Extensions Library provides some of the most useful of these missing routines, so you can take full advantage of them from your VBScript code. Supported functions include:

- Clipboard copy/paste
- VB's complete "Format" function
- The ShellExecute API -- lets you launch programs and open specific web pages!
- True Windows timer events
- VB's "Beep"

Foundation For TeamWork™ Workflow system

The TeamWork™ workflow system makes extensive use of the Extensions Library to provide all of the necessary supporting functions.

For More Information

To speak with a sales representative about The Extensions Library *for Outlook* and how it can benefit your organization, please contact us:

TeamScope Software
23679 Calabasas Road, #140
Calabasas, CA 91302
United States of America

Phone: (818) 876-0776

Fax: (818) 876-0779

E-mail: info@teamscope.com

Web: <http://www.teamscope.com>