Help for Regicon

Properties Methods

How To Buy This Software

Order Form

Getting Custom Controls Written

Licensing Information

Description

Take advantage of the Windows Registry with Mabry's RegiCon control. With the introduction of Windows 95, the use of the Windows Registry is now recommended over .INI files. RegiCon is a VB5 ActiveX control that allows your application to easily access the Windows Registry.

Creating or deleting registry keys and items is as easy as setting two or three properties and invoking a single method. You can also set, retrieve or change an item's value. When retrieving a value the control returns both the value and its data type. There is even a property that will tell you if a key or item exists.

The control can be used for any number of situations where you need to store or retrieve information for your applications. Use the control to store version information or configuration settings. Read the system configurations of the computer to customize or optimize your application for the computer. Control evaluation periods of shareware products. Maintain access control or encrypted security lists.

The control can be used in any 32 bit development environment and works with the Win95 or WinNT 4.0 registry. It supports the most common registry data types, including strings, multi-line strings, Dword (long integers), free form binary, expanded strings, and others.

General Registry Information

The Registry uses a heirarchical data structure that contains keys and items. It is configured very much like the directory and file structure of a hard disk. Think of keys as directories and items as files. Directories do not contain data, they only contain other directories (sub-directories) or the files which actually contain the data. Likewise, Registry keys can either contain other keys (sub-keys) or items. The actual data stored in the registry is not kept in keys, but in the items within the keys. When a key contains a sub-key, the combination is considered a multi-key.

Keys cannot begin or end with a backslash and must be 255 characters or less.

Important Note

The Mabry RegiCon OCX provides you with direct access to your system registry. This allows you to add, delete, and change the contents of the registry. Please be aware that altering or deleting registry settings can render your computer inoperable.

Also, when a key is deleted under Windows 95, any sub-keys or items within that key are also deleted. Windows NT prevents a key with sub-keys from being deleted. See the DeleteEntry method for additional details.

File Name

REGICON.OCX

ActiveX Compatibility

VB 4.0 (32-bit), 5.0 and 6.0

ActiveX Built With

Microsoft Visual Basic v5

ActiveX - Required DLLs

VB 5.0 Run-Time DLLs
MSVBVM50.DLL

Distribution Note When you develop and distribute an application that uses this control, you should install the control file into the user's Windows SYSTEM directory. The control file has version information built into it. So, during installation, you should ensure that you are not overwriting a newer version.

Close

Regicon Properties

Properties that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__IsEntry__ Property

*__ItemNames__ Property

*__ItemsCount__ Property

*__Key__ Property

*__KeyItemName__ Property

*__KeyItemType__ Property

*__KeyItemValue__ Property

*__RootKey__ Property

*__SubKey__ Property

*__SubKeyNames__ Property

*__SubKeysCount__ Property

*__Version__ Property

Close

Regicon Methods

Methods that have special meaning for this control or that only apply to this control are marked with an asterisk (*).

*__DeleteEntry__ Method
*__EnumerateEntry__ Method
*__GetEntry__ Method
*__SetEntry__ Method

How To Buy This Software

CREDITS

Regicon was written by Mark Lund.

CONTACT INFORMATION

Orders, inquiries, technical support, questions, comments, etc. can be sent to mabry@mabry.com on the Internet. Our mailing address/contact information is:

Mabry Software, Inc.
503 316th Street Northwest
Stanwood, WA 98292

Sales: 1-800-99-MABRY (U.S. Only)
Voice: 360-629-9278
Fax: 360-629-9278
Web: http://www.mabry.com

COST

The price of Regicon (control only) is US$25 (US$30 for International orders). The cost of Regicon and the Visual Basic source code (of the control itself) is US$75 (US$80 for International orders).

Prices are subject to change without notice.

Printed manuals are available at US$12.50 per copy.

PART NUMBERS

The product number for Regicon (control only) is 16303.

The product number for and the Visual Basic source code (of the control itself) is 16326.

DELIVERY METHODS

We can ship this software to you via air mail and/or e-mail.

Air Mail - you will receive diskettes, a printed manual (if purchased), and printed receipt if you choose this delivery method. The costs are:

US$10.00	US Priority Mail
US$15.00	Airborne Express 2nd Day (US deliveries only)
US$20.00	Airborne Express Overnight (US deliveries only)
US$20.00	Global Priority Mail (Int'l deliveries only; Western Europe, Pacific Rim and Canada only)
US$45.00	International Airborne Express (Int'l deliveries only)

E-Mail - We can ship this package to you via e-mail. You need to have an e-mail account that can accept large file attachments (which includes CompuServe, AOL, and most Internet providers). We will e-mail a receipt to you.

Be sure to include your full mailing address with your order. Sometimes (on the Internet) the package cannot be e-mailed, so we are forced to send it through the normal mails.

ORDER / PAYMENT METHODS

You can order this software by phone, fax, e-mail, mail. For your convenience, an order form has been provided that you can print out directly from this help file.

Please note that orders must include all information that is requested on our order form. Your shipment WILL BE DELAYED if we have to contact you for additional information (such as phone number, street address, etc.).

You can pay by credit card (VISA, MasterCard, American Express, Discover, NOVUS), check (U.S. dollars drawn on a U.S. bank), cash, International Money Order, International Postal Order, Purchase Order (established business entities only - terms net 30), or wire transfer.

WIRE TRANSFER INFORMATION

Here is the information you need regarding our account for a wire funds transfer:

Bank Name:	SeaFirst - Stone Way Branch
Bank Address:	3601 Stone Way North Seattle, WA 98103
Bank Phone:	206-585-4951
Account Name:	Mabry Software, Inc.
Routing Number:	12000024
Account Number:	16311706

If you are paying with a wire transfer of funds, please add US$25.00 to your order. This is the fee that SeaFirst Bank charges Mabry Software. Also, please ADD ANY ADDITIONAL FEES THAT YOUR BANK MAY CHARGE for wire transfer service. If you are paying with a wire transfer, we must have full payment deposited to our account before we can ship your order.

Regicon Order Form

Use the Print Topic... command from the File menu to print this order form.

Mail this form to:	Mabry Software, Inc. 503 316th Street Northwest Stanwood, WA 98292 Phone: 360-629-9278 Fax: 360-629-9278 Internet: mabry@mabry.com Web: www.mabry.com

Where did you get this copy of Regicon?

____

Name: ____

Ship to: ____

____

____

____

Phone: ____

Fax: ____

E-Mail: ____

Credit Card #: ______________________________ exp. ______________

P.O. # (if any): ______________ Signature ______________________

qty ordered	____	REGISTRATION $25.00 ($30.00 international). Check or money order in U.S. currency drawn on a U.S. bank. Add $10.00 per order for shipping and handling. Add $12.50 per printed manual.
qty ordered	____	SOURCE CODE AND REGISTRATION $75.00 ($80.00 international). Check or money order in U.S. currency drawn on a U.S. bank. Add $10.00 per order for shipping and handling. Add $12.50 per printed manual.

See Also

Key Property

KeyItemName Property

RootKey Property

SubKey Property

See Also

ItemNames Property

ItemsCount Property

Key Property

RootKey Property

SubKeyNames Property

SubKeysCount Property

See Also

Key Property

KeyItemName Property

KeyItemType Property

KeyItemValue Property

RootKey Property

SubKey Property

See Also

Key Property

KeyItemName Property

See Also

EnumerateEntry Method

ItemsCount Property

Key Property

RootKey Property

SubKeyNames Property

SubKeysCount Property

See Also

EnumerateEntry Method

ItemNames Property

Key Property

RootKey Property

SubKeyNames Property

SubKeysCount Property

See Also

DeleteEntry Method

EnumerateEntry Method

GetEntry Method

IsEntry Property

ItemNames Property

ItemsCount Property

KeyItemName Property

KeyItemType Property

KeyItemValue Property

RootKey Property

SetEntry Method

SubKey Property

SubKeyNames Property

SubKeysCount Property

See Also

DeleteEntry Method

GetEntry Method

IsEntry Property

Key Property

KeyItemType Property

KeyItemValue Property

SetEntry Method

See Also

GetEntry Method

Key Property

KeyItemName Property

KeyItemValue Property

RootKey Property

SetEntry Method

SubKey Property

See Also

GetEntry Method

Key Property

KeyItemName Property

KeyItemType Property

SetEntry Method

See Also

DeleteEntry Method

EnumerateEntry Method

GetEntry Method

ItemNames Property

ItemsCount Property

Key Property

KeyItemType Property

SetEntry Method

SubKey Property

SubKeyNames Property

SubKeysCount Property

See Also

Key Property

KeyItemName Property

KeyItemType Property

KeyItemValue Property

RootKey Property

SubKey Property

See Also

DeleteEntry Method

GetEntry Method

Key Property

KeyItemType Property

RootKey Property

SetEntry Method

See Also

EnumerateEntry Method

ItemNames Property

ItemsCount Property

Key Property

RootKey Property

SubKeysCount Property

See Also

EnumerateEntry Method
ItemNames Property
ItemsCount Property
Key Property
RootKey Property
SubKeyNames Property

DeleteEntry Method

See Also

Description

Deletes a key and all of its sub-keys or deletes an item from the registry.

Syntax

object.**DeleteEntry**

The syntax of the **DeleteEntry** method has these parts:

Part	Description
object	Required. A Regicon control.

Remarks

If KeyItemName is specified, the Item within the multi-key created by the RootKey and Key is deleted. When KeyItemName is specified, the SubKey property is not used by the DeleteEntry method.

When the KeyItemName is not specified, a key will be deleted. Since the nature of the registry tree results in most keys being multi-key, determining which part of a multi-key to delete can be ambiguous. The SubKey property is used to explicitly indicate which portion of the multi-key to delete.

To delete a key, set the Key property to the parent key of the key to delete and the SubKey property to the portion of the key which is to be deleted. If a SubKey is not specified, the Key is deleted from the RootKey (this should only occur when the RootKey is HKEY_CLASSES_ROOT).

When a key is deleted under Windows 95, any sub-keys or items within that key are also deleted. Windows NT prevents a key with sub-keys from being deleted.

Example #1:

```
'deletes the "My Configuration" key from
' the parent key "Software\My Company\My Application"
'
RegiCon1.RootKey = rkHKEY_LOCAL_MACHINE
RegiCon1.Key = "Software\My Company\My Application"
RegiCon1.SubKey = "My Configuration"
RegiCon1.DeleteEntry
```

Example #2:

```
'deletes the "My Data" item from the
'key "Software\My Company\My Application\My Configuration"
'
'SubKey is ignored when KeyItemName is specified
RegiCon1.RootKey = rkHKEY_LOCAL_MACHINE
RegiCon1.Key = "Software\My Company\My Application\My Configuration"
RegiCon1.KeyItemName = "My Data"
RegiCon1.DeleteEntry
```

Example #3:

```
    'under Win95, this will delete "My Application\My Configuration"
    'WinNT will not allow the deletion because "My Configuration" is a sub-
key of "My Application"
    RegiCon1.RootKey = rkHKEY_LOCAL_MACHINE
    RegiCon1.Key = "Software\My Company"
    RegiCon1.SubKey = "My Application\My Configuration"
    RegiCon1.DeleteEntry
```

EnumerateEntry Method

See Also

Description

Retrieves the names of all keys and items immediately under the key specified by the RootKey and Key properties.

Syntax

object.**EnumerateEntry**

The syntax of the **EnumerateEntry** method has these parts:

Part	**Description**
object	Required. A Regicon control.

Remarks

When the EnumerateEntry method is called, the names of all subkeys immediately under the multi-key created by the RootKey and Key properties are loaded into the SubKeyNames array. Also, the names of all item immediately under the multi-key are loaded into the ItemNames array.

The SubKeysCount property returns the number of items in the SubKeyNames array. The ItemsCount property returns the number of items in the ItemsNames array.

Example #1:

```
'
    'loads List1 with all subkeys under the parent
    'key "Software\My Company\My Application"
    '
    'loads List2 with all items under the parent
    'key "Software\My Company\My Application"
    RegiCon1.RootKey = rkHKEY_LOCAL_MACHINE
    RegiCon1.Key = "Software\My Company\My Application"
    RegiCon1.EnumerateEntry
    Dim l As Long
    List1.Clear
    List2.Clear
    l = 0
    For l = 0 To RegiCon1.SubKeysCount - 1
        List1.AddItem RegiCon1.SubKeyNames(l)
    Next l
    l = 0
    For l = 0 To RegiCon1.ItemsCount - 1
        List2.AddItem RegiCon1.ItemNames(l)
    Next l
```

GetEntry Method

See Also

Description

Returns the value and type for an item in the registry.

Syntax

object.**GetEntry**

The syntax of the **GetEntry** method has these parts:

Part	**Description**
object	Required. A Regicon control.

Remarks

After the GetEntry method has been invoked, the KeyItemValue and KeyItemType properties reflect the data and data type for the item specified. If the KeyItemName does not exist in the registry (as specified by the RootKey and Key properties), an ITEM_NOT_FOUND error is thrown, the KeyItemValue is set to "" and the KeyItemType is set to dtREG_NONE. The SubKey property is not used by the GetEntry method.

IsEntry Property

See Also

Description

Indicates whether a Key or Item exists.

Syntax

object.**IsEntry**

The syntax of the **IsEntry** property has these parts:

Part	Description
object	A Regicon control.

Remarks

If a KeyItemName is specified, the IsEntry returns True if the Item exists within the Key and False if the Item does not exist within the key. If the KeyItemName is not specified (empty), the IsEntry returns True if the key exists and False if the key does not exist.

This property is read-only.

Data Type

Boolean

ItemNames Property

See Also

Description

String array that returns the name of the item immediately under the key specified by RootKey and Key.

Syntax

object.**ItemNames(** *Index* **)** [= *ItemNames*]

The syntax of the **ItemNames** property has these parts:

Part	**Description**
object	A Regicon control.
Index	A long integer specifying the index for the array.
ItemNames	A string expression specifying the item's name.

Remarks

When the EnumerateEntry method is called, the names of all items immediately under the multi-key created by the RootKey and Key properties are loaded into the ItemNames array.

The ItemsCount property returns the number of items in the ItemNames array. The index can be set to a value between 0 and ItemsCount - 1.

Data Type

String

ItemsCount Property

See Also

Description

Returns the name of the number of elements in the ItemNames array property.

Syntax

object.**ItemsCount** [= *ItemsCount*]

The syntax of the **ItemsCount** property has these parts:

Part	**Description**
object	A Regicon control.
ItemsCount	A long integer specifying the number of elements in the ItemNames array.

Remarks

When the EnumerateEntry method is called, the names of all items immediately under the multi-key created by the RootKey and Key properties are loaded into the ItemNames array. The ItemsCount property returns the number of items contained in the ItemNames array.

Data Type

Long

Key Property

See Also

Description

Specifies a key's value.

Syntax

object.**Key** [= *key*]

The syntax of the **Key** property has these parts:

Part	**Description**
object	A Regicon control.
key	A string expression that specifies a key or multi-key.

Remarks

Keys specify the location of registry items, much like directories specify the location of files. A key can contain other keys (called sub-keys). When a key has sub-keys, it is called a multi-key. The entire length of the multi-key must be 255 characters or less and the Key property cannot begin or end with a "\" character.

All methods require a RootKey and a Key. Since the nature of the registry tree results in most keys being multi-key, determining a key from a sub-key can be ambiguous. When using the DeleteEntry method, set the Key property to the parent directory and use the SubKey property to explicitly indicate which portion of the multi-key to delete.

Data Type

String

KeyItemName Property

See Also

Description

Specifies the name of the Item within a key.

Syntax

object.**KeyItemName** [= *name*]

The syntax of the **KeyItemName** property has these parts:

Part	**Description**
object	A Regicon control.
name	A string expression that specifies an item name.

Remarks

Items are to keys, much like files are to directories. The item name specifies the actual entry that contains the registry data. A KeyItemName must be 255 characters or less. To retrieve the default item of a key, set the KeyItemName to an empty string ("").

Data Type

String

KeyItemType Property

See Also

Description

Sets or returns the type of data contained in KeyItemValue.

Syntax

object.**KeyItemType** [= *type*]

The syntax of the **KeyItemType** property has these parts:

Part	Description
object	A Regicon control.
type	An integer that specifies the data type to use.

Remarks

The registry supports several different data types. The most common are strings, long integers or free form binary data. For efficient registry performance, data which exceeds 2048 bytes should be stored in files outside the registry.   Use the registry to store the location of the files.

Constant	Value	Description
dtREG_SZ	0	String (cannot contain a Null CHR$(0))
dtREG_DWORD	1	Long Integer
dtREG_BINARY	2	Free-form binary data.
dtREG_MULTI_SZ	3	Multi-line string (cannot contain Nulls).
dtREG_EXPAND_SZ	4	String containing unexpanded environment variables, (for example, "%PATH%").
dtREG_NONE	5	No defined type.
dtREG_LINK	6	Unicode symbolic link.
dtREG_DWORD_LITTLE_ENDIAN	7	32-bit number. Same as REG_DWORD.
dtREG_DWORD_BIG_ENDIAN	8	32-bit number with the most significant byte of a word as the low-order word.
dtREG_RESOURCE_LIST	9	A device driver resource list.

Data Type

Integer

KeyItemValue Property

See Also

Description

Sets or returns the actual data value.

Syntax

object.**KeyItemValue** [= *data*]

The syntax of the **KeyItemValue** property has these parts:

Part	Description
object	A Regicon control.
data	A variant that holds the data to store or the data retrieved.

Remarks

The registry supports several different data types. The most common are strings, long integers or free form binary data. For efficient registry performance, data which exceeds 2048 bytes should be stored in files outside the registry.   Use the registry to store the location of the files.

When REG_EXPAND_SZ is specified, the registry does not expect the environment variables to be expanded nor will it expand the variables when returning data containing this data type.

When REG_DWORD_BIG_ENDIAN, the RegiCon control assumes the data is entered with the most significant byte of a word as the low order word.

Data Type

Variant

RootKey Property

See Also

Description

Determines the root level registry key to access.

Syntax

object.**RootKey** [= *rootkey*]

The syntax of the **RootKey** property has these parts:

Part	Description
object	A Regicon control.
rootkey	An integer that specifies the root key to use.

Remarks

The registry has six predefined keys called root keys. These keys are always open and cannot be deleted. Root keys can only contain other keys. The most common root keys are HKEY_LOCAL_MACHINE (used to hold information about the machine) and HKEY_USERS or HKEY_CURRENT_USER (used to hold user information).

Constant	Value	Constant
rkHKEY_LOCAL_MACHINE	0	Default. Used to store machine information.
rkHKEY_CURRENT_USER	1	Used to store user information.
rkHKEY_CURRENT_CONFIG	2	Used to store machine information.
rkHKEY_USERS	3	Used to store user information.
rkHKEY_CLASSES_ROOT	4	Used to store machine information.
rkHKEY_DYN_DATA	5	Used to store machine information.

Data Type

Integer

SetEntry Method

See Also

Description

Creates a registry key, or creates or sets an Item in the registry.

Syntax

object.**SetEntry**

The syntax of the **SetEntry** method has these parts:

Part	Description
object	Required. A Regicon control.

Remarks

If KeyItemName is specified, the data in KeyItemValue is stored in the registry under the KeyItemName. The location of the item is determined by the multi-key that RootKey and Key specify. If the data in the KeyItemValue property doesn't match the type specified by the KeyItemType property (for example, if the KeyItemType is set to REG_DWORD and the KeyItemValue is "foo"), an error occurs.

If the KeyItemName does not exist, the entry is created. If the KeyItemName previously existed, the entry is updated with the item value and type.

If the item specified by KeyItemName is not specified (empty), SetEntry will create a registry key according to the RootKey and Key settings. The SubKey property is not used by the SetEntry method.

Example #1:

```
    'to create an empty key
    RegiCon1.RootKey = rkHKEY_LOCAL_MACHINE
    RegiCon1.Key = "Software\My Company\My Application"
    RegiCon1.SetEntry
```

Example #2:

```
'to create a registry item
    RegiCon1.RootKey = rkHKEY_LOCAL_MACHINE
    RegiCon1.Key = "Software\My Company\My Application"
    RegiCon1.KeyItemName = "My Item"
    RegiCon1.KeyItemType= dtREG_SZ
    RegiCon1.KeyItemValue = "This is my data!"
    RegiCon1.SetEntry
```

SubKey Property

See Also

Description

Specifies the portion of a multi-key to delete.   Only used by the DeleteEntry method.

Syntax

object.**SubKey** [= *subkey*]

The syntax of the **SubKey** property has these parts:

Part	**Description**
object	A Regicon control.
subkey	A string expression that specifies a subkey or multi-key.

Remarks

Keys specify the location of registry Items, much like directories specify the location of files. A Key can contain other keys (called sub-keys). When a key has sub-keys, it is called a multi-key. The entire length of the multi-key must be 255 characters or less and neither keys nor sub-keys can begin or end with a "\" character.

All methods require a RootKey and a Key. Since the nature of the registry tree results in most keys being multi-key, determining a sub-key from a multi-key can be ambiguous. The DeleteEntry method may require an explicit sub-key value. The SubKey property is used to explicitly indicate which portion is the sub-key of a multi-key.

Data Type

String

SubKeyNames Property

See Also

Description

String array that returns the name of the sub-keys immediately under the key specified by RootKey and Key.

Syntax

object.**SubKeyNames(** *Index* **)** [= *SubKeyNames*]

The syntax of the **SubKeyNames** property has these parts:

Part	**Description**
object	A Regicon control.
Index	A long integer specifying the index for the array.
SubKeyNames	A string expression specifying the subkey's name.

Remarks

When the EnumerateEntry method is called, the names of all sub-keys immediately under the multi-key created by the RootKey and Key properties are loaded into the SubKeyNames array.

The SubKeysCount property returns the number of items in the SubKeyNames array. The index can be set to a value between 0 and SubKeysCount - 1.

Data Type

String

SubKeysCount Property

See Also

Description

Returns the number of elements in the SubKeyNames array property.

Syntax

object.**SubKeysCount** [= *SubKeysCount*]

The syntax of the **SubKeysCount** property has these parts:

Part	**Description**
object	A Regicon control.
SubKeysCount	A long integer specifying the number of elements in the SubKeyNames array.

Remarks

When the EnumerateEntry method is called, the names of all sub-keys immediately under the multi-key created by the RootKey and Key properties are loaded into the SubKeyNames array. The SubKeysCount property returns the number of items contained in the SubKeyNames array.

Data Type

Long

Version Property

Description

Returns the version of the control.

Syntax

object.**Version**

The syntax of the **Version** property has these parts:

Part	Description
object	A Regicon control.

Remarks

This property holds the current version of the control.   It is read-only and available at both design-time and run-time.

Data Type

String

Getting Custom Controls Written

If you or your organization would like to have custom controls written, you can contact us at the following:

Mabry Software, Inc.
503 316th Street Northwest
Stanwood, WA   98292

Phone: 360-629-9278
Fax: 360-629-9278

Internet: mabry@mabry.com

You can also contact Zane Thomas.   He can be reached at:

Zane Thomas
Post Office Box 121
Indianola, WA   98342

Internet: zane@mabry.com

Licensing Information

Legalese Version

Mabry Software grants a license to use the enclosed software to the original purchaser. Copies may be made for back-up purposes only. Copies made for any other purpose are expressly prohibited, and adherence to this requirement is the sole responsibility of the purchaser.

Customer written executable applications containing embedded Mabry products may be freely distributed, without royalty payments to Mabry Software, provided that such distributed Mabry product is bound into these applications in such a way so as to prohibit separate use in design mode, and that such Mabry product is distributed only in conjunction with the customers own software product. The Mabry Software product may not be distributed by itself in any form.

Neither source code for Mabry Software products nor modified source code for Mabry Software products may be distributed under any circumstances, nor may you distribute .OBJ, .LIB, etc. files that contain our routines. This control may be used as a constituent control only if the compound control thus created is distributed with and as an integral part of an application. Permission to use this control as a constituent control does not grant a right to distribute the license (LIC) file or any other file other than the control executable itself. This license may be transferred to a third party only if all existing copies of the software and its documentation are also transferred.

This product is licensed for use by only one developer at a time. Mabry Software expressly prohibits installing this product on more than one computer if there is any chance that both copies will be used simultaneously. This restriction also extends to installation on a network server, if more than one workstation will be accessing the product. All developers working on a project which includes a Mabry Software product, even though not working directly with the Mabry product, are required to purchase a license for that Mabry product.

This software is provided as is. Mabry Software makes no warranty, expressed or implied, with regard to the software. All implied warranties, including the warranties of merchantability and fitness for a particular use, are hereby excluded.

MABRY SOFTWARE'S LIABILITY IS LIMITED TO THE PURCHASE PRICE. Under no circumstances shall Mabry Software or the authors of this product be liable for any incidental or consequential damages, nor for any damages in excess of the original purchase price.

To be eligible for free technical support by telephone, the Internet, CompuServe, etc. and to ensure that you are notified of any future updates, please complete the enclosed registration card and return it to Mabry Software.

English Version

We require that you purchase one copy of a control per developer on a project. If this is met, you may distribute the control with your application royalty free. You may never distribute the LIC file. You may not change the product in any way that removes or changes the requirement of a license file.

We encourage the use of our controls as constituent controls when the compound controls you create are an integral part of your application. But we don't allow distribution of our controls as constituents of other controls when the compound control is not part of an application. The reason we need to have this restriction is that without it someone might decide to use our control as a constituent, add some trivial (or even non-trivial) enhancements and then sell the compound control. Obviously there would be little difference between that and just plain reselling our control.

If you have purchased the source code, you may not re-distribute the source code either (nor may you copy it into your own project). Mabry Software retains the copyright to the source code.

Your license is transferable. The original purchaser of the product must make the transfer request. Contact us for further information.

The sample versions of our products are intended for evaluation purposes only. You may not use the sample version to develop completed applications.