Advanced “Disk” Help Table of Contents

How To ...

Advanced **Disk** is a 32-bit OCX Custom Control providing applications with Disk information for Visual Basic 4.0 (32-bit applications) and 32-bit C++ applications.

This control has no methods with this control, but it has over twenty five (25) properties, two of which are changable. The properties provide a way of determining operating system, file system, disk system, drive system and some windows information. One of the two changable properties, allows you to read and change the Disk Volume Label.

Method(s)

None.

Properties

Quick List
File System
Disk
Windows
Operating System
Drive

Control Registration

Registration

Version

ADVDISK.OCX Version 4.0

Includes

ADVDISK.EXE	Control Support Application
ADVDISK.HLP	Help File

Other Controls by Advanced Applications

DAYTIP.OCX, a 32-bit Tip of the Day Control, see CompuServe WINSHARE, keyword is “DAYTIP”

Information in this document is subject to change without notice. Microsoft, MS-DOS, Visual Basic, Windows, Windows 95, and Windows NT are registered trademarks of Microsoft Corporation.

License

See the License Agreement License, see Registration form, print the form and send it in., see Ordering. which covers Payment options and other information. You can also register this control on CompuServe: GO SWREG, Registration ID 4061.

Note: Without the ADVDISK.LIC file your copy of ADVDISK.OCX will display a **Registered Property** equal to **False.**

Contacts

Advanced Applications

6700 North Tryon Street
Box 560991.
Charlotte, NC 28256-0991
USA

Compuserve	72713,2106
Internet	72713,2106@compuserve.com
Voice	(704) 597-3948

Copyright notice

Copyright notice.

AdvDisk Quick List Properties

This following list is a quick list table which discuss property capabilites.

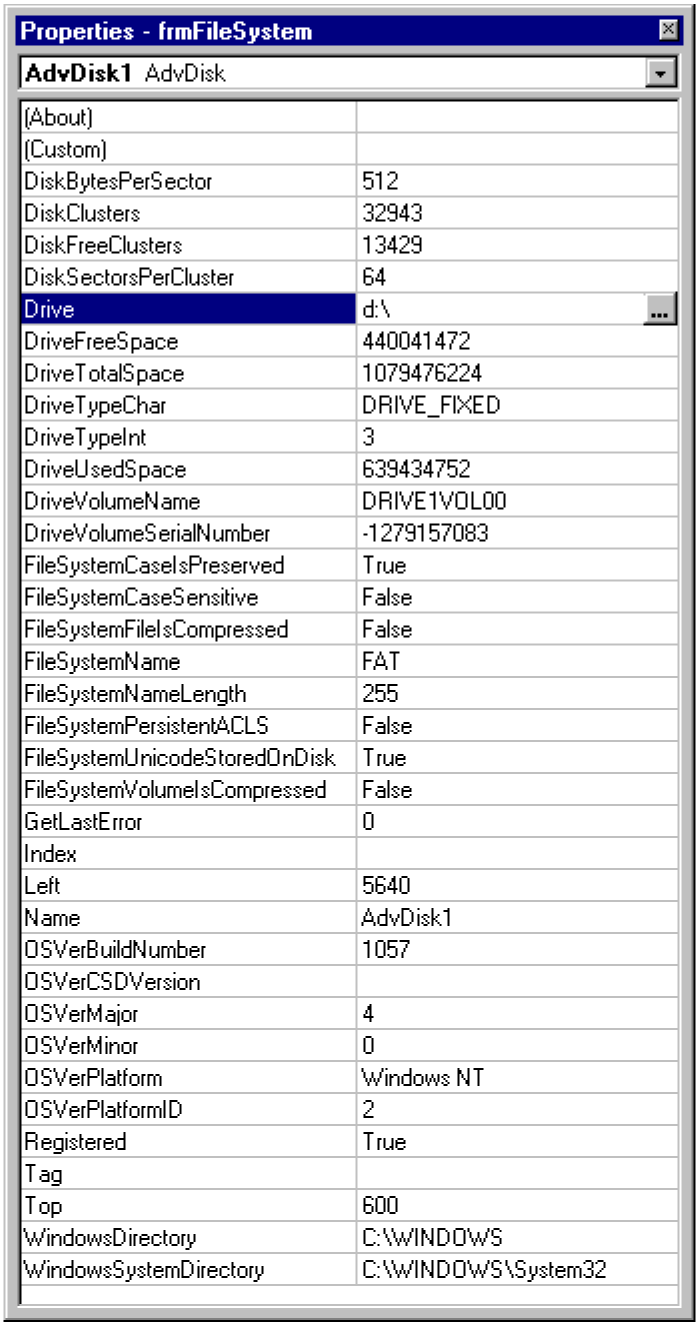
Property Name	Data Type	Setable	Description
Drive	String	**yes**	An Input Property which dictates the values of all the properties below.
DriveFreeSpace	double	no	Retrieves the amount of free space for the specified **Drive**.
DriveTotalSpace	double	no	Retrieves the amount of free space for the specified **Drive**.
DriveTypeChar	String	no	Determines whether the disk **Drive** is a removable, fixed, CD-ROM, RAM disk, or network drive returning a String value: `DRIVE_UNKNOWN` The drive type cannot be determined. `DRIVE_NOT_EXIST` The root directory does not exist. `DRIVE_REMOVABLE` The drive can be removed from the drive. `DRIVE_FIXED` The disk cannot be removed from the drive. `DRIVE_REMOTE` The drive is a remote (network) drive. `DRIVE_CDROM` The drive is a CD-ROM drive. `DRIVE_RAMDISK` The drive is a RAM disk.

DriveTypeInt	long	no	Determines whether the disk **Drive** is a removable, fixed, CD-ROM, RAM disk, or network drive returning a long value:
			`0 = DRIVE_UNKNOWN`
			`1 = DRIVE_NOT_EXIST`
			`2 = DRIVE_REMOVABLE`
			`3 = DRIVE_FIXED`
			`4 = DRIVE_REMOTE`
			`5 = DRIVE_CDROM`
			`6 = DRIVE_RAMDISK`

DriveUsedSpace	double	no	Retrieves the amount of used space for the specified **Drive**.
DriveVolumeName	String	**yes**	Retrieves the the name of the disk volume for the specified **Drive**. This property can be changed, which will change the disk volume for the specified **Drive**.
DriveVolumeSerialNumber	long	no	Retrieves the volume serial number for the specified **Drive**.
FileSystemCaseIsPreserved	BOOL	no	If this flag is set, the file system preserves the case of filenames when it places a name on disk, for the specified **Drive**.
FileSystemCaseSensitive	BOOL	no	If this flag is set, the file system supports case-sensitive filenames for the specified **Drive**.
FileSystemName	String	no	Retrieves the name of the file system (such as FAT or NTFS), for the specified **Drive**.
FileSystemNameLength	long	no	Retrieves the length, in characters, of the file system name for the specified **Drive**.
FileSystemPersistentACLS	BOOL	no	If this flag is set, the file system preserves and enforces ACLs. For example, NTFS preserves and enforces ACLs and FAT does not, for the specified **Drive**.
FileSystemUnicodeStoredOnDisk	BOOL	no	If this flag is set, the file system supports Unicode in

			filenames as they appear on disk, for the specifed **Drive**.
FileSystemVolumeIsCompressed	BOOL	no	The specified volume is a compressed volume; for example, a DoubleSpace volume, for the specified **Drive**.
FileSystemFileIsCompressed	BOOL	no	The file system supports file-based compression for the specified **Drive**.
GetLastError	long	no	Retrieves any Error Codes when determining Disk Information.
Registered	BOOL	no	Retrieves whether this control is registered or not (True/False).
DiskSectorsPerCluster	double	no	Retrieves the Disk sectors per cluster for the specified **Drive**.
DiskBytesPerSector	double	no	Retrieves the Disk bytes per sector for the specified **Drive**.
DiskFreeClusters	double	no	Retrieves the Disk free clusters for the specified **Drive**.
DiskClusters	double	no	Retrieves the Disk total clusters for the specified **Drive**.
OSVerMajor	long	no	Retrieves the Operating System's Major Verions Number.
OSVerMinor	long	no	Retrieves the Operating System's Minor Verions Number.
OSVerBuildNumber	long	no	Retrieves the Operating System's Build Number.
OSVerPlatformID	long	no	Retrieves the Operating System's Platform ID Number: 0 = Windows WIN32S 1 = Windows 95 2 = Windows NT
OSVerPlatform	String	no	Retrieves the Operating System's Platform ID as a String, the Platform strings are: Windows WIN32S Windows 95 Windows NT
OSVerCSDVersion	String	no	Retrieves a string that provides additional information about the Operating System, if any.
WindowsDirectory	String	no	Retrieves the path of the Windows directory. The Windows directory contains such files as Windows-based applications, initialization files, and Help files.
WindowsSystemDirectory	String	no	Retrieves the path of the Windows system directory. The system directory contains such files as Windows libraries, drivers, and font files.

The following Visual Basic Property Window displays AdvDisk Properties:



Properties - frmFileSystem

AdvDisk1 AdvDisk

(About)	
(Custom)	
DiskBytesPerSector	512
DiskClusters	32943
DiskFreeClusters	13429
DiskSectorsPerCluster	64
Drive	d:\
DriveFreeSpace	440041472
DriveTotalSpace	1079476224
DriveTypeChar	DRIVE_FIXED
DriveTypeInt	3
DriveUsedSpace	639434752
DriveVolumeName	DRIVE1VOL00
DriveVolumeSerialNumber	-1279157083
FileSystemCaseIsPreserved	True
FileSystemCaseSensitive	False
FileSystemFileIsCompressed	False
FileSystemName	FAT
FileSystemNameLength	255
FileSystemPersistentACLS	False
FileSystemUnicodeStoredOnDisk	True
FileSystemVolumeIsCompressed	False
GetLastError	0
Index	
Left	5640
Name	AdvDisk1
OSVerBuildNumber	1057
OSVerCSDVersion	
OSVerMajor	4
OSVerMinor	0
OSVerPlatform	Windows NT
OSVerPlatformID	2
Registered	True
Tag	
Top	600
WindowsDirectory	C:\WINDOWS
WindowsSystemDirectory	C:\WINDOWS\System32

AdvDisk File System Details

When developing an application for Windows NT, you should always keep in mind that the user might be using any combination of four (4) present file system (with new file systems in the future), FAT, HPFS, NTFS, and CDFS. AdvDisk provides eight (8) properties which return specific information about a users file system:

File System Properties	**Definitions**
FileSystemCaseIsPreserved	If True, the case of a filename is preserved when the name is put on disk.
FileSystemCaseSensitive	If True, the file system supports case-sensitive filename lookup.
FileSystemName	A String, which names the file system, (FAT, HPFS, NTFS, CDFS).
FileSystemNameLength	A long, which provides the maximum number of characters supported for directory names and filenames. For a FAT file system, the value is 12 and for both HPFS and NTFS, the value is 255. With this property, you no longer have to hardcode lengths in your source code for pathnames and filenames. For many applications everything may seem OK for applications to manipulate files and paths on a FAT system, but which cause stack overwrites, invalid memory accesses and other problems on NTFS system.
FileSystemPersistentACLS	If True, ACLS (Access Control List), In the Windows NT security model, all objects can be secured. These attributes are described by a security descriptor and an ACLS. The ACLS identifies the users and groups allowed or denied access to the object containing an entry for each user, global and local groups which are denied access to the object.
FileSystemUnicodeStoredOnDisk	If True, the file system supports Unicode in filenames as they appear on disk. If you are using Unicode in your application, your buffer needs to be twice as big. Windows NT knows if your application is Unicode. When you request paths and filenames, the system will perform all conversions, but you must ensure that your buffers will be large enough to hold the conversion results.
FileSystemVolumeIsCompressed	If True, the specified volume is a compressed volume; for example, a DoubleSpace volume.
FileSystemFileIsCompressed	If True, the file system supports file-based compression.

By setting the Drive Property, the File System Properties are automatically updated, no other user interaction are required.

AdvDisk Disk Details

These Disk Properties return disk statistics. You can either obtain this information by using the Drive Properties or you can perform your own calculations.

Disk Properties	**Definitions**
DiskSectorsPerCluster	Disk sectors per cluster.
DiskBytesPerSector	Disk bytes per sector, 512 bytes for a FAT system.
DiskFreeClusters	Disk free clusters.
DiskClusters	Disk total clusters.

To calculate Total Disk Space perform the following:

Total Disk Space = **DiskSectorsPerCluster * DiskBytesPerSector * DiskClusters**

To calculate Free Disk Space perform the following:

Total Free Space = **DiskSectorsPerCluster * DiskBytesPerSector * DiskFreeClusters**

To calculate Used Disk Space perform the following:

Used Disk Space = **DiskSectorsPerCluster * DiskBytesPerSector * (DiskClusters - DiskFreeClusters)**

All the bytes available on floppy disks and hard drives are packaged together into sectors, usually 512 bytes per sector. Sectors are then grouped together to form clusters. In the FAT file system the number of sectors per cluster can vary dramatically.   One disadvantage of clusters is that (when there are more than one sector per cluster), they increase the amount of disk space occupied by tiny files. If there are 512 bytes per sector and 8 sectors per cluster, then the minimum space allocated to a file is 4 kb, even for a file whose size is listed as one byte.

Sectors Per Cluster	**Drive Type**
1	1.44 mb floppy disk or it could be a RAM disk
2	360 kb floppy disk
4	1.2 mb floppy disk
8	200 mg hard disk
32	400 mb hard disk

AdvDisk Windows Details

These properties return Windows Directory Information without have to make API calls.

Windows Properties	**Definitions**
WindowsDirectory	Retrieves the path of the Windows directory. As an example, **c:\windows**.
WindowsSystemDirectory	Retrieves the path of the Windows system directory. As an example, **c:\windows\system32**.

AdvDisk Operating System Details

The Operating System Version Properties make it easy to find out which operating system your users are using. This is useful for source code Operating System dependency.

Operating System Properties	**Definitions**
OSVerMajor	The Operating System’s Major Verions Number.
OSVerMinor	The Operating System’s Minor Verions Number.
OSVerBuildNumber	The Operating System’s Build Number.
OSVerPlatformID*	The Operating System’s Platform ID Number: 0 = Windows WIN32S 1 = Windows 95 2 = Windows NT
OSVerPlatform*	The Operating System’s Platform ID as a String, the Platform strings are: Windows WIN32S Windows 95 Windows NT
OSVerCSDVersion	A string that provides additional information about the Operating System, Do not be suprised if this string is blank.

***Note**: You can use either this platform information as a number or a string.

AdvDisk Drive Details

The following two properties are the only properties which are changable. The **Drive** Property is that item which dictates all other AdvDisk Properties. And the **DriveVolumeName** Property displays the Volume Name for that drive. If you change the property, your Drive Volume Name is automatically updated.

Disk Properties **Definitions**

Drive An Input Property which dictates the values of all the properties below.

DriveVolumeName An Input Property which retrieves the the name of the disk volume. This property can be changed, which will change the disk volume for the specified **Drive** above.

Visual Basic Drive Example:

```
        Dim strDrive As String

        strDrive = “c”
        AdvDisk1.Drive = strDrive
or
        strDrive = AdvDisk1.Drive
        MsgBox strDrive
```

Visual Basic Volume Name Example:

```
        Dim strVolume As String

        strVolume = “Drive One”
        AdvDisk1.DriveVolumeName = strVolume
or
        strVolume = AdvDisk1.DriveVolumeName
        MsgBox strVolume
```

The following Properties are read-only properties.

Drive Properties **Definitions**

DriveFreeSpace Automatically update drive statics.

DriveTotalSpace Automatically update drive statics.

DriveTypeChar Automatically updated and provides informatino as to whether the drive is a removable, fixed, CD-ROM, RAM disk, or network drive returning a String value:

```
DRIVE_UNKNOWN   The drive type cannot be determined.
DRIVE_NOT_EXIST The root directory does not exist.
DRIVE_REMOVABLE The drive can be removed from the
                drive.
DRIVE_FIXED     The disk cannot be removed from the
                drive.
DRIVE_REMOTE    The drive is a remote (network)
                drive.
DRIVE_CDROM     The drive is a CD-ROM drive.
DRIVE_RAMDISK   The drive is a RAM disk.
```

This provides string infomation, you may want to use this to display to your user as a string.

DriveTypeInt Automatically updated and provides informatino as to whether the drive is a removable, fixed, CD-ROM, RAM disk, or network drive returning a long value:

```
0 = DRIVE_UNKNOWN
1 = DRIVE_NOT_EXIST
2 = DRIVE_REMOVABLE
3 = DRIVE_FIXED
4 = DRIVE_REMOTE
5 = DRIVE_CDROM
6 = DRIVE_RAMDISK
```

This is useful for your source code to check an long value.

DriveUsedSpace — Automatically update drive statics.

DriveVolumeSerialNumber — The Volume Serial Number is most useful when another disk has been inserted into the drive. Starting with MS-DOS 4.0, the FORMAT command puts unique serial number information on the disk. In this way, even if two disks have the same volume label, each will have its own unique serial number. Your application can check to see if the user has changed the diskettes within the drive.

Control Registration

AdvDisk.ocx is a self-registration file, but before a control can be used, appropriate entries must be created for it in the Windows registration database. REGISTER.EXE provides a dialog item for users to register new controls (the files will default to ADVDISK.OCX, but can be used for other controls), but this feature may not be useful for your application. Therefore, you may want your setup program to register the controls when they are installed. Microsoft provided a redistributable program, REGSVR32.EXE, which can be used to register controls. Just pass the complete path and filename of the control .OCX file as an argument to REGSVR32:

```
regsvr32 c:\AdvDisk\AdvDisk.ocx
```

If you prefer, you can write your setup program to register the control directly instead.

Use the **LoadLibrary** Windows API to load the control DLL. Next, use **GetProcAddress** to obtain the address of the “DllRegisterServer” function. Finally, call the DllRegisterServer function. The following code sample demonstrates one possible method, where stores the handle of the control library, and lpDllEntryPoint stores the address of the “DllRegisterServer” function. The following is a “C” code process (see “C” Code Example below):

```
HINSTANCE hLib = LoadLibrary(pszDllName);

if (hLib < (HINSTANCE)HINSTANCE_ERROR)
{
        DisplayMessage(IDS_LOADLIBFAILED, pszDllName);        //unable to load DLL
         iReturn = FAIL_LOAD;                                 //unable to load DLL
}

// Find the entry point.
(FARPROC&)lpDllEntryPoint = GetProcAddress(hLib, “DllRegisterServer”);
if (lpDllEntryPoint != NULL)
         (*lpDllEntryPoint)();
else
        //unable to locate entry point
```

The advantage of registering the control directly is that you don’t need to invoke and load a separate process (namely, REGSVR32), lessening installation time. In addition, because registration is an internal process, the setup program can handle errors and unforeseen situations better than an external process can.

Note: Before your setup program installs an OLE control, it should call **OleInitialize**. When your setup program is finished, call **OleUnitialize**. This ensures that the OLE system DLLs are in the proper state for registering an OLE control.

When you install and register a control, you should also register OLEPRO32.DLL. Use the same procedure for registering this DLL as you did for the ADVDISK .OCX file. Perform this registration step only if you need to install OLEPRO32.DLL. If the DLL is installed already, you should assume that it has been registered.

You should also register MFC40.DLL. Unlike OLEPRO32.DLL, you should always register this DLL, even if it is already installed.

List of Redistributable Files

This section lists the files that must accompany the AdvDisk OLE control. When you redistribute any of these files, you should copy them from the setup distribution medium. This ensures that you are redistributing the correct version of the files.

Table 1 Redistributable Files

Files	**Description**
MFC40.DLL	MFC DLL (ANSI)
OLEPRO32.DLL	OLE property frame and standard types support
REGSVR32.EXE	Control registration utility
ADVDISK.OCX	Control File

Note: ANSI controls, which work on all Win32 operating systems, allow for maximum portability between the various Win32 operating systems. Unicode controls work on only Windows NT (version 3.51 or later), but not on Windows 95. AdvDisk.ocx is an ANSI Control.

Registration Code Example:

```
static char _szDllRegSvr[]      = "DllRegisterServer";
static char _szDllUnregSvr[] = "DllUnregisterServer";

pszDllEntryPoint = _szDllRegSvr;
TCHAR szMessage[255];

HRESULT (FAR STDAPICALLTYPE * lpDllEntryPoint)(void);

if (m_strFileName.GetLength() == 0)
{
        wsprintf(szMessage, "Cannot perform %s task without control file name.", strTitle);
        MessageBox(     szMessage,
                        strTitle,
                        MB_ICONSTOP + MB_OK);

        return;
}

// Initialize OLE.
if (FAILED(OleInitialize(NULL)))
{
        wsprintf(szMessage, "OleInitialize failed for %s.", m_strFileName);
        MessageBox(     szMessage,
                        strTitle,
                        MB_ICONSTOP + MB_OK);
}

// Load the library.
HINSTANCE hLib = LoadLibrary(m_strFileName);
```

```
if (hLib < (HINSTANCE)HINSTANCE_ERROR)
{
        TCHAR szError[12];

        wsprintf(szError, _T("0x%08lx"), GetLastError());
        wsprintf(szMessage,
                "LoadLibrary(""%s"") failed.\nGetLastError returns %s.",
                m_strFileName,
                szError);

        MessageBox(szMessage, strTitle, MB_ICONSTOP + MB_OK);
        goto CleanupOle;
}

// Find the entry point.
(FARPROC&)lpDllEntryPoint = GetProcAddress(hLib, pszDllEntryPoint);

if (lpDllEntryPoint == NULL)
{
        TCHAR szExt[_MAX_EXT];
        _tsplitpath(m_strFileName, NULL, NULL, NULL, szExt);

        if ((stricmp(szExt, ".dll") != 0) && (stricmp(szExt, ".ocx") != 0))
        {
                wsprintf(szMessage, "%s was loaded, but the %s' entry point was not
                        found.\n\n%s does not appear to be a .DLL or .OCX file.",
                        m_strFileName, m_strFileName, m_strFileName);
                MessageBox(     szMessage,
                                strTitle,
                                MB_ICONSTOP + MB_OK);
        }
        else
        {
                wsprintf(szMessage, "%s was loaded, but the %s' entry point was not
                        found.\n\n%s may not be exported, or a corrupt version of %s may be
                        in memory. Consider using PView to detect and remove it.",
                        m_strFileName, m_strFileName, m_strFileName, m_strFileName);
                MessageBox(     szMessage,
                                strTitle,
                                MB_ICONSTOP + MB_OK);
        }
        goto CleanupLibrary;
}

// Call the entry point.
if (FAILED((*lpDllEntryPoint)()))
{
        wsprintf(szMessage, "%s of %s failed.", strTitle, m_strFileName);
        MessageBox(szMessage, strTitle, MB_ICONSTOP + MB_OK);
        goto CleanupLibrary;
}

wsprintf(szMessage, "%s of %s succeeded.", strTitle, m_strFileName);
MessageBox(szMessage, strTitle, MB_OK);
```

```
CleanupLibrary:
        FreeLibrary(hLib);

CleanupOle:
        OleUninitialize();
```

No Help Available

No help is available for this area of the window.

Software Ordering

AdvDisk is distributed as for evaluation for thirty days (License). Please remember that using unlicensed shareware past the evaluation period is unethical and illegal. To register and receive the latest version of AdvDisk, please complete the order form and include **US$15.00**. Foreign orders add **US$0.00** for international postage and handling.

Any questions about the status of shipment, refunds, registration options, product details, technical support, volume discounts, dealer pricing, site licenses, etc., must be directed to Advanced Applications directly (see below). Multiple, network and site licenses are available. Contact Advanced Applications for terms and conditions. Prices subject to change without notice.

Contacts

Advanced Applications
6700 North Tryon Street
Box 560991.
Charlotte, NC 28256-0991
USA

Compuserve	72713,2106
Voice	(704) 597-3948

Ordering Information

Select Registration form, and then after printing the form, send it in.

License Agreement

This software is licensed shareware. Select Registration form, and then after printing the form, send it in. Or you may register this software on CompuServe “ GO SWREG”, ID 6041

AdvDisk is supplied "as is". The author disclaims all warranties, expressed or implied, including, without limitation, the warranties of merchant ability and of fitness for any purpose. The author assumes no liability for damages, direct or consequential, which may result from the use of AdvDisk. You also agree to not hold Advanced Applications liable for any damages, direct or consequential which may result from the use of AdvDisk. By purchasing, using, distributing AdvDisk, you are agreeing to all of these conditions.

AdvDisk is provided free of charge for a thirty day evaluation period. Any use beyond this period requires that AdvDisk be registered with the author, Advanced Applications. This registration fee will license one copy of AdvDisk for use on a single computer. Multi-user, network and site licenses are available; contact Advanced Applications for terms.

Government users: This software and documentation are subject to restrictions set forth in **The Rights in Technical Data and Computer Software clause at 252.227-7013 and elsewhere.**

If you believe AdvDisk is valuable and useful, please give it to anyone else you think would be interested, and encourage them to register their copy. Copy registration also provides notification of Revision Updates, these upgrades will be supplied at a upgrade price.

Note: Without the ADVDISK.LIC file your copy of ADVDISK.OCX will display **False** in the **Registered Property** if the control is not registered.

Copyright

Registration Form

To: Advanced Applications
6700 North Tryon Street
Box 560991
Charlotte, NC 28256-0991
Attn.: **ADVDISK.OCX**, Revision 4.0

Fm: (Name)

(Company)

(Address)

(City, State)

(Country)

(ZIP/Post)

(Phone)

(Fax)

(Software) **AdvDisk Software.**

Price	**$15.00**	
Copies		(Number of copies).
Subtotal $		($15.00 x Copies).
Tax	$	(6% North Carolina State).
Shipping	$**00.00**	
Total	$	(Latest Release/updates).

(Do not write below this line)

Receive Date:

Serial Number:

Check/MO Number:

Notification Sent:

Sent by/Date: