Apr - 06

Application Note #4423

Getting Started with Galil's .NET API (DMCdNet.dll) for Visual Studio 2003 and 2005

Galil's .NET API (Application Program Interface) provides a set of classes to aid in the development of Windows based .NET software applications that communicate to Galil controllers. This document describes how to get started with a simple "hello controller" program, error handling and the method list. Benefits of using the .NET API include:

a) .NET compliant – data types/structures take advantage of .NET programming environment. All of the core functions to communicate to Galil controllers (DMCWin32 API) have been ported to the .NET development platform.
b) Ease of programming – Online help (F1) is available for quick access to the help files and "Intellisense" auto-completion of functions.
c) Improved error handling – exception based error handling that simplifies error checking with try and catch statements.

The Galil namespace contains classes that define methods and properties used to communicate with Galil motion controllers from a .NET project. Programmers familiar with the Galil DMCWin32 API (DMC32.dll) should find this .NET API familiar. Here is a list of classes in the Galil namespace:

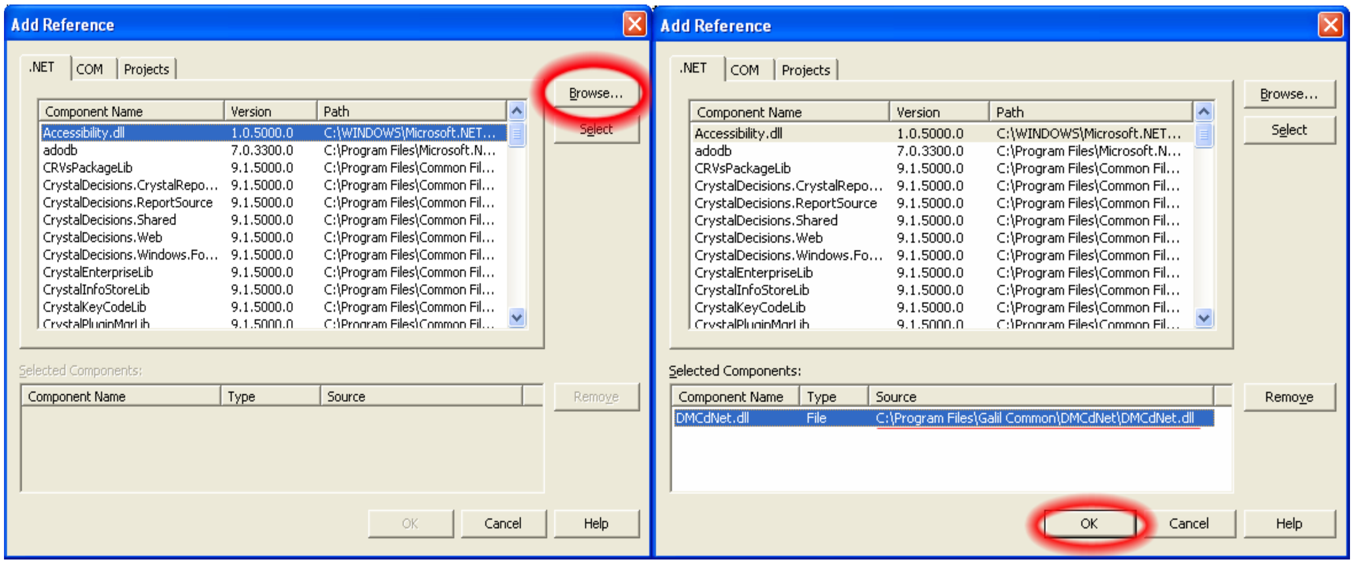
DMCAPI	The DMCAPI class is the core of the Galil namespace and supports all communication with Galil controllers. Each DMCAPI object represents a connection with a controller (use two DMCAPI objects to communicate with two controllers).
DMCArray	The DMCArray class allows the programmer to easily utilize arrays on the motion controller.
DMCDR	The Data Record class represents the record of binary axis and I/O data that most Galil Motion Controllers can produce. The data record is received from the controller in binary form and used to fill in the members of this class for easy access to the information. DMCAPI contains a property called dr which is an instance of DMCDR.
DMCException	An exception class derived from System.Exception that adds Galil specific information such as the DMCErrorCode property.
DMCGalilRegistry	The DMCGalilRegistry class supports the use of the Galil Registry. The Galil Registry is a series of keys within the Windows registry that is used to store a controller's communication configuration. When a connection is opened to a controller using the DMCAPI class, the information stored in the Galil Registry is used to configure the connection.

Getting Started

Here are the steps necessary to build a simple VB.NET and C# application with the Galil .NET API:

1. Download and install Galil software including the .NET API installation from http://www.galilmc.com/support/download.html and verify communication to the controller through "Smart-Term" or "WSDK" software.
2. Open up Visual Studio .NET and select "New Project". Choose the type of project you would like (ie: VB or C#). A "Windows Application" was chosen for this example.

   Click "Project – Add Reference". A dialog box will open. Click on "Browse" and open the file: "C:\Program Files\Galil Common\DMCdNetFW2\DMCdNet.dll" for .NET 2005 or "C:\Program Files\Galil Common\DMCdNet\DMCdNet.dll" for .NET 2003 and then hit OK as shown here:

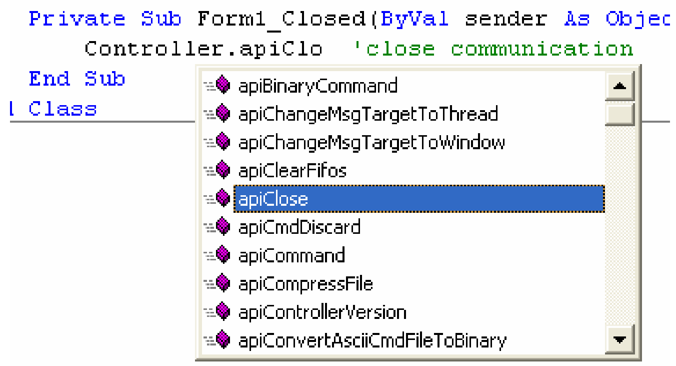


Component Name	Version	Path
Accessibility.dll	1.0.5000.0	C:\WINDOWS\Microsoft.NET...
adodb	7.0.3300.0	C:\Program Files\Microsoft.N...
CRVsPackageLib	9.1.5000.0	C:\Program Files\Common Fil...
CrystalDecisions.CrystalRepo...	9.1.5000.0	C:\Program Files\Common Fil...
CrystalDecisions.ReportSource	9.1.5000.0	C:\Program Files\Common Fil...
CrystalDecisions.Shared	9.1.5000.0	C:\Program Files\Common Fil...
CrystalDecisions.Web	9.1.5000.0	C:\Program Files\Common Fil...
CrystalDecisions.Windows.Fo...	9.1.5000.0	C:\Program Files\Common Fil...
CrystalEnterpriseLib	9.1.5000.0	C:\Program Files\Common Fil...
CrystalInfoStoreLib	9.1.5000.0	C:\Program Files\Common Fil...
CrystalKeyCodeLib	9.1.5000.0	C:\Program Files\Common Fil...
CrystalPluginMgrLib	9.1.5000.0	C:\Program Files\Common Fil...

Selected Components:

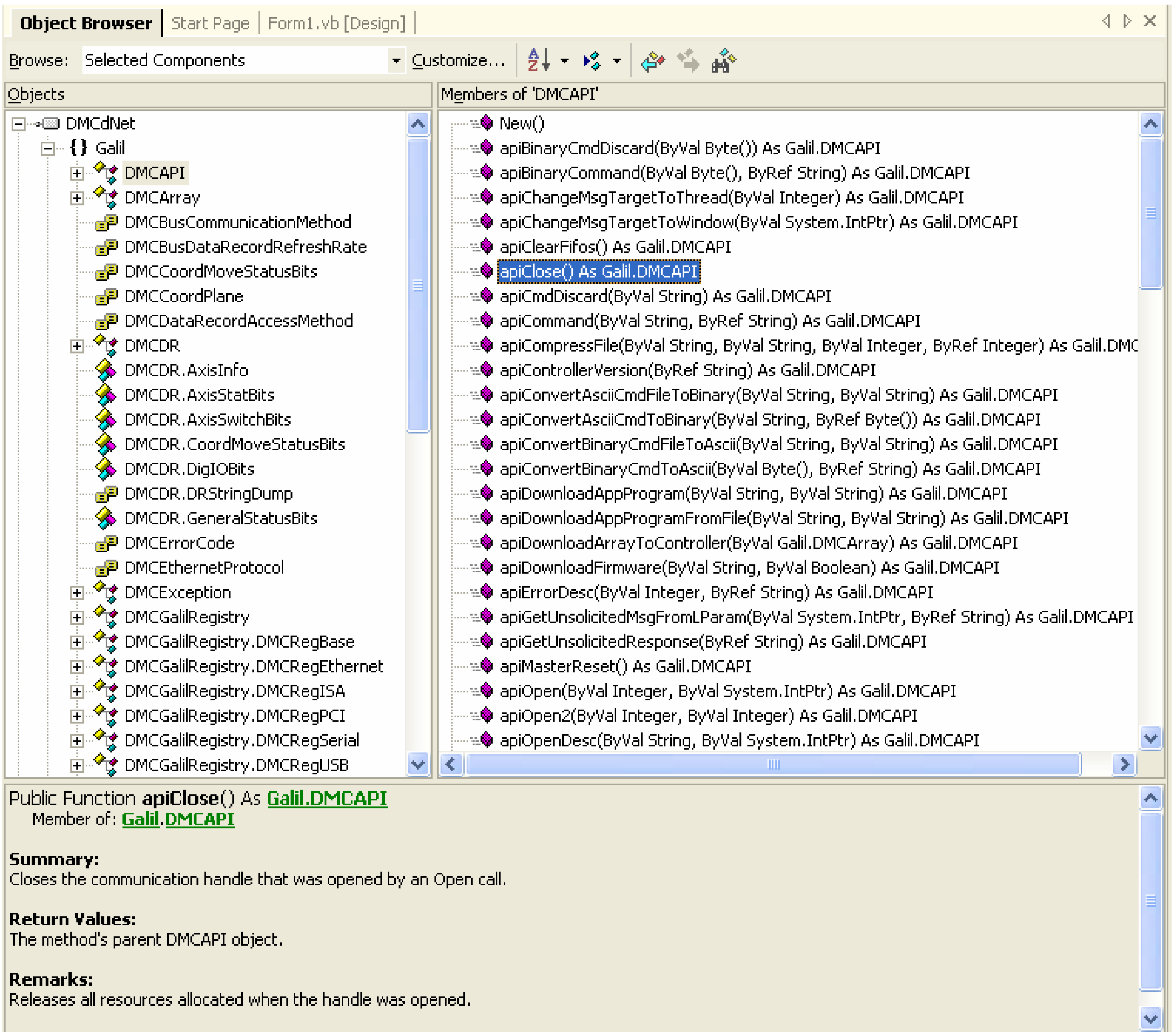
Component Name	Type	Source
DMCdNet.dll	File	C:\Program Files\Galil Common\DMCdNet\DMCdNet.dll

3. Click on the menu item "View – Object Browser" and expand DMCdNet and **{}**Galil (see next page). This will show all of the classes, structures, and enumerations available in the API. Click on DMCAPI and the list of functions included will show up on the right hand side. Click on any one of the functions and hit **F1** to open the help file on it. While typing, the "Intellisense" will suggest a function as shown here:



```
Private Sub Form1_Closed(ByVal sender As Objec
    Controller.apiClo  'close communication
End Sub
l Class
```

Hit the Tab key to have it auto-complete the function. Then hit **F1** if you want to bring up the help page on that function. Place the cursor on the code for any function and hit **F1** to open the help page on it.


Object Browser | Start Page | Form1.vb [Design]
Browse: Selected Components
Customize...
Objects
Members of 'DMCAPI'

```
DMCdNet
  {} Galil
    DMCAPI
    DMCArray
    DMCBusCommunicationMethod
    DMCBusDataRecordRefreshRate
    DMCCoordMoveStatusBits
    DMCCoordPlane
    DMCDataRecordAccessMethod
    DMCDR
    DMCDR.AxisInfo
    DMCDR.AxisStatBits
    DMCDR.AxisSwitchBits
    DMCDR.CoordMoveStatusBits
    DMCDR.DigIOBits
    DMCDR.DRStringDump
    DMCDR.GeneralStatusBits
    DMCErrorCode
    DMCEthernetProtocol
    DMCException
    DMCGalilRegistry
    DMCGalilRegistry.DMCRegBase
    DMCGalilRegistry.DMCRegEthernet
    DMCGalilRegistry.DMCRegISA
    DMCGalilRegistry.DMCRegPCI
    DMCGalilRegistry.DMCRegSerial
    DMCGalilRegistry.DMCRegUSB
```

```
New()
apiBinaryCmdDiscard(ByVal Byte()) As Galil.DMCAPI
apiBinaryCommand(ByVal Byte(), ByRef String) As Galil.DMCAPI
apiChangeMsgTargetToThread(ByVal Integer) As Galil.DMCAPI
apiChangeMsgTargetToWindow(ByVal System.IntPtr) As Galil.DMCAPI
apiClearFifos() As Galil.DMCAPI
apiClose() As Galil.DMCAPI
apiCmdDiscard(ByVal String) As Galil.DMCAPI
apiCommand(ByVal String, ByRef String) As Galil.DMCAPI
apiCompressFile(ByVal String, ByVal String, ByVal Integer, ByRef Integer) As Galil.DMC
apiControllerVersion(ByRef String) As Galil.DMCAPI
apiConvertAsciiCmdFileToBinary(ByVal String, ByVal String) As Galil.DMCAPI
apiConvertAsciiCmdToBinary(ByVal String, ByRef Byte()) As Galil.DMCAPI
apiConvertBinaryCmdFileToAscii(ByVal String, ByVal String) As Galil.DMCAPI
apiConvertBinaryCmdToAscii(ByVal Byte(), ByRef String) As Galil.DMCAPI
apiDownloadAppProgram(ByVal String, ByVal String) As Galil.DMCAPI
apiDownloadAppProgramFromFile(ByVal String, ByVal String) As Galil.DMCAPI
apiDownloadArrayToController(ByVal Galil.DMCArray) As Galil.DMCAPI
apiDownloadFirmware(ByVal String, ByVal Boolean) As Galil.DMCAPI
apiErrorDesc(ByVal Integer, ByRef String) As Galil.DMCAPI
apiGetUnsolicitedMsgFromLParam(ByVal System.IntPtr, ByRef String) As Galil.DMCAPI
apiGetUnsolicitedResponse(ByRef String) As Galil.DMCAPI
apiMasterReset() As Galil.DMCAPI
apiOpen(ByVal Integer, ByVal System.IntPtr) As Galil.DMCAPI
apiOpen2(ByVal Integer, ByVal Integer) As Galil.DMCAPI
apiOpenDesc(ByVal String, ByVal System.IntPtr) As Galil.DMCAPI
```

Public Function **apiClose**() As Galil.DMCAPI
Member of: Galil.DMCAPI

Summary:
Closes the communication handle that was opened by an Open call.

Return Values:
The method's parent DMCAPI object.

Remarks:
Releases all resources allocated when the handle was opened.

4. Next, bring up the "Form1 [Design]" and add a TextBox to it. Double-click anywhere on the form (not on the TextBox). This will bring up the code view of the form.

For **VB.Net**, add the lines of code that are shown prefixed with a Red Arrow (→):

```
→Imports Galil                                          'Import Galil namespace
 Public Class Form1
   Inherits System.Windows.Forms.Form
→    Dim Controller As DMCAPI                            'Declare controller object

   Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
→        Controller = New DMCAPI                         'Allocate memory for controller object
→        Controller.apiOpen(1, System.IntPtr.Zero)       'Open communications

→        Controller.sCommand("SHX;PRX=1000;BGX")        'Send Position Relative move command
→        Controller.apiWaitForMotionComplete("X", True)  'Wait for motion to complete on X axis
→        TextBox1.Text = Controller.sCmdTrim("TPX")      'Display X axis Position in TextBox1

   End Sub
```

[Note for VB.NET 2005, the Form1_Closing event below changes to Form1_FormClosing()]

```
   Private Sub Form1_Closing(ByVal sender As Object, ByVal e As
     System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
→        Controller.apiClose()                            'Close communication
```

```
     End Sub
  End Class
```

For **C# .Net**, add the lines of code that are shown prefixed with a Red Arrow (→):

```
   using System;
   using System.Drawing;
   using System.Collections;
   using System.ComponentModel;
   using System.Windows.Forms;
   using System.Data;
 →using Galil; //use Galil namespace

   namespace Galil_CdotNetAPI_example
   {
        public class Form1 : System.Windows.Forms.Form
        {
                private System.ComponentModel.Container components = null;
                private System.Windows.Forms.TextBox textBox1;
               →private DMCAPI Controller; //declare controller object

                public Form1()
                {

                        InitializeComponent();
                }

                . . .

                [STAThread]
                static void Main()
                {
                        Application.Run(new Form1());
                }

                private void Form1_Load(object sender, System.EventArgs e)
                {
                        →Controller = new DMCAPI(); //allocate memory for controller object
                        →Controller.apiOpen(1, System.IntPtr.Zero); //open communications

                        →Controller.sCommand("SHX;PRX=1000;BGX"); //Send Position Relative move command
                        →Controller.apiWaitForMotionComplete("X",true); //Wait for motion to complete on X axis
                        →textBox1.Text = Controller.sCmdTrim("TPX"); //Display X axis Position in textBox1

                }
```

[Note for C#.NET 2005, the Form1_Closing event below changes to Form1_FormClosing()]

```
                private void Form1_Closing(object sender, System.EventArgs e)
                {
                        →Controller.apiClose(); //Close Communication
                }


        }
   }
```

⚡***Note: To get the Form1_Closing event procedure, go to the Form1.cs [Design] view and click on the form. Go to the "Properties" toolbar and click on the "Events" image (shown as lightning bolt). Double click on the "Closing" (or "FormClosing" for 2005) Event and this will add the event into the code.

Error Handling

The DMCdNet API allows for two methods of error handling. The functions with “ec” as a prefix use the same method as the older DMCWIN API. A variable is put in front of the function (rc) and the “return code” is reported by the function. A zero means the function was successful and any other value signifies there was an error. The new method of error handling uses exceptions and “try” and “catch” blocks. With this method, all the code that needs to be evaluated is put between the curly brackets {} (in C#) of the try block. In The error handling is then done using the code contained in the “catch” block. These three examples illustrate the different methods:

```
****************************************************************************
//OLD API in Visual C 6
rc = DMCOpen(1, 0, &hDmc); //attempt to open controller 1
if(rc != 0) //if there's an error
{
       printf("Error %i occurred\n", rc); //print the error number
}
****************************************************************************
//NEW .NET API in C# with “ec” functions
rc = Controller.ecOpen(1, System.IntPtr.Zero); //open communications
if(rc != 0) //if there's an error
{
       textBox1.Text = "Error" + rc.ToString() + “ occurred”;
}
****************************************************************************
//NEW .NET API in C# with “api” functions
try
{
       Controller.apiOpen(1, System.IntPtr.Zero); //attempt to open controller 1
}
catch (DMCException MyException)
{
       textBox1.Text = "Error" + MyException.DMCErrorCode + " occurred";
}
****************************************************************************
```

```
   List of Error codes.
0 No error occured
-1 A time-out occurred while waiting for a response from the controller.
-2 There was an error with a command sent to the controller.  Send “TC1” to get error code.
-3 Controller could not be found in Windows registry.
-4 File could not be opened.
-5 Device driver could not be opened, or a read or write error occured.
-6 Invalid controller handle.
-7 Support dynamic link library could not be loaded.
-8 Out of memory.
-9 Response from the controller was larger than the response buffer supplied.
-10 Response from the controller overflowed the internal additional response buffer.
-11 Could not communicate with DMA channel.
-12 One or more required arguments to a DMC API function call was NULL.
-13 Could not access DMC data record.
-14 File Download Failed.  Total number of lines or line length restriction may have been exceeded.
-15 Could not update the controller's firmware.
-16 Could not convert DMC command (ASCII to binary or binary to ASCII).
-17 Windows reports a resource conflict with the current hardware configuration.
-18 Could not access or modify the controller's registry information.
-19 Controller is busy and not ready to accept commands.
-20 Controller has been disconnected from the communications channel (USB or Ethernet).
-21 Data is not being transfered to controller fast enough to maintain time synchronization.
-22 The user supplied buffer is too large. Must be < 1024 bytes.
-23 Registry modification of PnP controllers is not allowed.
-24 This function is obsolete.
-25 A different process is using a streaming command(LS,UL,ED,QD,QU).
-26 The device driver needed to communicate with the selected controller is too old for this dll.
-27 Streaming commands (LS,UL,ED,QD,QU) cannot be mixed with other commands on the command line.
-28 The dll/driver has been configured to use a feature that is not supported by this (older) f/w version.
-29 The dll/driver is trying to connect to an ethernet controller that doesn’t have enough available handles.
-30 The dll/driver may be attempting to connect to an ethernet controller with an IP address that is
     not part of the network adapter's subnet.
```

Function List

Here is the list of available functions (methods) in the Galil namespace. Some functions have multiple prefixes that can be used to change the way the function works. A list of prefixes and their purpose is given at the end of the table.

COMMUNICATION

Class	Possible Prefixes			Function
Galil.DMCAPI.	api	ec		ChangeMsgTargetToThread
Galil.DMCAPI.	api	ec		ChangeMsgTargetToWindow
Galil.DMCAPI.	api	ec		ClearFifos
Galil.DMCAPI.	api	ec		Close
Galil.DMCAPI.	api	ec		CmdDiscard
Galil.DMCAPI.	api	ec	s	Command
Galil.DMCAPI.	api	ec	s	ControllerVersion
Galil.DMCAPI.	api	ec	s	ErrorDesc
Galil.DMCAPI.	api	ec	s	GetUnsolicitedMsgFromLParam
Galil.DMCAPI.	api	ec	s	GetUnsolicitedResponse
Galil.DMCAPI.	api	ec		MasterReset
Galil.DMCAPI.	api	ec		Open
Galil.DMCAPI.	api	ec		Open2
Galil.DMCAPI.	api	ec		OpenDesc
Galil.DMCAPI.	api	ec		OpenDesc2
Galil.DMCAPI.	api	ec	s	ReadData
Galil.DMCAPI.	api	ec		Reset
Galil.DMCAPI.	api	ec		WaitForMotionComplete
Galil.DMCAPI.	api	ec		WriteData
Galil.DMCAPI.	bool			ThreadWaitForUnsolicitedMsg*
Galil.DMCAPI.	bool			ThreadWaitForUserInterruptMsg*
Galil.DMCAPI.	bool			ThreadWaitForUnsolicitedOrInterruptMsg*
Galil.DMCAPI.				FromHandle
Galil.DMCAPI.				sarCmd
Galil.DMCAPI.				sarCmdTrim
Galil.DMCAPI.			s	CmdTrim
Galil.DMCAPI.	api	ec		DiagnosticsOff
Galil.DMCAPI.	api	ec		DiagnosticsON
Galil.DMCAPI.	api			Sleep
Galil.DMCAPI.	api			OpenWithEventsEnabled*
Galil.DMCAPI.	api			ReEnableEvents*

*Added or Modified Functions for .NET 2005. See Help files for complete documentation.

ARRAYS

Class	Possible Prefixes			Function
Galil.DMCAPI.	api	ec		DownloadArrayToController
Galil.DMCAPI.	api	ec		UploadArrayFromController
Galil.DMCArray.				arControllerAllocate
Galil.DMCArray.				arControllerDeallocate
Galil.DMCArray.				arDownloadToController
Galil.DMCArray.				arLoadFromFile
Galil.DMCArray.				arSaveToFile
Galil.DMCArray.				arUploadFromController
Galil.DMCArray.				GetValue
Galil.DMCArray.				arLoadArrayValuesFromTextFile
Galil.DMCArray.				arSaveArrayValuesToTextFile
Galil.DMCArray.				#ctor
Galil.DMCArray.				#ctor
Galil.DMCArray.				#ctor
Galil.DMCArray.				SetValue
Galil.DMCArray.				ZeroArray

FILES

Class	Possible Prefixes			Function
Galil.DMCAPI.	api	ec		CompressFile
Galil.DMCAPI.	api	ec		DownloadAppProgram
Galil.DMCAPI.	api	ec		DownloadAppProgramFromFile
Galil.DMCAPI.	api	ec		DownloadFirmware
Galil.DMCAPI.	api	ec		SendCmdFileToController
Galil.DMCAPI.	api	ec	s	UploadAppProgram
Galil.DMCAPI.	api	ec		UploadAppProgramToFile

DATA RECORD

Class	Possible Prefixes			Function
Galil.DMCDR.	dr	ec	api	AxisData
Galil.DMCDR.	dr	ec	api	CoordMoveDistance
Galil.DMCDR.	dr	ec	api	CoordMoveSegmentCount
Galil.DMCDR.	dr	ec	api	CoordMoveStatBits
Galil.DMCDR.	dr	ec	api	CoordMoveStatBytes
Galil.DMCDR.	dr	ec	api	DigInputBit
Galil.DMCDR.	dr	ec	api	DigInputByte
Galil.DMCDR.	dr	ec	api	DigOutputBit
Galil.DMCDR.	dr	ec	api	DigOutputByte
Galil.DMCDR.	dr	ec	api	ErrorCode
Galil.DMCDR.	dr	ec	api	GenStatBits
Galil.DMCDR.	dr	ec	api	GenStatByte
Galil.DMCDR.	dr	ec	api	Refresh
Galil.DMCDR.	dr	ec	api	SampleNum
Galil.DMCDR.	dr	ec	api	SetDRIndex
Galil.DMCDR.				AxisData
Galil.DMCDR.	bool	byte		DigInputBit
Galil.DMCDR.	bool	byte		DigOutputBit
Galil.DMCDR.				byteGenStat
Galil.DMCDR.				CoordMoveStatBits
Galil.DMCDR.				DigInput
Galil.DMCDR.				DigOutput
Galil.DMCDR.				GenStatBits
Galil.DMCDR.				intCoordMoveDistance
Galil.DMCDR.				intCoordMoveSegmentCount
Galil.DMCDR.				intCoordMoveStat
Galil.DMCDR.				intErrorCode
Galil.DMCDR.				intGetDRCount
Galil.DMCDR.				intRefresh
Galil.DMCDR.				intSampleNum
Galil.DMCDR.				op_Increment
Galil.DMCDR.				sDumpDR
Galil.DMCDR.	dr	ec	api	AnalogOutInfo
Galil.DMCDR.	dr	ec	api	AnalogOutput
Galil.DMCDR.	dr	ec	api	ContourBufferSpace
Galil.DMCDR.	dr	ec	api	ContourSegmentCount
Galil.DMCDR.	dr	ec	api	CoordMoveBufferSpace
Galil.DMCDR.	dr	ec	api	ThreadStatBits
Galil.DMCDR.	dr	ec	api	ThreadStatByte
Galil.DMCDR.				intAnalogOutput
Galil.DMCDR.				intContourBufferSpace
Galil.DMCDR.				intContourSegmentCount
Galil.DMCDR.				intCoordMoveBufferSpace
Galil.DMCDR.				byteThreadStat

Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007AnalogInput
Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007AnalogInputData
Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007AnalogVoltage
Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007AnalogVoltageData
Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007DigBit
Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007DigData
Galil.DMCDR.IOCDR.	dr	ec	api	IOC7007SlotData
Galil.DMCDR.IOCDR.			bool	IOC7007DigBit
Galil.DMCDR.IOCDR.				fIOC7007AnalogVoltage
Galil.DMCDR.IOCDR.			int	IOC7007AnalogInput

REGISTRY

Class	**Possible Prefixes**			**Function**
Galil.DMCGalilRegistry.				AddController
Galil.DMCGalilRegistry.				CreateRegObject
Galil.DMCGalilRegistry.				DeleteController
Galil.DMCGalilRegistry.				EditRegistryDlg
Galil.DMCGalilRegistry.				GetAllControllersFromRegistry
Galil.DMCGalilRegistry.				GetControllerFromRegistry
Galil.DMCGalilRegistry.				GetControllerFromRegistry
Galil.DMCGalilRegistry.				GetControllerModelDesc
Galil.DMCGalilRegistry.				ModifyController
Galil.DMCGalilRegistry.				SelectControllerDlg

BINARY COMMANDS

Class	**Possible Prefixes**			**Function**
Galil.DMCAPI.	api	ec		BinaryCmdDiscard
Galil.DMCAPI.	api	ec		BinaryCommand
Galil.DMCAPI.	api	ec		ConvertAsciiCmdFileToBinary
Galil.DMCAPI.	api	ec	byte	ConvertAsciiCmdToBinary
Galil.DMCAPI.	api	ec		ConvertBinaryCmdFileToAscii
Galil.DMCAPI.	api	ec	s	ConvertBinaryCmdToAscii
Galil.DMCAPI.	api	ec		ReadSpecialBinaryCmdConversionFile
Galil.DMCAPI.	api	ec		SendBinaryCmdFileToController
Galil.DMCAPI.	sar	s		BinaryCmd
Galil.DMCAPI.	sar	s		BinaryCmdTrim

Prefix Designations

The Galil class methods use a prefix notation to indicate the method return type. Most methods that do not begin with an ec can throw exceptions.

"api" - returns the parent DMCAPI object, allowing methods to be concatenated. Use try{} and catch{} blocks for error handling. Here is a VB.NET example of concatenating:

```
Controller.apiOpen(1,System.IntPtr.Zero).apiCmdDiscard("SHX").apiClose()
```

"ec" - returns a Galil error code. Use these methods if porting an application that used the DMC32.dll API and switching to exception-based error handling is not desired. Use a variable in front of the function to get the error code that is returned. Concatenation is not allowed with these functions. Here is a VB.NET example:

```
Dim RC As Integer
RC = Controller.ecCmdDiscard("SHX")
RC = Controller.ecClose()
```

"s" - returns a System.String. Use this to avoid two steps as shown here:

```
Controller.apiCommand("MG _BN", sResponse) 'get serial number
```

```
        TextBox1.Text = "Serial number= " + sResponse 'display response

        'Alternate method using "s" instead of "api" prefix
        TextBox1.Text = "Serial number= " + Controller.sCommand("MG_BN")
```

"sar" - returns an array of System.Strings.
"bool" - returns a System.Boolean.
"byte" - returns a System.Byte array.
"dr" - returns the parent DMCDR object, allowing methods to be concatenated.
"int" - returns a System.Int32
"ar" - returns a DMCArray object

Methods with no prefix return something other than the above types.

Visual Studio .NET 2005

It is important to note that the Visual Studio .NET 2005 API has a different installation than the previous .NET 2003 API. Please verify that you install the correct version of the API. If you were previously working with VS .NET 2003, then you will need to follow the procedure below to update your project to 2005:
Step 1: Uninstall Galil DMC.NET 2003
Step 2: Install DMC.NET 2005
Step 3: Open up the solution and have .NET 2005 automatically convert it.
Step 4: Open up the Solution Explorer and click "Show All Files". Expand "References" and delete DMCdNet.
Step 5: Select Project – Add Reference and browse to "C:\Program Files\Galil Common\DMCdNetFW2\DMCdNet.dll"

You should now be able to run your existing solution written in .NET 2003 with .NET 2005.

Events were added to the DMCdNet 2005 API. The following functions take advantage of events:
DMCAPI.apiOpenWithEventsEnabled
DMCAPI.ReEnableEvents
Please see the Help files for documentation and examples of using Events.

Note: Windows XP 64bit O/S with .NET Express should follow the procedure below to use Galil's .NET API.
When Microsoft stripped the IDE to make the Visual Studio Express interfaces, they removed the "PlatformTarget" property from the compiler settings. As a result, the Express versions default to target "AnyCPU", which means on a 64 bit machine it compiles a 64 bit application. When the 64 bit app tries to load Galil 32 bit dlls, a BadImageFormatException is thrown

To work around this, users must open the project file in a text editor (notepad works just fine) and add the following line:

```
<PlatformTarget>x86</PlatformTarget>
```

to all <PropertyGroup> nodes in the file. As an example, the following text is a VBasic Express project file listing through the first <PropertyGroup> section. The <PlatformTarget> node has been added.

Listing for ConsoleApplication1.vbproj:

```
<?xml version="1.0" encoding="utf-8"?>
<Project DefaultTargets="Build"
xmlns="http://schemas.microsoft.com/developer/msbuild/2003">
   <PropertyGroup>
     <Configuration Condition=" '$(Configuration)' == '' ">Debug</Configuration>
     <Platform Condition=" '$(Platform)' == '' ">AnyCPU</Platform>
     <ProductVersion>8.0.50727</ProductVersion>
     <SchemaVersion>2.0</SchemaVersion>
     <ProjectGuid>{B6DF5666-2217-4A6F-9B57-9077562C1415}</ProjectGuid>
```

```
    <OutputType>Exe</OutputType>
    <StartupObject>Sub Main</StartupObject>
    <RootNamespace>ConsoleApplication1</RootNamespace>
    <AssemblyName>ConsoleApplication1</AssemblyName>
    <MyType>Console</MyType>
    <PlatformTarget>x86</PlatformTarget>
  </PropertyGroup>
```