

AN00127-001 - Using the MintMT ActiveX Control in Microsoft Visual Studio .NET

Overview

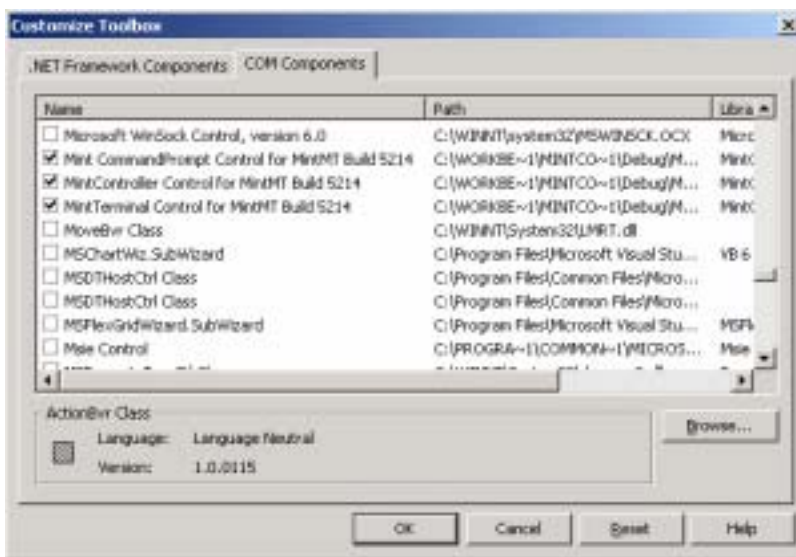
Visual Studio .NET is the latest development environment from Microsoft. It supports a variety of different languages, including Visual Basic, Visual C++ and C#. All these languages fully support ActiveX controls, and hence the development of host applications using the MintMT ActiveX control. This document aims to show the reader how to get started with automating Baldor's MintMT motion control and drive products using Microsoft Visual Studio .NET. Later on we will also discuss some of the differences that may be encountered when moving from Visual Basic 6 to Visual Studio .NET.

Supported Controllers

NextMovePCI	<input checked="" type="checkbox"/>
NextMoveBX^{II}	<input checked="" type="checkbox"/>
NextMoveST	<input checked="" type="checkbox"/>
NextMoveESB	<input checked="" type="checkbox"/>
NextMoveES	<input checked="" type="checkbox"/>
MintDrive^{II}	<input checked="" type="checkbox"/>
Flex+Drive^{II}	<input checked="" type="checkbox"/>
FlexDrive^{II}	<input checked="" type="checkbox"/>

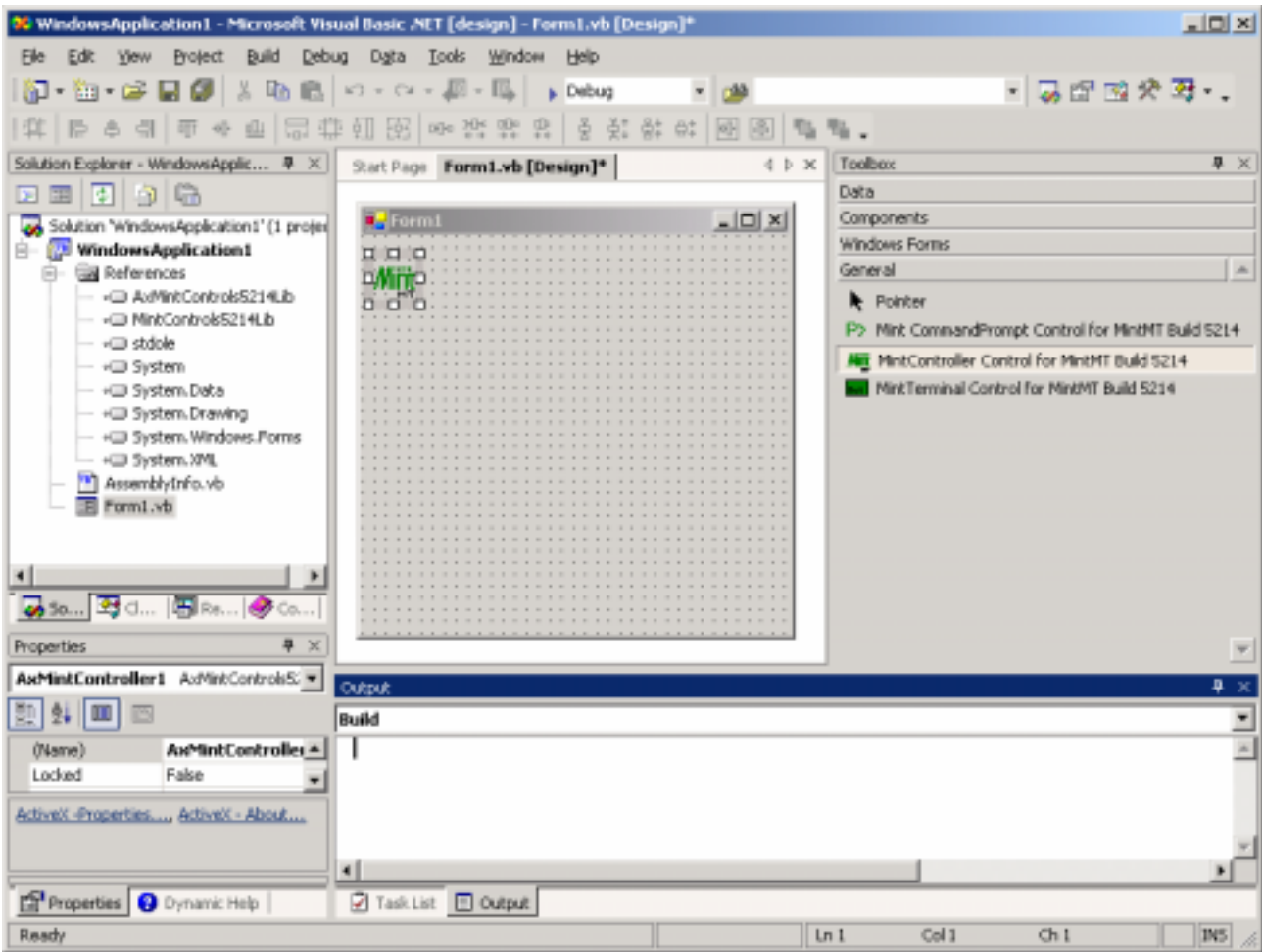
Getting Started

The first step to using the MintMT ActiveX control in Microsoft Visual Studio .NET is to add the control to the toolbox. To do this, select "Add/Remove Toolbox Items..." from the Tools menu. In the "Customize Toolbox" dialog, select the "COM Components" tab, which will list all COM components (an ActiveX control is a COM component) that are registered on your computer. Listed amongst these should be the Mint CommandPrompt control, the MintController control and the MintTerminal control. There will probably be several versions of each control registered, just choose the version you want to work with (if you're not sure then choose the latest) by clicking in the small box to the left of the name (this will become ticked). When you have added the controls you wish to use press OK.



MintMT Application Note

Once you have opened a Visual Studio .NET project, you will find the ActiveX controls on the General tab of the toolbox. You can add a control to your project by selecting it in the toolbox and drawing it on your form.



Differences between Visual Basic 6 and Visual Studio .NET

If you have previously written host applications with the MintMT ActiveX control in Visual Basic 6 then you will notice a few differences when using Microsoft Visual Studio .NET.

Property Naming

In Visual Studio .NET, ActiveX properties that have parameters (e.g. Speed) now have a set_ or get_ in front of the property name.

For example:

VB6

```
Controller.SPEED(0) = 20
```

```
value = Controller.SPEED(0)
```

VB .NET

```
Controller.set_SPEED(0, 20)
```

```
value = Controller.get_SPEED(0)
```

So when using the MintMT ActiveX help file with Visual Studio .NET, remember to add a set_ or get_ to the start of the name of any property with a parameter in the VB examples.

Method Calling

In Visual Basic 6, method calls could be made without using brackets:

VB6

```
Controller.SetNextMoveESLink 2, 1, 57600, True
```

This is no longer supported, so all method calls must include brackets around the arguments:

VB .NET

```
Controller.SetNextMoveESLink( 2, 1, 57600, True )
```

Note that if you accidentally use the VB6 syntax, the Visual Basic .NET editor will automatically add the brackets to it.

MintMT Application Note

Capture

One of the MintMT ActiveX properties, CAPTURE causes a naming conflict in Visual Basic .NET and so is not accessible (note that other Visual Studio .NET languages are not affected by this). From version 5212 of the MintMT ActiveX control, MintController has a new property CAPTURESTATE which is functionally equivalent to CAPTURE but can be called from Visual Basic .NET:

VB6

Controller.CAPTURE = 1

VB .NET

Controller.CAPTURESTATE = 1

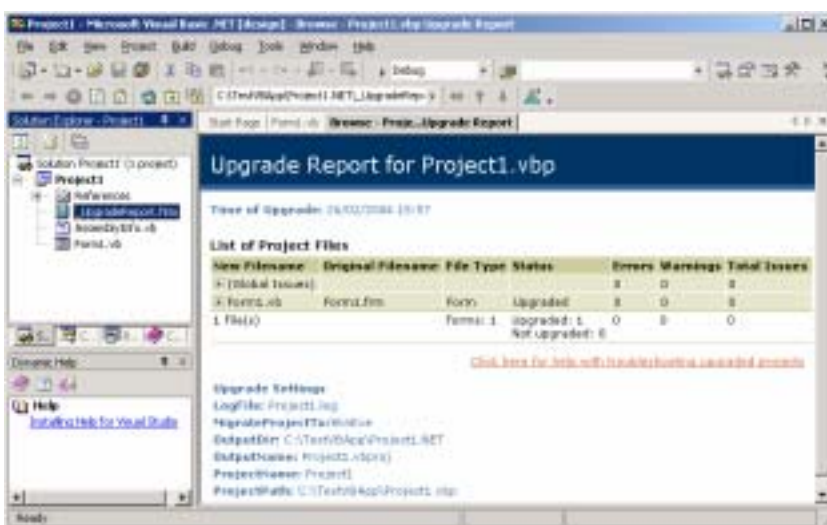
Note that CAPTURESTATE doesn't have any parameters, so it isn't prefixed with set_.

Upgrade Wizard

If you have a Visual Basic 6 application that you would like to move to Visual Basic .NET then there is an Upgrade Wizard in Visual Studio .NET(Professional, Enterprise Developer, and Enterprise Architect Editions only).

From the File menu, select Open and Project, and navigate to the Visual Basic 6 project file (vbp). Visual Studio .NET will realize you are trying to open an old Visual Basic project and will launch the Upgrade Wizard. The wizard will ask you to choose the application type (EXE or DLL/Custom Control) and to specify where you would like the new Visual Basic .NET project to be created (note the original project will not be modified during this process, so you will still be able to open it in Visual Basic 6).

The new project contains a report outlining the results of the upgrade. This may include details of areas where the wizard encountered problems, for example, code that the wizard wasn't able to upgrade, which will need attention.



Using the Command Prompt and Terminal ActiveX controls

In Visual Basic 6 you could easily connect the Terminal or Command Prompt controls to the MintController control using the setMintController method:

VB6

Controller.SetMintDrivellink 2, 1, 57600, True ' create a controller connection

Terminal.setMintController Controller ' connect the controller to the terminal

Unfortunately this no longer works in Visual Studio .NET. Instead, the old setMintControllerID function should be used:

VB .NET

Controller.SetMintDrivellink(2, 1, 57600, True)

Terminal.setMintControllerID(Controller.ControllerID)

And for the command prompt you also need to set up the compiler and symbol table:

VB .NET

Controller.SetMintDrivellink(2, 1, 57600, True)

Prompt.setMintControllerID(Controller.ControllerID)

Prompt.setCompiler(10) ' See below. Must call this before setting symbol table.

Dim str As String = ""

Controller.GetSymbolTableForController(False, str)

Prompt.setSymbolTable(str)

The Compiler Target Format that is written with the setCompiler() call is linked to the version of the ActiveX control being used. The following table shows which compiler target format to use for a given ActiveX control.

ActiveX Control	Compiler
Builds 50xx	8
Builds 51xx	9
Builds 52xx	10