

Graphics Class Libraries for MFC

Revision 3.0

Windows 95
Windows 98
Windows NT

Used as an Add-On to the *Charting Tools* and *Real-Time Graphics Tools for Windows*.

Compatible with 32-bit MFC Programs.

32-bit - for Visual C++5.0 - used to create 32-bit applications which will run under Windows NT, Windows 95 and Windows 98.

No Runtime Royalties.

Source Code Included - the source code to the *GCL* is included. This product does not include the source code to the Quinn-Curtis *Charting* and *Real-Time Graphics* DLLs. See the product descriptions for the *Charting Tools* and *Real-Time Graphics Tools* for source code options for our DLLs.

Encapsulates both the *Charting Tools* and the *Real-Time Graphics Tools for Windows*.

Print Preview - view your graphs as they will be printed using print preview.

Base MFC Classes - CWnd, CView, CScrollView and CFormView.

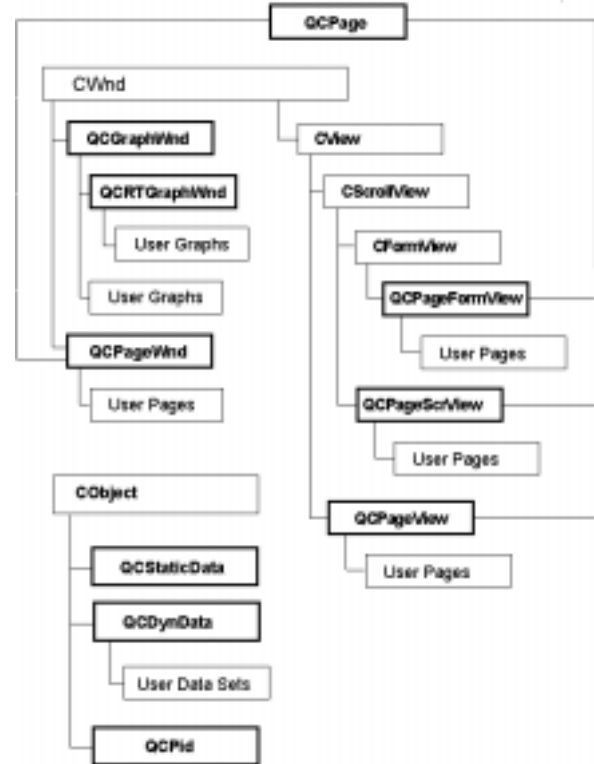
Data Set Classes - static and dynamic data set classes.

Compatible with both Static and Dynamic MFC libraries.

Documentation - a separate 500 page manual, organized by class, documents all of the *Charting Tools* and *Real-Time Graphics Tools* functions in the class libraries.

Example Programs - 20 MFC-based example programs demonstrate the proper use of the software in SDI, MDI and Form applications.

AppWizard Tutorial - a tutorial teaches you step by step how to integrate *GCL* with MFC applications created using the Visual C++ App Wizard.



Class Hierarchy Chart for the *Graphics Class Libraries for MFC*

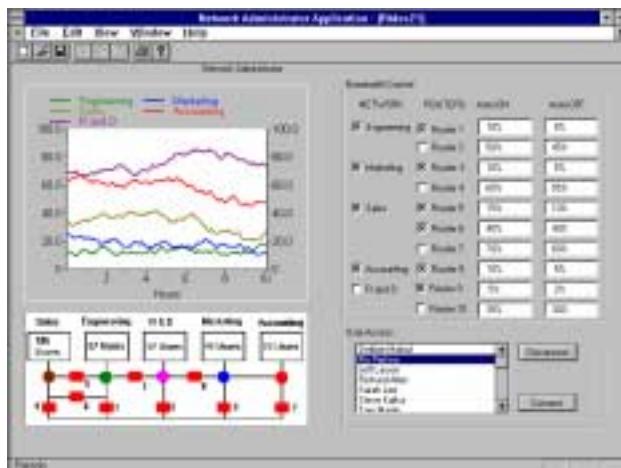
The Quinn-Curtis *Graphics Class Libraries* (GCL) for MFC is an add-on product to our *Charting Tools* and *Real-Time Graphics Tools for Windows*. The *Graphics Class Libraries* integrate our tools with the MFC Document/View architecture. These libraries include classes for both the *Charting Tools* and *Real-Time Graphics Tools*. It provides a convenient object-oriented interface that significantly reduces the effort needed to create graphics applications. With these libraries you can build displays that inherit all of the behavior of CWnd, CView, CScrollView, and CFormView classes, including print preview. The price of the Quinn-Curtis *Graphics Class Libraries for MFC* is \$200. The full source for the class libraries are included.

ORDERING INFORMATION

PART #	DESCRIPTION	PRICE
WIN-MFC-210	Graphics Class Libraries for MFC	\$200

SHIPPING CHARGES

UPS Ground	UPS Blue	UPS Red	DHL	Canada
12	20	30	48	20



Combine real-time graphics and dialog controls using CFormView

The primary classes of the *Graphics Class Libraries* are defined in one of the following major categories: Page Classes, Graph Classes, Data Set Classes and PID Control Classes.

Page Classes

Pages are described by the classes **QCPageWnd**, **QCPageView**, **QCPageScrView**, and **QCPageFormView**. Pages are derived from the abstract base page class, **QCPage** and one of the MFC window classes: **CWnd**, **CView**, **CScrollView**, or **CFormView**. Page classes can be used in both SDI and MDI applications. Page classes have a virtual function, **BuildPage**, that must be overridden to create graphs in the page window.

Graph Classes

Graphs are represented by the classes **QCGraphWnd** for static graphs, and **QCRTGraphWnd** for real-time graphs. These classes encapsulate all of the drawing logic for graphs and their components. They have a virtual function **BuildGraph** that must be overridden to draw a graph in the graph window. Detailed function descriptions are found in the *Charting Tools* and *Real-Time Graphics Tools for Windows* product descriptions.

Data Set Classes

Static and dynamic data sets are described by the classes **QCStaticData** and **QCDynData**. The class **QCStaticData** combines data for independent and dependent variables. It represents data in a graph that does not require continuous real-time updates. The class **QCDynData** manages dynamic floating point and binary data used to monitor alarms and update real-time graphs.

PID Control Class

The class **QCPid** provides PID (Proportional-Integral-Derivative) control functionality. See the description of the PID control routines in the *Real-Time Graphics Tools* for Windows product description.

Example Code Listings

Code listings extracted from a simple static charting example program created using the Visual C++ AppWizard with the *GCL* are found to the right. A complete, and more complicated real-time example can be downloaded from our WWW, FTP and BBS sites. Download the file *GCL_CODE.ZIP*. The compressed file also includes an executable of the program.

WWW <http://www.quinn-curtis.com>
 FTP at <ftp.webcom.com> in /pub/quinn/demofiles
 BBS 617/449-4783

Code Segment from Document CPP File

```
#define NUMP1 1024

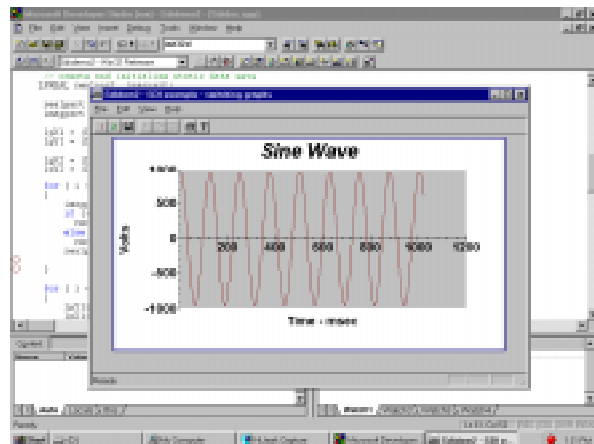
CSdidDoc::CSdidDoc()
{
    int i;
    // create and initialize static data sets
    lpX1 = (LPREAL) calloc(NUMP1, sizeof(realtype));
    lpY1 = (LPREAL) calloc(NUMP1, sizeof(realtype));
    // Create simulated data sine wave
    for ( i = 0; i < NUMP1; i++)
    {
        lpY1[i] = 15.0 * cos ((realtype) / 20.0);
        lpX1[i] = (realtype) i;
    }
    pDataSet = new QCStaticData ("60 Cycle Noise",
        lpX1, lpY1, NUMP1, FALSE);
}
}
```

Code Segment from View CPP file

```
// Page building procedure
// Called from QCPageView::OnInitialUpdate()
void CSdiPageView::BuildPage()
{
    // create graph object
    pGraph1= new CGraph1 (this, GetDocument());
    pGraph1->Create (0.05, 0.01, 0.95, 0.8,
        C_WHITE, C_BLUE, 2) ;
}
}
```

Code Segment from Graph Class File

```
void CGraph1::BuildGraph()
{
    CSdidDoc *pDoc = (CSdidDoc *) GetDocument();
    ASSERT(pDoc) ;
    // get data set object
    QCStaticData *pDataSet =
    pDoc->GetDataSet ( 0 );
    HGOBJ hAxisX, hAxisY; // axes handles
    // define plot area
    WGSetPlotArea (0.15, 0.15, 0.9, 0.80, C_LIGHTGRAY) ;
    // Draw axes
    WGSetLineStyle(PS_SOLID, 1, C_BLACK) ;
    WGSetTextByName(C_BLACK, "Arial", 10, TEXT_BOLD);
    WGAutoAxes(pDataSet, AS_ROUNDCLOSE,
        INTF_ZERO, &hAxisX, &hAxisY, NULL, NULL);
    WGSetLineStyle(PS_SOLID,0,C_RED) ;
    // Plot Data
    WGLinePlot(pDataSet,FALSE,FALSE) ;
    // Title axes and graph
    WGTtitleAxis(hAxisX,POS_BELOW,"Time - msec") ;
    WGTtitleAxis(hAxisY,POS_BELOW,"Volts") ;
    WGSetTextByName (C_BLACK, "Arial",
        16, TEXT_BOLD | TEXT_ITAL) ;
    WGTtitleGraph("Sine Wave") ;
}
}
```



The example code segments above produced this graph in an MFC-AppWizard-GCL application.