

```

/*-----*
* File Name: ChangePlotFormat.c                                     *
* Creation: ER, 01/17/05                                           *
* Purpose: Programming Example                                     *
* Copyright (c) OriginLab Corp.2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010 *
* All Rights Reserved                                             *
*                                                                    *
* Modification Log:                                              *
*-----*/

#include <Origin.h>

////////////////////////////////////
// This example shows how to change the attributes of an existing plot.
// NOTE: It is assumed that a graph with a dataplot in layer1 is active.
//
void ChangePlotFormat ()
{
    // Declare graph layer as 1st layer of active graph page
    GraphPage gpg = Project.Pages();
    GraphLayer gly = gpg.Layers(0);
    if( !gly )
    {
        out_str("Active layer is not a graph layer!");
        return;
    }

    // Declare dataplot as first plot in layer and check validity
    DataPlot dp = gly.DataPlots(0);
    if( !dp )
    {
        out_str("No valid data plot found!");
        return;
    }

    // Get name of dataset associated with dataplot and report
    string strDatasetName = dp.GetDatasetName();
    printf("Dataset associated with the dataplot is: %s\n", strDatasetName);

    // Get the type of the dataplot and report
    int nPlotId = dp.GetPlotType();
    out_int("Plot type is ", nPlotId);

    // If plot is not of type line+symbol, then change the plot type
    // by adding the plot to with new plot type
    if( nPlotId != IDM_PLOT_LINESYMB )
    {
        Curve crv(strDatasetName);
        int nPlotIndex = gly.AddPlot(crv, IDM_PLOT_LINESYMB, GAP_ALLOW_DUPLICATE_COL);
        legend_update(gly); // Refresh legend for the newly added plot

        // Redefine data plot object
        dp = gly.DataPlots(nPlotIndex);
    }

    // Get format of the data plot to tree and output tree
    Tree trFormat;
    trFormat = dp.Curve;
    out_str("DataPlot format:");
    out_tree(trFormat);
    // The format tree will have branches such as Line, Symbol etc

    // Change the symbol type, color, and size
    dp.Curve.Symbol.Shape.nVal = 2; // filled circles
    dp.Curve.Symbol.EdgeColor.nVal = 1; // red color
    dp.Curve.Symbol.Size.nVal = 12; // size 12

    // One can also first copy part of the format to a tree,
    // then change the tree and set the tree back to apply
    // the new format.
    trFormat = dp.Curve.DropLines;
    // Dump tree to script window:
    out_str("Current format for drop lines is:");
    out_tree(trFormat);
    // Turn on vertical drop lines
    trFormat.Vertical.nVal = 1; // turn on vertical drop lines
    // Set tree back to apply format change
    dp.Curve.DropLines = trFormat;
}
////////////////////////////////////

```