

```

/*-----*
* File Name: ManipulateCurves.c *
* Creation: ER, 02/10/05 *
* Purpose: Programming Example *
* Copyright (c) OriginLab Corp.2003, 2004, 2005, 2006, 2007, 2008, 2009, 2010 *
* All Rights Reserved *
* *
* Modification Log: *
*-----*/

#include <Origin.h>

////////////////////////////////////
// This function shows how to create curve objects and then call a lower level
// function that works with the curve data.
//
void manipulate_curves()
{
    // Assumes worksheet is active with 4 cols set as XYXY and first two
    // columns have some data
    // Declare worksheet object
    Worksheet wks = Project.ActiveLayer();
    // Declare two curve objects, one for input based on first two cols
    // and one for output based on 3rd, 4th cols of wks
    Curve crvIn(wks, 0, 1);
    Curve crvOut(wks, 2, 3);
    if( !crvIn || !crvOut )
    {
        out_str("Could not declare curves!");
        return;
    }

    // Call lower level funtion to compute
    int iRet = my_curve_func(crvIn, crvOut, 100, 0.1);
    if( 0 != iRet )
        printf("Lower level function call returned error code: %d\n", iRet);
}

// This lower level function accepts reference two curvebase objects, one for
// input curve and one for output curve. The Y value of the output curve is
// set to dPar * Y of input curve. The X value of output curve is set to
// nPar + X as input curve. This is just an example to show how to access X and Y
// of the curves.
static int my_curve_func(curvebase& crvInput, curvebase& crvOutput, int nPar, double dPar)
{
    // Manipulating the curve directly is equivalent to manipulating the Y
    crvOutput = crvInput * dPar;

    // To manipulate the X, need to first attach datasets to X part of the curves
    Dataset dsXInput, dsXOutput;
    crvInput.AttachX(dsXInput);
    crvOutput.AttachX(dsXOutput);
    // Check validity - return error if could not attach X
    if( !dsXInput || !dsXOutput ) return 1;

    // Set X of output curve
    dsXOutput = dsXInput + nPar;

    // X and Y can be manipulated using datasets such as above.
    // If however it is desired to get the curve X, Y data into vectors and
    // manipulate the vectors instead, the following can be done:
    // Get copies of X, Y of input curve into vectors
    vector vecX, vecY;
    crvInput.CopyData(vecX, vecY);

    // Manipulate the vectors
    vecY *= dPar;
    vecX += nPar;

    // The vector values can be set back to output curve as follows:
    // For Y of curve, assign directly
    crvOutput = vecY;
    // For X, need to get X dataset of curve (which we did earlier)
    dsXOutput = vecX;

    return 0;
}

```

////////////////////////////////////