

```

/*-----*
* File Name: ReadScreenPoints.c *
* Creation: ER, 02/11/05 *
* 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 get x, y coordinates of points clicked by user
// on a graph layer.
// Number of points to be fetched is passed as parameter. Default is 2.
// So for 5 points, type this command in the script window:
// read_screen_points 5
//
void read_screen_points(int nPts = 2)
{
    // Declare vectors to hold coordinates
    vector vecX, vecY;
    // Call function to get points
    int iRet = get_screen_reader_points(nPts, vecX, vecY);
    if( iRet < 0 )
        printf("Function call to get screen points returned error: %d\n", iRet);

    printf("User clicked on %d points\n", iRet);
    if( 0 != iRet )
    {
        printf("X, Y coordinates of points are:\n");
        for(int ii = 0; ii < iRet; ii++)
            printf(" %d: X = %f, Y = %f\n", ii, vecX[ii], vecY[ii]);
    }
}

// This static function handles calling LabTalk to get user to click on the screen.
// Parameters:
// nPts: no. of points that user should click
// vecX: vector to hold x coordinates of points, passed by reference
// vecY: vector to hold y coordinates of points, passed by reference
// Return:
// -1: no. of points inappropriate
// >=0: no. of points user actually clicked
static int get_screen_reader_points(int nPts, vector& vecX, vector& vecY)
{
    // Return error if number of points not appropriate
    // Arbitrary upper limit of 99 - change as desired
    if( nPts < 1 || nPts > 99) return -1;

    // Run the LabTalk section in this file to give user control to click points
    string strThisFile = __FILE__;
    string strCMD;
    strCMD.Format("run.section(%s,GetScreenPts,%d)", strThisFile, nPts);
    LT_execute(strCMD);

    // On return, the LabTalk variable tmp_scr_count has number of points user clicked + 1
    double dCount;
    LT_get_var("tmp_scr_count", &dCount);
    int nCount = (int) (dCount - 1);
    if( 0 == nCount ) return 0;

    // Get values of x, y coordinates from LabTalk variables and delete the vars
    vecX.SetSize(nCount);
    vecY.SetSize(nCount);
    string strX, strY;
    double dX, dY;
    for(int ii = 1; ii <= nCount; ii++)
    {
        strX.Format("tmp_scr_pos_x%d", ii);
        strY.Format("tmp_scr_pos_y%d", ii);
        LT_get_var(strX, &vecX[ii - 1]);
        LT_get_var(strY, &vecY[ii - 1]);
        strCMD.Format("del -v %s", strX);
        LT_execute(strCMD);
        strCMD.Format("del -v %s", strY);
        LT_execute(strCMD);
    }
}

```

```

}

// Delete the count LabTalk variable
LT_execute("del -v tmp_scr_count;");

// Return the number of points clicked by user
return nCount;
}

// This section of LabTalk will be ignored by compiler since it is contained within an ifdef block.
// The section of script within this block can then be successfully called by a run.section LabTalk com
#ifdef LABTALK
[GetScreenPts]
// Number of points to get is passed as 1st parameter
// Select screen reader tool
dotool 2;

// Main loop to accept points
tmp_scr_count = 1;
def quittoolbox
{
// Check if user has clicked required number of points
if( 1 == tmp_scr_count )
done = 1;
if( %1 != tmp_scr_count )
return;
}

// Get points and store in temp string variables
def pointproc
{
tmp_scr_pos_x$(tmp_scr_count)=x;
tmp_scr_pos_y$(tmp_scr_count)=y;
if( tmp_scr_count >= %1 )
{
dotool 0;
// Set the variable to break the infinite loop
done = 1;
}
tmp_scr_count += 1;
}
// Set up an "infinite loop" so code waits for user to click points
for( done = 0; done == 0; )
{
// wait a while
sec -p 0.1;
}
#endif
////////////////////////////////////

```