

```
using System;
using System.Collections.Generic;
//using System.Text;
using System.Windows.Forms;
using System.IO;
using Autodesk.AutoCAD.ApplicationServices;
//using Autodesk.AutoCAD.DatabaseServices;
//using Autodesk.AutoCAD.EditorInput;
using Autodesk.AutoCAD.Runtime;
//using Autodesk.AutoCAD.PlottingServices;
//using Autodesk.AutoCAD.Interop;
//using Ini;

namespace QuickP
{
    public partial class clsQuickP
    {
        //public enum plot_scale
        //{
        //    // This enum is used to multiply the DWGSCALE to get the
        //    // (hopefully) proper scale.
        //    Extents,
        //    Scale,
        //    Half_Scale,
        //    Third_Scale,
        //    Quarter_Scale
        //}

        //This structure holds all the pertinent plot information
        public struct plot_info
        {
            //Info to write to log file
            public string directory;
            public string drawing;
            public string printer;
            public int HalfSize;
            public string PaperSize;
            public bool Portrait;
            public bool CenterPlot;
            public string CTBFile;
        }

        public string printing_queue;
        public plot_info file_to_plot = new plot_info();
        public bool drawings_waiting;
        public string dwgfilename;
        public bool Check;
        public int lwt;
        public int swt;

        //public void add_the_file(string filename) { }
        public bool run_autocad()
        {
            // This is where we will do all the processing of the file
            dwgfilename = file_to_plot.directory + file_to_plot.drawing;

            DocumentCollection dm = Autodesk.AutoCAD.ApplicationServices.Application.DocumentManager;
            //Document doc = dm.MdiActiveDocument;
            //Editor ed = doc.Editor;
            //Database db = doc.Database;

            try
            {
                // Open the drawing as read-only just in case someone is in the drawing
                dm.Open(dwgfilename, true);
            }
        }
    }
}
```

```
//Autodesk.AutoCAD.ApplicationServices.Application.SetSystemVariable("SDI", 0);
//Document doc = Autodesk.AutoCAD.ApplicationServices.DocumentCollectionExtension.Open(dm, ↵
dwgfilename, true);

    // Perform plotting actions here
    clsPlotting pltwdwg = new clsPlotting();
    pltwdwg.SimplePlot(file_to_plot);

    dm.MdiActiveDocument.CloseAndDiscard();

    //dm.CloseAll();
    //dm.Open(Properties.Settings.Default.default_dwg, false);
    //Autodesk.AutoCAD.ApplicationServices.Application.SetSystemVariable("SDI", 0);
    //doc.CloseAndDiscard();

    return true;
}
catch (System.IO.FileNotFoundException)
{
    return false;
}
catch (Autodesk.AutoCAD.Runtime.Exception Ex)
{
    Autodesk.AutoCAD.ApplicationServices.Application.ShowAlertDialog("The following exception ↵
was caught:\n" + Ex.Message);
    return false;
}
catch
{
    MessageBox.Show("Caught an error");
    return false;
}
}

public void CheckTheQueue()
{
    lstPrintQueue.Items.Clear();
    if (File.Exists(printing_queue)){
        if (ListDrawings(printing_queue) > 0)
            drawings_waiting = true;
        else
            drawings_waiting = false;
    }
    else
        drawings_waiting = false;
}

public int ListDrawings(string qfilename)
{
    StreamReader sr = new StreamReader(qfilename);
    while (!sr.EndOfStream)
    {
        try
        {
            string myString = sr.ReadLine();
            string[] queuestring = myString.Split(',');
            lstPrintQueue.Items.Add(queuestring[1]);
        }
        catch
        {
            lblStatus.Text = "Error";
            lstPrintQueue.Clear();
            sr.Close();
            return 0;
        }
    }
}
```

```
        sr.Close();
        //sr.Dispose();

        lblQCount.Text = Convert.ToString(lstPrintQueue.Items.Count);
        return lstPrintQueue.Items.Count;
    }

    public void Timer_Loop()
    {
        if (get_the_next_drawing())
        {
            if (run_autocad())
                success(true);
            else
                success(false);

            lblTimer.Text = Convert.ToString(swt);
        }
        else
        {
            if (!drawings_waiting)
                lblTimer.Text = Convert.ToString(lwt);
            else
                lblTimer.Text = Convert.ToString(swt);
        }
    }

    public void success(bool yn)
    {
        if (yn)
            lblStatus.Text = String.Concat(file_to_plot.directory, "\\",
SUCCCESSFUL");
        else
            lblStatus.Text = String.Concat(file_to_plot.directory, "\\",
UNSUCCCESSFUL");
    }

    public bool get_the_next_drawing()
    {
        int count;
        List<string> drawing_buffer = new List<string>();
        StreamReader sr = new StreamReader(printing_queue);

        count = 0;

        while (!sr.EndOfStream)
        {
            drawing_buffer.Add(sr.ReadLine());
            count += 1;
        }

        sr.Close();
        //sr.Dispose();

        //public string directory;
        //public string drawing;
        //public string printer;
        //public plot_scale HalfSize;
        //public string PaperSize;
        //public bool Portrait;
        //public bool CenterPlot;
        //public string CTBFile;
        if (count > 0)
        {
            StreamWriter wpq = new StreamWriter(printing_queue, false);
            wpq.WriteLine(drawing_buffer[0]);
            wpq.Close();
        }
    }
}
```

```
        try
        {
            string[] line = drawing_buffer[0].Split(',');
            file_to_plot.directory = line[0];
            file_to_plot.drawing = line[1];
            file_to_plot.printer = line[2];
            file_to_plot.HalfSize = /*(plot_scale)*/(Convert.ToInt32(line[3]));
            file_to_plot.PaperSize = line[4];
            file_to_plot.Portrait = Convert.ToBoolean(line[5]);
            file_to_plot.CenterPlot = Convert.ToBoolean(line[6]);
            file_to_plot.CTBFile = line[7];

            // Rewrite the queue minus the first line
            for (int i = 1; i < (drawing_buffer.Count); i++)
                wpq.WriteLine(drawing_buffer[i]);
            wpq.Flush();
            wpq.Close();
            //wpq.Dispose();

            drawings_waiting = true;
            return true;
        }
        catch
        {
            lblStatus.Text = "Error with the print queue.";
            // Rewrite the queue minus the first line
            for (int i = 1; i < (drawing_buffer.Count); i++)
                wpq.WriteLine(drawing_buffer[i]);
            wpq.Flush();
            wpq.Close();
            drawings_waiting = true;
            return false;
        }
    }
    else
    {
        lblStatus.Text = "No drawings are waiting.";
        drawings_waiting = false;
        return false;
    }
}
}
```