Autodesk University 2012

Using .NET Programming to Create New Possibilities with the AutoCAD® Core Console

Augusto Goncalves – Autodesk

CP3338 Starting with AutoCAD 2013, it is now possible use AutoCAD software from the Core Console, which can help us automate tasks with a more stable and controllable headless AutoCAD. This class will explore how to use and integrate the Core Console using .NET programming. Prior knowledge of .NET programming and the AutoCAD API are required.

Learning Objectives

At the end of this class, you will be able to:

- Use the AutoCAD Console with not programming skills
- Create custom commands for the console
- Speed up with threads
- Integrate with Windows APIs

About the Speaker

Augusto is member of Autodesk DevTech team since 2008 based at Sao Paulo office. He is Civil Engineer with a Master in Computer Science, also a specialist in AutoCAD, Civil3D, Revit and Inventor APIs.
augusto.goncalves@autodesk.com

Overview

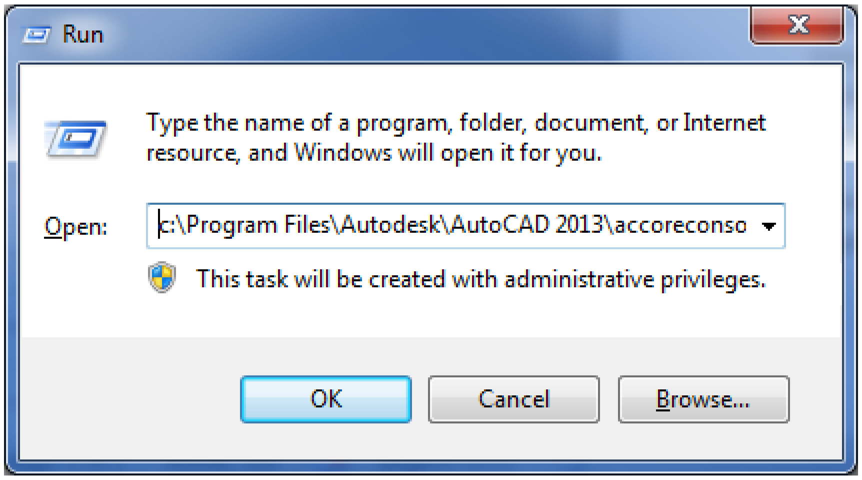
The AutoCAD Console feature can be understood as an ‘AutoCAD for developers’ version. It is there since version 2013 and is really fast. This class will present the basic and how use it with or without APIs.

AutoCAD Console with not API

It is possible to use it only with scripts. You might say that script is an API, but in fact we can create a script as simple as a sequence of commands. To create a script, simply open the Notepad, write a sequence of commands with all required parameters and save it with .scr extension.

The basic syntax is call **accoreconsole.exe** with ***/i*** and ***/s*** for file name and script file. It will launch, open the file and execute the sequence of commands determined by the script. This is just a start, not effective, but go to Windows *Start* button, then select *Run*, the dialog below will open. For a file, let’s say **c:\temp\myFile.dwg**, and a list of command as a script file at **c:\temp\myScript.scr**, try type (in a single line) at the Run dialog:

```
c:\Program Files\Autodesk\AutoCAD 2013\accoreconsole.exe
/i c:\folder\myFile.dwg
/s c:\temp\myScript.scr
```

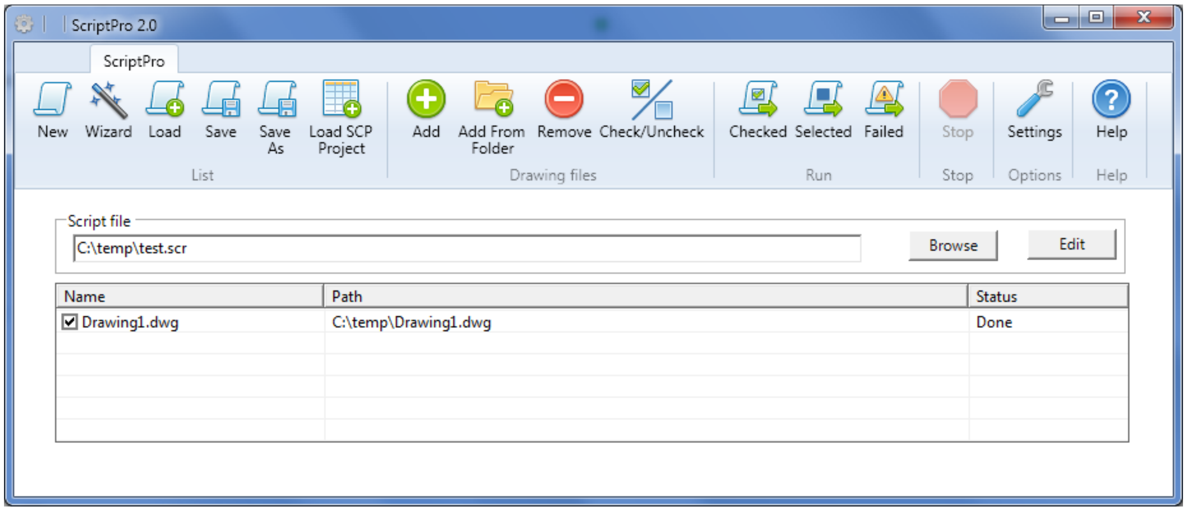


Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

c:\Program Files\Autodesk\AutoCAD 2013\accoreconso

This task will be created with administrative privileges.

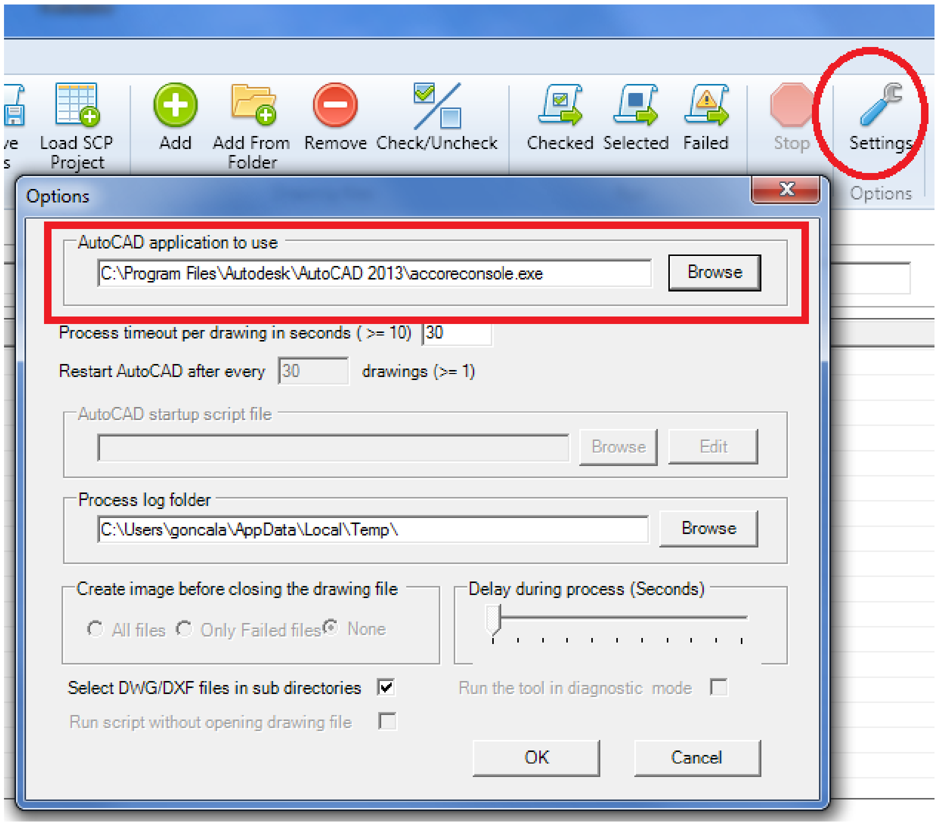
To improve that, there is a tool called ScriptPro 2.0, available for free at the Autodesk Labs (http://labs.autodesk.com/utilities/scriptpro/). Basically, this tool is a batch processing utility that applies a set of commands to multiple drawings. Simply specify a script file that contains the commands you want to run on a single drawing, and then use ScriptPro 2.0 to apply that script to as many drawings as you like.



Script file: C:\temp\test.scr

Name	Path	Status
✓ Drawing1.dwg	C:\temp\Drawing1.dwg	Done

And the new version released with AutoCAD 2013 support the **AcCoreConsole**. To use it, go to *Settings*, the click *Browse* and select the appropriate path. The image below demonstrates it, take a look. Later on this class we'll expand these features.



Options

AutoCAD application to use: C:\Program Files\Autodesk\AutoCAD 2013\accoreconsole.exe

Process timeout per drawing in seconds (>= 10) 30

Restart AutoCAD after every 30 drawings (>= 1)

AutoCAD startup script file

Process log folder: C:\Users\goncala\AppData\Local\Temp\

Create image before closing the drawing file: All files Only Failed files [✓ None]

Delay during process (Seconds)

Select DWG/DXF files in sub directories ✓

Run the tool in diagnostic mode ☐

Run script without opening drawing file ☐

Sure the script can contain a call to NETLOAD, the assembly path, and then the custom command. So, even a simple no-API approach can evolve to a more customized one. Let's move further.

AutoCAD Console with APIs

Now starting with programming languages, we open space for a lot of new and cool possibilities. First let's create samples with .bat files, and then move forward to full applications that take advantage of the **AcCoreConsole**.

Using .bat files

Sure this is not .NET programming, but is so easy and powerful that is worth it. Create a simple .bat file allow us to open **AcCoreConsole** for each file at a specific folder and execute a script file. The basic syntax for a .bat file is as follows: for each file at a specific folder, launch the console with this file and execute the script.

```
FOR %%f IN (c:\DwgFolder\*.dwg) DO
"C:\Program Files\Autodesk\AutoCAD 2013\accoreconsole.exe"
/i "%%f" /s "c:\MyScript.scr" /l en-US
```

Moving further, it is possible to include custom commands on the script, written in .NET, C++ or LISP.

Calling AcCoreConsole from API

The .NET Framework has the answer under **System.Diagnostics.Process** object. Create this object with the path of the .exe and the arguments. The arguments should include the .dwg file to be processed and the script that will be executed. Finally redirect the output and hide any window that may appear, so the execution is completely silent.

The following function summarizes how call it with .NET. Note how **CreateNoWindow** make it not visible with **UseShelExecute** property. Also, the **StartInfo**.**Arguments** captures the output of the command line so we can get the results. The full sample is available with this class.

```
public static string RunCommands(string fileName, string scriptFile)
{
  // no window and redirect the output
  Process process = new Process();
  process.StartInfo.UseShellExecute = false;
  process.StartInfo.RedirectStandardOutput = true;
  process.StartInfo.CreateNoWindow =true;

  // parameters to execute the script file
  process.StartInfo.FileName = AcadConsoleExePath;

  // build the parameters
  StringBuilder param = new StringBuilder();
  if (!string.IsNullOrWhiteSpace(scriptFile))
    param.AppendFormat(" /s \"{0}\"", scriptFile);
  if (!string.IsNullOrWhiteSpace(fileName))
    param.AppendFormat(" /i \"{0}\"", fileName);

  process.StartInfo.Arguments = param.ToString();
  string output = string.Empty;
  try
  {
    using (process)
    {
      // run it!
      process.Start();

      // read the output to return
      // this will stop this execute until AutoCAD exits
      StreamReader outputStream = process.StandardOutput;
      output = outputStream.ReadToEnd();
      outputStream.Close();
    }
  }
  catch (Exception ex)
  {
    output = ex.Message;
  }

  return output;
}
```

Using .NET Plug-ins

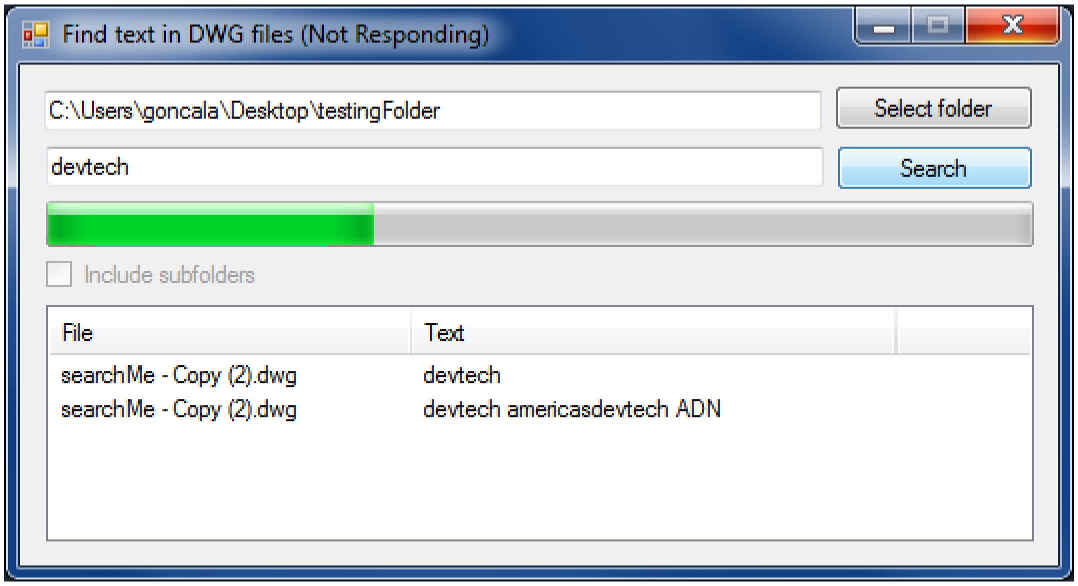
Develop for AutoCAD Console is basically the same as develop for regular AutoCAD, but as there is no User Interface, we cannot use any UI, except AutoCAD Editor command line. So write the code using just the **AcMgd** and **AcCoreMgd** references. Do not use **AcMgd** as this reference is used to create interface and it's not available on the console. Additionally, any Windows related interface is also not supported, such as Windows Forms and Windows Presentation Foundation (WPF).

The **FindText** sample demonstrate how a plug-in called **FindTextAcadModule** can be loaded and its command *FindText* be executed. The *RunCommand* method was developed for this class. The output result is stored for later use.

```
string[] files = Directory.GetFiles(txtFolder.Text);
foreach (string f in files)
{
  prgBar.PerformStep();
  Refresh();

  string res = AcadConsoleProcess.RunCommands(
    f,
    "NETLOAD",
    GetPathForDLL("FindTextAcadModule.dll"),
    "FindText",
    txtTextToSearch.Text);
}
```

During the execution, it will launch **AcCoreConsole** for each file at a specific folder and use the custom command to search DBText and MText that contains a given text. Below is a part of the code followed by the sample form during execution.



C:\Users\goncala\Desktop\testingFolder

devtech

File	Text
searchMe - Copy (2).dwg	devtech
searchMe - Copy (2).dwg	devtech americasdevtech ADN

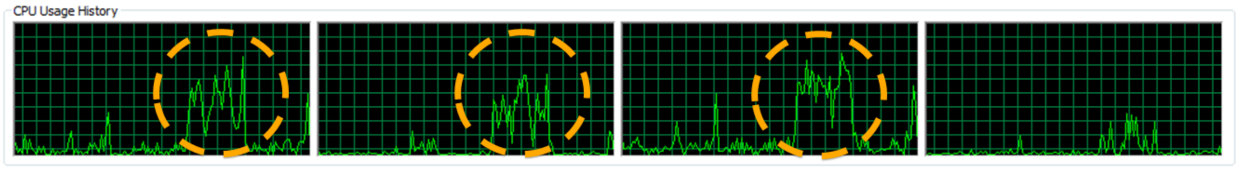
Improving the speed

During the execution of the samples demonstrated until this point, you may notice that only a small part of the CPU capability is used (depending on the machine). Basically we launch one **AcCoreConsole** and wait for its response, then start the second, and so forth. How can this be improved?

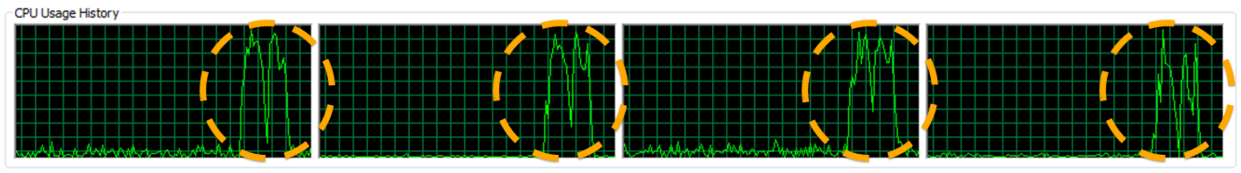
<u>Important:</u> AutoCAD is a single thread application. It is not safe use multithread while developing for it and any success on that may be a matter of luck.

That said, here we'll not do multithread inside AutoCAD, but in fact start several **AcCoreConsole** instances. That can be interesting as most of modern machines are multi core, so we're looking for a way to use them all.

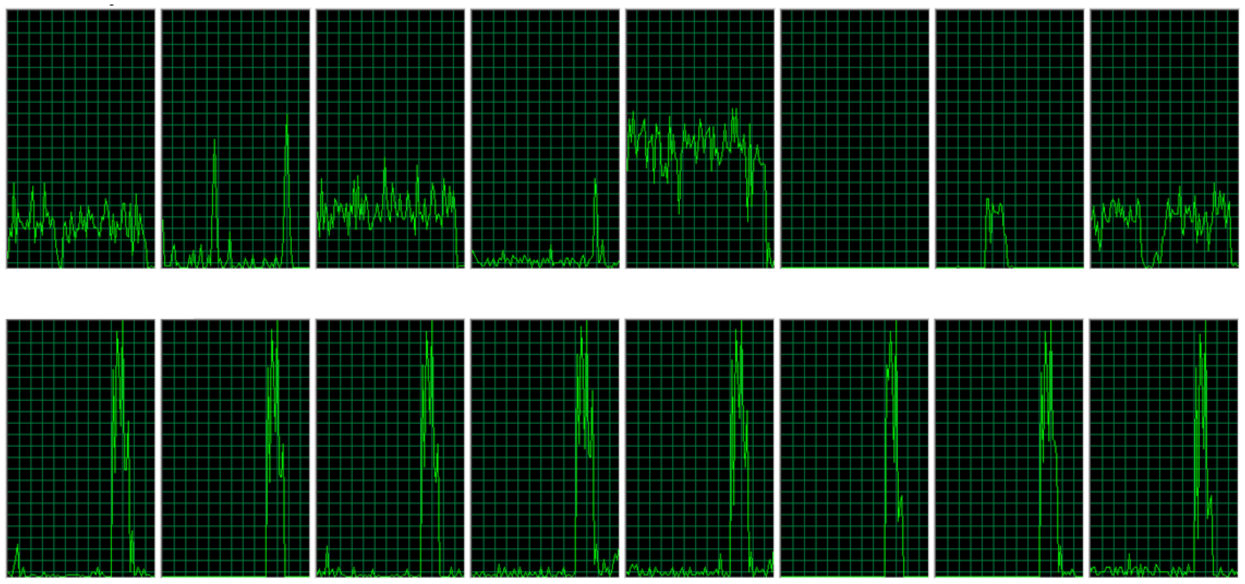
The sample project **Threads** is prepared to open all files inside a folder and process them with a script file. It can process the files sequentially, as described on the image below. Notice that on a 4-core machine, only 3 are used, but not at the limit.



Now the **Threads** sample have another option: run in parallel. Below is an image of the same process (same files and script of the previous sequential sample). Note that all four cores are used almost at the limit. Also, this approach executes around 30% to 50% faster than in sequence. This is a pure empirical result and varies with several variables, such as memory and other tasks being executed. For a better result, run this sample with files you commonly use.



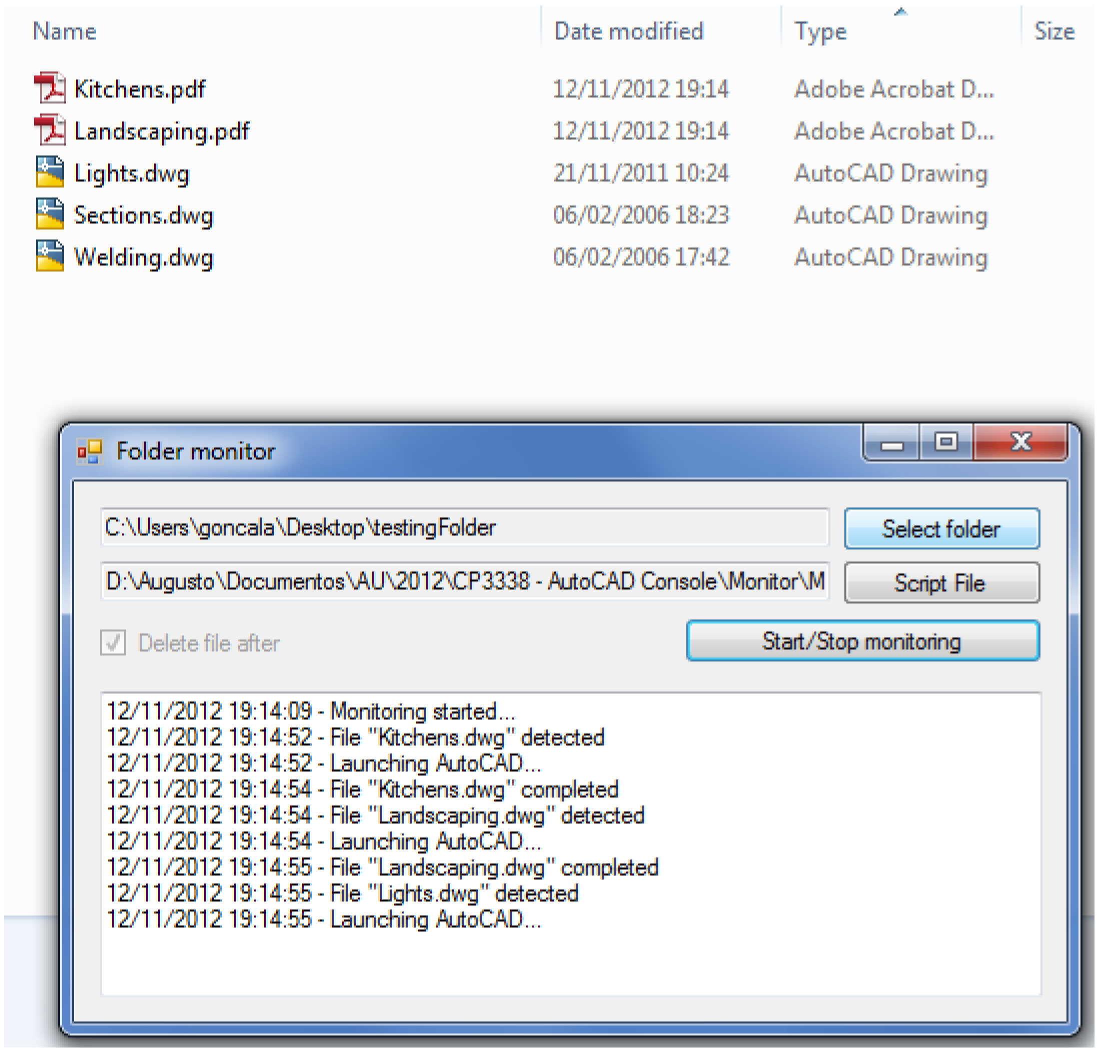
Just for fun, I tried this sample with more files on an 8-core machine. The results are very similar, check out below: sequential on the top, parallel on the bottom. Keep in mind that executing several processes in parallel may use the entire machine, which may make the system unresponsive during the execution.



Integrating with Windows APIs

Now it is possible integrate with the system so we can have more interesting results. The Monitor sample project uses .NET FileSystemWatcher object to listen to changes on a specific folder. When a new file is placed on the folder, it will launch **AcCoreConsole** and execute a script for that file, in this sample, convert to PDF.

The image below shows this sample in action. Note that each new .dwg file will trigger a new instance of the console, which will generate the PDF and finally erase the .dwg. Keep in mind that it may open several instances of the console (multithread by Windows API).



Name	Date modified	Type	Size
Kitchens.pdf	12/11/2012 19:14	Adobe Acrobat D...	
Landscaping.pdf	12/11/2012 19:14	Adobe Acrobat D...	
Lights.dwg	21/11/2011 10:24	AutoCAD Drawing	
Sections.dwg	06/02/2006 18:23	AutoCAD Drawing	
Welding.dwg	06/02/2006 17:42	AutoCAD Drawing	

C:\Users\goncala\Desktop\testingFolder

D:\Augusto\Documentos\AU\2012\CP3338 - AutoCAD Console\Monitor\M

[✓] Delete file after

```
12/11/2012 19:14:09 - Monitoring started...
12/11/2012 19:14:52 - File "Kitchens.dwg" detected
12/11/2012 19:14:52 - Launching AutoCAD...
12/11/2012 19:14:54 - File "Kitchens.dwg" completed
12/11/2012 19:14:54 - File "Landscaping.dwg" detected
12/11/2012 19:14:54 - Launching AutoCAD...
12/11/2012 19:14:55 - File "Landscaping.dwg" completed
12/11/2012 19:14:55 - File "Lights.dwg" detected
12/11/2012 19:14:55 - Launching AutoCAD...
```

And you? What other Windows APIs can be used here? System activities can be monitored or triggered by the user to launch the console to execute tasks. Note that most of the simple process are taking just a few seconds, which is a good user experience.

Further reading

Some additional information you might consider useful:

- Through the Interface blog - http://through-the-interface.typepad.com
  Kean Walmsley's .NET focused blog includes several example codes for .NET

- AutoCAD.NET Developer's Guide - http://www.autodesk.com/autocad-net-developers-guide

.NET documentation with .NET (C# and VB.NET) samples with the equivalent VBA code

- AutoCAD Developers Center – http://www.autodesk.com/developautocad
  Training material, recorded presentations, and our AutoCAD .NET Wizards.

- Discussion Groups - http://discussion.autodesk.com/forums/category.jspa?categoryID=8

- Information about the Autodesk Developer Network – http://www.autodesk.com/joinadn
  ADN members can ask unlimited API questions through our DevHelp Online interface

- API Training – http://www.autodesk.com/apitraining
  Information about upcoming training classes and webcasts, also download of webcasts

- Watch out for our regular ADN DevLab events. DevLab is a programmers' workshop (free to ADN and non-ADN members) where you can come and discuss your AutoCAD programming problems with the ADN DevTech team.

Conclusion

Thank you for attending this session on AutoCAD Console with .NET. I hope you found the class enjoyable and valuable. The collection of samples is meant to be used with this handout and the presentation.

We started with basic steps of the console, then how integrate with .NET plug-ins and custom commands, and how speed up the process with threads. Finally a sample on how integrate this with Windows APIs.

The Console is a great new feature, take advantage of it! Good luck!