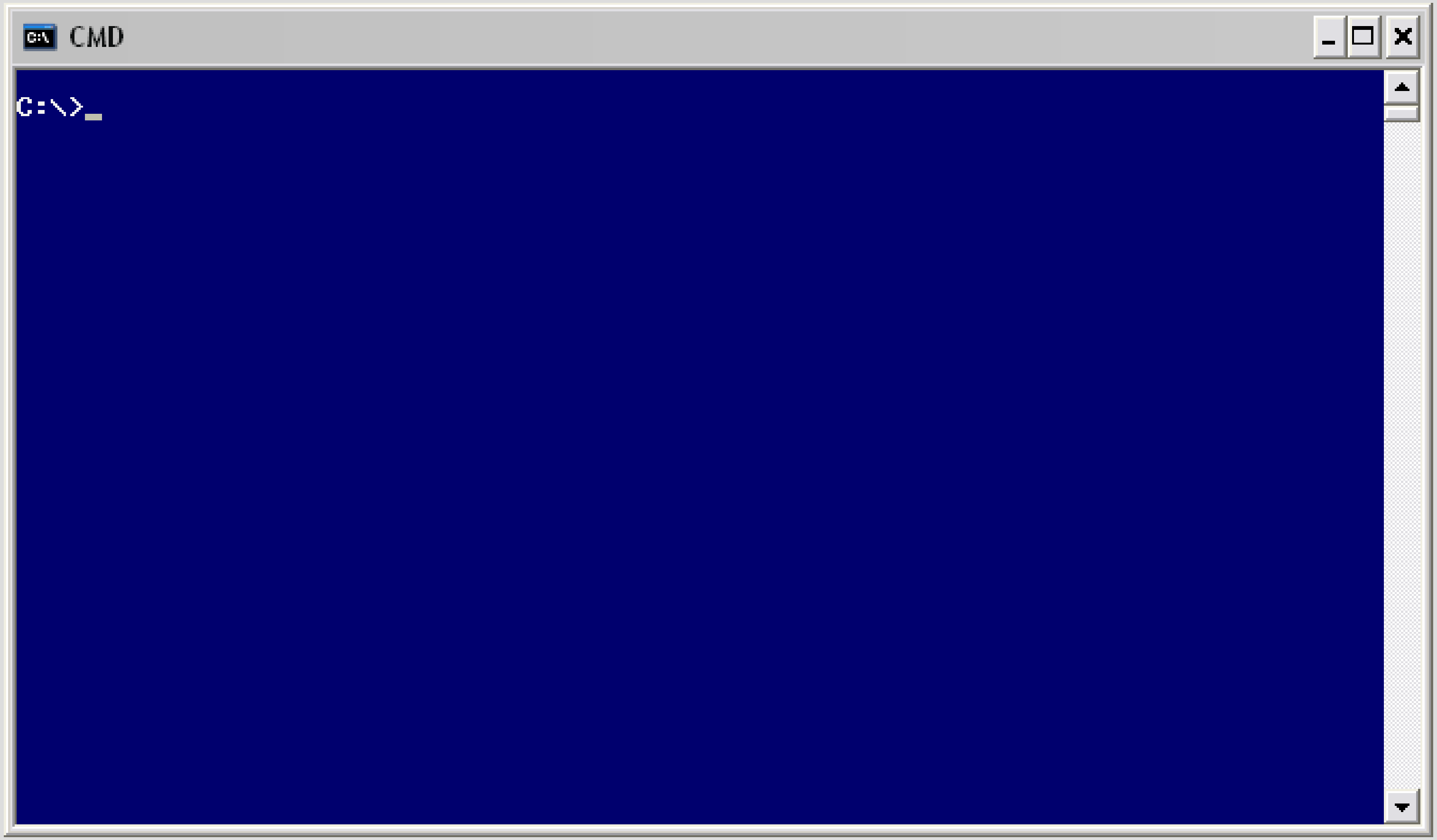
Applications of the win32console Module

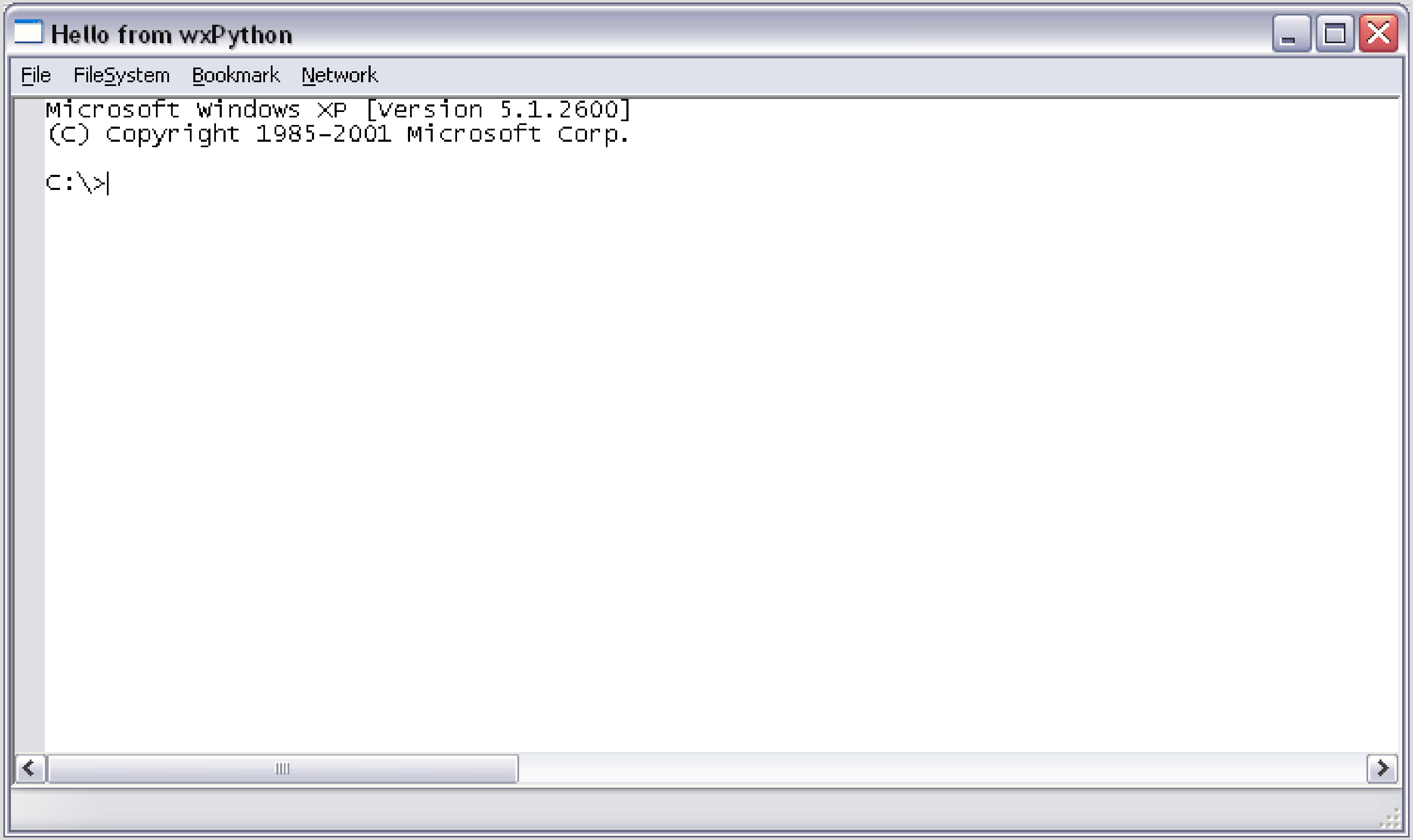
A Merging of Character-mode Applications and Graphical User Interfaces

A Basic Command Shell



```
C:\>_
```

A wxPython Console Wrapper



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\>
```

The win32console module

- Microsoft win32 console API
  - MSDN ms682010
  - Low level API to connect to an actual console
- Python extension module: win32console.pyd
  - Functions such as AllocConsole, ReadConsoleOutputCharacter, WriteConsoleInput, ...
  - Objects such as PyCOORDType, PyINPUT_RECORDType, ...

pyconsole.py

- A simple “Pythonic” wrapper
  - Abstracts away complexity of win32console.pyd
  - Treats console memory buffer as a kind of scratch-pad for interprocess communication
- Implemented as a high level class

```
class ConsoleProcess:
    def __init__ (self, cmd_line,
        console_update=None,
        console_update_many=None,
        console_process_end=None,
        ...
    def write (self, text):
        ...
    def writeline (self, text):
        ...
```

Simple Example

```
cmd = pyconsole.ConsoleProcess
  ('cmd.exe', console_update=output)
```

Multiplexing processes

```
import time, pyconsole

class SampleConsoleProcess (pyconsole.ConsoleProcess):
    def __init__ (self, cmd_line):
        pyconsole.ConsoleProcess.__init__ (self, cmd_line,
            console_update=self.console_update,
            console_process_end=self.console_process_end)
        self.running = True

    def console_update (self, x, y, text):
        print 'output:', text.strip()

    def console_process_end (self):
        self.running = False
```

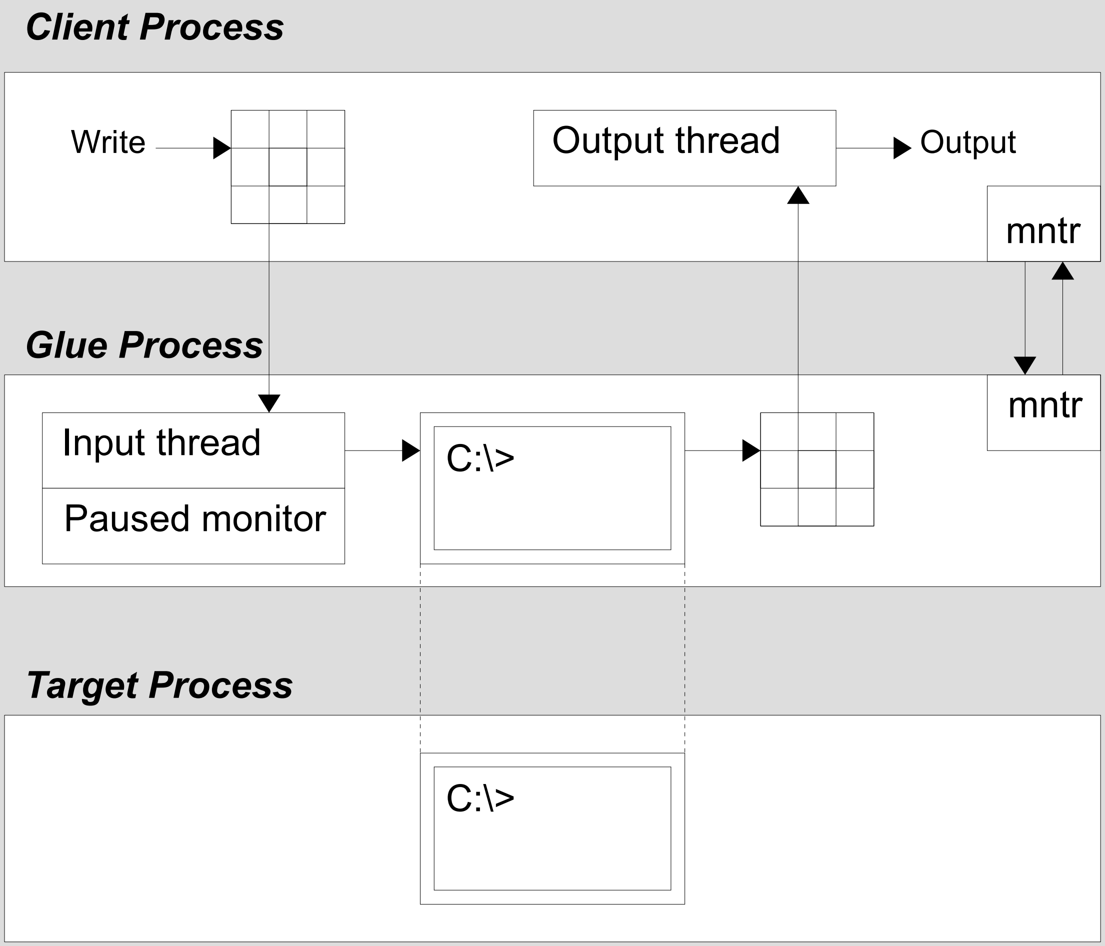
Multiplexing processes, 2

```
p1 = SampleConsoleProcess('python lrpwio.py One')
p2 = SampleConsoleProcess('python lrpwio.py Two')
p3 = SampleConsoleProcess('python lrpwio.py Three')

while p1.running or p2.running or p3.running:
    time.sleep (0.1)
```

Generator Approach

```
import time, Queue, pyconsole

def ConsoleProcessGenerator (cmd_line):
    queue = Queue.Queue()
    class CP (pyconsole.ConsoleProcess):
        def __init__ (self, cmd_line):
            ...

    process = CP (cmd_line)
    while process.running:
        yield queue.get()

for line in ConsoleProcessGenerator('python lrpwio.py Queue'):
    print 'output:', line
```

Console Process Interaction



Client Process

Write

Output thread

Output

mntr

Glue Process

mntr

Input thread

Paused monitor

C:\>

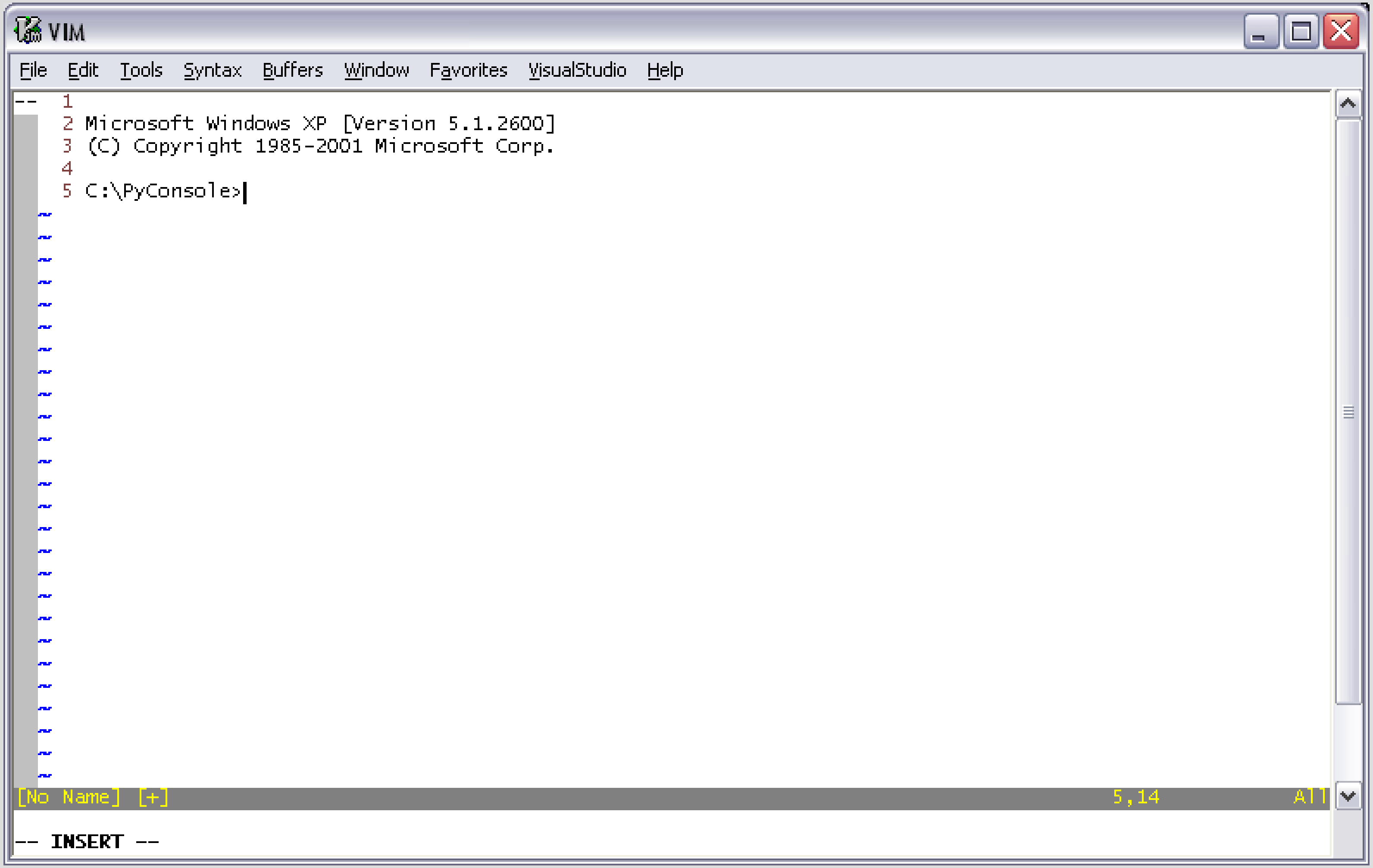
Target Process

C:\>

Design Challenges

- Hooking windows kernel with python.exe
- No virtual coordinate system
- Need to clear screen
- Synthetic pause keypress for flow control
- Batching updates/output

Vim Demo

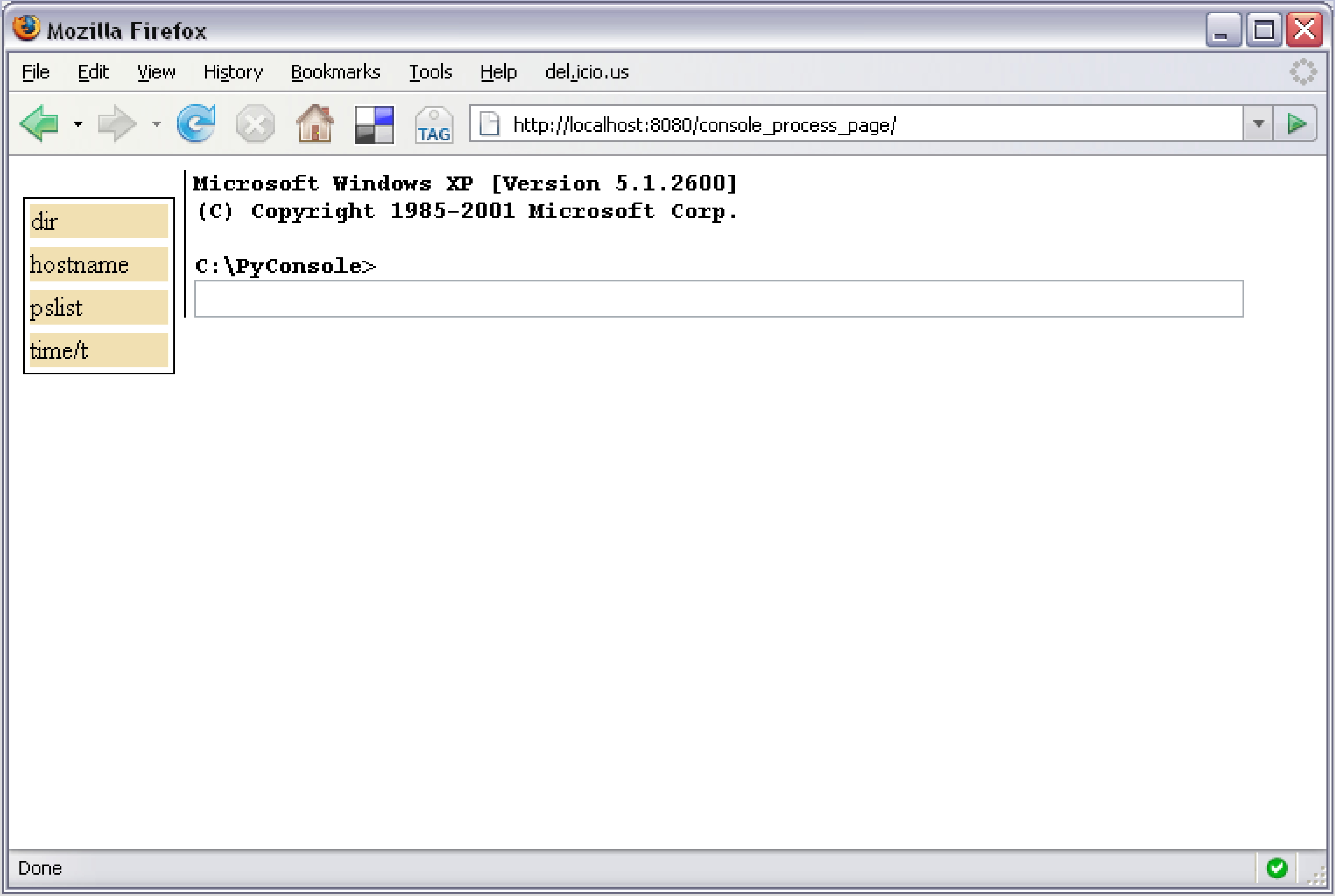


```
--
   Microsoft Windows XP [Version 5.1.2600]
   (C) Copyright 1985-2001 Microsoft Corp.

   C:\PyConsole>
[No Name] [+]                                             5,14     All

-- INSERT --
```

Web 2.0

- The "Divmod Nevow" python web framework
  - Also called "Live Page"
  - Has advanced Ajax and Comet capabilities
    - Ajax - the ability for a browser to initiate actions on a web server
    - Comet - the ability for a server to initiate actions in a web browser
- Built on top of the python Twisted framework
  - Simplifies networking
  - Concept of "Deferreds"
- Consists of python and javascript components

Nevow Demo


Mozilla Firefox
File Edit View History Bookmarks Tools Help del.icio.us
TAG
http://localhost:8080/console_process_page/
dir
hostname
pslist
time/t
Done

```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\PyConsole>
```

Influences

- Emacs embedded shell
- Python pyreadline module
- Console2 project

Future Directions

- Stability
- Optimization
- Command line handling
- Cursor control
- Screen attributes such as colors
- Interfacing to legacy applications
- Unix/Linux based solution