Advanced cx_Freeze - Using a setup.py File

WE'LL COVER THE FOLLOWING

- Wrapping Up

First off we need a script to use. We will use the wxPython form example from the previous chapters.

```
import wx

class DemoPanel(wx.Panel):
    """"""

    def __init__(self, parent):
        """Constructor"""
        wx.Panel.__init__(self, parent)

        labels = ["Name", "Address", "City", "State", "Zip",
                  "Phone", "Email", "Notes"]

        mainSizer = wx.BoxSizer(wx.VERTICAL)
        lbl = wx.StaticText(self, label="Please enter your information here:")
        lbl.SetFont(wx.Font(12, wx.SWISS, wx.NORMAL, wx.BOLD))
        mainSizer.Add(lbl, 0, wx.ALL, 5)
        for lbl in labels:
            sizer = self.buildControls(lbl)
            mainSizer.Add(sizer, 1, wx.EXPAND)
        self.SetSizer(mainSizer)
        mainSizer.Layout()

    def buildControls(self, label):
        """
        Put the widgets together
        """
        sizer = wx.BoxSizer(wx.HORIZONTAL)
        size = (80,40)
        font = wx.Font(12, wx.SWISS, wx.NORMAL, wx.BOLD)

        lbl = wx.StaticText(self, label=label, size=size)
        lbl.SetFont(font)
        sizer.Add(lbl, 0, wx.ALL|wx.CENTER, 5)
        if label != "Notes":
            txt = wx.TextCtrl(self, name=label)
        else:
            txt = wx.TextCtrl(self, style=wx.TE_MULTILINE, name=label)
```

```
        sizer.Add(txt, 1, wx.ALL, 5)
        return sizer

class DemoFrame(wx.Frame):
    """
    Frame that holds all other widgets
    """

    def __init__(self):
        """Constructor"""
        wx.Frame.__init__(self, None, wx.ID_ANY,
                          "Py2Exe Tutorial",
                          size=(600,400)
                          )
        panel = DemoPanel(self)
        self.Show()

if __name__ == "__main__":
    app = wx.App(False)
    frame = DemoFrame()
    app.MainLoop()
```

Now let's create a **setup.py** file in the cx_Freeze style:

```
# setup.py
from cx_Freeze import setup, Executable

setup(
    name = "wxSampleApp",
    version = "0.1",
    description = "An example wxPython script",
    executables = [Executable("sampleApp.py")]
    )
```

As you can see, this is a pretty simple one. We import a couple classes from cx_Freeze and pass some parameters into them. In this case, we give the setup class a name, version, description and Executable class. The Executable class also gets one parameter, the script name that it will use to create the binary from.

Alternatively, you can create a simple setup.py using cx_Freeze's quickstart command (assuming it's on your system's path) in the same folder as your code:

> cxfreeze-quickstart

> To get the setup.py to build the binary, you need to do the following on

the command line:

```
python setup.py build
```

After running this, you should end up with the following folders: **buildexe.win32-2.7**. Inside that last folder I ended up with 15 files that total 16.6 MB. When you run the sampleApp.exe file, you will notice that we've screwed something up. There's a console window loading in addition to our GUI! To rectify this, we'll need to change our setup file slightly. Take a look at our new one:

```
from cx_Freeze import setup, Executable

exe = Executable(
    script="sampleApp.py",
    base="Win32GUI",
    )

setup(
    name = "wxSampleApp",
    version = "0.1",
    description = "An example wxPython script",
    executables = [exe]
    )
```

First off, we separated the Executable class from the setup class and assigned the Executable class to a variable. We also added a second parameter to the Executable class that is key. That parameter is called **base**. By setting **base="Win32GUI"**, we are able to suppress the console window. The documentation on the cx_Freeze website shows the many other options that the Executable class takes.

Wrapping Up

Now you should know how to create binaries with cx_Freeze. It's pretty easy to do and it ran a lot faster than bbfreeze did in my testing. If you have the need to create binaries for both Python 2.x and 3.x on all major platforms, then this is the tool for you!