Objektorientierung mit Python

OO Grafik mit wxPython

OO Grafik mit wxPython

- Zu Python gibt es mehrere Grafik-Toolkits
- Eine Erläuterung findet man im Python-Buch von Galileo-Computing
  [ISBN 978-3-8362-1110-9]
  das man auch als e-book herunterladen kann.
  [http://www.galileo-computing.de]
- Dort sind etwas ausführlicher beschrieben
  - Tkinter
  - PyGtk
  - PyQt

OO Grafik mit wxPython

In diesem Workshop soll deutlich werden, weshalb ich mich für

wxPython

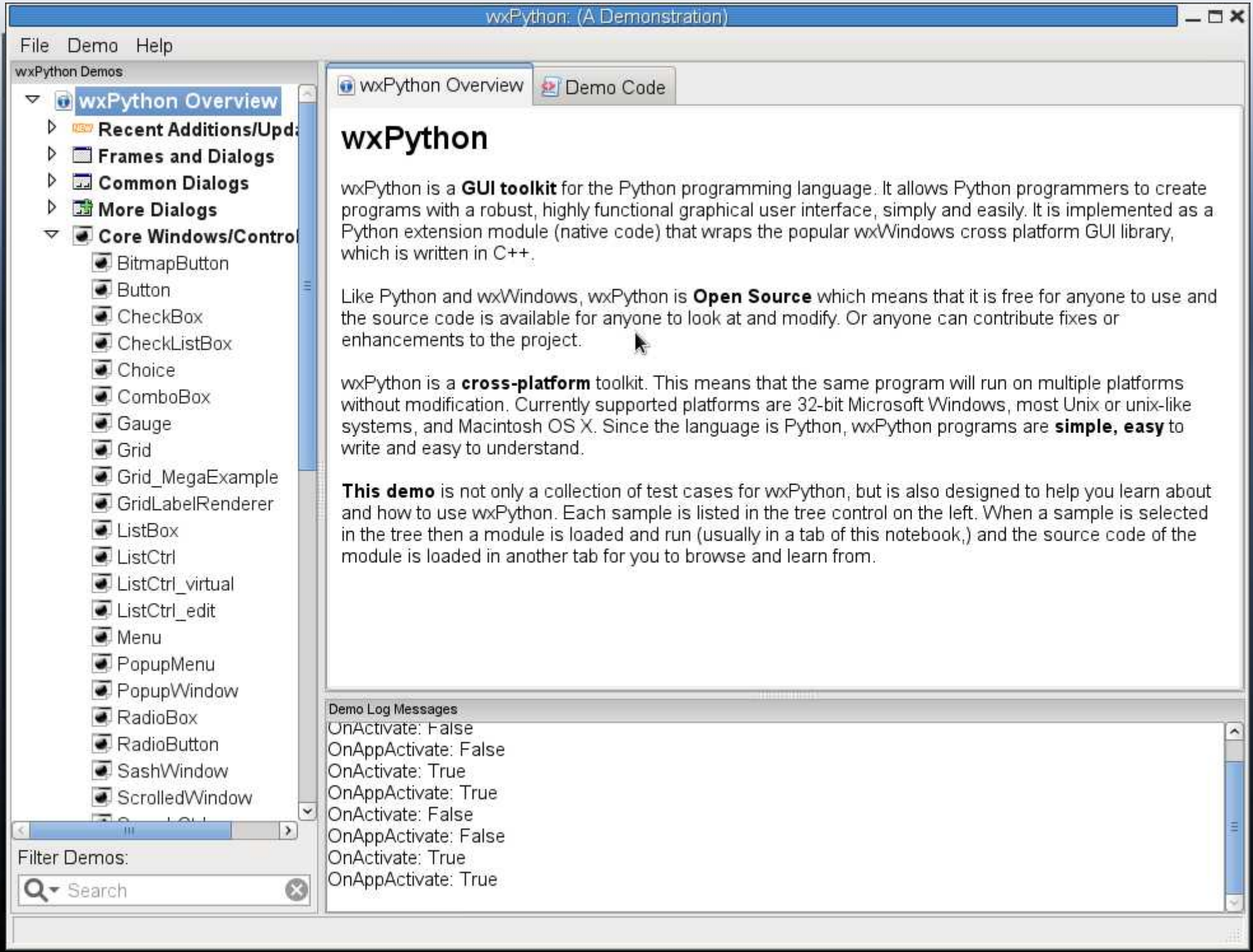
entschieden habe.

OO Grafik mit wxPython

- Für Windows Herunterladen von http://wxpython.org/
  - Passende Version zur eigenen Python-Version verwenden [Python 3 geht nicht!]
  - Installation möglichst in den Ordner <Python>/Lib/sitepackages
- Unbedingt auch die wxPython docs und demos herunterladen und installieren
- linux-Distributionen installieren beide in der Regel in die richtigen Verzeichnisse.

OO Grafik mit wxPython

- wxPython Demos parallel zu IDLE starten.
- Wir arbeiten nicht im Abschnitt "using images", da wir uns bei der OO nicht mit Pixelgrafik, sondern mit Vektorgrafik beschäftigen.
- Wir benötigen "GraphicsContext"
  [bei mir im Abschnitt Recent Additions/Updates]


wxPython: (A Demonstration)
File
Demo
Help
wxPython Demos
wxPython Overview
Recent Additions/Upda
Frames and Dialogs
Common Dialogs
More Dialogs
Core Windows/Control
BitmapButton
Button
CheckBox
CheckListBox
Choice
ComboBox
Gauge
Grid
Grid_MegaExample
GridLabelRenderer
ListBox
ListCtrl
ListCtrl_virtual
ListCtrl_edit
Menu
PopupMenu
PopupWindow
RadioBox
RadioButton
SashWindow
ScrolledWindow
Filter Demos:
Search
wxPython Overview
Demo Code
wxPython
wxPython is a GUI toolkit for the Python programming language. It allows Python programmers to create programs with a robust, highly functional graphical user interface, simply and easily. It is implemented as a Python extension module (native code) that wraps the popular wxWindows cross platform GUI library, which is written in C++.
Like Python and wxWindows, wxPython is Open Source which means that it is free for anyone to use and the source code is available for anyone to look at and modify. Or anyone can contribute fixes or enhancements to the project.
wxPython is a cross-platform toolkit. This means that the same program will run on multiple platforms without modification. Currently supported platforms are 32-bit Microsoft Windows, most Unix or unix-like systems, and Macintosh OS X. Since the language is Python, wxPython programs are simple, easy to write and easy to understand.
This demo is not only a collection of test cases for wxPython, but is also designed to help you learn about and how to use wxPython. Each sample is listed in the tree control on the left. When a sample is selected in the tree then a module is loaded and run (usually in a tab of this notebook,) and the source code of the module is loaded in another tab for you to browse and learn from.
Demo Log Messages
OnActivate: False
OnAppActivate: False
OnActivate: True
OnAppActivate: True
OnActivate: False
OnAppActivate: False
OnActivate: True
OnAppActivate: True

OO Grafik mit wxPython

- Die Lasche *Demo Code* zeigt den vollständigen Python-Code dieses Anwendungsfensters.
- Mit copy and paste holen wir den Code in das Editorfenster von Python und versuchen ihn mit F5 auszuführen.

OO Grafik mit wxPython

- Das Ausführen schlägt natürlich fehl, da sich viele Anweisung allein auf die Integration in das Demoprogramm beziehen.
- Bevor wir jetzt fortfahren, versuchen wir erst einmal ein einfaches Fenster zu erzeugen. Abschnitt:
  - Frames und Dialogs
    - Frame

OO Grafik mit wxPython

- Abschnitt:
  Frames und Dialogs
  Frame
- Die Probleme sind die selben, das Programm ist aber überschaubarer:
- Es enthält eine Klasse
  - MyFrame, die von wx.Frame abgeleitet ist
  - TestPanel, die von wx.Panel abgeleitet ist und als Inhaltsfläche für das Demo-Projekt dient.

OO Grafik mit wxPython

- Es wird nun klar, was zu tun ist:
- Es muss ein MyFrame-Objekt erzeugt werden, die Teile für die Panelklasse werden nicht benötigt.
- Die Versuche führen schließlich auf eine Fehlermeldung, dass ein Application-Objekt benötigt wird.
- An dieser Stelle ist nun zusätzliche Hilfe notwendig, die man nicht im Demo-Tool findet.

OO Grafik mit wxPython

```
class MyApp(wx.App):
    def OnInit(self):
        fenster = MyFrame(None, -1, "Test")
        fenster.Show(True)
        return True

if __name__ == '__main__':
    app = MyApp(redirect=False)
    app.MainLoop()
```

OO Grafik mit wxPython

- <<<ggf. zur Übung:
  Bauen Sie mit Hilfe der demo
  - einen Label [StaticText] und
  - ein Textfeld [TextCtrl]

  in die einfache Anwendung ein.>>>
- Nach dieser Hilfe sollte es gelingen, Schritt für Schritt die Grafik zu integrieren.