

Desarrollar aplicaciones desktop con Python

¿Por qué y cómo hacerlas?

Henry Tong – Software Architect/Backend Developer en Patagonian
PyDay Neuquén

Haciendo aplicaciones desktop

- ¿Por qué?
- ¿Cómo?
- Ejemplos

¿Por qué hacer aplicaciones desktop?

- Necesidades especiales (Música, IA, Machine Learning, Computer Vision, videojuegos, seguridad).
- Mercados alternativos: Ubuntu shop, gnome app store, Elementary app store, flatpack app store, Pop app store, ...
- Mejor accesibilidad (discapacidad visual, motora, etc...)

Referencias de interés

Accessibility design is good for business (Mejorar la marca, llegar a mas mercados, minimizar riesgo legal por leyes de accesibilidad...)	https://www.w3.org/WAI/business-case/
React Native para... desktop y xbox	https://github.com/microsoft/react-native-windows
Linux se utiliza (y se usará) en cada vez más en otras plataformas (y esto requerirá aplicaciones “desktop” para ellas)	Raspberry PI Chome OS tablets/laptops Celulares no-Android: https://puri.sm/products/librem-5/
Aumenta demanda de uso de linux de escritorio (no todos pueden migrar de window XP/7 a windows 10...)	https://techqoon.net/2019/05/01/demand-for-enterprise-ubuntu-linux-on-the-rise/

Tipos de librería gráfica/UI

Tipo “Controles UI”

Usado principalmente en apps “clásicas” o de negocio, como utilitarios. Donde es más útil tener un conjunto de componentes UI para reusar.

Tipo “Renderizado”

Usado cuando se requiere un control a nivel de píxeles (Juegos, visualizaciones 3D)

Tipo “Híbrido”

Si quieres usar html para el UI y python para la lógica detrás (estilo Electron)

QT

Ejemplo:
Kubuntu
(Ubuntu con escritorio KDE)

GTK

Ejemplo:
Ubuntu con escritorio gnome

También:

wxWidgets
Tkinter

Kivy, PyGame,
Panda3D, RenPy

Python CEF
Python EEL
Qt+WebView

Lo mejor de front + lo mejor de back



Detalles de librerías Python para UI

CEF Python Python EEL QT+WebView	https://github.com/cztomczak/cefpython https://github.com/samuelhwilliams/Eel EEL/CEF: Ok para cosas simples. Cuidado: CEF usa un browser viejo. QT+WebView: Recomendable para aplicaciones de mayor complejidad.
PyQT	https://riverbankcomputing.com/software/pyqt/ Maduro con muchos años. Muchos libros y tutoriales Cuidado: Licencia GPL para aplicaciones comerciales
QT for python	https://www.qt.io/qt-for-python Reciente (lanzado el 2018). Poca documentación (Pero puedes usar la documentación de PyQt) Es gratuito para aplicaciones comerciales
Tkinter / wxPython	https://wxpython.org Tkinter está incluido por defecto en Python Fáciles, populares, mucha documentación. Uso libre. Cuidado: Tkinter no genera aplicaciones accesibles
GTK	https://pygobject.readthedocs.io Documentación y apoyo de la comunidad GNOME Si tu target es exclusivamente Linux va genial, sino cuidado con GTK para Windows.

Editores visuales

The image displays three different visual editors for creating user interfaces:

- GTK tiene: Glade** (top left): Shows the 'gedit-preferences-dialog.ui' window with tabs for View, Editor, Font & Colors, and Plugins. The Font & Colors tab is active, showing settings for the editor font (Sans, size 12) and a color scheme.
- QT tiene: Qt Creator** (top right): Shows the 'mainwindow.ui - hellomobile - Qt Creator' window. It features a central design view with a grid layout containing a label 'Hello friend!' and a button 'You can click the button above.'. The left sidebar shows a list of widgets like Vertical Layout, Horizontal Layout, Grid Layout, Form Layout, Spacers, and Buttons. The right sidebar shows the Object Tree and the Object Properties panel.
- wx tiene: wxFormBuilder** (bottom): Shows the 'wxMCVE - wxFormBuilder v3.9.0' window. It features a central design view with a grid layout containing a label 'Hello friend!' and a button 'You can click the button above.'. The left sidebar shows the Object Tree with a tree structure: Test: Project > MyFrame: Frame > topSizer: wxBoxSizer > m_radioBox1: wxRadioBox. The right sidebar shows the Component Palette and the Object Properties panel.

GTK tiene: Glade
QT tiene: Qt Creator
wx tiene: wxFormBuilder

Extras

- Base de datos embebida: Sqlite, tinydb...
- Distribución: py2exe, pyinstaller, cx_freeze..

Ejemplos: Vamos al código!