MAKE YOUR APPLICATION MULTI-PLATFORM!

Linux App Summit 2021
Dan Yeaw & Arjan Molenaar
WHAT ARE WE TALKING ABOUT

1. Why go multi-platform
2. The case: Gaphor
3. Building for Linux (Flatpak / AppImage)
4. Building for Windows
5. Building for macOS
6. Take aways
A VIBRANT COMMUNITY

We want the apps we build to be useful for others!

- Users who get value out of using them 🧑‍💻
- Diverse contributors who want to make them better 🔧
Desktop Market Share

- Windows: 75.6%
- macOS: 16.5%
- Linux-based: 4.06%
- Unknown: 3.89%

Developer's Primary OS

- Windows: 45.8%
- macOS: 27.5%
- Linux-based: 26.6%

Sources:
- https://gs.statcounter.com/os-market-share/desktop/
- https://insights.stackoverflow.com/survey/2020
WHY GO MULTI-PLATFORM

1. Broader & more inclusive user base
2. Helps introduce people to Open Source 🤝
3. More future proof 🧪
4. Improves adaptability - not bound to a specific OS
WHAT ARE WE TARGETING

- All major desktop platforms: Windows, macOS, Linux
- Automate as much as possible🤖
- No scary warnings during install⚠️
THE CASE: GAPHOR
SETUP FOR SUCCESS
PAINFUL UPGRADES

• Highly customized widgets can be a nightmare
• Instead try to use out of the box solutions

```python
class CompactButton(gtk.Widget):
  __gtype_name__ = "EtkCompactButton"
  __gsignals__ = {
    "clicked": (gobject.SIGNAL_RUN_FIRST | gobject.SIGNAL_ACTION, gobject.TYPE_NONE, tuple()),
  }

  __gproperties__ = {
    "icon-name-normal": (gobject.TYPE_STRING, "icon name normal", "icon name normal", "", gobject.PARAM_READWRITE),
  }
```
KEEP THINGS SIMPLE

• Grab a great GUI toolkit
• Use out of the box widgets and other components
• Use a few key libraries if needed
• Ensure dependencies are cross-platform

```python
[tool.poetry.dependencies]
python = "^3.7"
PyGObject = "^3.30"
pycairo = "^1.18"
gaphas = "^3.1.0"
generic = "^1.0.0"
tinycss2 = "^1.0.2"
```
STAY TRUE TO THE ECOSYSTEM

- Follow the modern best practices for the language you are using
- Use Cookiecutter to get started quickly
- These solutions will be tried and true
- For Python: pyproject.toml and a Python build tool
PACKAGING
PACKAGING IN LINUX WITH FLATPAK

• Flatpak is 😎
• Provides universal and sandboxed distribution for Linux
• For app developers, runtimes are a strong foundation to build on
BUILDING FLATPAKs

• Make builds reproducible by building from Python wheels
• Uses a separate repository in flathub

```
pip3 download --dest ${BUILD} gaphor=="${GAPHOR_VERSION}"  
find ${BUILD} -type f -printf '%P\n' | awk -F- '{ print $1 " " $0 }' | \ 
while read -r DEP FILE do  
curl -sSfL https://pypi.org/pypi/"${DEP}"/json | jq -r '.releases[][] | select(.filename == "'"${FILE}"'") | "\(.digests.sha256) \(.url)"'  
done
```
APPIMAGE: ONE APP = ONE FILE

- Another great format for distributing apps
- Also universal and sandboxed with a single file executable
- Challenges for app developers to use the latest GUI toolkits while maintaining compatibility for users
Windows

- MSYS2 provides a familiar environment, but also some challenges
- Cooperate with upstream projects to help improve things
- Code signing

'C:\Program Files (x86)\Windows Kits\10\bin\10.0.17763.0\x86\signtool.exe' sign ^
/f 'certificate.pfx' /tr http://timestamp.digicert.com /td sha256 /fd sha256 ^
/p $PASSWORD gaphor-installer.exe
MACOS

- Homebrew
- Signing
MACOS PACKAGING

• A .app file in a DMG (disk image, think ISO)
• Apps have a predefined directory structure

Gaphor.app/Contents/Info.plist
Gaphor.app/Contents/MacOS/gaphor
Gaphor.app/Contents/Resources

• Library references are absolute - need relocating
• Update environment variables
• Used our own script, now rely on PyInstaller
MACOS SIGNING

• Both app and dmg need signing
• All performed from build pipeline
TAKE AWAYS

- Approach each platform separately
- Work with upstream projects
- Integrate platform builds in the build pipeline
QUESTIONS?

Dan: @danyeaw / dan@yeaw.me
Arjan: @ajmolenaar / gaphor@gmail.com