Biting the Apple

Unlocking macOS with Python

About Me

- Hobbyist programmer
- First code in Tandy BASIC on a TRS-80 Model III
- Reformed Perl hacker
- Pythonista since 2018
- github.com/RhetTbull

Installing Python on macOS

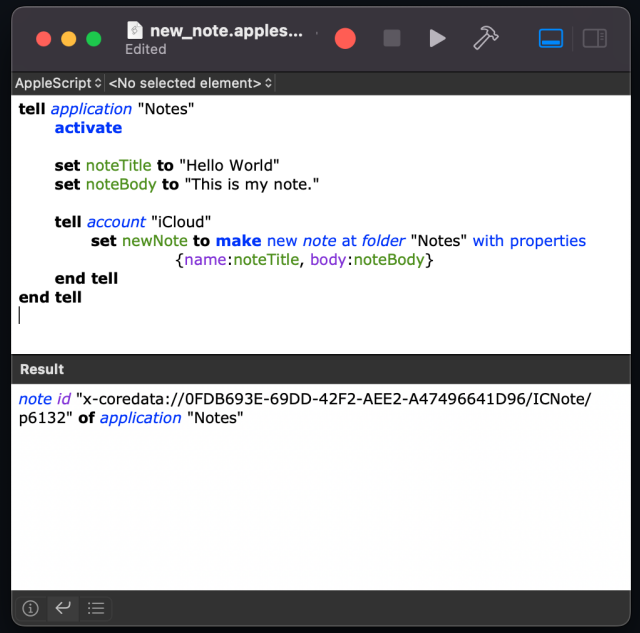
- Homebrew: Homebrew Python is Not For You
- Conda
- Python.org
- uv: `uv python install 3.13`

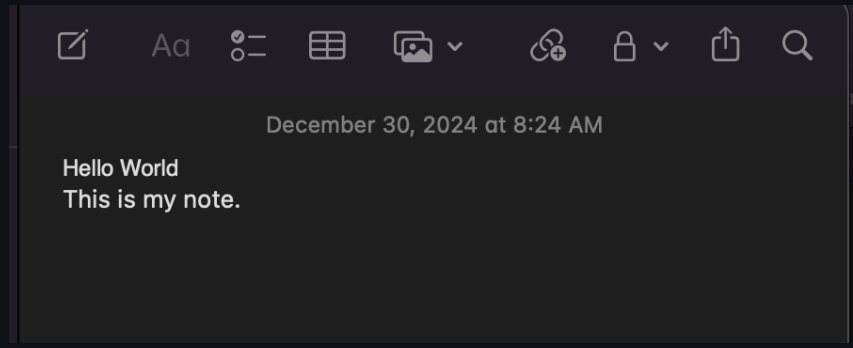
Just use uv or python.org

Automating Mac Apps

Many Mac apps are scriptable using AppleScript

```
tell application "Notes"
        activate

        set noteTitle to "Hello World"
        set noteBody to "This is my note."

        tell account "iCloud"
                set newNote to make new note at folder "Notes" with properties
                        {name:noteTitle, body:noteBody}
        end tell
end tell
```





Isn't This Talk Supposed To Be About Python?

```
pip install py-applescript
```

```python
import applescript

def create_note(title: str, body: str) -> str:
    """Create a new note in Notes.app"""
    script = """
    on createNote(noteTitle, noteBody)
        tell application "Notes"
            activate

            tell account "iCloud"
                set newNote to make new note at folder "Notes" with properties {name:noteTitle, body:noteBody}
                return id of newNote
            end tell
        end tell
    end createNote
    """
    ascript = applescript.AppleScript(script)
    return str(ascript.call("createNote", title, body))

note_id = create_note("Hello World", "This is my note.")
print(f"New note created with id {note_id}")
```

There's a Package For That!

```
pip install macnotesapp
```

```
import macnotesapp

new_note = macnotesapp.NotesApp().make_note("Hello World", "This is my note.")
print(f"New note created with ID {new_note}")
```

```
pip install photoscript
```

```
import photoscript

for photo in photoscript.PhotosLibrary().selection:
    photo.keywords += ["Travel"]
```

PyXA: Python for Automation

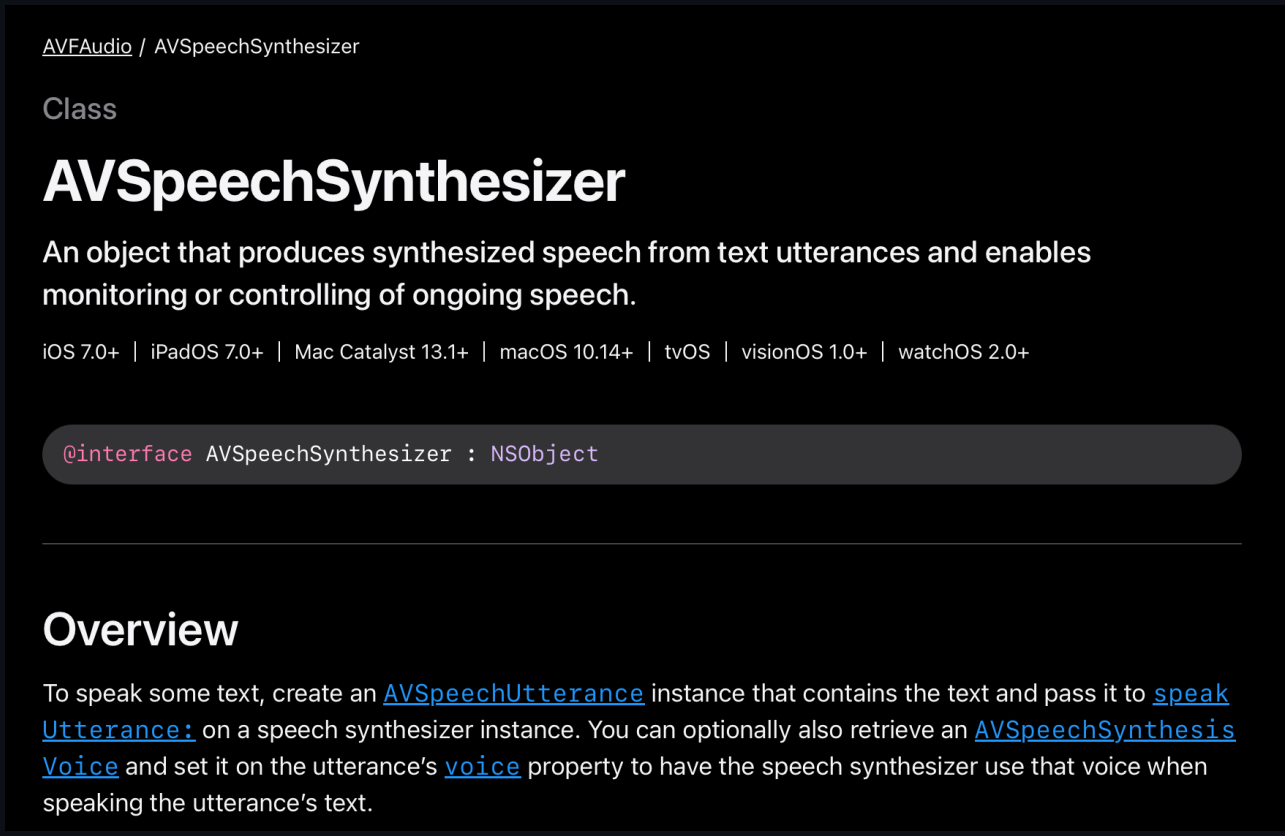
`pip install mac-pyxa` (Doesn't work yet with Python 3.13)

```
from datetime import datetime
import PyXA

def create_reminder(name, due_date=None):
    """Creates a new reminder in the Mac Reminders app's default list."""
    try:
        reminders_app = PyXA.Application("Reminders")
        due_date = datetime.fromisoformat(due_date) if due_date else None
        reminder = reminders_app.new_reminder(name=name, due_date=due_date)
        print(f"Reminder '{name}' created successfully.")
    except Exception as e:
        print(f"Failed to create reminder: {e}")

create_reminder("Finish HSV.py talk", due_date="2025-02-01T17:00:00")
```

Beyond Scripting: Accessing the Power of Native APIs

Apple provides useful APIs in Objective-C and Swift that enable features like extracting text from images, speech synthesis, accessing the Mac's camera, etc.

Can we use these from Python?



AVFAudio / AVSpeechSynthesizer

Class

AVSpeechSynthesizer

An object that produces synthesized speech from text utterances and enables monitoring or controlling of ongoing speech.

iOS 7.0+ | iPadOS 7.0+ | Mac Catalyst 13.1+ | macOS 10.14+ | tvOS | visionOS 1.0+ | watchOS 2.0+

```
@interface AVSpeechSynthesizer : NSObject
```

Overview

To speak some text, create an AVSpeechUtterance instance that contains the text and pass it to speakUtterance: on a speech synthesizer instance. You can optionally also retrieve an AVSpeechSynthesisVoice and set it on the utterance's voice property to have the speech synthesizer use that voice when speaking the utterance's text.

Yes! Python to Objective-C Bridge

PyObjC

Appears to have at least tacit blessing from Apple. Stable, offers support for most macOS APIs. Does not support iOS.

Rubicon

Part of the BeeWare project. Supported by Anaconda. Supports both macOS and iOS.

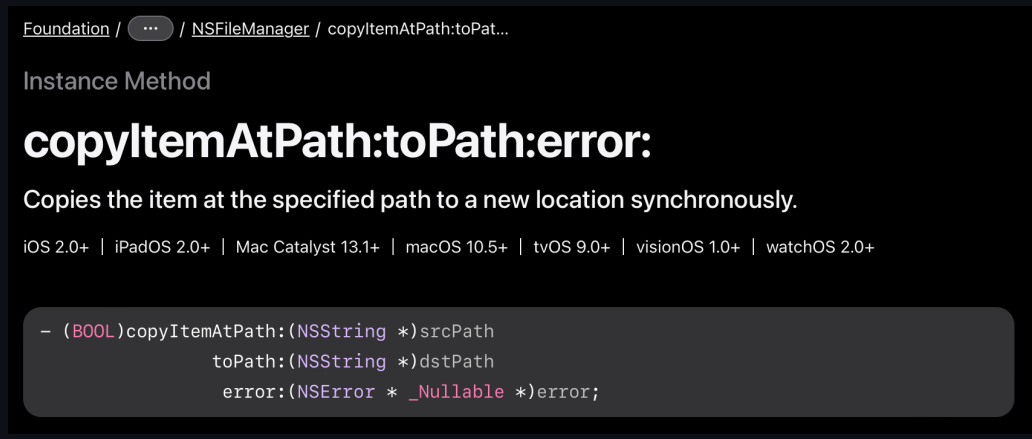
Simple PyObjC Example

Use native copy-on-write for APFS file systems.

- Python's native file copy functions cannot take advantage of copy-on-write; also do not copy all metadata.
- Apple's NSFileManager provides copyItemAtPath:toPath:error: method that does use copy-on-write

Python implementation:

```python
import Foundation

def copyfile(src: str, dest: str):
    """Copy file from src to dest using NSFileManager"""
    filemgr = Foundation.NSFileManager.defaultManager()
    success, error = filemgr.copyItemAtPath_toPath_error_(src, dest, None)
    if not success:
        raise OSError(error)
```



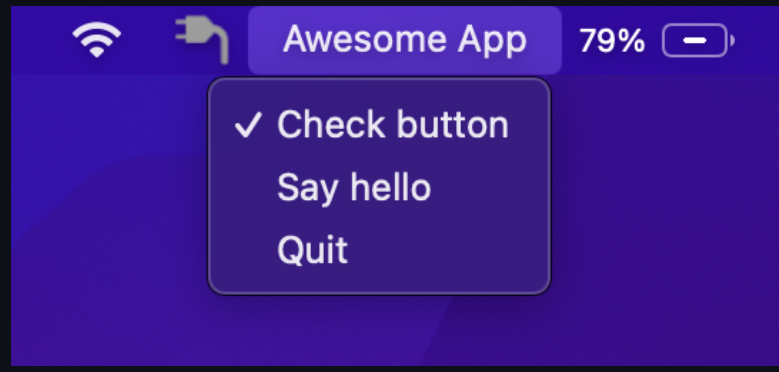
Speech Synthesis Example

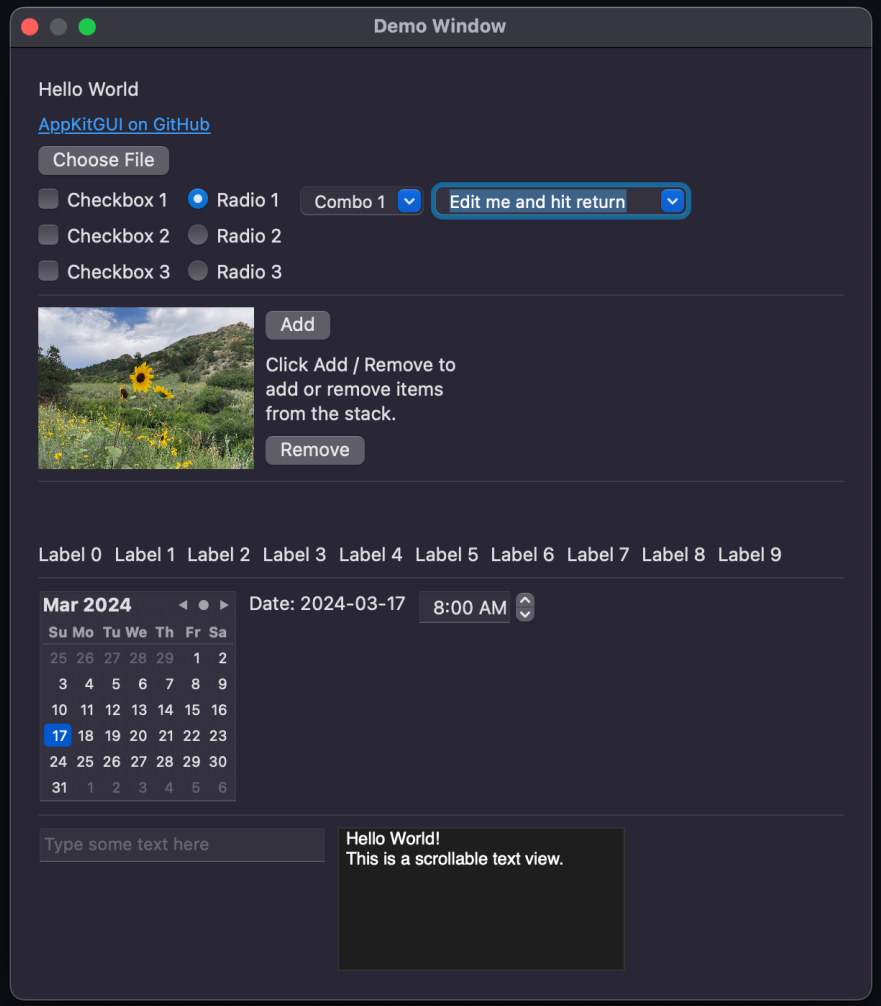
```
import AVFoundation
from Foundation import NSObject
from PyObjCTools.AppHelper import runEventLoop, stopEventLoop

class SpeechSynthesizerDelegate(NSObject):
    def speechSynthesizer_didFinishSpeechUtterance_(self, synthesizer, utterance):
        stopEventLoop()

def speak_string(text: str) -> None:
    synthesizer = AVFoundation.AVSpeechSynthesizer.alloc().init()
    utterance = AVFoundation.AVSpeechUtterance.speechUtteranceWithString_(text)
    voice = AVFoundation.AVSpeechSynthesisVoice.voiceWithLanguage_("en-US")
    utterance.setVoice_(voice)
    delegate = SpeechSynthesizerDelegate.alloc().init()
    synthesizer.setDelegate_(delegate)
    synthesizer.speakUtterance_(utterance)
    runEventLoop()
```

Status Bar Apps

Rumps: Ridiculously Uncomplicated macOS Python Statusbar

```
import rumps

class AwesomeStatusBarApp(rumps.App):
    @rumps.clicked("Check button")
    def onoff(self, sender):
        sender.state = not sender.state

    @rumps.clicked("Say hello")
    def sayhello(self, _):
        rumps.alert("Hello", "Hello HSV.py!", "Goodbye")

if __name__ == "__main__":
    AwesomeStatusBarApp("Awesome App").run()
```





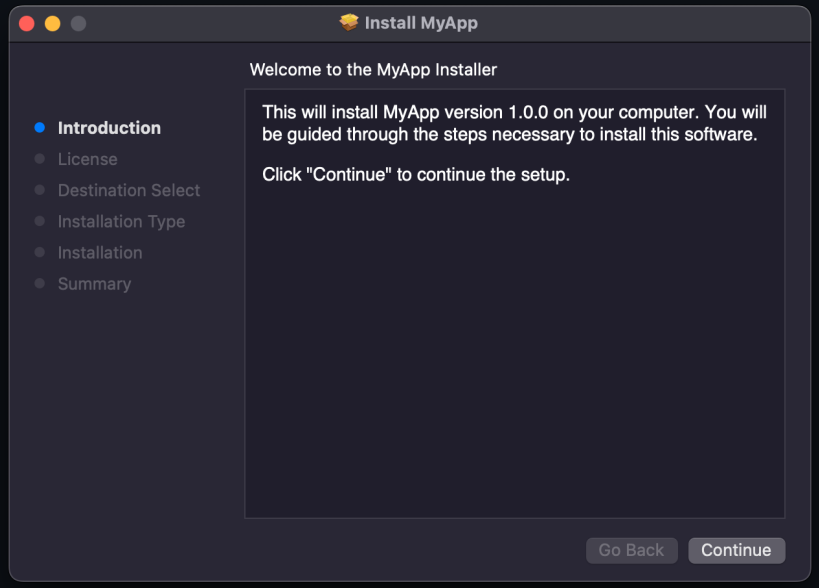
What about a GUI?

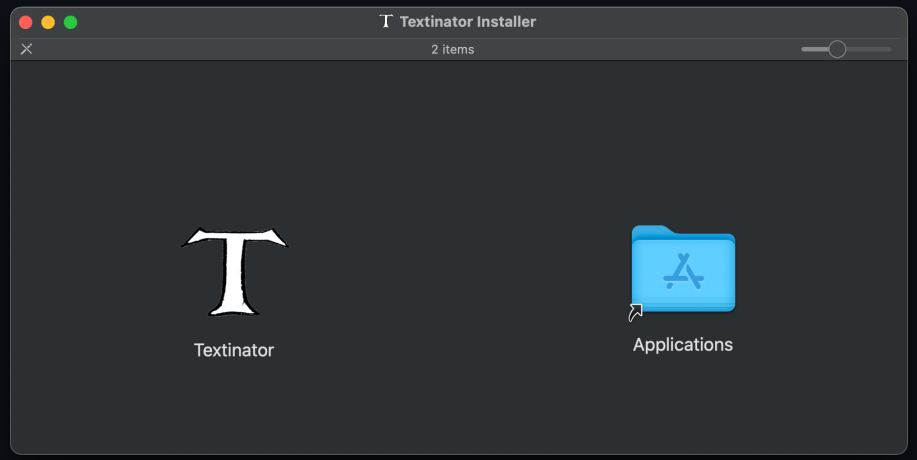
That's a whole other talk

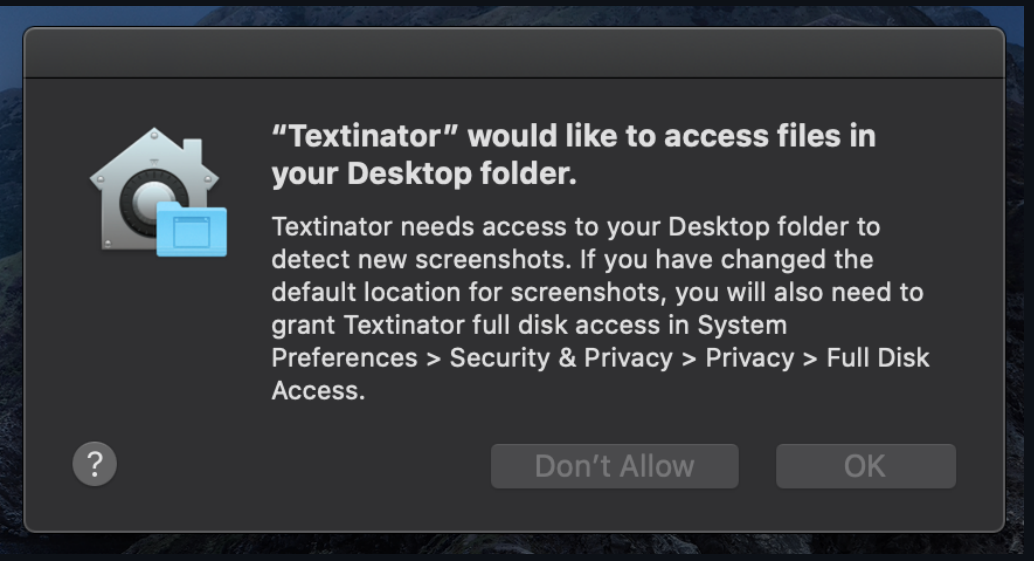
- Most Python GUI frameworks work fine with macOS
- Tkinter: Ships with Python standard library
- Toga: From BeeWare, native widgets, early development
- AppKitGUI: My own mini-framework, shows how to use AppKit from Python

Creating a Standalone App

- PyInstaller: Good for command line tools
- Py2App: Standalone Mac apps (like py2exe but for macOS)
- PyApp: Self-bootstrapping apps with self-update, very fast, written in Rust
- Briefcase: From BeeWare, similar to Py2App
- AppleCrate: Create installers for your app







Permissions & Entitlements

macOS has a ~~naggy~~ robust security model

- Certain features require user approval
- Entitlements: executable permission to use a technology / service
- Permissions: control run-time access to sensitive user data (e.g. location, files)

Command line apps: the Terminal app is responsible for requesting permission.

GUI apps: the application is responsible for requesting permission.

Resources

Automation

- AppleScript: Apple's scripting language
- macnotesapp: Automate Apple Notes
- PhotoScript: Automate Apple Photos
- PyXA: Python for automation

Native APIs

- PyObjC: Python to Objective-C bridge
- Rubicon: Alternate Python to Objective-C bridge
- Rumps: macOS statusbar apps

GUIs

- Tkinter: Standard library
- Toga: Native widgets
- AppKitGUI: My own macOS native mini-framework

Installers / App Packaging

- PyInstaller: Good for command line tools
- Py2App: Standalone Mac apps
- PyApp: Self-bootstrapping apps
- Briefcase: Similar to Py2App
- AppleCrate: Create installers

Questions?

This talk: https://github.com/RhetTbull/biting-the-apple

GitHub: https://github.com/RhetTbull

LinkedIn: https://www.linkedin.com/in/rhettbull/