Forester

sustainably manage your ham radio logs

RMHAM University Nerdfest

Chris Keller, K0SWE
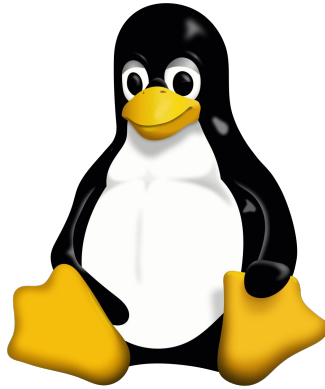February 13, 2021

Your Host

Chris Keller, K0SWE. Active in RMHAM and Colorado ARES R1D6, as well as software development for amateur radio. (That's where I got my vanity call: SoftWare Engineer, SWE)

Linux

If you've heard me talk before, you probably know I'm a Linux nut and avoid Windows and Mac where I can. That makes it challenging sometimes to find quality software, especially for niches like ham radio.

CQRLog

I built my first HF station in March 2020 when the pandemic started. CQRLog was the first logging application I used seriously. I was disappointed by the lack of being able to import logs and have them de-duplicated, and the lack of sync between independent machines, but it was good enough. That is, until I got a HiDPI monitor...

CQRLog HiDPI (Retina)

Yikes, that's awful! With a HiDPI (Retina-class) monitor, CQRLog is unusable. Without scaling, it's tiny; with scaling, the UI is completely distorted.

Buy vs **Build**

Requirements:

- Merge on import with smart deduplication
- Sync between clients
- Sync with QSL services & other online logbooks (QRZ, LotW, eQSL, etc.)
- Award progress with maps
- Offline
- WSJT-X and rig control integration
- At least Linux, Windows and Mac
- Open source

Make it a web app

I fretted over whether to make it a native app like [CQRLog](#) (or contribute to that in Pascal), a hybrid like [K4CPO Field Day logger](#) (or contribute to that in PHP), or a pure web app. Ultimately, I decided I didn't want to make a slightly better CQRLog or a slightly better K4CPO. Nor did I want to contribute to Pascal or PHP programs where their authors already had a vision.

I decided to play to my personal strengths as a software developer and build a web app, and rely on cutting-edge techniques to overcome traditional web limitations.

Rig Control via Web Page

WebUSB

- Direct device control, no need for any installed software
- But device (radio) firmware has to support it. Not going to happen!

Agent program

- Translate local program APIs (WSJT-X, rig control) to web sockets

I researched WebUSB first for rig control. I've used WebUSB for things like my Ergodox EZ mechanical keyboard; it can connect to the web-based key-mapping tool so the web UI can show you what the current key layout is. I imagined being able to transpile hamlib into Javascript with WebAssembly; transpiling is ugly, but it does work. Unfortunately, I found that the USB device must specify its WebUSB domain name during enumeration. In other words, it has to come out of the factory knowing that it's WebUSB compatible and exactly which website is going to connect to it. Not an option for hundreds of ham radio transceivers from dozens of manufacturers connecting to arbitrary third-party web sites.
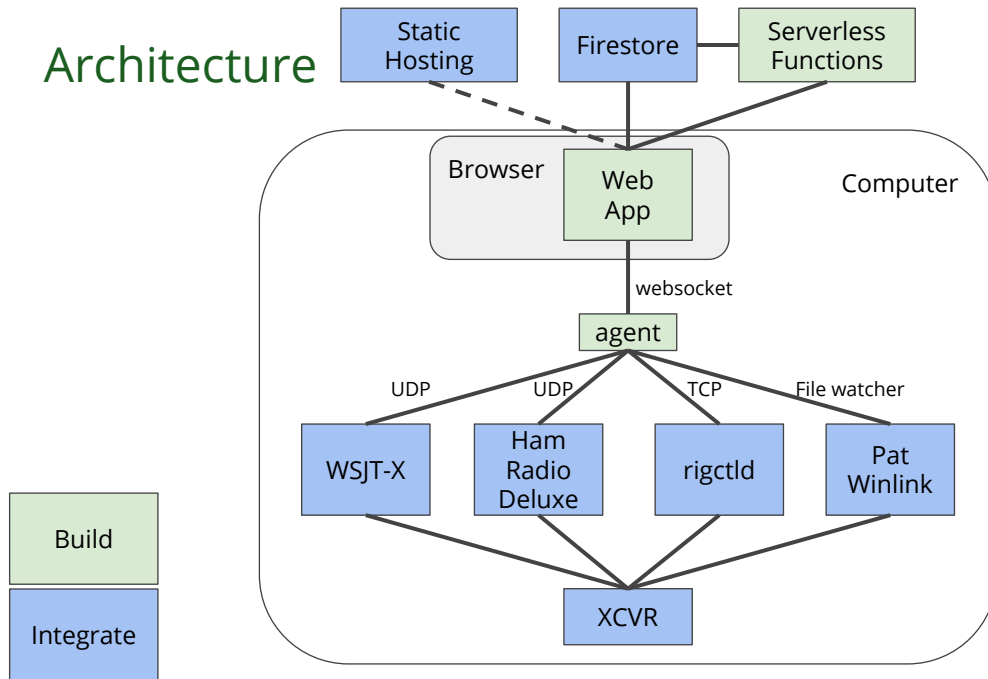
Instead, I looked at making an agent, a small installed program that mediates between the web page and the stuff on the computer. This technique is used by the Arduino Create web IDE. Create and compile your program in the web browser, then upload it to the Arduino board via the agent.

Prototype
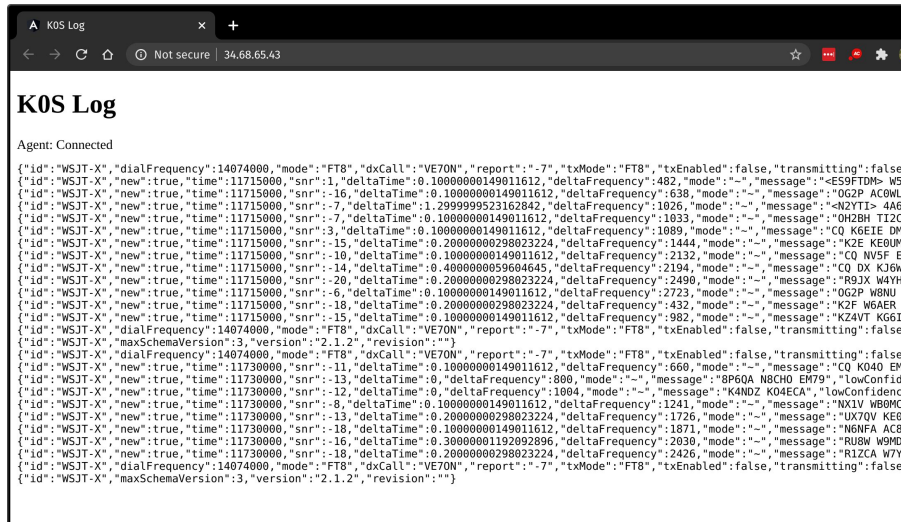
But does this really work? I deployed a prototype web app to an internet host and had it connect to a simple agent on my machine. I discovered (the hard way) a couple of browser security checks that had to be satisfied, but after some trial and error I got it to work! I got WSJT-X on my computer to stream information into a web application in my browser!
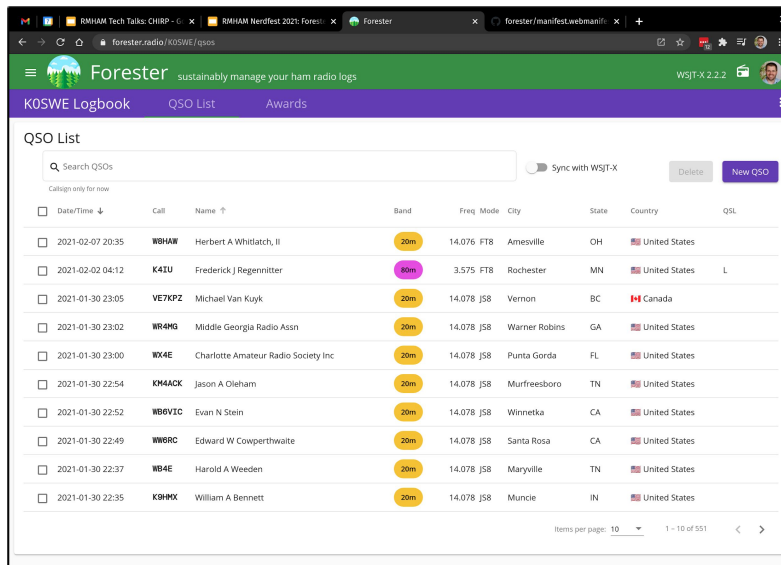
Architecture

Static Hosting

Firestore

Serverless Functions

Browser

Web App

Computer

websocket

agent

UDP

UDP

TCP

File watcher

WSJT-X

Ham Radio Deluxe

rigctld

Pat Winlink

Build

Integrate

XCVR

Demo: The current state

Subsequent slides are for talking points

Forester

Forester is a [hosted web application](#) written in the Angular framework in the Typescript language. It uses Google Firestore as an underlying database which gives it fast real-time sync between devices as well as being robust to going offline for a short while. The offline capability is probably not appropriate for a DXpedition, but perfect for field day or a POTA/SOTA activation.

To connect to installed ham programs, it uses [kel-agent](#), written in golang. Installers are available for Windows, Mac, Linux and Raspberry Pi, and will soon be available in the Debian/Ubuntu APT repositories. Right now, kel-agent can connect to WSJT-X as a backend, with planned support for rigctld, Ham Radio Deluxe, JS8Call, and whatever else people ask for. Eventually, kel-agent will be a tray-icon program, but right now it's just a command-line utility.

For certain tasks like syncing with other internet services (QRZ.com, LotW, etc), Forester uses Serverless Functions. This is faster and uses less client bandwidth than running these tasks in the browser. However, I've been careful to build anything functionality that's needed offline into the web app.

All three facets of Forester are [open source](#), and kel-agent is written to be used by other applications. (I hesitate to say "free" because I pay for the cloud services; if I get enough users, I'll probably have to charge or show ads to break even.)

Progressive Web App
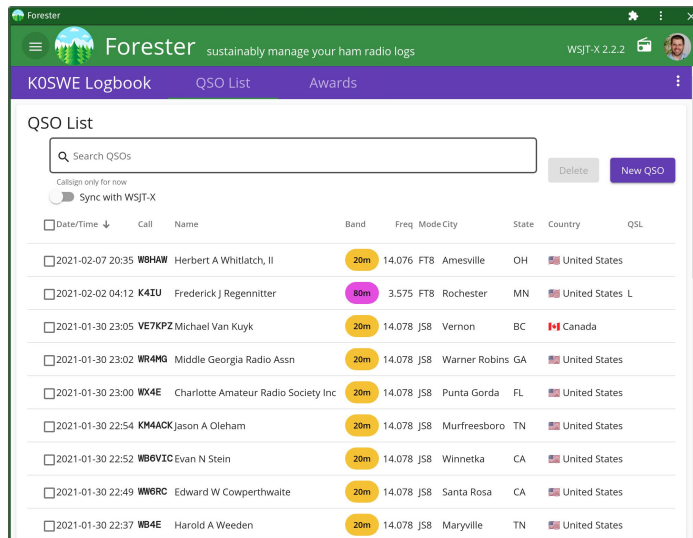
PWA enables offline support, "add to home screen" and "installed" experience. It adds an icon to the application menu on desktops, phones and tablets in all major operating systems. The app will run and most things will function even when offline. (Support for PWAs in web browsers is [not universal yet](#); a Chromium-based browser like Chrome or Edge is your best bet.)

QRZ.com Logbook 3 beta

The next generation QRZ.com logbook (currently in beta) is also being built as a PWA and will have offline support.

Under Construction

Current state of Forester is "alpha"

- Works well enough for some tasks
- Plenty of missing features and sharp edges

https://forester.radio/

I'm eager to hear feedback!

Contact Me

k0swe@arrl.net