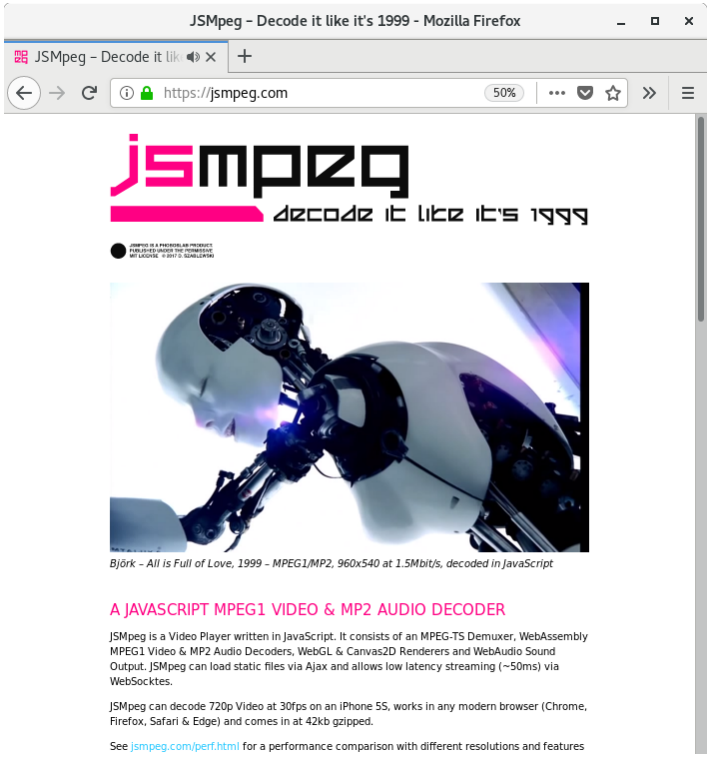
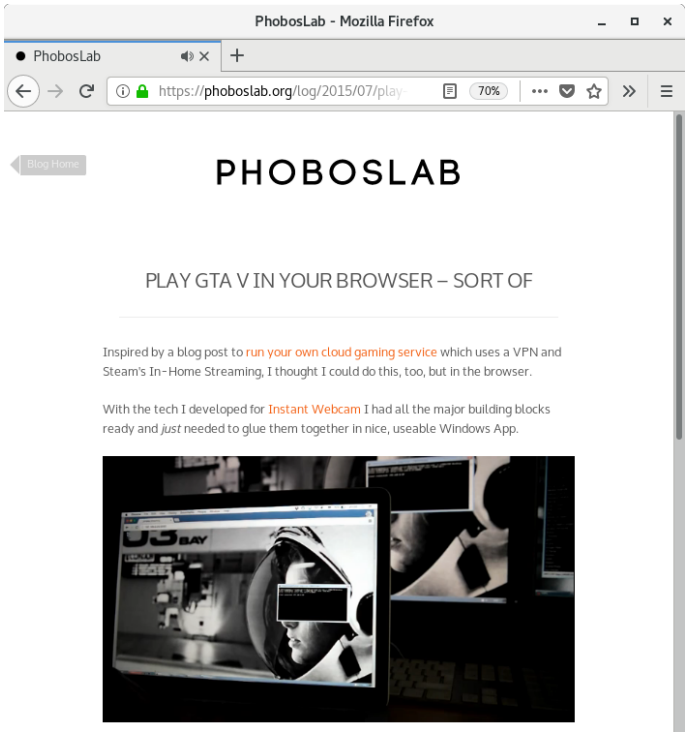
# A new video driver for undroidwish (and other goodies)

http://www.androwish.org

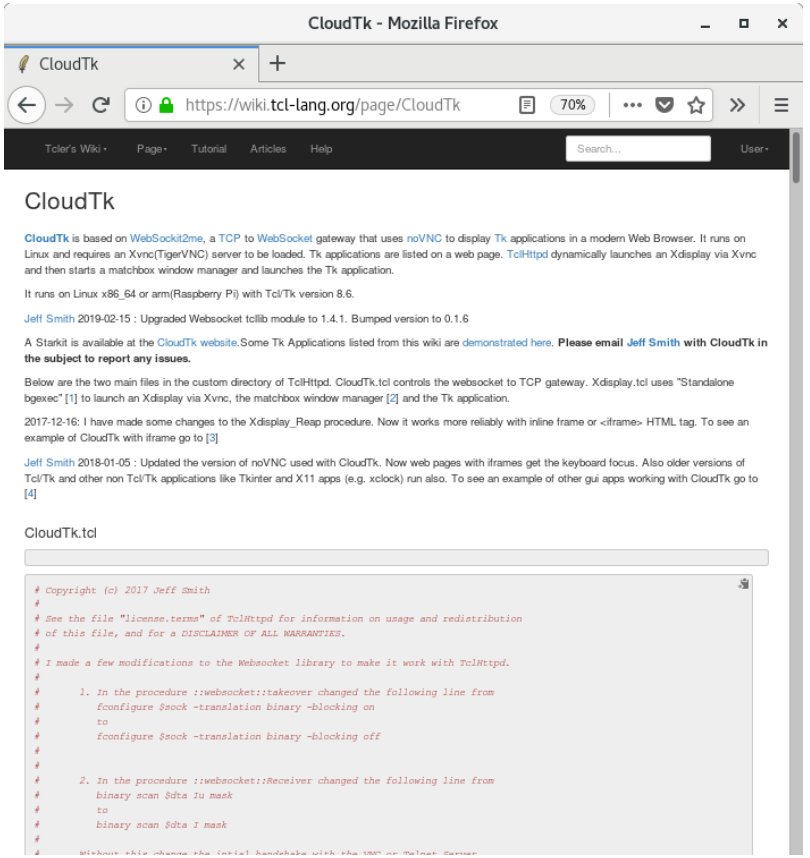
# The inspiration ...

https://jsmpeg.com and based on it jsmpeg-vnc, refer to the links on jsmpeg.com's main page.



# The inspiration (continued) ...




https://wiki.tcl-lang.org/page/CloudTk which uses Xvnc and a window manager to present that X11 session in a browser using the noVNC client from https://novnc.com/

However, this solution is not light weight due to requiring a full X11 environment, OTOH provides all infrastructure to serve arbitrary other X11 clients via browser.

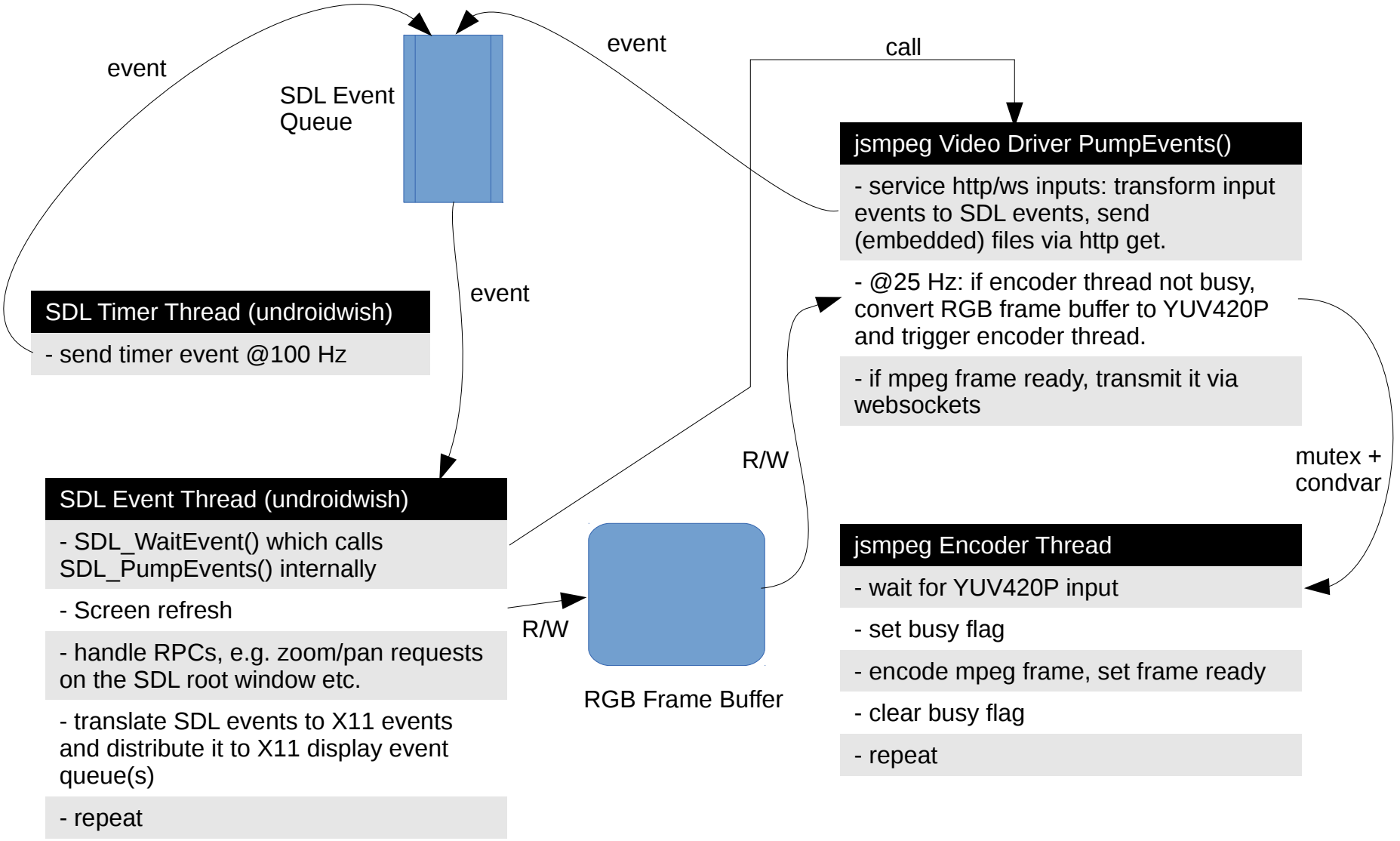
But I want undroidwish being as self contained as possible ...

# What is required?

- Some facilities of the ffmpeg libraries.
- Some facilities of the libwebsockets library.
- Some JavaScript for mpeg decoding to a HTML5 canvas optionally using WebGL.
- Some JavaScript for event reporting (mouse, keyboard, touch).
- A modern web browser (Firefox, Chrome, Safari, Edge, IE10/11).
- All server side stuff mixed into an SDL2 video driver plus some http/websockets facility to deliver the required JavaScript/HTML components to the browser.

➔ undroidwish's screen/keyboard/mouse is a web browser.

# Driver Architecture



# jsmpeg supported platforms

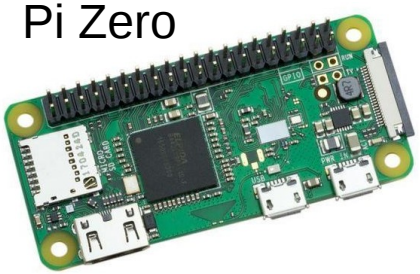
- **Linux**: OpenGL is supported when an X11 display connection is available.
- **Windows**: OpenGL is supported, thanks to Windows' refusal being window(s)less.
- **MacOSX**: no OpenGL support yet (but most likely can be done).
- ***BSD**s: in theory (i.e. port needs be done, no technical obstacles expected)

# undroidwish X11 emulation (almost the same blurb as last year)

- Multi-threaded Tk applications are supported.
- SDL2 (and browser) supported input devices work OOTB (touch screens, joysticks?).
- Many (non-trivial) Tk extensions are working (platform dependent): Canvas3D, tkpath, tkimg, TkZinc, tktable, BLT, tktreectrl.
- A server less static Tcl/Tk binary can still be made in about 6 Mbyte (excluding required shared libraries and fonts). Overall, the jsmpeg video driver adds 120 kByte excluding dependencies.

# Where jsmpeg (somewhat) fails ...

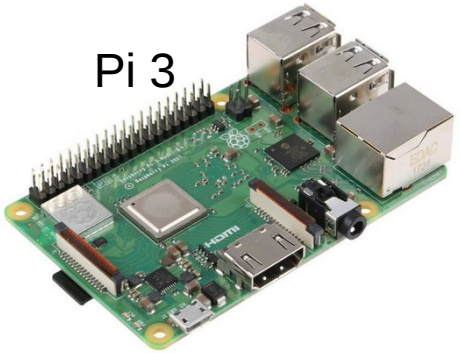
Pi Zero



- Single core
- Sub surface performance
- Interplanetary latency

➔ “The discovery of slowness”

Pi 3



- Four cores
- But 1.2 cores constantly @ 100%

➔ Almost usable

# Alternatives to jsmpeg

- libvncserver allows to export a frame buffer to VNC clients (e.g. noVNC in web browsers). However, this library is GPL.
- freerdp allows to export a frame buffer to RDP clients. Licensing unclear. No web based RDP client known.
- A radical different approach like Wtk or GNOME's broadway, i.e. render directly to JavaScript/HTML5 canvas in Tk. Much work ahead; OTOH, in GNOME still a broadwayd (daemon, like an X server?) seems to be required which is not exactly self contained.

# Other goodies 1 (tkvlc)

https://github.com/ray2501/tkvlc is an interface to use libVLC (the core of the VLC media player) in Tcl/Tk for video playback. My contribution adds an event callback and play back into photo images in order to use it e.g. as textures in a Canvas3D or in undroidwish like environments with frame buffers. Example:

```
package require Tk
package require tkvlc

set photo [image create photo -width 320 -height 240]
set display [label .tkvlc -image $photo -bg white]
pack $display -fill both -expand 1

tkvlc::init tkvlc0 $photo
tkvlc0 open "video.mp4"
if {![tkvlc0 isplaying]} {
    tkvlc0 play
}

vwait forever
```

# Other goodies 2 (topcua)

Tcl binding to OPC Unified Architecture (OPC UA), a machine to machine communication protocol for industrial automation developed by the OPC Foundation.

Refer to https://en.wikipedia.org/wiki/OPC_Unified_Architecture for a detailed overview.

The Tcl binding uses the C based OPC UA implementation from https://open62541.org/ and can be found in https://www.androwish.org/index.html/dir?name=jni/topcua.

Documentation is in https://www.androwish.org/index.html/wiki?name=topcua.

Sample code is in the wiki: https://wiki.tcl-lang.org/page/topcua.

# topcua – Quick demo

- The Tcl server `uacam.tcl` implements a little webcam acquiring images using `tcluvc`. It maps the camera image and some camera controls (brightness, contrast etc.) to data variables in its own namespace in the OPC UA address space.
- A generic OPC UA client can access these variables.
- The Tcl client `uacam_client.tcl` displays the image variable as a photo image in a label widget. The image update is done using a subscription and monitor in OPC UA speak, i.e. a periodic activity expressed in terms of OPC UA communication.

# WIP: Taygete Scrap Book

- Taygete (Ταϋγέτη): a small retrograde irregular satellite of Jupiter, aka Jupiter XX.
- Idea: take a webview (the rendering component/library of a web browser), add a Tcl interface, and mash it up with some Tcl and JavaScript to provide an UX somewhat resembling Jupyter Notebooks.
- No browser and webserver required, one binary, zero installation, unclouded.
- There's a "Tiny cross-platform webview library for C/C++/Golang. Uses WebKit (Gtk/Cocoa) and MSHTML (Windows)" in https://github.com/zserge/webview which has a Python binding which inspired the Tcl binding.
- The Tcl binding is about 650 LOC in https://www.androwish.org/home/dir?name=undroid/twv
- The UI/engine of "Taygete Scrap Book" is an about 1300 LOC mixture of Tcl and JavaScript in https://www.androwish.org/home/dir?name=undroid/tsb

# TSB: How it works

In Tcl a webview is created

```
set W [twv:new -width 800 -height 600 -url … -callback …]
```

From Tcl JavaScript code can be evaluated using

```
$W eval JavaScript-code-string
```

```
$W call JavaScript-function ?string-argument …?
```

From JavaScript Tcl code can be evaluated using

```
window.external.invoke(Tcl-callback-argument);
```

The bootstrap in the webview constructor is an URL which contains JavaScript to evaluate Tcl code, which writes the skeleton of a HTML document including the necessary JavaScript functions to interface with the rest of the Tcl code, i.e. calls

```
document.write(HTML+CSS+JavaScript-code);
```

```
document.close();
```

# TSB: Try it for yourself

A ZIP kit is available in

`http://www.ch-werner.de/AndroWish/TSB.kit`

It supports the three major desktop platforms

- Windows (32 and 64 bit)
- MacOS (Intel, 64 bit)
- Linux (Intel, 32 and 64 bit, distro agnostic but a decent version with gnome 3 runtime is required)

Questions?