Tcl/Tk on Android

http://www.ch-werner.de/sdltk/AndroWish

Icon kindly donated by Jorge Raul Moreno
AndroWish

- Native Tcl/Tk port for Android ≥ 2.3.3 on ARM and x86 processors. Top goal: execute existing Tcl/Tk scripts on Android without modification.
- Uses Tim Baker's SDLTK project for graphics rendering, see http://members.shaw.ca/tnbaker/SDLTk
  - X11 emulation based on AGG (Anti-Grain-Geometry, http://www.antigrain.com) and SDL 2.0 (http://libsdl.org)
  - freetype font engine (http://www.freetype.org)
- Mounts its constituting APK (Android Package, i.e. the app) using a built-in ZIP file system based on mmap(2)
- “Batteries included” like TclKits (i.e. many ready-to-use Tcl extensions already bundled)
- Tcl commands available to use specific Android facilities
AndroWish, the big picture
AndroWish, limitations

- X11 emulation is not thread-safe, thus it is impossible to `package require Tk` from another thread. But multiple Tcl `interp`s in the main thread work.

- Due to Android process start up with respect to the window system `exec(n)` is limited to non-Tk processes, i.e. you can't `exec wish`.

- The bandwidth of device screen resolutions is broad (140 dpi ... 500 dpi) compared to usual desktop systems. But many elements of `Tk` widgets are pixel based.
SDLTK

- Partial Xlib replacement to implement rendering of standard Tk widgets using SDL and AGG and to support most features of extensions (treeview, ttktable, TIX, BLT).
- Event handling by translating SDL events to X11 events plus additional virtual events
  - App life cycle (<<WillEnterBackground>>, <<WillEnterForeground>>, etc.)
  - Accelerometer is handled by SDL and mapped to an SDL joystick, and translated to Tk virtual event <<Accelerometer>>
  - Multi-touch events <<FingerUp>>, <<FingerDown>>, <<FingerMotion>>, <<PinchToZoom>>
- Other SDL goodies available through sdltk command
- Can be built standalone on other platforms (currently tested on Linux x86/Raspberry with X11 and DirectFB), see http://www.ch-werner.de/sdltk
sdltk command

- Obtain power management information (sdltk powerinfo)
- Control accelerometer (sdltk accelerometer)
- Control screensaver (sdltk screensaver)
- Show/hide virtual keyboard (sdltk textinput)
  - built into standard bindings of entry and text widgets
- Control emulation of middle and right mouse buttons for context menus and panning/scrolling (sdltk touchtranslate)
borg [ˈbɔːrk]

• German imperative of “borgen” (to borrow)
• Inspired by Scripting Layer for Android (SL4A) and PhoneGap (now Apache Cordova)
• Junction to connect the (native) Tcl/Tk with the (Java) Android universe
• Use Android facilities from Tcl scripts
  – Information about device
  – Activities (Android apps)
  – Alarms, Notifications
  – Content (Android databases)
  – Location (GPS)
• Report device events using Tk virtual events (network status, location updates, sensor events)
borg command examples

Obtain information about the display

borg displaymetrics
density 1.33125 densitydpi 213 \width 800 height 1216 \xdpi 216.17021 ydpi 215.31126 \scaleddensity 1.33125
borg command examples

Use Android's text-to-speech facility for speech output

    borg speak "resistance is futile" \ en_US
borg command examples

Add an icon to Android's desktop which launches a Tcl script contained in AndroWish's ZIP mounted /assets directory

borg shortcut add \nfile://assets/sdl2tk8.6/demos/widget
Use other Android activities (parts of other apps). Example: read a bar code using the [http://code.google.com/p/zxing](http://code.google.com/p/zxing) bar code scanner app.

```bash
proc barcode_read {code action uri type cat data} {
    array set result $data
    if {[info exists result(SCAN_RESULT)]} {
        # that is the barcode
        # result(SCAN_RESULT_FORMAT) is the barcode format
    }
}

borg activity com.google.zxing.client.android.Scan \
    {} {} {} {} {} barcode_read
```
borg virtual events

- **<<LocationUpdate>>**: location information is available and can be read using

  `borg location get`

- **<<SensorUpdate>>**: a sensor can be read using

  `borg sensor get ...`

- **<<NetworkInfo>>**: network status was updated and can be read using

  `borg networkinfo`

- **<<Bluetooth>>**: Bluetooth status was updated and can be read using

  `borg bluetooth state`
Other AndroWish goodies

- `bluetooth` command allows to create Bluetooth client and server sockets providing the serial port profile (SPP). Those sockets are normal Tcl channels.

- `usbserial` command allows to use certain USB to serial converters (FTDI, Prolific) similar to normal Tcl serial channels, i.e. `fconfigure` can be used to control the baud rate etc.
## Anatomy of AndroWish's APK

<table>
<thead>
<tr>
<th>Directory within APK</th>
<th>Description</th>
<th>Size uncompressed</th>
</tr>
</thead>
<tbody>
<tr>
<td>assets/</td>
<td>Application auxiliary files, Tcl libraries</td>
<td>36 MByte</td>
</tr>
<tr>
<td>tcl8.6/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>sdltk8.6/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>fonts/</td>
<td>Default fonts, Deja Vu</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>“Batteries included” (SQLite, tcllib, tls, treectl, ...)</td>
<td></td>
</tr>
<tr>
<td>app/</td>
<td></td>
<td></td>
</tr>
<tr>
<td>lib/</td>
<td>Native code, shared libraries</td>
<td></td>
</tr>
<tr>
<td>armeabi/</td>
<td>... for ARM processors</td>
<td>10 MByte</td>
</tr>
<tr>
<td>x86/</td>
<td>... for x86 processors</td>
<td>15 MByte</td>
</tr>
<tr>
<td>res/</td>
<td>Android Resources</td>
<td>few kByte</td>
</tr>
<tr>
<td>drawable/</td>
<td>... icons</td>
<td></td>
</tr>
<tr>
<td>layout/</td>
<td>... layout, styles</td>
<td></td>
</tr>
<tr>
<td>resources.asrc</td>
<td>Application resources</td>
<td>few kByte</td>
</tr>
<tr>
<td>AndroidManifest.xml</td>
<td>Application descriptor</td>
<td>few kByte</td>
</tr>
<tr>
<td>classes.dex</td>
<td>Compiled Java code for Android's Dalvik VM</td>
<td>&lt; 1 MByte</td>
</tr>
<tr>
<td>META-INF/</td>
<td>JAR Manifest, signatures of all files in APK</td>
<td>&lt; 1 MByte</td>
</tr>
</tbody>
</table>
Batteries included
How to roll your own app

- It is possible to re-use AndroWish's infrastructure (/assets directory with Tcl runtime, native shared libraries) from other apps.
- A slightly modified Java glue is required (about 300 kByte compressed, compared to ≥ 22 MByte of the complete AndroWish APK).
- Due to APK building (various Android tools and Java jarsigner needed) this must be done using Android's SDK and optionally Eclipse.
- A demo project is available on http://www.ch-werner.de/sdltk/AndroWish/HelloTclTk
20 minutes into the future

Tcl/Tk is quite portable and SDL already runs on many platforms. So let's think of ...
... throwing the feather at
Thank you.
Questions?