net Application Layer

Browser Components

Bernd Paysan

EuroForth 2013, Hamburg

Overview

Purpose?

People want to share information *(share* means *copy)*

- Texts, photos, videos, music
- longer, structured documents
- Real–time media (chat, videos, video conferences)
- collaborative gaming

Things I want to show

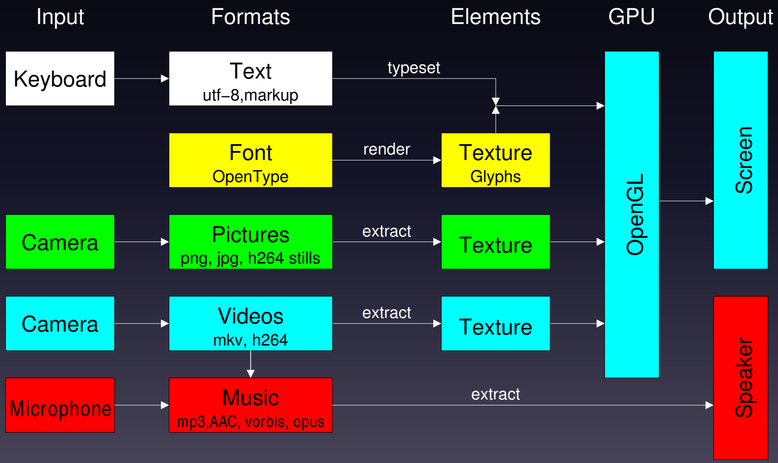
- In 2011 I did a presentation with the same title — back then, this was complete vaporware: the plan.
- Now there are components which need to be put together
- and there's a concept how to do that
- needs to work on PCs and mobile platforms like Android, which are sometimes "a bit strange".

Sidetracking: Impact of Snowden Leaks

- Encryption now uses Keccak (SHA–3) as primitive. Universal crypto primitive, faster than Wurstkessel at same level of security, chosen in an open competition.
- ECDHE for connection setup in a way that doesn't reveal identities (metadata!)
- Made sure the random numbers use entropy of the system, but not directly system random numbers
- Secure internet more important than ever!

Formats and IO

How to display things



Why OpenGL?

OpenGL can do everything

OpenGL renders:

1. Triangles, lines, points — simple components
2. Textures and gradients
3. and uses shader programs — the most powerful thing in OpenGL from 2.0.

Real requirement: visualization of *any* data. OpenGL can do that.

How to connect the media?

Lemma: every glue logic will become Turing complete

- currently used glue: HTML+CSS+JavaScript
- containers with Flash, Java, ActiveX, PDF, Google's NaCl...
- conclusion: use a powerful tool right from start!
- browser: run–time and development tool for applications

Security

Lemma: every sufficiently complex format can be exploited

Java's approach to secure the language from the inside can be seen as a failure. Java is now malware entry door number 1.

Sandbox

- sandbox the process that interprets network apps
- funnel network connections through a proxy — a shared memory module for net2o is missing
- encryption (key access!) outside the sandbox
- „same–origin"–policies don't work in a P2P cloud

Slideshow

I use the slide-how for this presentation

Fader

```
: fade { n1 n2 f: delta-time -- } n1 n2 = ?EXIT
    ftime { f: startt }
    BEGIN  ftime startt f- delta-time f/ fdup 1e f<  WHILE
            <draw-slide
            1e blend n1 draw-slide
            ( time ) blend n2 draw-slide
            draw-slide>  REPEAT
    <draw-slide 1e blend n2 draw-slide draw-slide>
    fdrop ;
```

Slideshow 2

Even more effects

Hslide

```
: hslide { n1 n2 f: delta-time -- } n1 n2 = ?EXIT
    ftime { f: startt }
    BEGIN  ftime startt f- delta-time f/ fdup 1e f<  WHILE
            <draw-slide
            pi f* fcos 1e f-
            [ pi f2/ fnegate ] FLiteral f* fcos 1e f-
            fdup n1 n2 > IF fnegate  THEN xshift n1 draw-slide
            2e f+ n1 n2 > IF fnegate  THEN xshift n2 draw-slide
            draw-slide>  REPEAT
    <draw-slide n2 draw-slide draw-slide>
    fdrop ;
```

Approach and Problems

libSOIL: Simple API to load images

libjpeg and libpng have a very complcated AP
Other option: libSOIL:

libSOIL load texture

```
: >texture ( addr w h -- )
    2 pick >r rgba-texture wrap nearest r> free throw ;
: mem>texture ( addr u -- addr w h )
    over >r  0 0 0 { w^ w w^ h w^ ch# }
    w h ch# SOIL_LOAD_RGBA SOIL_load_image_from_memory
    r> free throw w @ h @  2dup 2>r >texture 2r> ;
: load-texture ( addr u -- w h )
    open-fpath-file throw 2drop slurp-fid mem>texture ;
```

Onion–Programming

Looks big from the outside

Onion–Programming

Use of martial tools recommended

Onion–Programming

Onion “all the way down”

Videos

OpenMAX AL

- Android uses OpenMAX AL as video framework — similar to gstreamer, but slightly different...
- renders video into a texture, but can also record videos from the camera
- input: MPEG transport stream
- C++–like C API (vtable implemented as function pointer struct)
- only half–hearted implemented, needs Java via JNI, can't handle resizes
- four languages for video player: Forth, C, Java, OpenGL shader language

JNI declarations

MediaPlayer

```
jni-class: android/media/MediaPlayer

jni-new: new-MediaPlayer ()V
jni-method: prepare prepare ()V
jni-method: start start ()V
jni-method: setSurface setSurface (Landroid/view/Surface;)V
jni-method: setVolume setVolume (FF)V
```

JNI declarations II

SurfaceTexture

```
jni-class: android/graphics/SurfaceTexture

jni-new: new-SurfaceTexture (I)V
jni-method: updateTexImage updateTexImage ()V
jni-method: getTimestamp getTimestamp ()J
jni-method: setDefaultBufferSize setDefaultBufferSize (II)V
jni-method: getTransformMatrix getTransformMatrix ([F)V
```

JNI calls

get timestamp

```
: get-deltat ( -- f )
    media-sft >o getTimestamp o> d>f 1e-9 f*
    first-timestamp f@ f- ;
```

Java–Calls integrate seamless into Mini–OOF2 (Mini–OOF with current object)

MTS? All videos today are MKV!

"Matroska" sounds like onion programming, too...

Container — what for?

- Usual explanation: several files too difficult to handle. IMHO, directories with multiple files are better than containers.
- Videos and audio stored as single frames and short packets
- Timestamps for synchronized playback
- Index for random access

Matroska interpreter

Binary XML format

Solution: read MKV, convert to MTS

- Matroska parser uses a hash table for the tags
- each tag has an associated Mini OOF2 method
- different classes for different purposes: dump for inspection, MTS converter class

Fonts rendering

Freetype–GL renders OpenType fonts into OpenGL–Textures

- OpenType is state of the art
- we render textures, so the vector font needs to go into a texture
- FreeType–GL uses a texture as glyph cache
- 1 glyph: 2 triangles

Text Render Demo Code

Fonts and Texts

```
48e FConstant fontsize#
atlas "/system/fonts/DroidSans.ttf\0" drop
fontsize# texture_font_new Value font1
atlas "/system/fonts/DroidSansFallback.ttf\0" drop
fontsize# texture_font_new Value font2
Variable text1$ "Dös isch a Tägscht." text1$ $!
Variable text2$ "这是一个文本：我爱彭秀清。" text2$ $!
```

Text Render Demo Code II

Fonts und Texte

```
: glyph-demo ( -- )  hidekb
    1 level# +!  BEGIN
        <render
        0. penxy 2!
        font1 to font  text1$ $@ render-string
        -100e penxy sf! -60e penxy sfloat+ sf!
        font2 to font  text2$ $@ render-string
        render>
        sync >looper
    level# @ 0= UNTIL ;
```

Outlook

This presentation has been rendered with LaTeX Beamer...

- The next presentation should be rendered with MINOΣ2
- Texts, videos, and images should be get with net2o, shouldn't be on the device
- Typesetting engine with boxes and glues, line breaking and hyphenation missing
- a lot less classes than MINOΣ — but more objects
- add a zbox for vertical layering
- integrate animations
- combine the GLSL programs into one program?

Literature&Links

- Bernd Paysan
  net2o fossil repository
  http://fossil.net2o.de/net2o/

- Bernd Paysan
  minos2 fossil repository
  http://fossil.net2o.de/minos2/