



Game Programming with GLC for Smart Device

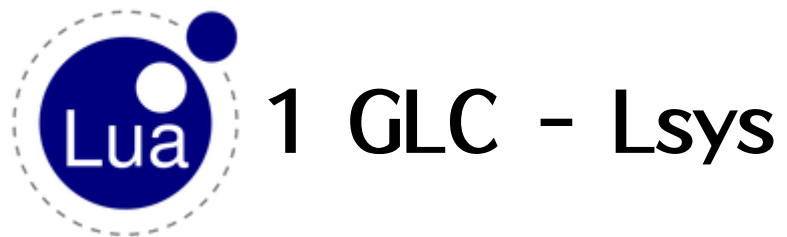
afewhee@gmail.com





- System – Lsys
- 2D – Ltex
- Font – Lfont
- Input – Lin
- Media – Lsmd
- Etc
 - ◆ Util – Lutil
 - ◆ File – Lfile
 - ◆ Database - Lstdb
 - ◆ Hard ware device - Lhw







- LUA API
 - ◆ Host Program에서 호출하는 LUA 함수: 반드시 정의해야 할 함수
 - ◆ 사용자가 작성
 - ◆ 성공 – return 0, 실패 – return -1

- Lua_Create()
 - ◆ 시스템 초기화

- Lua_Init()
 - ◆ 데이터 초기화

- Lua_Destroy()
 - ◆ 데이터 해제
 - ◆ 내용은 비워 둠 → 자동으로 해제함

- Lua_FrameMove()
 - ◆ 데이터 갱신

- Lua_Render()
 - ◆ Rendering





● LUA Glue

- ◆ Script에서 호출하는 Host 함수
- ◆ 엔진에서 제공

● GLC – LUA Glue

- ◆ Lsys - system
- ◆ Ltex - 2d texture
- ◆ Lfont - font
- ◆ Lin - input device
- ◆ Lsmd - sound, music, video
- ◆ Lutil - utility module
- ◆ Lfile - file i/o
- ◆ Lstdb - simple data base
- ◆ Lhw - hard ware device





● CreateWindow

- ◆ 윈도우 생성
- ◆ 시작 위치 $x, y \rightarrow$ center $-1, -1$
- ◆ 윈도우 너비, 높이 \rightarrow default 640, 480
- ◆ 윈도우 클래스 이름

● ScriptFile

- ◆ Phase에 대한 Script 파일 설정
- ◆ Lua_FrameMove()의 return 값과 연결





- Sleep

- ◆ Process 대기 함수

- ShowCursor

- ◆ Mouse Cursor: Show → 1, Hide → 0

- ShowState

- ◆ Device, Rendering Frame 상태 보기 설정

- ◆ Show→1, Hide →0

- {Set|Get}ClearColor

- ◆ 디바이스 배경색상





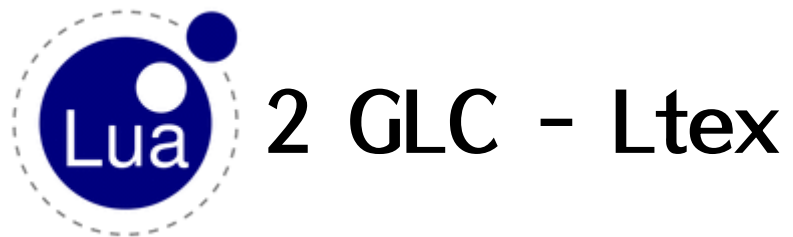
- GetTime, GetTickCount
 - ◆ 프로그램 시작 후 시간 가져오기
 - ◆ 1/1000 초 (Millisecond)
- WindowTitle
 - ◆ 윈도우 타이틀 문자열 설정
- DoFile
 - ◆ Lua의 dofile 대체
 - ◆ Embedded 에서 반드시 필요





```
-- main.lua: simple GLC Project  
function Lua_Create()  
    Lsys.ScriptFile(1, "main.lua")  
    Lsys.SetClearColor("0xFF006699")  
    return Lsys.CreateWindow(-1, -1, 480, 320, "My New Lua with GLC", 0)  
end  
  
function Lua_Init()  
    return 0  
end  
  
function Lua_Destroy()  
    return 0  
end  
  
function Lua_FrameMove()  
    return 0  
end  
  
function Lua_Render()  
    return 0  
end
```







- Ltex
 - ◆ 2D, 3D Texture 객체
- New
 - ◆ 텍스처 생성
 - ◆ PNG 권장
 - ◆ 이미지 파일 이름, 16bit(→ 2), ColorKey(16진수)
- Width
 - ◆ 이미지 너비
- Height
 - ◆ 이미지 높이





- Draw
 - ◆ 이미지 그리기
- DrawOne
 - ◆ 회전, 크기 변환이 없는 이미지 그리기
 - ◆ 가장 빠름
- DrawEx
 - ◆ 회전, 크기 변환이 있는 이미지 그리기
 - ◆ 가장 느림
- SetTexture
 - ◆ Device에서 텍스처 설정 → 3D





● Blend

- ◆ 알파 블렌딩 설정
- ◆ Source, Dest 동시에 설정

● Blend Option

- "ZERO"
- "ONE"
- "SRC_COLOR"
- "ONE_MINUS_SRC_COLOR"
- "SRC_ALPHA"
- "ONE_MINUS_SRC_ALPHA"
- "DST_ALPHA"
- "ONE_MINUS_DST_ALPHA"
- "DST_COLOR"
- "ONE_MINUS_DST_COLOR"
- "SRC_ALPHA_SATURATE"





```
function Lua_Init()  
    g_tex = Ltex.New("texture/mario.png")  
    ...  
    return 0  
end  
  
function Lua_Render()  
    -- 원본 이미지 영역 0, 0, 128, 64  
    -- 화면 200, 100 위치에 렌더링  
    Ltex.Draw(g_tex , 0, 0, 128, 64, 200, 100)  
    ...  
    return 0  
end
```







● New

- ◆ 폰트 파일에서 폰트 생성
- ◆ 이름, 폰트 높이
- ◆ 폰트 이름이 "default" 이면 GLC 라이브러리에 내장된 폰트 사용

● Draw

- ◆ 문자열 출력
- ◆ 문자열의 인덱스를 지정하면 해당 문자 영역만 렌더링

● Setup

- ◆ 문자열, 위치(x, y), 색상(16진수), 정렬 위치 설정
- ◆ 처음 설정에 유리





- Position

- ◆ 문자열 출력 위치(x, y)

- String

- ◆ 문자열 설정

- Color

- ◆ 색상 (16진수) 설정

- Scale

- ◆ 크기 변환(x, y) 설정
- ◆ 음수 값은 반전(Inversion)





- Align

- ◆ 화면에서 정렬

- Count

- ◆ 출력 문자 개수 반환

- Rect

- ◆ 해당 인덱스의 문자열 출력 영역 반환(left, top, right, bottom)





```
function Lua_Init()  
    g_Font = {}  
    g_Font[1] = Lfont.New("seoulHangangL.ttf", 24) -- Lfont.New("default", 24)  
    Lfont.Scale(g_Font[1], 1.0, 1.0);  
    ...  
end
```

```
function Lua_Render()  
    local x=0; local y=0; local r=0; local b=0  
    -- 문자열 설정  
    Lfont.Setup(g_Font[1], "안녕하세요 반갑습니다", 5, 50, "0xFF00FFFF")  
  
    local index = 2  
    x, y, r, b = Lfont.Rect(g_Font[1], index)  
    count = Lfont.Count(g_Font[1])  
    idx = -1  
    Lfont.Draw(g_Font[1])  
    Lfont.Draw(g_Font[1], idx, x, y + 15, "0XFF0000FF")  
    Lfont.Draw(g_Font[1], idx, r, y + 45, "0XFF0000FF")  
    ...  
end
```







- Lin

- ◆ Input Device Control
- ◆ Keyboard
- ◆ Mouse → Touch screen

- KeyboardAll

- ◆ [0, 255] 키보드 이벤트 → PC

- KeyboardOne

- ◆ [0, 255] 의 가상 키 중에 하나의 키에 대한 이벤트 → PC





- MousePos

- ◆ 마우스 위치: x: L-button, y: R-button, z: M-button

- MouseDelta

- ◆ 마우스의 이동 변화량: x, y, z

- MouseEvt

- ◆ 마우스 또는 터치 스크린 이벤트 → PC, Smart Device

- Key, Mouse Event

- ◆ NONE → 0

- ◆ DOWN → 1 한 번 누름

- ◆ UP → 2

- ◆ PRESS → 3 한 번 누름 포함 안함. 계속 눌려진 상태





```
function InitFrameMove()
```

```
    M.px, M.py, M.pz = Lin.MousePos()    -- 위치 받음
```

```
    M.lb, M.rb = Lin.MouseEvt()          -- left, right 이벤트 받음
```

```
    for i=1, 256 do                        --키보드 이벤트 받음
```

```
        K[i] = Lin.KeyboardOne(i-1)
```

```
    end
```

```
    ...
```

```
end
```





5 GLC – Lsmd





● LSmd

- ◆ Audio, Video Device Control
- ◆ Smart Device → Sound만 지원
- ◆ PC → DirectSound, 안드로이드 → OpenSL, iPhone → OpenAL

● New

- ◆ 사운드 생성

● 안드로이드 환경에서 bit rate, Hz의 영향이 심함

- ◆ 게임 사운드는 고 품질이 필요한 경우가 거의 없음
- ◆ 16bit - 22050Hz가 가장 적합





- Play
 - ◆ 사운드 재생
 - ◆ 무한 재생: -1
- Stop
 - ◆ 재생 정지
- Pause
 - ◆ 일시 정지
- Reset
 - ◆ 정지, 재생 위치 초기화





- Volume

- ◆ 볼륨 배율 [0.0, 1.0]
- ◆ float 형

- Repeat

- ◆ 반복 회수
- ◆ 무한: -1

- State

- ◆ 사운드 객체 상태
- ◆ STOP → 0, Play → 3
- ◆ Reset → 1, Pause → 2





```
function SmdLoad()
```

```
    smd = {}
```

```
    ...
```

```
    smd[3] = Lsmd.New( "bgm_03.wav")
```

```
end
```

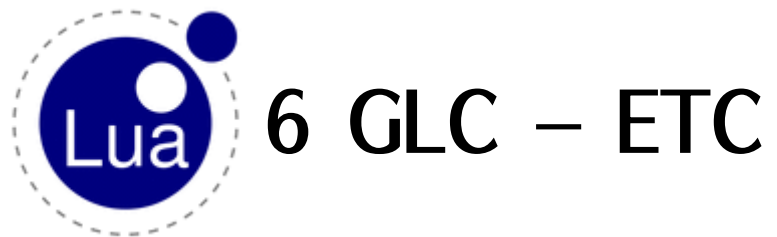
```
function Lua_Init()
```

```
    --Sound Load  
    SmdLoad()
```

```
    -- BGM Play  
    Lsmd.Play(smd[31])
```

```
    return 0  
end
```







- Rand
 - ◆ Random number
- GetTickCount
 - ◆ 프로그램 시작부터의 시간 반환 millisecond
- StrLen
 - ◆ 문자열 길이
- StrCmp
 - ◆ 문자열 비교





- 파일 입/출력
 - ◆ Smart Device는 파일 I/O 제한이 엄격
 - ◆ 파일의 사용에 따라 특정 폴더로 옮겨서 사용해야 함
 - ◆ "save" 문자열이 포함된 파일은 rw가 가능한 폴더로 옮김

- 함수
 - ◆ New/Release

 - ◆ Read, ReadLine
 - ◆ Write





```

function ReadCsv(fileName)                                -- CSV 파일 읽기
    local fr = Lfile.New(fileName)
    local cs_lst = {}
    local n = 0

    while true do
        local line = Lfile.ReadLine(fr)                  -- 한 줄 읽기
        if nil == line then break end

        local v = {}
        local i = 1
        for w in string.gmatch(line, "([^\r\n]+)") do
            v[i] = w; i = i + 1
        end

        --[[
        for i=1, #v do
            print(v[i] .. " ")
        end
        print("\n")
        --]]

        n = n + 1
        cs_lst[n] = v
    end

    Lfile.Release(fr)
    if 0 == n then
        return nil
    end

    return cs_lst
end
    
```





● 기타 하드웨어 장치

◆ Sensor

- 중력, 가속, Gyroscope, ...
- 함수: `Lhw.Sensor()`, `Lhw.Sensor("센서이름")`

◆ Haptic

- 진동 장치
- 함수: `Lhw.HapticPlay(milliseconds)`, `Lhw.Stop()`





-- default is "GRAVITY"

x, y, z = Lhw.Sensor()

senidx =1

sensor_type =

```
{  
    "ACCELEROMETER"  
    , "MAGNETIC_FIELD"  
    , "ORIENTATION"  
    , "GYROSCOPE"  
    , "LIGHT"  
    , "PRESSURE"  
    , "TEMPERATURE"  
    , "PROXIMITY"  
    , "GRAVITY"  
    , "LINEAR_ACCELERATION"  
    , "ROTATION_VECTOR"  
}
```

-- other sensor values

x, y, z = Lhw.Sensor(sensor_type[senidx])





millisecond = 100

if 1 == g_mouse.e then
 Lhw.HapticPlay(millisecond)
end

...

Lhw.HapticStop()



