ASM.JS

powered Application

오픈프론티어 1기 A.J @andrwj

> ASM.JS 개념 이해 및 실제 적용

> ASM.JS as Browser plugin

ASM.JS 101

논리 연산

논리 연산 &

```
> var A = 1611;

> console.log( A & 0);

0
```

논리 연산 &

0110 0100 1011 (1611)

& 0000 0000 0000 (0)

> 0

논리 연산 &

```
> var A = 1611;
> console.log( A & 1);
```

?

논리 연산 &

0110 0100 1011 (1611)

& 0000 0000 0001 (1)

> 1

논리 연산 | (or)

```
> var A = 1611;
```

```
> console.log( A | 1);
```

?

논리 연산 | (or)

```
0110 0100 1011  (1611)
|    0000 0000 0001  (   1)
-----------------------
>            1611
```

논리 연산 | (or)

```
> var A = 1611;
> console.log( A | 0 );
```

?

논리 연산 | (or)

```
0110 0100 1011  (1611)
|  0000 0000 0000  (   0)
---------------------
>          1611
```

논리 연산 | (or)

what ever ...

| 0000 0000 0000

> itself

Type ?

- Integer
- Float
- Pointer
- Array
- String

Size !

- Integer 1 ~ 4 byte
- Float 4 / 8 byte
- Pointer 4 / 8 byte
- Array ...
- String ...

Type Conversion

```
> int8_t   age;
> int16_t  year = 1611;
> age = year;
```

Type Conversion

0000 0110 0100 1011 (1611)

0000 0000 0100 1011 (75)

Type Conversion

```
> int8_t   age;
> int16_t  year = 1611;
> age = (int8_t)year;
```

Type Conversion

- Size match

Number in JS

> **double precision float**

bitwise ops

- **>>**, **<<**, **&**, **|**, **~**
- **32bit** 정수 대상

convert to Integer

>> 0 **(signed)**

<< 0

| 0

>>> 0 **(unsigned)**

<<< 0

Array index

```
> var array = [ ];

> array = [ 1.5 ];    // X

> array = [ NaN ];    // X

> array = [ -1 ];     // X
```

> index ==> 32bit unsigned integer
(§ 15.4.4.14)

convert to Integer

```
1      >>>0                  === 1
-1     >>>0                  === 0xFFFFFFFF
-1     >>0                   === -1
1.7    >>>0                  === 1
0x100000002 >>>0             === 2
1e21 >>>0                    === 0xDEA00000
1e21 >>0                     === -0x21600000
Infinity >>>0                === 0
NaN >>>0                     === 0
null >>>0                    === 0
‘1' >>>0                     === 1
‘x' >>>0                     === 0
Object >>>0                  === 0
```

Why ?

> 속도

> 최적화

> what else...

var type ?

- Number
- Boolean
- Undefined
- Null
- Array
- Object
- Date
- Function
- Regex

Runtime checking

```
> var something = {};
> [ ].push.call(something, 1611);
> something[‘show’] = function() { … }
> something = undefined;
```

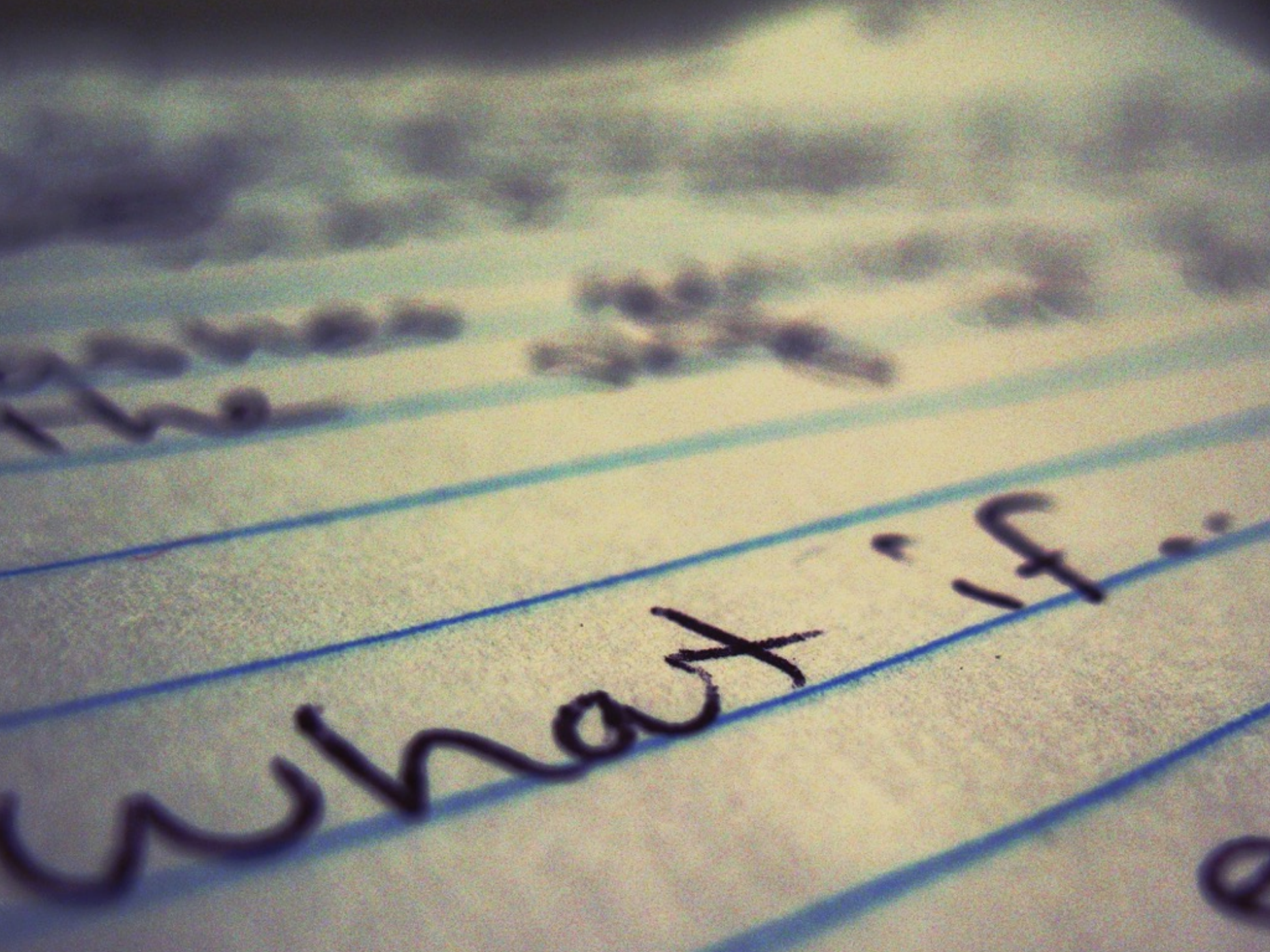
Numbers

- NaN
- double
- signed
- int
- signed
- unsigned

Reference Counting & GC

Closure

```
> for(var idx=0; idx<1000; idx++) {
    function(cnt)(
      setTimeout( function() {
        console.log( cnt );
      }, 500 );
    }(idx);
  }
```



can predict ..

```
> var year = 1611.0;
> var age = year >>> 0;
```

even return value

```
> function getYear(){
    return 1611.0;
  };

> console.log( getYear() | 0 );
```

double

```
> function getYear(){
    return +1611.0;
  };

> console.log( +getYear() );
```

also, array index

```
> var seasons = [“봄”, “여름”, 미세”, “먼지”]
> index = 3;
> console.log( seasons[ index >>> 0 ] );
```

먼지

array as Memory

```
> var BUFF  = new ArrayBuffer( 4 );

> var year = new Int16Array( BUFF );

> var age = new Int8Array( BUFF );
```

array as Memory

```
> var BUFF  = new ArrayBuffer( 24 );
> var v1 = new Uint32Array( BUFF, 0, 1 );
> var v2 = new Int8Array( BUFF, 4, 16 );
> var v3 = new Float32Array( BUFF, 20, 1 );
```

array as Memory

- NO reference counting
- predict Type

Emscripten

- Alon Zakai
- 2010년 8월 26일, 첫 커밋
- Transcompiler
- github.com/kripen/emscripten
- LLVM optimizer for JS

Emscripten processing overview

ASM JS

- Mozilla
- 2013년 5월 21일
- JavaScript Subset
- asmsj.org
- No Plug, No Extension
- C/C++
- Assembly for Web

```
function asmCode(global, env, buffer) {
  'use asm';
  var HEAP = new global.Uint8Array(buffer);
  function fib_like(x) {
    x = x|0;
    if ((x >>> 0) < 2) return HEAP[x]|0;
    return ((fib_like((x-2)|0)|0) + (fib_like((x-1)|0)|0))|0;
  }
  return fib_like;
}
```

```c
/* C or C++ */
int func(int *p) {
    intr = *p;
    return calc(r, r << 16);
}
```

```c
/* ASM.JS */
function func(p) {
    var r = HEAP32[ p >> 2 ];
    return calc(r, r << 16);
}
```

```
function tsa(d,e,f,g,h,j){d=d|0;e=e|0;f=f|0;g=g|
0;h=h|0;j=j|0;var
k=0,l=0,m=0,n=0,o=0,p=0,q=0,r=0,s=0,t=0,u=0,v=0,w
=0,x=0,y=0,z=0,A=0,B=0,C=0,D=0,E=0,F=0,G=0,H=0,I=
0,J=0;k=i;i=i+160|0;l=k+144|0;m=k+128|0;n=k+112|
0;o=k+96|0;p=k+80|0;q=k+40|0;r=k+32|0;s=k+16|
0;t=k;u=e+60|0;v=e+4|0;a:do{if((c[u>>2]|0)==3)
{c[l+0>>2]=c[563920>>2];c[l
+4>>2]=c[563924>>2];c[l+8>>2]=c[563928>>2];w=l
+12|0;a[w]=0;
}
```

Ports

https://github.com/emscripten-ports

SDL2

- Simple DirectMedia Layer
- Cross-platform development library
- Provide low level access:

  audio, keyboard, mouse, joystick
  graphics hardware
  via OpenGL and Direct3D.

- http://apps.webrox.fr/?page_id=8Change

Font

- SDL2_ttf
- FreeType

> http://coolwanglu.github.io/PDFium.js/viewer/

Sound

- Vorbis
- Ogg

Image

- SDL2_image
- libpng

others

- Bullet
- SDL2_net
- zlib

usage

```
> emcc -s USE_SDL_IMAGE=2    ...
> emcc -s -s USE_SDL=2 ...
> emcc —show-ports
Available ports:
   zlib (USE_ZLIB=1)
   libpng (USE_LIBPNG=1)
   SDL2 (USE_SDL=2)
   SDL2_image (USE_SDL_IMAGE=2)
   ogg (USE_OGG=1)
   vorbis (USE_VORBIS=1)
   bullet (USE_BULLET=1)
   freetype (USE_FREETYPE=1)
   SDL2_ttf (USE_SDL_TTF=2)
```

Pebble on Browser

http://localhost:8000


```
[INFO] main() 시작
[DEBUG] app_event_loop() ...
[INFO] 윈도우 로드: Alignment
[WARN] [NOOP] window_stack_push()
[DEBUG] 메인루프로 제어권 넘김
```

Emscripten limitations

- multi threads
- shared states
- aligned behaviour
- function pointer
- Exception
- setjump/longjump with stack replacing
- 64-bit int, C++ Exception (slow)

ASM.JS, Web Assembly

JS

DCMScope

http://github.com/andrwj/dcmscope

DCMScope v0.6.10 - NWJS/ASM.JS Powered DICOM Viewer

FAVORITES

- Home
- 샘플 파일
- Documents
- Pictures

root / Users / andrwj / Develops / node-webkit.applications / 00-DCMScope / dcm-files / CT-1

IM-0001-0063.dcm

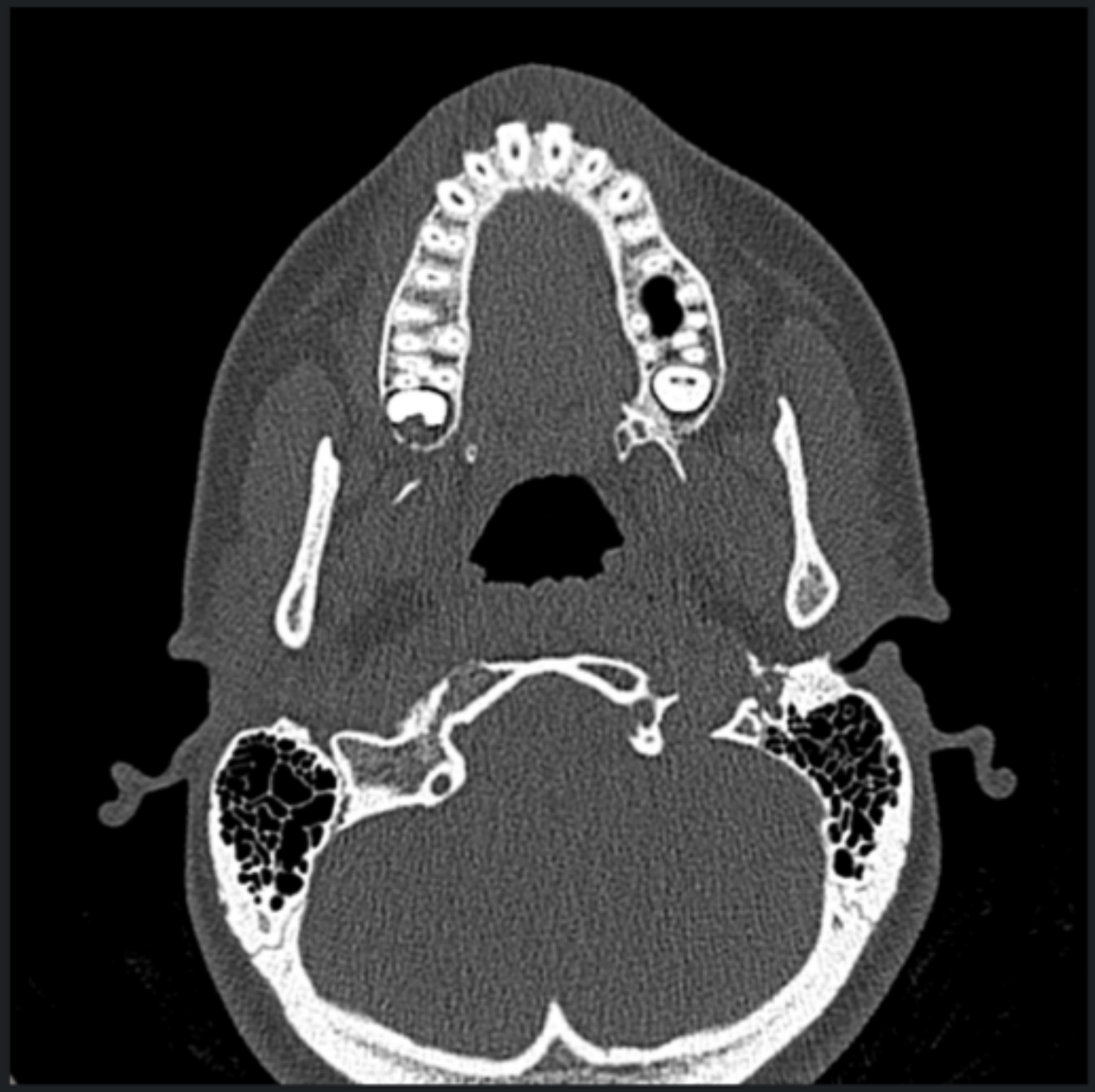
IM-0001-0064.dcm

IM-0001-0065.dcm

IM-0001-0066.dcm

Image size: 512 x 512
WL: 400 WW: 2000 (52 / 166)
X: 0 px Y: 0 px Value: 999

SOtNwu
T걜^Dental (Adulte)
Dental



문제점

- 웹에 최적화 되기 힘들다.
- 모바일보다 데스크탑용에 적합하다.
- **CPU** 자원을 활용하기 힘들다.
- 컴파일러와 툴킷에 의존성이 크다.

개선 방안

- **CPU**자원활용! (최우선)
- 모바일 환경을 위해 소스 크기 최소화

> ASM.JS as Browser plugin

?

오픈프론티어 1기 A.J @andrwj

2015.12.22