UBJSON

Submit Solution

Il formato di salvataggio di un software per la creazione di immagini memorizza i dati utilizzando una codifica binaria di JSON detta Universal Binary JSON. Rispetto alle specifiche di Universal Binary JSON che potete trovare qui, non è necessario gestire i tipi `int64`, gli high-precision number e l'*optimized format* per il tipo object.
L'optimized format per gli array deve essere gestito almeno per il caso di interi a 8 bit senza segno.

In particolare il formato prevede un oggetto che contiene un oggetto "canvas", ovvero l'area in cui si disegna, di cui vengono specificate "width" e "height", due interi che indicano rispettivamente la larghezza e l'altezza in pixel. Inoltre viene fornito un "background", un array di tre valori RGB da 0 a 255 che rappresenta il colore con cui riempire il canvas.

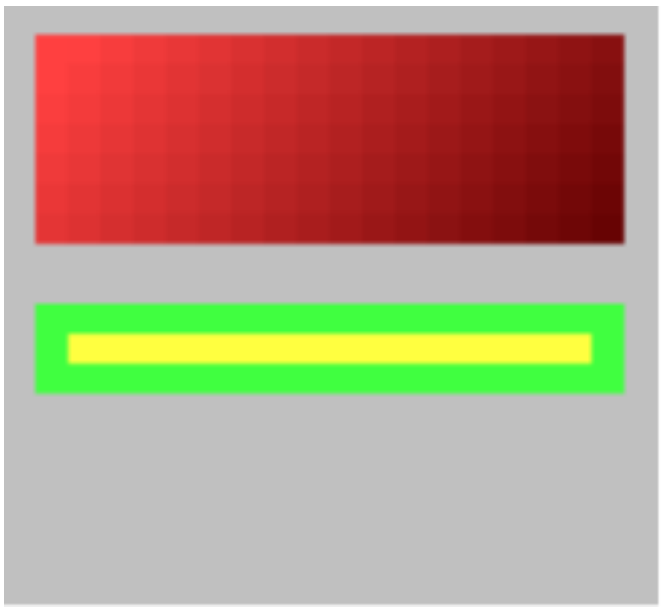
Segue poi un oggetto `elements`, in cui **l'ordine è importante**, che contiene una serie di elementi il cui tipo è definito dal nome. Sono definite diverse primitive grafiche, ad esempio "rectangle", "ellipse", "text", "line", ecc. Ognuna di queste primitive è poi un oggetto JSON che contiene diversi campi che ne definiscono gli attributi.

L'unica primitiva che ci interessa è la primitiva "image" che contiene i campi "x", "y", "width" e "height", che indicano la posizione in cui applicare l'angolo in alto a sinistra dell'immagine e le sue dimensioni tutte in pixel. Segue poi il campo "data", un array di width*height terne RGB (3 interi da 0 a 255).

Ad esempio, il file seguente:

```
{
    "canvas" : {
        "width" : 20,
        "height" : 20,
        "background" : [ 192, 192, 192]
    },
    "elements" : {
        "rectangle" : {
            "x" : 1,
            "y" : 10,
            "width" : 18,
            "height" : 3,
            "border-color" : [ 64, 255, 64],
            "fill-color" : [ 255, 255, 64]
        },
        "image" : {
            "x" : 1,
            "y" : 1,
            "width" : 18,
            "height" : 7,
            "data" : [
                255,64,64,254,64,64,247,61,61,239,58,58,232,55,55,225,52,52,217,49,49,210,46,46,202,
                43,43,195,40,40,188,37,37,180,34,34,173,31,31,165,28,28,158,25,25,151,22,22,143,19,19,136,16,16,
                255,64,64,248,61,61,241,58,58,234,55,55,226,52,52,219,50,50,212,46,46,204,44,44,197,
                41,41,189,38,38,182,35,35,175,32,32,167,29,29,160,26,26,153,23,23,145,20,20,138,17,17,130,14,14,
                250,62,62,243,59,59,236,56,56,228,53,53,221,50,50,213,47,47,206,44,44,199,41,41,191,
                38,38,184,35,35,176,32,32,169,29,29,162,26,26,154,23,23,147,20,20,139,18,18,132,15,15,125,12,12,
                245,60,60,237,57,57,230,54,54,223,51,51,215,48,48,208,45,45,200,42,42,193,39,39,186,
                36,36,178,33,33,171,30,30,164,27,27,156,24,24,149,21,21,141,18,18,134,15,15,127,12,12,119,9,9,
                239,58,58,232,55,55,224,52,52,217,49,49,210,46,46,202,43,43,195,40,40,187,37,37,180,
                34,34,173,31,31,165,28,28,158,25,25,151,22,22,143,19,19,136,16,16,128,13,13,121,10,10,114,7,7,
                234,55,55,226,52,52,219,49,49,212,46,46,204,44,44,197,41,41,189,38,38,182,35,35,175,
                32,32,167,29,29,160,26,26,152,23,23,145,20,20,138,17,17,130,14,14,123,11,11,115,8,8,108,5,5,
                228,53,53,221,50,50,213,47,47,206,44,44,198,41,41,191,38,38,184,35,35,176,32,32,169,
                29,29,162,26,26,154,23,23,147,20,20,139,18,18,132,14,14,125,12,12,117,9,9,110,6,6,103,3,3
            ]
        }
    }
}
```

descrive l'immagine seguente:



Il corrispondente file UBJ (visto in un editor esadecimale) diventa:

```
Offset(h) 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  7B 69 06 63 61 6E 76 61 73 7B 69 05 77 69 64 74  {i.canvas{i.widt
00000010  68 69 14 69 06 68 65 69 67 68 74 69 14 69 0A 62  hi.i.heighti.i.b
00000020  61 63 6B 67 72 6F 75 6E 64 5B 24 55 23 69 03 C0  ackground[$U#i.À
00000030  C0 C0 7D 69 08 65 6C 65 6D 65 6E 74 73 7B 69 09  ÀÀ}i.elements{i.
00000040  72 65 63 74 61 6E 67 6C 65 7B 69 01 78 69 01 69  rectangle{i.xi.i
00000050  01 79 69 0A 69 05 77 69 64 74 68 69 12 69 06 68  .yi.i.widthi.i.h
00000060  65 69 67 68 74 69 03 69 0C 62 6F 72 64 65 72 2D  eighti.i.border-
00000070  63 6F 6C 6F 72 5B 24 55 23 69 03 40 FF 40 69 0A  color[$U#i.@ÿ@i.
00000080  66 69 6C 6C 2D 63 6F 6C 6F 72 5B 24 55 23 69 03  fill-color[$U#i.
00000090  FF FF 40 7D 69 05 69 6D 61 67 65 7B 69 01 78 69  ÿÿ@}i.image{i.xi
000000A0  01 69 01 79 69 01 69 05 77 69 64 74 68 69 12 69  .i.yi.i.widthi.i
000000B0  06 68 65 69 67 68 74 69 07 69 04 64 61 74 61 5B  .heighti.i.data[
000000C0  24 55 23 49 01 7A FF 40 40 FE 40 40 F7 3D 3D EF  $U#I.zÿ@@þ@@÷==ï
000000D0  3A 3A E8 37 37 E1 34 34 D9 31 31 D2 2E 2E CA 2B  ::è77á44Ù11Ò..Ê+
000000E0  2B C3 28 28 BC 25 25 B4 22 22 AD 1F 1F A5 1C 1C  +Ã((¼%%´""...¥..
000000F0  9E 19 19 97 16 16 8F 13 13 88 10 10 FF 40 40 F8  ž..—.....ˆ..ÿ@@ø
00000100  3D 3D F1 3A 3A EA 37 37 E2 34 34 DB 32 32 D4 2E  ==ñ::ê77â44Û22Ô.
00000110  2E CC 2C 2C C5 29 29 BD 26 26 B6 23 23 AF 20 20  .Ì,,Å))½&&¶##¯
00000120  A7 1D 1D A0 1A 1A 99 17 17 91 14 14 8A 11 11 82  §.. ..™..‘..Š..,
00000130  0E 0E FA 3E 3E F3 3B 3B EC 38 38 E4 35 35 DD 32  ..ú>>ó;;ì88ä55Ý2
00000140  32 D5 2F 2F CE 2C 2C C7 29 29 BF 26 26 B8 23 23  2Õ//Î,,Ç))¿&&¸##
00000150  B0 20 20 A9 1D 1D A2 1A 1A 9A 17 17 93 14 14 8B  °  ©..¢..š..“..‹
00000160  12 12 84 0F 0F 7D 0C 0C F5 3C 3C ED 39 39 E6 36  ..„..}..õ<<í99æ6
00000170  36 DF 33 33 D7 30 30 D0 2D 2D C8 2A 2A C1 27 27  6ß33×00Ð--È**Á''
00000180  BA 24 24 B2 21 21 AB 1E 1E A4 1B 1B 9C 18 18 95  º$$²!!«..¤..œ..•
00000190  15 15 8D 12 12 86 0F 0F 7F 0C 0C 77 09 09 EF 3A  .....†.....w..ï:
000001A0  3A E8 37 37 E0 34 34 D9 31 31 D2 2E 2E CA 2B 2B  :è77à44Ù11Ò..Ê++
000001B0  C3 28 28 BB 25 25 B4 22 22 AD 1F 1F A5 1C 1C 9E  Ã((»%%´""...¥..ž
000001C0  19 19 97 16 16 8F 13 13 88 10 10 80 0D 0D 79 0A  ..—.....ˆ..€..y.
000001D0  0A 72 07 07 EA 37 37 E2 34 34 DB 31 31 D4 2E 2E  .r..ê77â44Û11Ô..
000001E0  CC 2C 2C C5 29 29 BD 26 26 B6 23 23 AF 20 20 A7  Ì,,Å))½&&¶##¯  §
000001F0  1D 1D A0 1A 1A 98 17 17 91 14 14 8A 11 11 82 0E  .. ..˜..‘..Š..,.
00000200  0E 7B 0B 0B 73 08 08 6C 05 05 E4 35 35 DD 32 32  .{..s..l..ä55Ý22
00000210  D5 2F 2F CE 2C 2C C6 29 29 BF 26 26 B8 23 23 B0  Õ//Î,,Æ))¿&&¸##°
00000220  20 20 A9 1D 1D A2 1A 1A 9A 17 17 93 14 14 8B 12    ©..¢..š..“..‹.
00000230  12 84 0E 0E 7D 0C 0C 75 09 09 6E 06 06 67 03 03  .„..}..u..n..g..
00000240  7D 7D 7D                                        }}}
```

Notate che gli array di interi tra 0 e 255 vengono codificati con il formato ottimizzato (con tipo e numero di elementi).

Si scriva un programma a linea di comando che accetti le seguenti opzioni:

```
ubj2ppm <input .UBJ> <output .PPM>
```

Il programma deve:

1. estrarre il canvas e produrre in output l'immagine PPM con il nome `canvas.ppm` e le dimensioni specificate, riempita con il colore di background indicato;
2. estrarre le immagini presenti all'interno del file e salvarle come PPM (`image1.ppm`, `image2.ppm`, ...);
3. applicare all'immagine di output le immagini estratte producendo una sola immagine finale ;
4. elencare tutti gli elementi presenti nel file con le rispettive proprietà e produrne un dump su console secondo il formato di esempio riportato di seguito.

Ad esempio, il caso precedente è nel file "caso01.ubj". Chiamando il programma con:

```
ubj2ppm caso01.ubj caso01.ppm
```

l'output su console sarebbe:

```
rectangle : x,y,width,height,border-color,fill-color,
image : x,y,width,height,data,
```

Nel file "caso01.ppm" si vedrebbe l'immagine seguente:

dove ovviamente manca il rettangolo, che non deve essere disegnato, come tutte le altre primitive grafiche eventualmente incontrate.