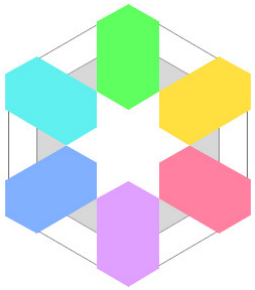
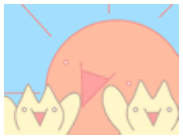
JPEG メタデータについて (バイナリを眺めながら)

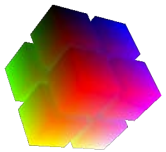


2015/10/16(Fri)

yoya@awm.jp



自己紹介



- プロフィール (https://osdn.jp/users/yoya/)
- Flash 動的生成を頑張ってたけど今は下火
  - https://osdn.jp/projects/swfed/
- 最近は ImageMagick の動向を追ってます
  - http://labs.gree.jp/blog/2013/12/9290/
- JavaScript 少し分かります
  - http://app.awm.jp/wafrag/debug.html
  - http://app.awm.jp/windchime/windchime.html

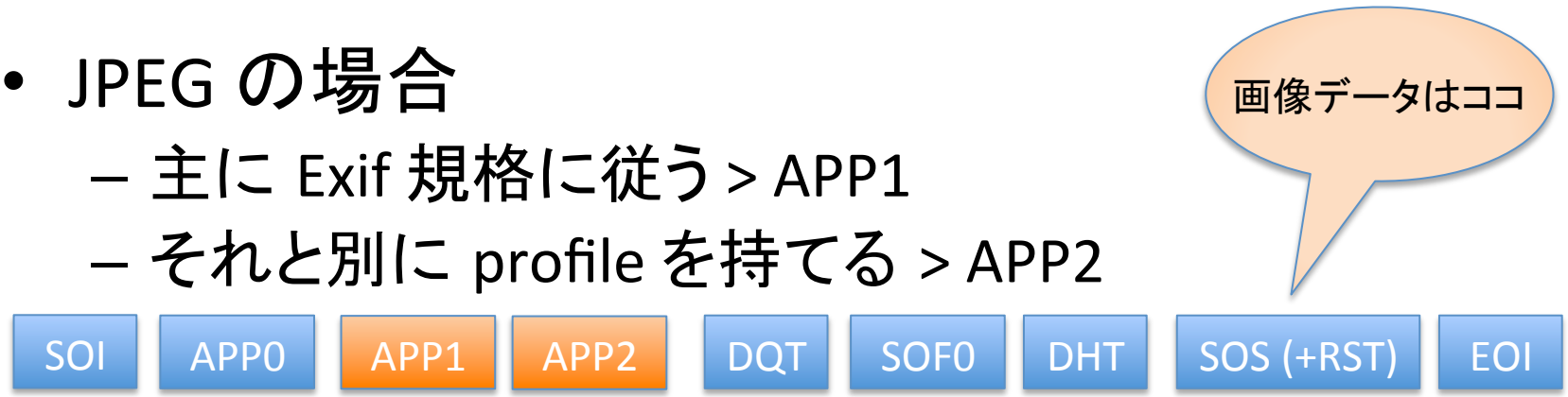
今日のお話

- (バイナリという言葉を知ってる前提で)
  - hexdump がお友達。0xED(Mac)や Stirling (Win) も
- JPEG のメタデータについて
  - プライバシーへの影響　　(こっちは余談)
    - 撮影日時と場所。サムネールにも注意
  - 表示への影響　　← こちらが本題
    - Exif Orientation
    - ICC Profile

画像ファイルは大抵メタデータを持つ

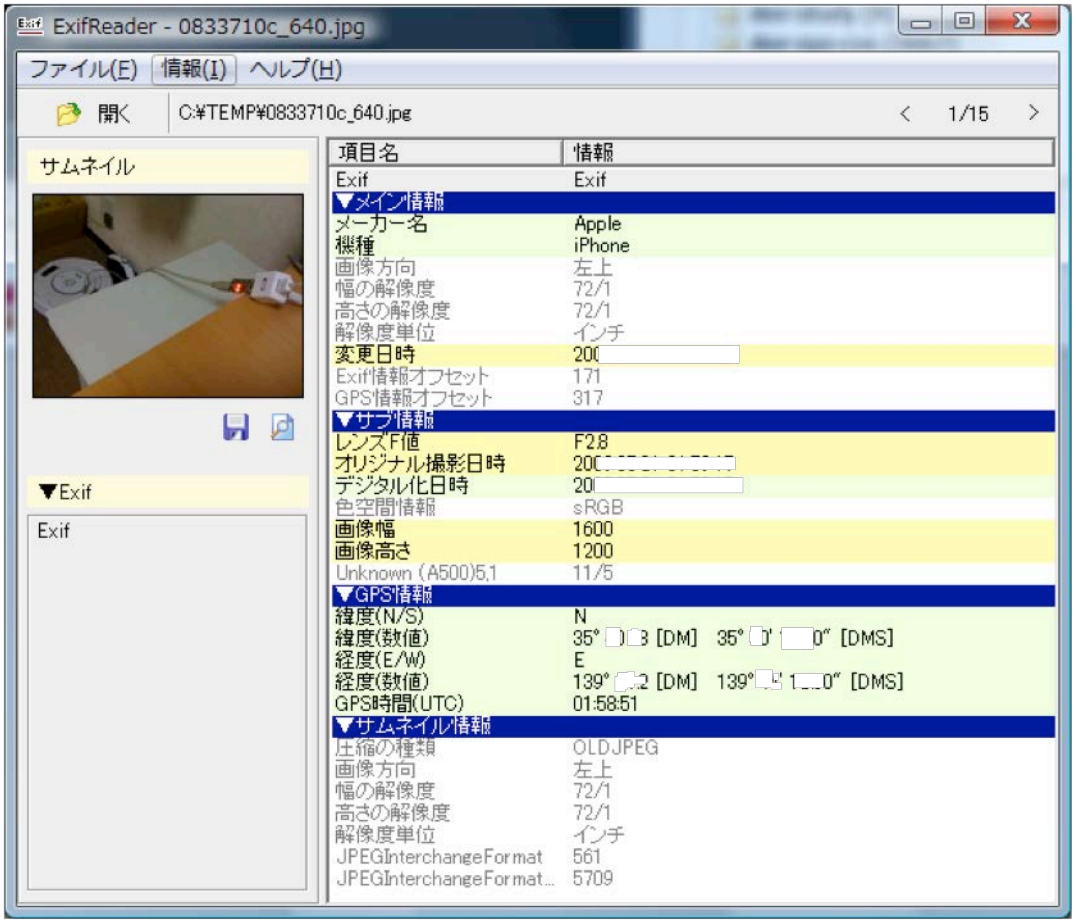
- 画像データ(RGB)だけでは足りない
  - 作成日付 (秒単位で)
  - 場所 (GPSで計測した緯度経度)
  - カメラの機種や撮影条件(露光とか)
  - サムネール画像

- JPEG の場合
  - 主に Exif 規格に従う > APP1
  - それと別に profile を持てる > APP2



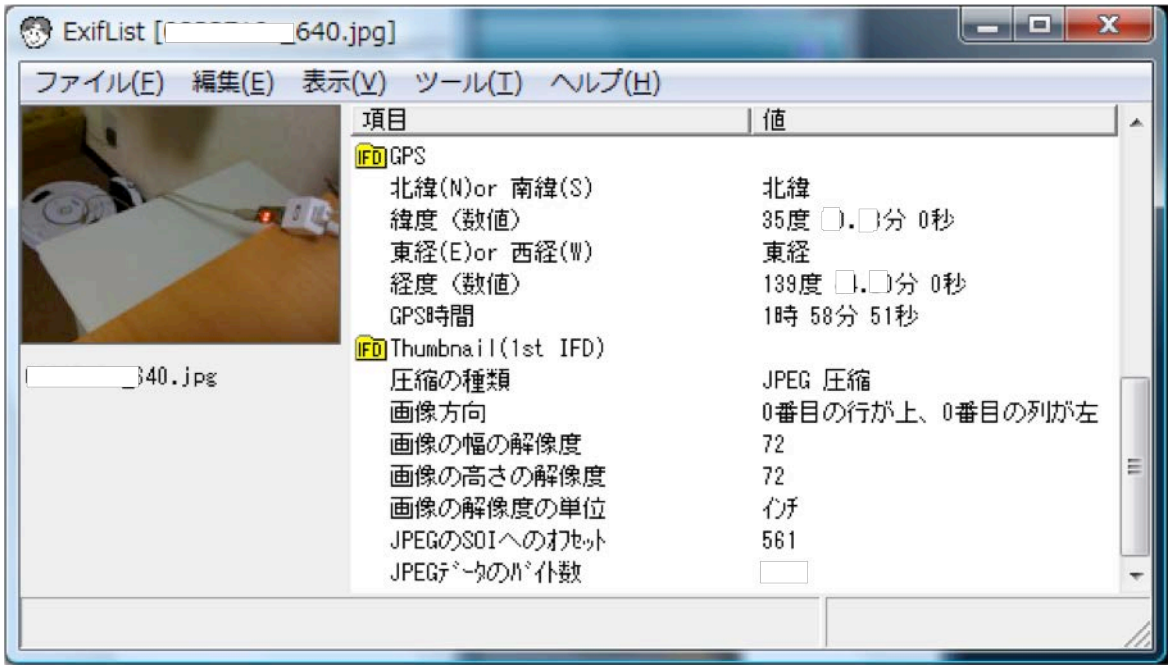
JPEG と Exif とプライバシー

- JPEGファイルは Exif という形式で画像以外の情報を付加できます



http://www.rysys.co.jp/exifreader/jp/

JPEG で位置バレ



- N35.XX.XX,
  E139.XX.XX

↓

(dms(60進)>degree変換)

=> 35.XXXXXX, 139.XXXXXX

※イメージです　→

(実際の場所から変えてます)

JPEG で顔バレ

- 編集して保存した時に、サムネールだけ前のままになる事がある。
  - paint.exe で顔を削ったけど。。

サムネを抽出すると…

JPEG って…

危険 …？

- ImageMagick で対処できる？ ↓

convert –strip Meta.jpg Output.jpg

正解はこっち

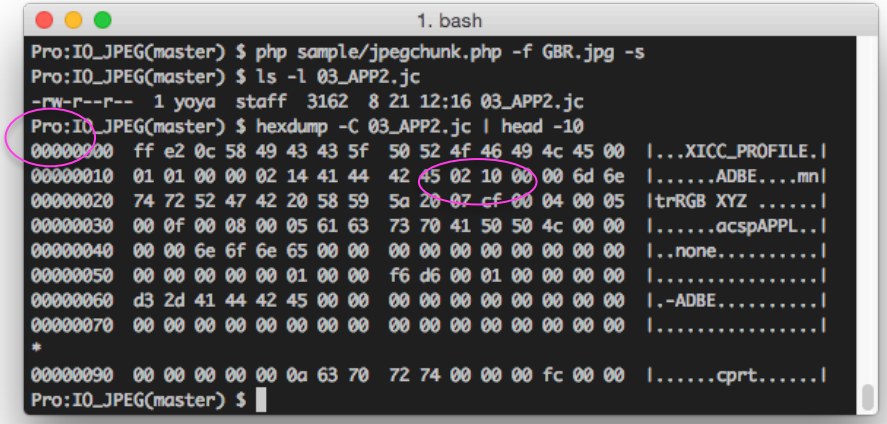
```
convert. Meta.jpg –auto-orient –profile sRGB.icc–strip Output.jpg
```

- −auto-orient ← 向きを補正する
- -profile sRGB.icc ← 色を補正する

これらについて詳しく説明

メタデータの中身を知る

- メタデータを無視するビューアがいる。(スマートフォンは色補正を完全無視。。)
- メタデータもそこそこサイズあるしプライバシーが危ないので削りたいけど、単に削ると表示が変わる



メタデータの表示への影響

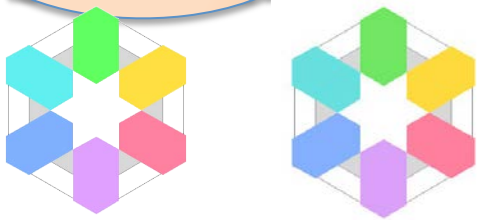
- JPEG は画像データ(YCbCr, RGB)をメタデータで補正して表示する必要あり
  - (無視するビューアもある)

©ricoh-imaging.co.jp

- 主に以下の 2つ。
  - Orientation (表示の向き)
  - ICC Profile (色補正)

撥影した時の
カメラの向き

メーカーや機器
によって
色の定義が違う

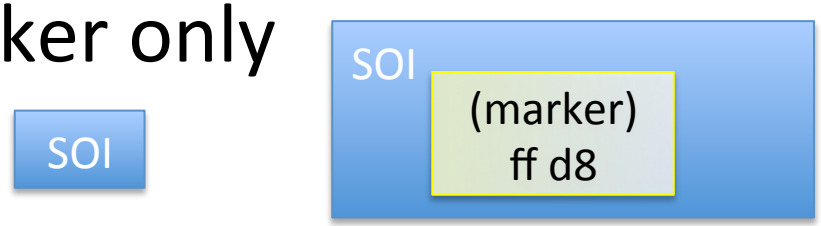


ガンマ値が違う
基準とする照明が違う
等々

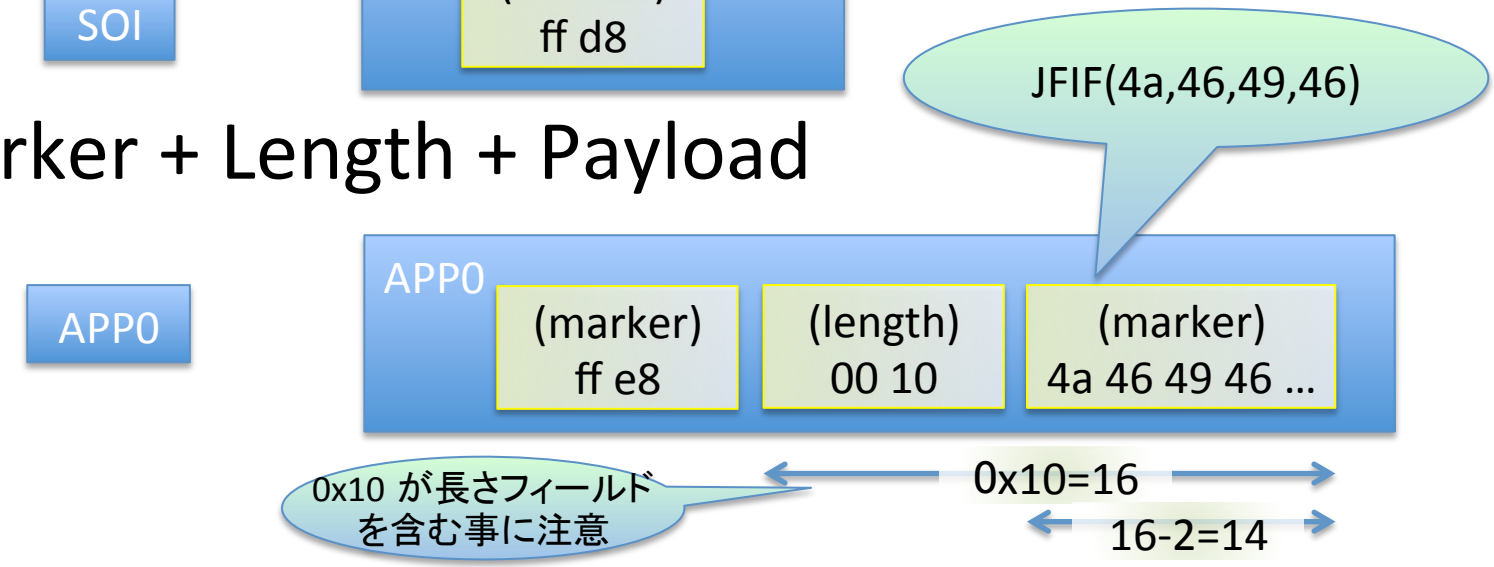
JPEG チャンク構造

- 実例

- Marker only

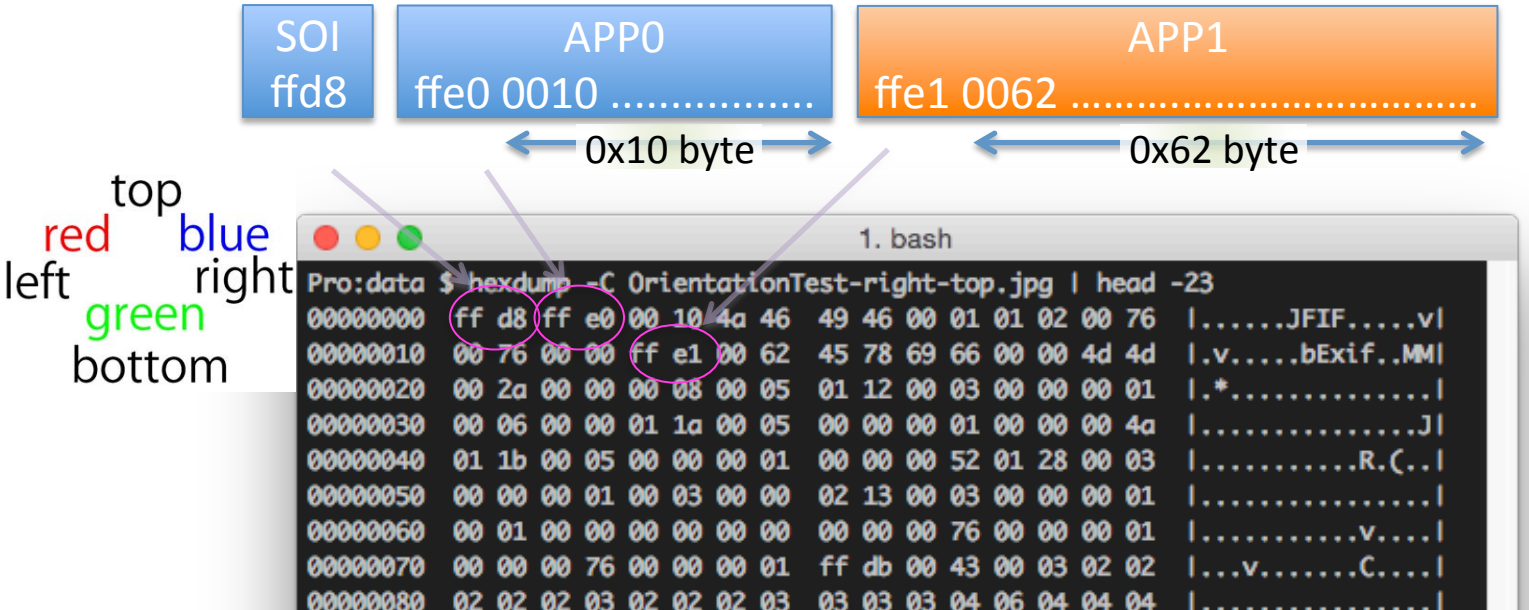


- Marker + Length + Payload



JPEGバイナリを見てみよう

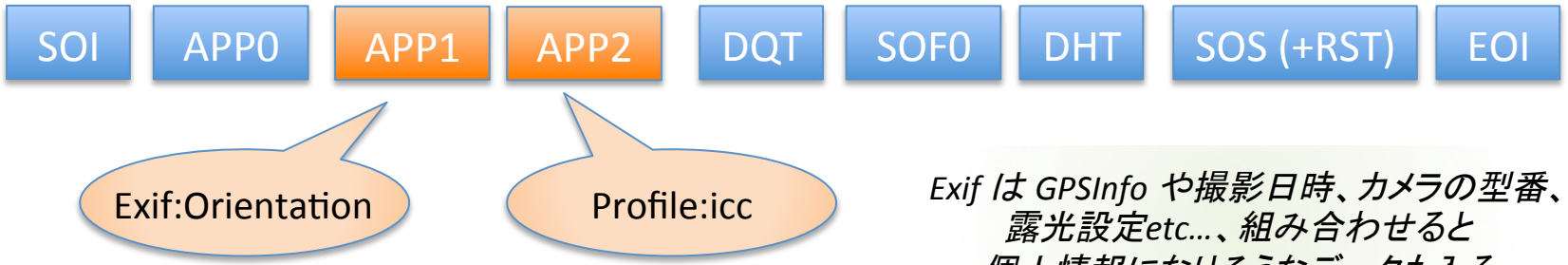
- SOI(開始),EOI(終了)はタグだけ
- それ以外はタグ＋データ長＋データ
  - データ長の数値はビッグエンディアン
  - 〃　　はデータ長フィールド分も含む



JPEG 構造

- JPEG バイナリフォーマット (チャンク構造)
  - 最小構成 (実画像データのみ)

  - メタデータ付き

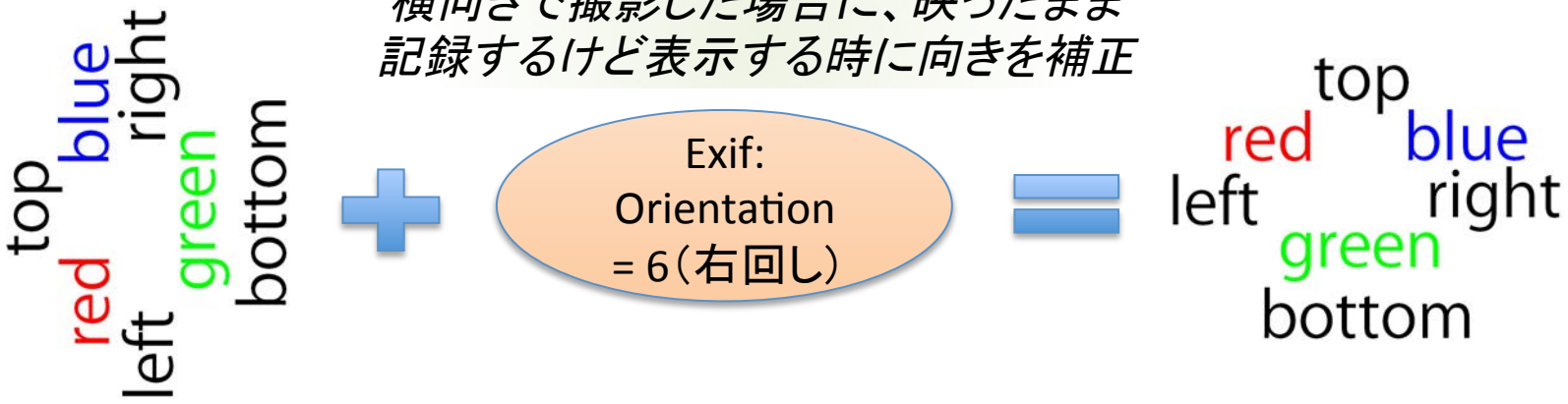


Exif は GPSInfo や撮影日時、カメラの型番、露光設定etc…、組み合わせると個人情報になりそうなデータも入る

Exif:Orientation とは？

- APP1 chunk の Exif:Orientation で撮影した時の向きを指定できる

横向きで撮影した場合に、映ったまま記録するけど表示する時に向きを補正

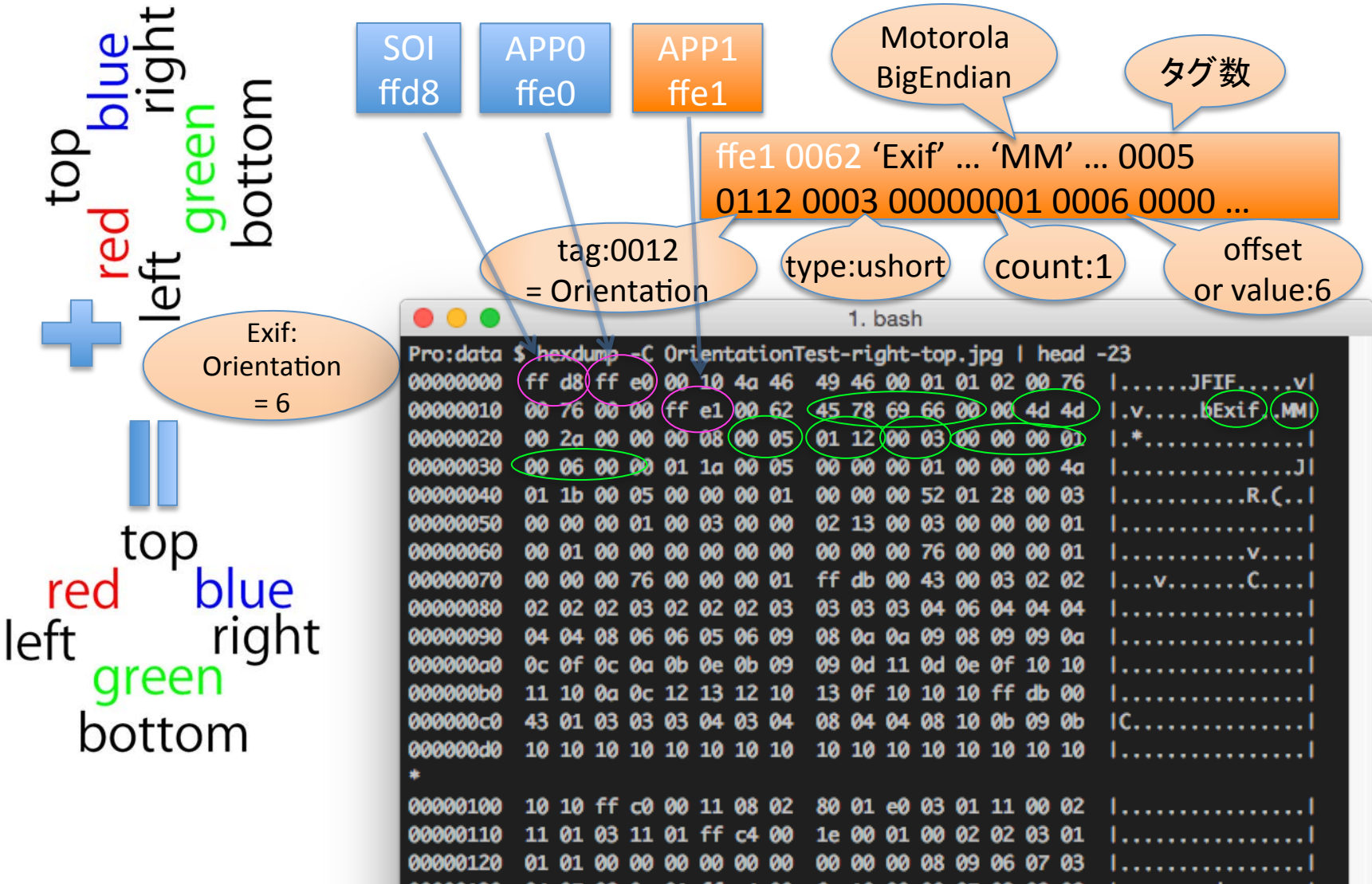


Exif Orientation の操作

- exiftool が便利
  - Orientation=6 を設定

```
$ exiftool -Orientation=6 -n OrientationTest-right-top.jpg
    1 image files updated
$
```

JPEGバイナリを見てみよう (Exif)



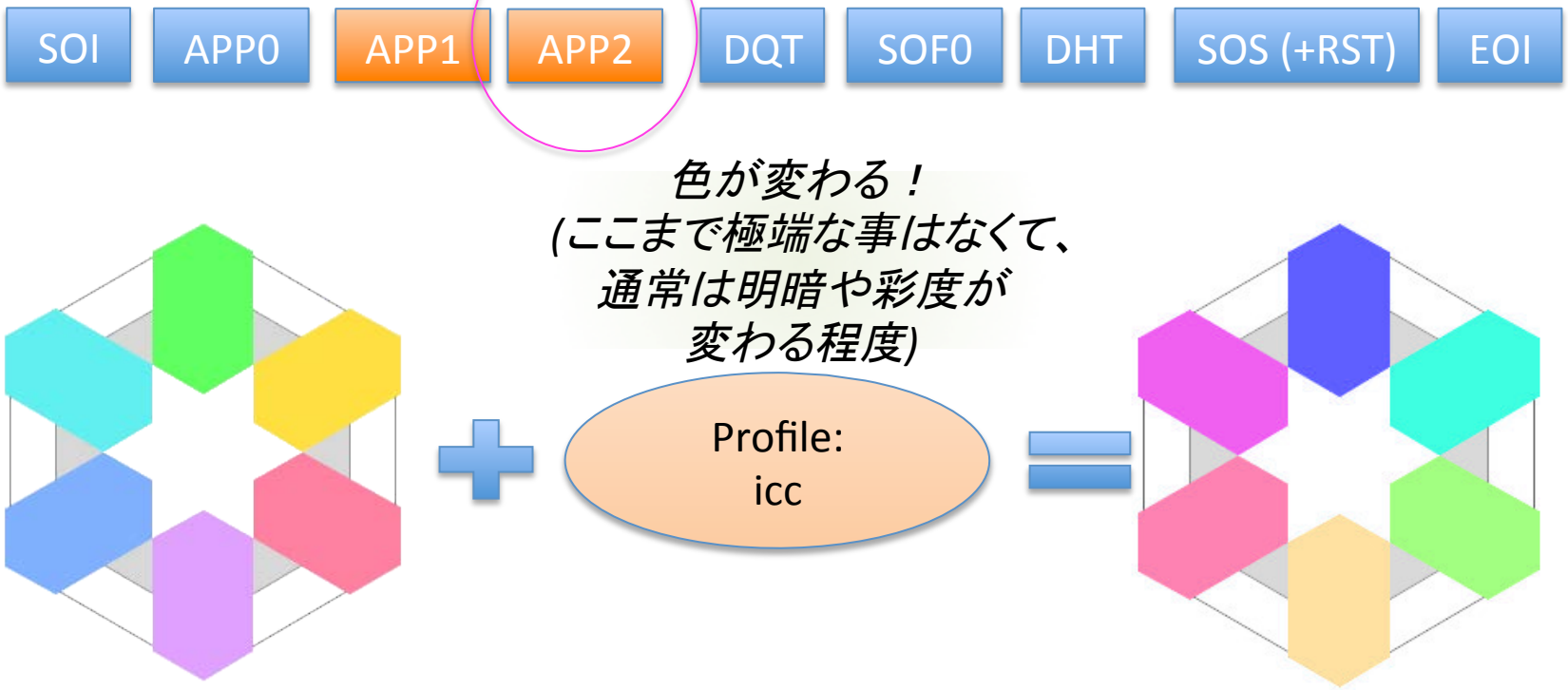
Exif の回転補正

- Exif Orientation に応じて画像を回転してくれる

```
$ convert Rotate6.jpg –auto-orient RotateNone.jpg
```

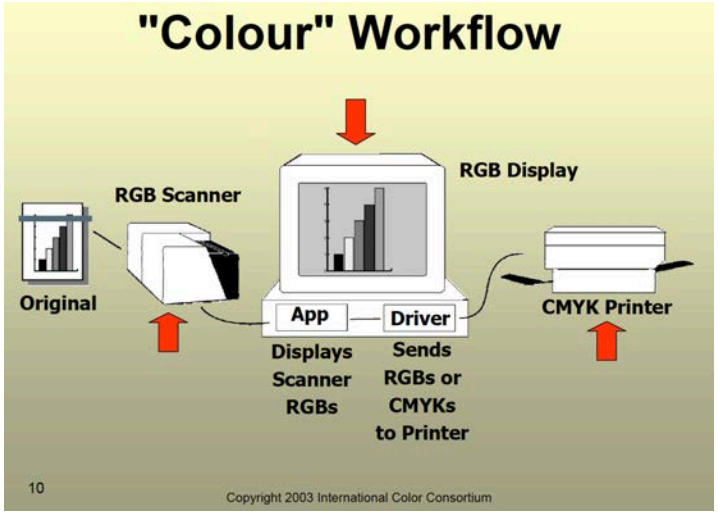
Profile:icc とは？

- APP2 の ICC プロファイルで色調補正



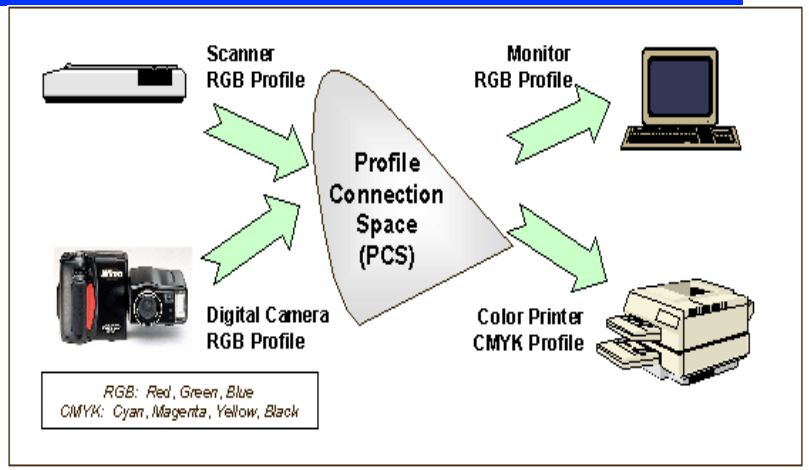
ICC プロファイルうんちく

- カラーマネジメントの規格
  - www.icc.org
  - デバイス毎に(主にセンサーやライトの物理特性の都合で) R,G,B と実際の色の対応が異なるので、それを吸収
  - モニタで確認した色と印刷した色が“機種によって”違うのは困る



ICC プロファイルの色変換

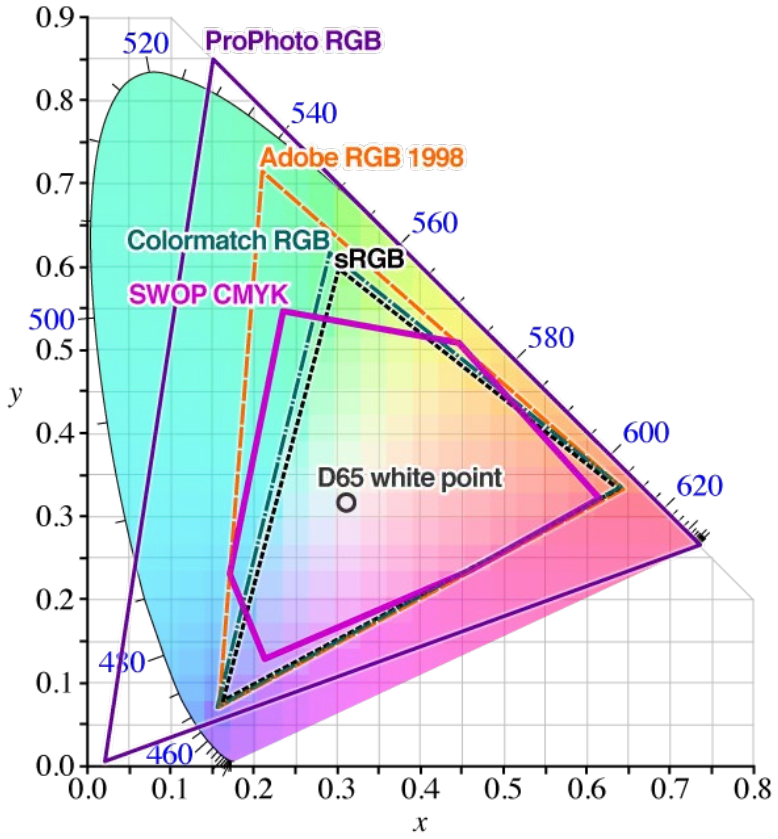
- カラーマネジメントの規格
  - デバイス間の全変換テーブルをメッシュで持たず、共通の色空間を定義して、そこ(PCS)との変換テーブルだけ持てば、相互接続できる
    - http://www.adobe.com/jp/support/techguides/color/colormodels/ciexyz.html



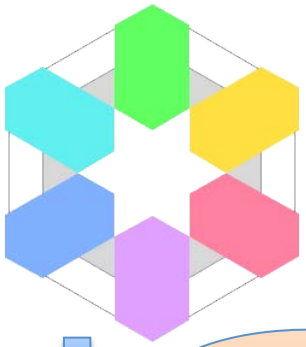
色域 (Gamut)

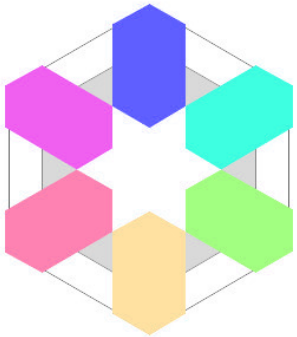
デジカメのセンサーや
モニタの発色が人間の
視覚の限界にどこまで
近づけるか

一般に普及している
モニタは sRGB なので
Web の画像はこれに
合わせてる (狭い)



JPEGバイナリを見てみよう (ICC)





SOI
ffd8

APP0
ffe0

APP2
ffe2

Profile:ICC_PROFILE

ffe2 0c58 ICC_PROFILE ….
00000214 ‘ADBE’ 0210 0000 ‘mntr’ …

Size
0x214

Type:
Adobe

Version
2.1

ディスプレイ
出力

Profile:icc

1. bash

```
Pro:data $ hexdump -C awmcorp_logo-GBR.jpg | head -10
00000000  ff d8 ff e0 00 10 4a 46  49 46 00 01 01 01 00 01  |......JFIF......|
00000010  00 01 00 00 ff e2 0c 58  49 43 43 5f 50 52 4f 46  |.......XICC_PROF|
00000020  49 4c 45 00 01 01 00 00  02 14 41 44 42 45 02 10  |ILE.......ADBE..|
00000030  00 00 6d 6e 74 72 52 47  42 20 58 59 5a 20 07 cf  |..mntrRGB XYZ ..|
00000040  00 04 00 05 00 0f 00 08  00 05 61 63 73 70 41 50  |..........acspAP|
00000050  50 4c 00 00 00 00 6e 6f  6e 65 00 00 00 00 00 00  |PL....none......|
00000060  00 00 00 00 00 00 00 00  00 01 00 00 f6 d6 00 01  |................|
00000070  00 00 00 00 d3 2d 41 44  42 45 00 00 00 00 00 00  |.....-ADBE......|
00000080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  |................|
*
Pro:data $
```

ICC Profile の抽出

- 抽出方法が見つからないので自作
  - https://github.com/yoya/misc/blob/master/go/imagick_profiles.go (Go から ImageMagick を呼ぶ)

```
$ go build imagick_profiles.go
$
$ imagick_profiles GBR.jpg
[app12 icc]
$ imagick_profiles GBR.jpg icc > GBR.icc
```

ICC Profile の設定

- ImageMagick の convert で可能 (MacOS で homebrew だと動かない、ports だと OK)

```
$ convert RGB.jpg -profile GBR.icc GBR.jpg
```

- 注意
  - 変換元JPEG の icc プロファイル有無で動作が変わる
  - ない場合
    - 単純に icc プロファイルを挿入するだけ
  - あった場合
    - 既に入っていた icc の色空間かsら新しく指定した icc の色空間に変換(補正)する

自分で色補正をやりたい

- ImageMagick だと遅い
- 以下の2つのツールが有名

- Little-CMS
  - http://www.littlecms.com/

- QCMS
  - https://people.mozilla.org/~jmuizelaar/qcms.git/
  - http://rockridge.hatenablog.com/entry/20090606/1244265122

Little-CMS

- ImageMagick のカラマネ色空間補正は実は Little-CMS を使ってるだけ
- 使い方超簡単

Step-by-step Example

Here is an example to show, step by step, how a client application can transform a bitmap between two ICC profiles using the lcms API.

```
#include "lcms2.h"

int main(void)
{
     cmsHPROFILE hInProfile, hOutProfile;
     cmsHTRANSFORM hTransform;
     int i;

     hInProfile  = cmsOpenProfileFromFile("HPSJTW.icc", "r");
     hOutProfile = cmsOpenProfileFromFile("sRGBColorSpace.icc", "r");

     hTransform = cmsCreateTransform(hInProfile,
                                     TYPE_BGR_8,
                                     hOutProfile,
                                     TYPE_BGR_8,
                                     INTENT_PERCEPTUAL, 0);

     cmsCloseProfile(hInProfile);
     cmsCloseProfile(hOutProfile);

     for (i=0; i < AllScanlinesTilesOrWatseverBlocksYouUse; i++)
     {
           cmsDoTransform(hTransform, YourInputBuffer,
                                      YourOutputBuffer,
                                      YourBuffersSizeInPixels);
     }

     cmsDeleteTransform(hTransform);

     return 0;
}
```

This is slightly different from the sample on 1.xx series, as Little CMS 2 allows you to close the profiles after creating the transform. On 1.xx you have to keep profiles open on all transform life, that is no longer required in Little CMS 2.x

QCMS

- Little-CMS が重たいので QCMS を作った
  - FireFox のエンジニアさんが作った。凄い。
  - Chrome もこれを使っているっぽい。
- 問題
  - ICC v2 のみ対応。(v4 未対応)
  - RGB 以外の色空間も微妙。(多分Webに特化)

- Safari は QCMS でなく ColorSync を使う

まとめ

- JPEG のメタデータに注意
  - プライバシー情報が入る事がある
  - 単純に削ると表示が乱れる
    - 回転するかも
    - 色味が変わるかも
- Imagemagick 使うと対処が簡単！
  - 高速化したい場合は LCMS や QCMS を利用して自前で頑張る

次回予告

- ピクセルの座標系について
  - グリッドの取り方 (ラスターとベクターで違う事が多い)
- 色の基本
  - 量子化 – A-D 変換
  - RGB, CMYK, HSL, YCbCr, CIE XYZ 等々 (虚色とは？)
- 減色
  - 均等法、中央値頻度法、頻度均等化法、etc…
- 補間アルゴリズム (賢いリサイズを)
  - NearestNeighbor, Bilinear, Cubic, etc…

お わ り

- 質問ください

質問まとめ

- PNG にも ICC が入る事があるの？
  - 入れる方法はあるけど自分は見た事がない
    - http://www.libpng.org/pub/png/spec/1.2/PNG-Chunks.html#C.iCCP
- 印刷で色がくすむけど良い方法は？
  - 色域外警告 (YMCK で表現できない領域)
    - http://psgips.blog24.fc2.com/blog-entry-406.html
  - 印刷の色域を広げる (高演色インク)
    - http://www.atomi.co.jp/h-uv/ink/index.html
    - 顔料やライトインクのマニアックな話は次回にでも

以上