Digitally signed by 佐藤 豊
Date: 2020.07.03 23:17:57 +09'00'

2020年7月3日 投稿者: SATOXITS

世界QRカウンター（への一歩）

社長：何か面白い事をしたいですね。

開発：たとえば。

社長：たとえば、わざわざそれを人間の手でやるの？という面白さが点描にはあるわけです。その逆に、普通は人間が手作業でやるものだと思われているものを機械的にミリ秒でやってしまう。

開発：まあ実用という言う意味では、設立当初から懸案の電子署名ですね。機械がハンコを押す。でもそれって、すでにSSLとともに実用化しているとは言えるかなと思います。サーバがミリ秒でコネクションにハンコを押すという。

社長：結果が人間にとって面白いというか、見える化されることが重要だと思うんです。結果が揮発性では無く、グローバルであることも。

社長：たとえば、以前少しかじったQRですが、あれは生成するのは今や簡単なんですが、通常は人間の手でパラメータを決めて生成するのに、そこそこの手間をかけてい

る。そうでは無く、たとえばHTTPでリクエストしたら、そこに書いたパラメータを反映したQRのPNGが返る。ミリ秒で。

開発：まあ、我が社の先端CGi技術で（笑）。50ミリ秒で返せると思います。工期は2時間くらいですかね。どういう付加価値を付けるのかが問題かなと思いますが。

社長：そこで思ったのが「世界でひとつのカウンター」です。世界で起きたことの全てに一連番号を付与する。タイムスタンプと電子署名付きです。で、そのシリアル番号と署名情報をQRで返す。署名情報はPNGのコメント的チャンクに入れる。

基盤：日付入りのスタンパーって味があって好きなんですが、あれがQRで出来ているという感じですかね。

社長：それで私は qrstamp というドメイン名を取りました。

経理：.online で経費節約でした。

開発：しかし、我社の500円サーバでは、1秒に10回程度、一日に100万回がせいぜいだと思いますけどね。

社長：one small step for man ですよ。百里の道も一歩から。

開発：じゃま、とりあえずやってみますか。

社長：というか私、お昼にビール2本はちょっときつかったです。また眠くなってしましました。寝てる間にやっといてくれるといいな。

開発：それがそうはいかないのが当社の仕組みになっております。

＊　＊　＊

開発：さて、QRの生成は go のパッケージで簡単にできることが以前の学習調査でわかっています。なので、この go スクリプトをコピペして、Go run！

開発：あ、パッケージが無いですね。こういう時は、go get パッケージ名、でゲットできることも知っております。go get。

開発：それでは再び Go！？ no such file … ああ、stap じゃなくて stamp ですね。では Go！

```
ns0% time go run qrstamp.go

real    0m0.258s
user    0m0.190s
sys     0m0.066s
```

開発：所要時間260ms。これは想定どおりです。では go build のちに qrstamp！

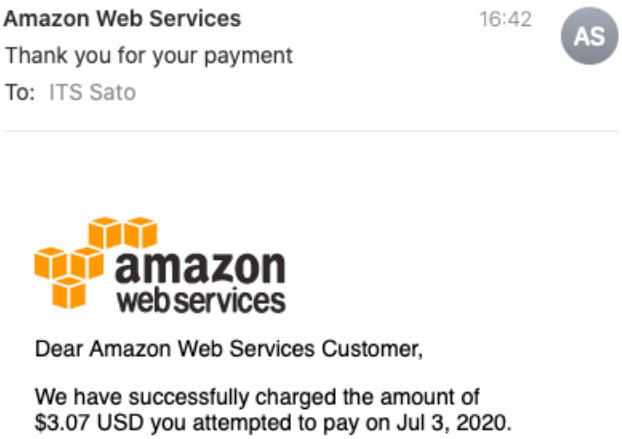
```
ns0% time ./qrstamp

real    0m0.006s
user    0m0.006s
sys     0m0.000s
```

開発：ということで、10ms以内の生成、png のファイルサイズは825バイトなので1パケットです。

開発：で、これを CGI から呼ぶわけですが…　うーん、人間がテストするにはいきなりPNGでダウンロードになっちゃうのはイマイチですね。やはりHTMLの中に表示したい。あと、.cgi で呼び出してダウンロードするとそのもののファイル名になるから .cgi って何型式ですかって怒る人もいると。そもそも何という名前のファイルにするかというのも考えどころです。これもパラメタで伝えるんでしょうね…

経理：ポーン。アマゾンからメールが来ました。



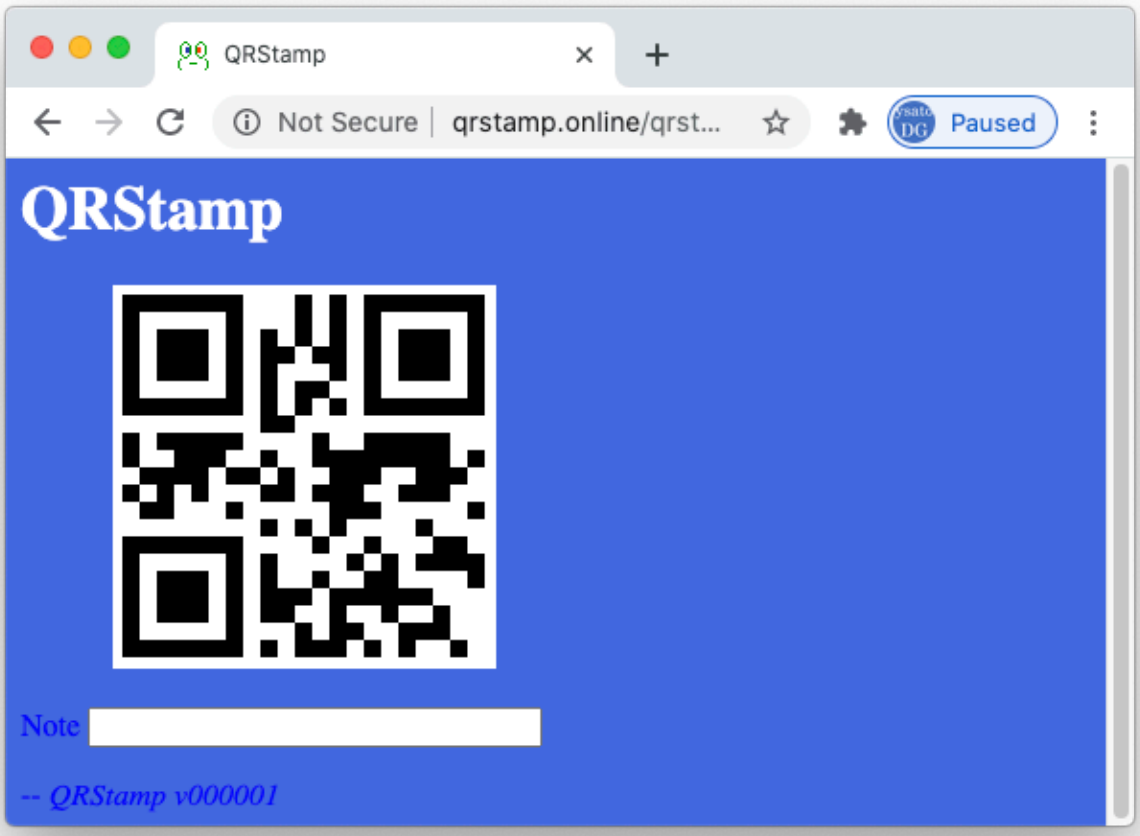
基盤：めっちゃ色々 attempt しましたしねー。よろこびもひとしおです。

開発：うーん、昨日作った点描のを流用しているんですが、このフォームはめちゃダサいな。やっぱ、点描でもそうだし、色のピッキングの共通ライブラリが必要ですね。

基盤：いくらでも既製のがあるんじゃないですかね。

社長：何でもいいので、とりあえず結果が見たいです。

開発：それではミニマルで。こんな感じですかね。



社長：おお、どれどれ、iPhoneのカメラで覗く…

社長：なんか Hello World のまんまですが。

開発：Goでコマンド引数を受け取る方法をまだ知らないのです。というか、タバコが切れました。買ってきましょうか。

社長：いやその前にそこをなんとか。コーヒーを入れて置きますから。

開発：えー、go command line argument で検索。。ああ、os.Args[] という配列ですね。これでどうでしょう？

開発：あれ？なんか反転してしまいましたね。使える文字に制約があるとか？

開発：おや？

開発：はてな？

社長：あ、未開封タバコ一箱発見。一服しましょう。

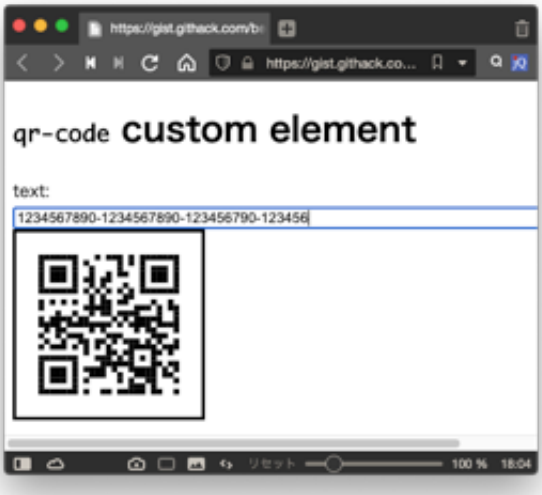
＊　＊　＊

社長：長さ制限とか文字セットでは無いですね。構文？

開発：手動テストに切り替えましょう。go run qrstamp.go 1234567890-… うーん、不思議。どういう規則でしょう？ 200×200 なのがまずいとか。400×400。変わらないですね。まあ200×200で入りきならいはずが無い。os.Argsの問題？でも無いですね。定数で与えても同じ。

社長：以前 JavaScriptでPNG 生成というのを見ましたね。あれではどうでしょう？

開発：問題なしですね。



基盤：それにしてもこの、一文字入れるごとに生成されるデモ、何度見てもインパクトありますね。

開発：ということは、Go の QR パッケージに何かありますね。MacOS版ではどうかな…

基盤：なんと、問題なし。

開発：go のバージョンは…

```
MacMini% uname -a
Darwin its02-macmini.local 19.5.0 Darwin Kernel
 Version 19.5.0: Tue May 26 20:41:44 PDT 2020;
root:xnu-6153.121.2~2/RELEASE_X86_64 x86_64
MacMini% go version
go version go1.14.3 darwin/amd64
```

```
ns0% uname -a
Linux ip-172-26-8-187 5.3.0-1028-aws #30~18.04.
1-Ubuntu SMP Mon Jun 22 15:48:21 UTC 2020 x86_6
4 x86_64 x86_64 GNU/Linux
ns0% go version
go version go1.10.4 linux/amd64
```

基盤：気が遠くなるほど古い？てか日付がわからない。

開発：一応 apt upgrade。うーん、1.10.4 が最新でございます。というか、こういう基本的っぽいプログラムにとって、昔の版とか関係しますかねぇ。

基盤：Goの本家を見てみます。うーん、go1.14.4 が最新だとありますね。tarball を落としてインストールしましょう。っていうか、これなら xso にもそのまま持ってけますね。おやー、scp めっちゃ遅い。1.1MB/s。ライトセールはポートごとにチョークを変えてるんですかね？

基盤：ではインストール手引の仰せのとおりに。というか、今現在の Go って何者？

```
ns0% which go
/usr/bin/go
ns0% ls -lL /usr/bin/go
-rwxr-xr-x 1 root root 7367408 Sep 27  2018 /usr/bin/go
```

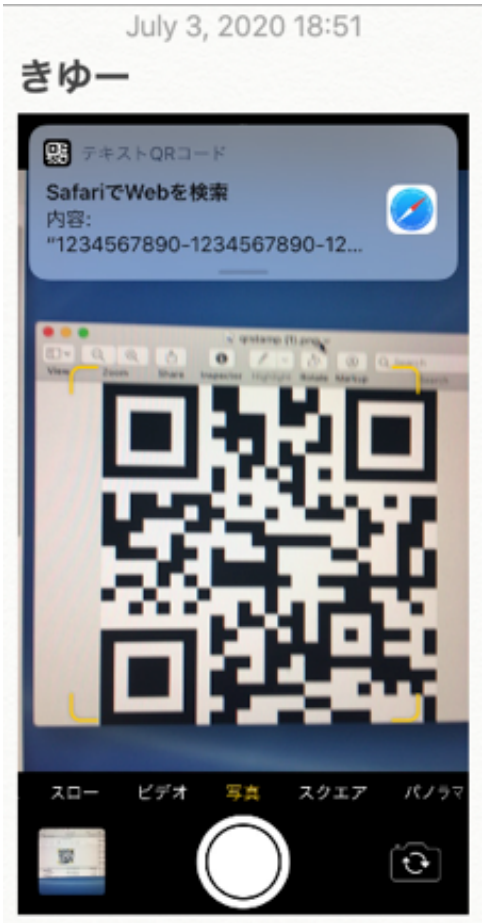
開発：骨董品ですね。

基盤：では仰せのとおりに /usr/local/go にインストールをば…　てか、-C ってオプション。こういうのがあるといいなというか、あるはずだとは思ってたんですよね…

```
ns0% /usr/local/go/bin/go version
go version go1.14.4 linux/amd64
```

基盤：では。

開発：では、まずは /usr/local/go/bin/go run qrstamp.go 1234…。できたかな？ダウンロードして見る。



基盤：ばっちりですね。

開発：じゃCGIに。

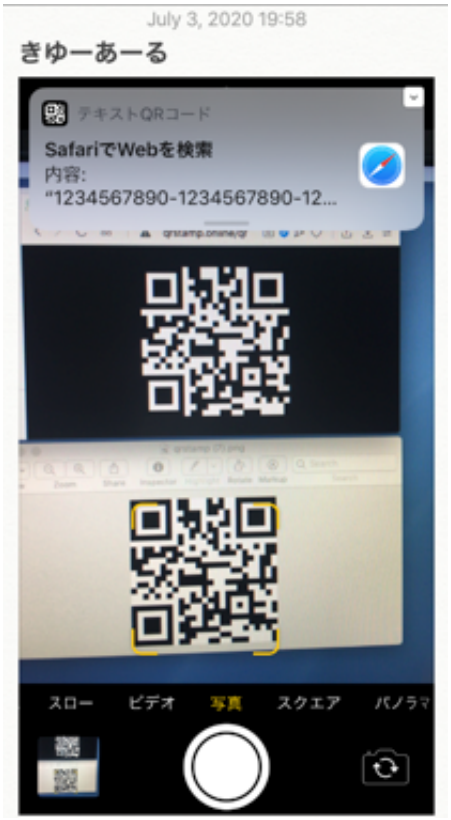
開発：・・・　あれー、だめだな。ライブラリとの整合性？GOROOT環境変数とかかな…

開発：だめだ。本家の人の解説が見つからない。わからん。

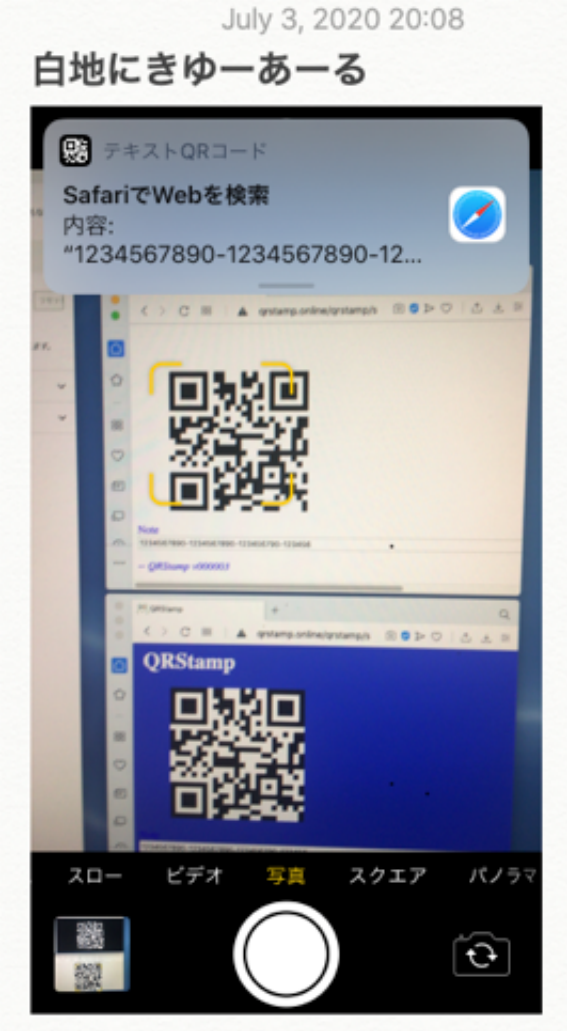
社長：ちょっといっぷくしましょう。

＊　＊　＊

開発：さっきうまく行ったように見えたのはダウンロードしたやつで…　カメラで撮ってみる。



開発：う、そうか。枠が無いと、データによっては識別できないということか。つまり、あの青背景が邪魔をしていたのではあるまいか。白背景にしてみるとどうか。



基盤：当たり。

開発：例題をもっと簡単にしましょう。1234567890abcde。

1234567890abcde のQRコード

開発：これは枠なしで生成されてしまうので、カメラで認識できない。一方、1234567890abcd。

1234567890abcd のQRコード

開発：これは枠が生成されるので、カメラで認識される。

基盤：QRって枠というか外との境界に弱いんですかね。意外。

開発：問題は、この Go の QR パッケージは外枠を生成する場合としない場合があること。意図したものでは無いように思われます。いわゆるバグってやつ？

基盤：常に枠を出さないというのなら、それはそれで一貫していると思うんですけどね。

社長：だいぶ見えてきましたね。いっぷくしましょう。

＊　＊　＊

開発：それにしても意外なところでつまづきました。

基盤：Goの版とか、寄り道をしました (^-^;

社長：でもそれが面白いです。自力インストールの勉強にもなったし。

基盤：まあ、ダウンロードして展開するだけということがわかりましたw

開発：最初、枠なしのやつが「反転している」ように見えたのですが、枠なしだとそれだけ人間の視覚的にも認識が動揺するってことですかね。

開発：Goのパッケージの仕様としては、枠あり、無しを明示指定できるのが良いのでしょうね。というかおそらく、QRの仕様として枠の仕様が規定されているんだろうと思います。

基盤：どういうコーディングなんですかね。っと。

```
ns0% cd ~/go/src/github.com/boombuler/barcode/qr
ns0% wc *
   66   186  1426 alphanumeric.go
   44   184   978 alphanumeric_test.go
   23    87   573 automatic.go
   30    90   806 automatic_test.go
   59   195  1784 blocks.go
   36   482  2332 blocks_test.go
  416  2321 15495 encoder.go
  134   298  2529 encoder_test.go
   29    79   639 errorcorrection.go
   60  1981 10157 errorcorrection_test.go
   56   160  1136 numeric.go
   26   132   783 numeric_test.go
  166   508  3309 qrcode.go
  126   340  2353 qrcode_test.go
   27    84   681 unicode.go
   18    80   473 unicode_test.go
  310  1540  9417 versioninfo.go
  157   562  4310 versioninfo_test.go
 1783  9309 59181 total
```

基盤：コンパクトですね。テスト用のコードを除けば 1,152行。

```
ns0% wc *go
   66   186  1426 alphanumeric.go
```

```
   23    87   573 automatic.go
   59   195  1784 blocks.go
  416  2321 15495 encoder.go
   29    79   639 errorcorrection.go
   56   160  1136 numeric.go
  166   508  3309 qrcode.go
   27    84   681 unicode.go
  310  1540  9417 versioninfo.go
 1152  5160 34460 total
```

開発：うーん、しかしこれ、仕様書を読みながらゼロからCで書いてたらと思うと、ぞっとしますね。

社長：そういえば、スケスケで色付きのとか虹色のQRってどう作るんでしょうね。

開発：PNGにする前にいじるか、PNGにしてからいじるかですね。二値画像ですから、どっちにしても簡単だと思いますが。使いでがあるっていう意味では、PNGの加工のほうが何にでも使えて良いですね。でも、せっかくだからQRのレベルでやるとすると…

```
func main() {
        // Create the barcode
        qrCode, _ := qr.Encode("Hello World", qr.M, qr.Auto)

        // Scale the barcode to 200x200 pixels
        qrCode, _ = barcode.Scale(qrCode, 200, 200)

        // create the output file
        file, _ := os.Create("qrcode.png")
        defer file.Close()

        // encode the barcode as png
        png.Encode(file, qrCode)
}
```

開発：qrCode というのを最終的に png.Encode してます。これは barcode.Barcode という型ですね。これは png.Encode が入力とする image.Image をラップしたもののようです。

開発：ところでこのパッケージ全体は barcode という名前なんですが、QRの他に通常のバーコードとか、PDF417とかを実装しているようです。え、PDFってあのAdobe

のPortable Document Format？って一瞬ドキドキしたのですが、Portable Data File の略だそうです。

https://ja.wikipedia.org/wiki/PDF417

社長：そういえば海外から何か購入した時に見たような記憶もあります。

開発：1991年に発案されたそうですから、2次元バーコードとしては、1994年発明のQRコードより少し先輩ですね。

開発：それで、image.Image というのは何かというと、こういう定義です。

```
type Image interface {
    // ColorModel returns the Image's color model.
    ColorModel() color.Model
    // Bounds returns the domain for which At can return non-zero col
    // The bounds do not necessarily contain the point (0, 0).
    Bounds() Rectangle
    // At returns the color of the pixel at (x, y).
    // At(Bounds().Min.X, Bounds().Min.Y) returns the upper-left pixe
    // At(Bounds().Max.X-1, Bounds().Max.Y-1) returns the lower-right
    At(x, y int) color.Color
}
```

開発：この辺を Go QR がどうしているかというと、qrcode.go のこのあたりのようです。

```
func (qr *qrcode) ColorModel() color.Model {
        return color.Gray16Model
}

func (qr *qrcode) Bounds() image.Rectangle {
        return image.Rect(0, 0, qr.dimension, qr.dimension)
```

```
}

func (qr *qrcode) At(x, y int) color.Color {
        if qr.Get(x, y) {
                return color.Black
        }
        return color.White
}
```

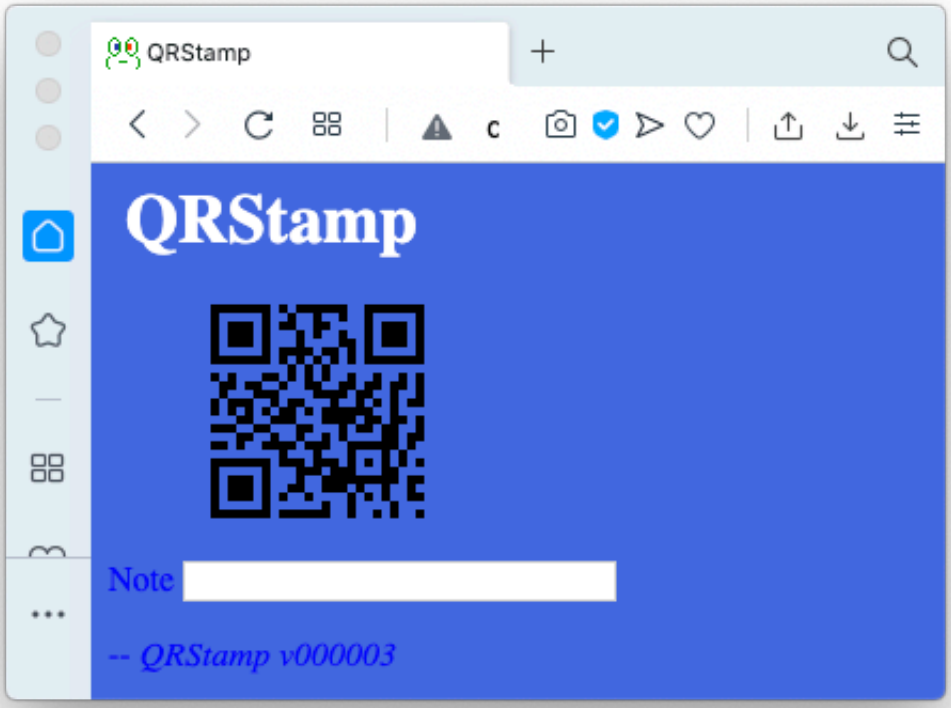
開発：これを見ると、At で color.White を返しているところを透明？にすれば良いのかなという気がします。その前にまず、Black を返しているところを Blue にしたらどうなるか。…　ああ、Blue なんて undefined だと。というか、そもそも Gray16Modelです。では White にしてみる。ムジナになりました（笑）では、White と Black を反転してみるとどうか。変な感じになりました。

「ITS more」のQRコードとそのネガ

開発：Black と White の他に color.Transparent というのがあるようです。これじゃないですかね。

```
Standard colors.
var (
    Black       = Gray16{0}
    White       = Gray16{0xffff}
    Transparent = Alpha16{0}
    Opaque      = Alpha16{0xffff}
)
```

開発：うーん、真っ黒になっちゃいますね。というか、カラーモデルが Gray16Model なんでどうにもならないのかも。color.RGBAModel というのにしてみます。おっ！



基盤：大成功、ぱちぱちぱちっ。

社長：これだと何とか、iPhone のカメラも認識します。

開発：白抜きではどうでしょう。

社長：iPhoneのカメラはこっちのほうが得意ですね。

基盤：うーん、面白すぎ。

社長：面白くて疲れたのでひと休みしましょう。

＊　＊　＊

社長：重ねてみましょう。

社長：ずうっとこれがやりたかったのです。構想6週間。

開発：制作半日、改変コード3行。

基盤：ほとんど関係ないところで迷って時間を潰してしまいましたね。

開発：世界征服カウンターまでたどり着けませんでした。

基盤：た〜か〜の〜つ〜め〜

社長：良いではないですか。我社にとって one giant leap です。

開発：点描空間の素材にも使いたいですね。

基盤：ていうか、しょっちゅう外でゴホゴホ咳をしているおばさん何者ですかね？気になるというか気に触るというか。

開発：あと、枠がなくなっちゃう件は、qr.Encode でパターンを作った後に、barcode.Scale ていうので指定サイズに拡大フィットさせているようなので、ちょうどパディング無しでサイズにフィットすることがあるということかなと思います。もともとフチを作るという考えは無かったのではないかとも思います。構造上昔のバーコードには要らなかったのかなと。

— 2020-0703 SatoxITS