리얼리티 스톤 앱 만들기3

Elly

변조 - LSB를 변경

Cover pixel: (167, 93, 27)
(**10100**111, **01011**101, **00011**011)

Secret pixel: (67, 200, 105)
(**010**00011, **110**01000, **011**01001)

Output pixel: (162, 94, 27) == (10100**010**, 01011**110**, 00011**011**)

코드로 작성해보자

1. ~~이미지를 RGB값으로 가져오기 (coverImage, secureImage)~~

2. coverImage를 원하는 bit만큼 RGB값 치환하기

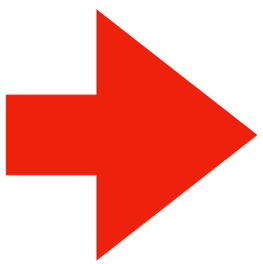
3. 치환한 pixel을 다시 이미지로 만들어서 view에서 보여주기

문제점

- 이미지 pixel 치환 식 오류

```
r: 11111111, g: 11111111, b: 11111111, a: 11111111
imageView clicked
2018-05-28 18:56:03.774903+0900 Just[45343:2306401] [discovery] errors encountered while
discovering extensions: Error Domain=PlugInKit Code=13 "query cancelled"
UserInfo={NSLocalizedDescription=query cancelled}
r: 11001101, g: 11010111, b: 11100000, a: 11111111
0
shiftBitMask: 22222222, imageAMasked: 65670, imageBShifted: 85946
tmp: 85950, r: 85950, g: 86150, b: 86718, a: 11111111
```

```swift
class RGBData {
    var r: CFBit
    var g: CFBit
    var b: CFBit
    var a: CFBit
    init(r: String, g: String, b: String, a: String) {
        self.r = CFBit(r)!
        self.g = CFBit(g)!
        self.b = CFBit(b)!
        self.a = CFBit(a)!
    }
```



```swift
class RGBData {
    var r: UInt8
    var g: UInt8
    var b: UInt8
    var a: UInt8
    init(r: UInt8, g: UInt8, b: UInt8, a: UInt8) {
        self.r = r
        self.g = g
        self.b = b
        self.a = a
    }
```

2. coverImage의 RGB값 하위 bit를 치환하기

```
func makebitMixing(imageA: [RGBData], imageB: [RGBData], bit: Int) -> [RGBData] {
    // 두 이미지 중 작은 값을 기준으로
    var imageOfMixing: [RGBData] = []
    let bitMask: UInt8 = 255

    for index in 0..<imageA.count {
        let resultR = imageA[index].r & bitMask << bit | imageB[index].r >> (8 - bit)
        let resultG = imageA[index].g & bitMask << bit | imageB[index].g >> (8 - bit)
        let resultB = imageA[index].b & bitMask << bit | imageB[index].b >> (8 - bit)
        let resultA = imageA[index].a
        imageOfMixing.append(RGBData.init(r: resultR, g: resultG, b: resultB, a: resultA))
    }
    return imageOfMixing
}
```

ex, **imageA r : 11011011** **bitMask : 11111111** **bit : 3**
imageB r : 01000011

```
imageA[index].r & bitMask << bit
```
: 11011000

```
imageB[index].r >> (8 - bit)
```
: 00000010

Result : 11011010

3. 치환한 pixel을 다시 이미지로 만들어서

ImageView에서 보여주기

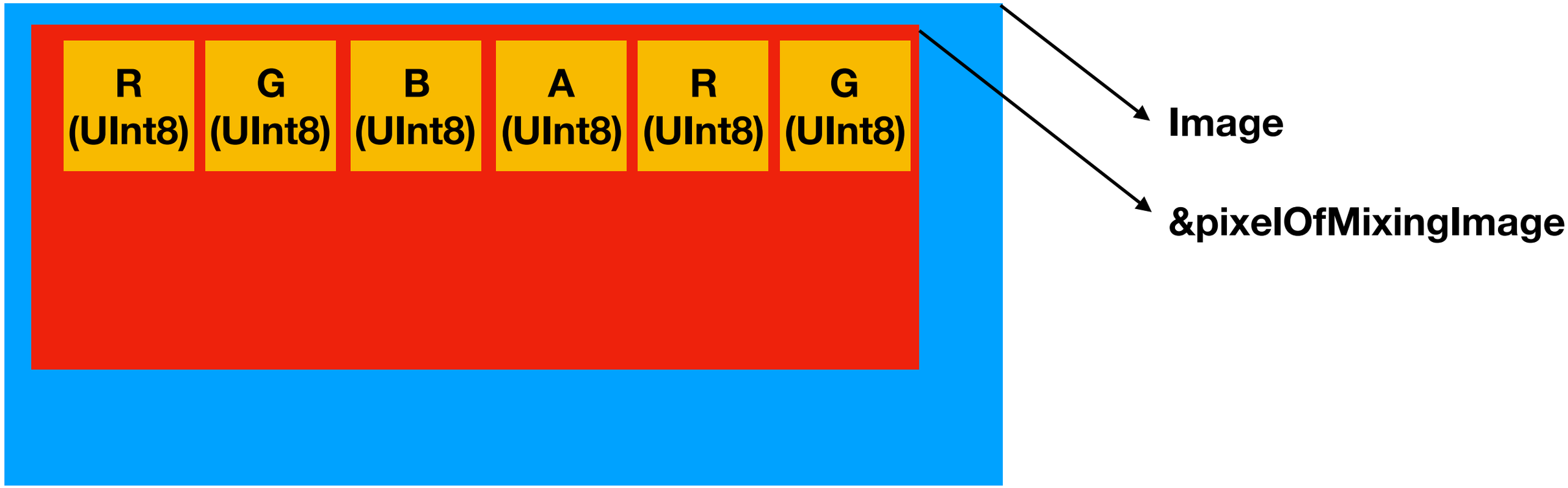
- bitmap의 정보를 담고 있는 CGContext를 만들고

```
guard let bitmapContext = CGContext(data: &pixelsOfMixingImage,
                                    width: width,
                                    height: height,
                                    bitsPerComponent: Int(bitsPerComponent),
                                    bytesPerRow: Int(bytesPerRow),
                                    space: colorSpace,
                                    bitmapInfo: CGImageAlphaInfo.premultipliedLast.rawValue) else {
    return coverImage
}
```

- CGImage를 생성 후, ImageView에 보여주기 위해 UIImage로 변환

```
guard let image = bitmapContext.makeImage() else {
    return coverImage
}
cgImage = image
return UIImage(cgImage: cgImage!)
```

여기서 배운점

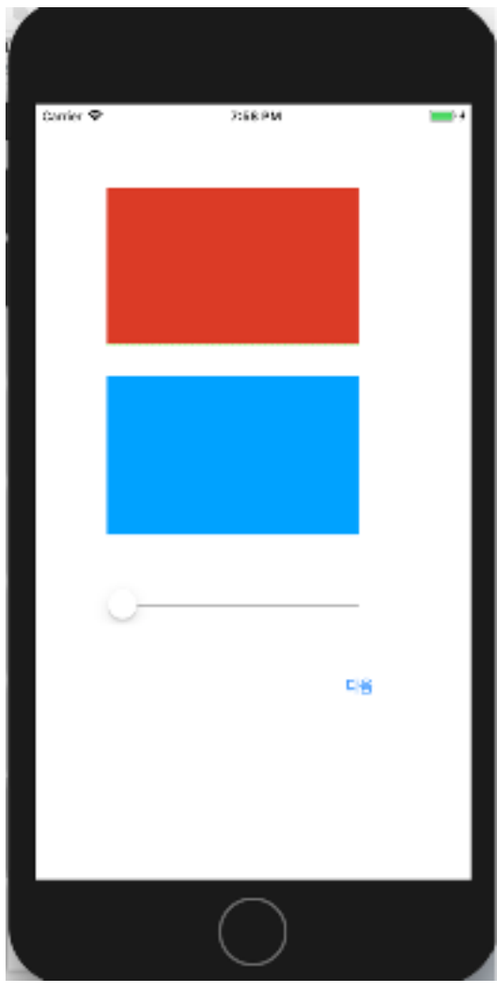



```swift
func makeRGBAPixel(rgbData: [RGBData]) -> [UInt32] {
    var pixelsOfMixingImage: [UInt32] = []
    for data in rgbData {
        let mixingImage: UInt32 = UInt32(UInt32(data.r)<<24 |
            UInt32(data.g)<<16 | UInt32(data.b)<<8 | UInt32(data.a))
        pixelsOfMixingImage.append(mixingImage)
    }
    return pixelsOfMixingImage
}
```

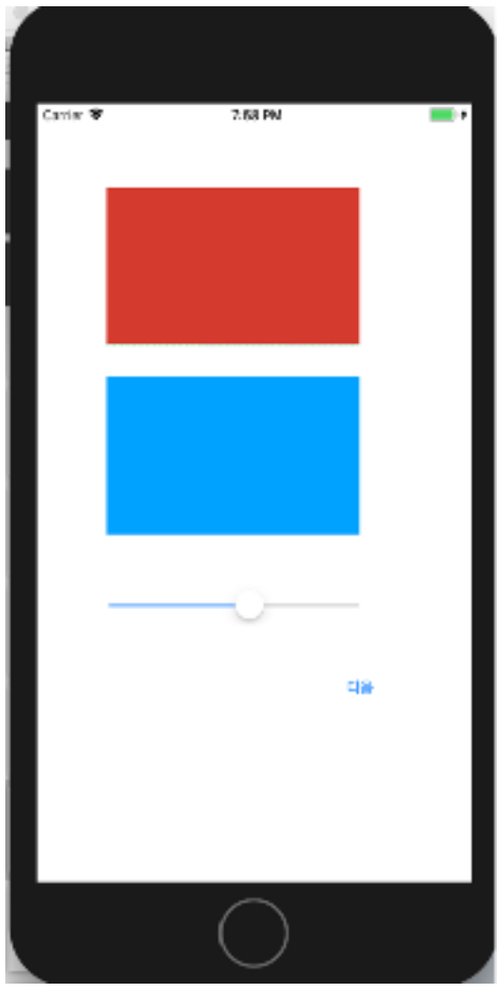
```swift
func makeRGBAPixel(rgbData: [RGBData]) -> [UInt8] {
    var pixelsOfMixingImage: [UInt8] = []
    for data in rgbData {
        pixelsOfMixingImage.append(data.r)
        pixelsOfMixingImage.append(data.g)
        pixelsOfMixingImage.append(data.b)
        pixelsOfMixingImage.append(data.a)
    }
    return pixelsOfMixingImage
}
```

실행 화면

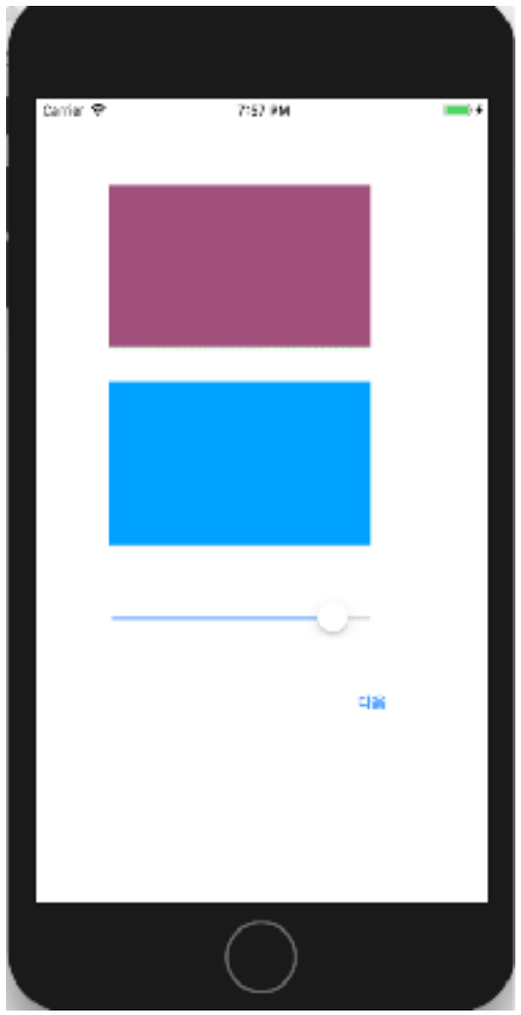
```
0
r: 11011011, g: 00111011, b: 00100110
r: 01000011, g: 10100001, b: 11111000
r: 11011011, g: 00111011, b: 00100110
```



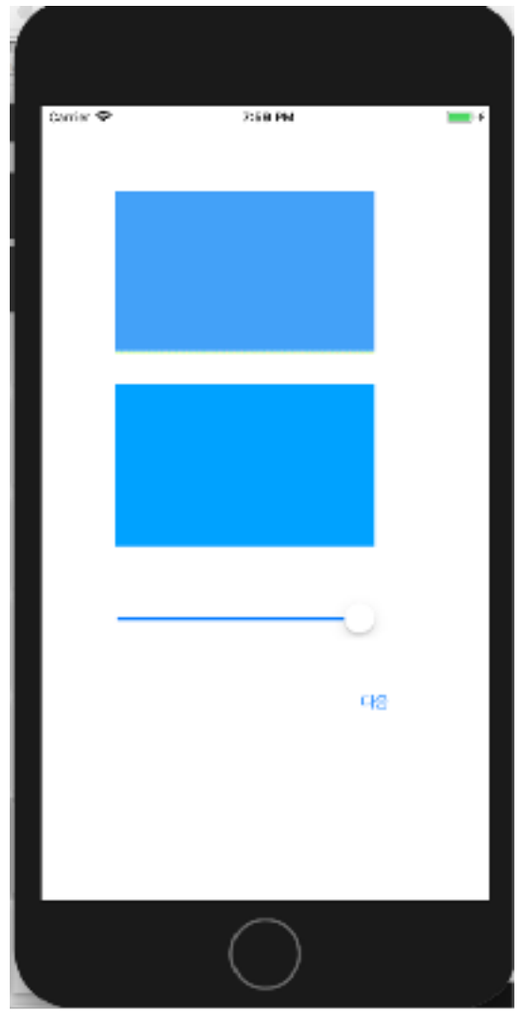
```
4
r: 11011011, g: 00111011, b: 00100110
r: 01000011, g: 10100001, b: 11111000
r: 11010100, g: 00111010, b: 00101111
```



```
6
r: 11011011, g: 00111011, b: 00100110
r: 01000011, g: 10100001, b: 11111000
r: 11010000, g: 00101000, b: 00111110
```



```
8
r: 11011011, g: 00111011, b: 00100110
r: 01000011, g: 10100001, b: 11111000
r: 01000011, g: 10100001, b: 11111000
```



본격 App 개발 Start!

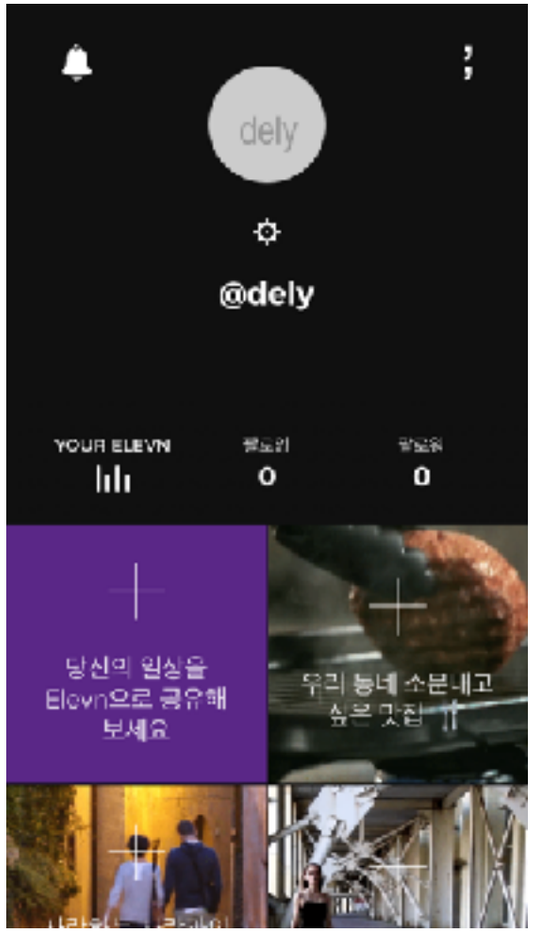
하기 전에.....

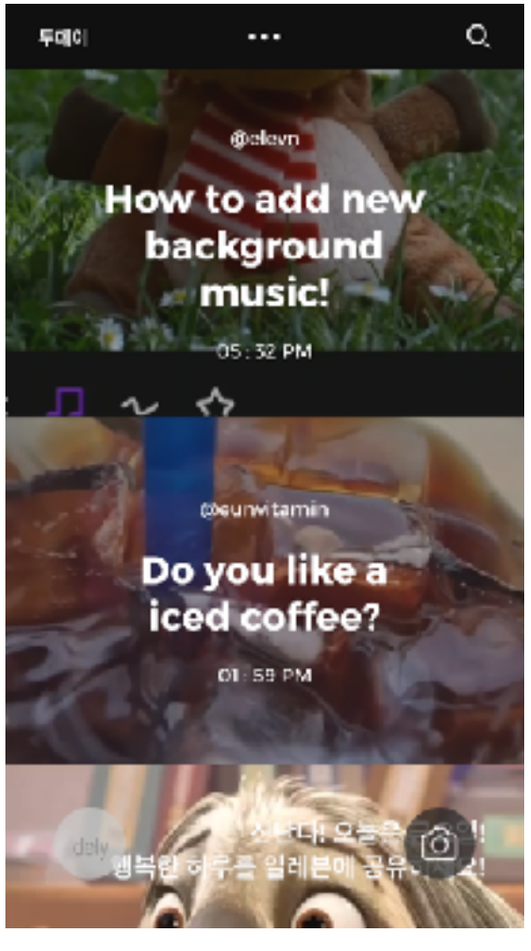
1. 비밀번호 추가 (이미지 헤더나 끝에 문자 추가로?)
2. 이미지 복호화도 만들어야지.....
3. 이미지 크기 다르면 오류.....

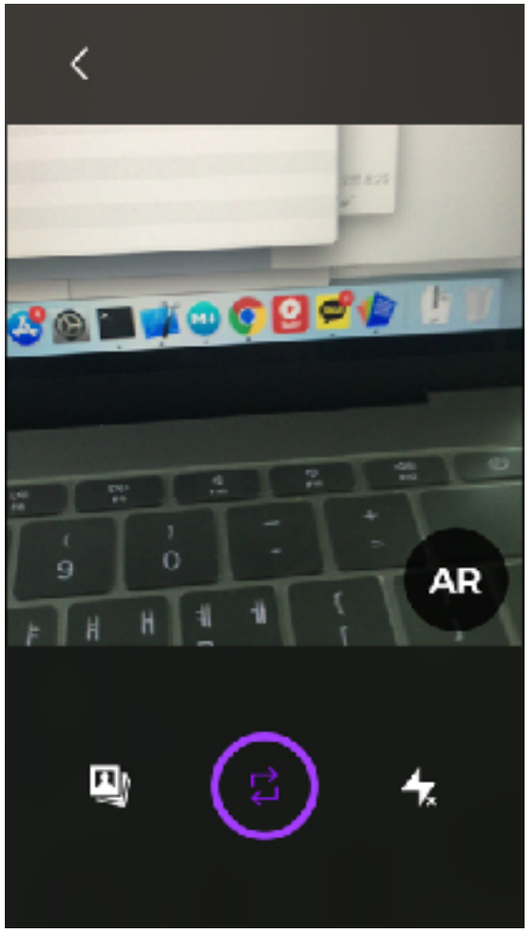
UX

- 개인이 하고싶은 앱 1-2개 선정해서 UI적용하고 싶은 부분이랑 빼고 싶은 부분?

- Elevn







끝 >_< -