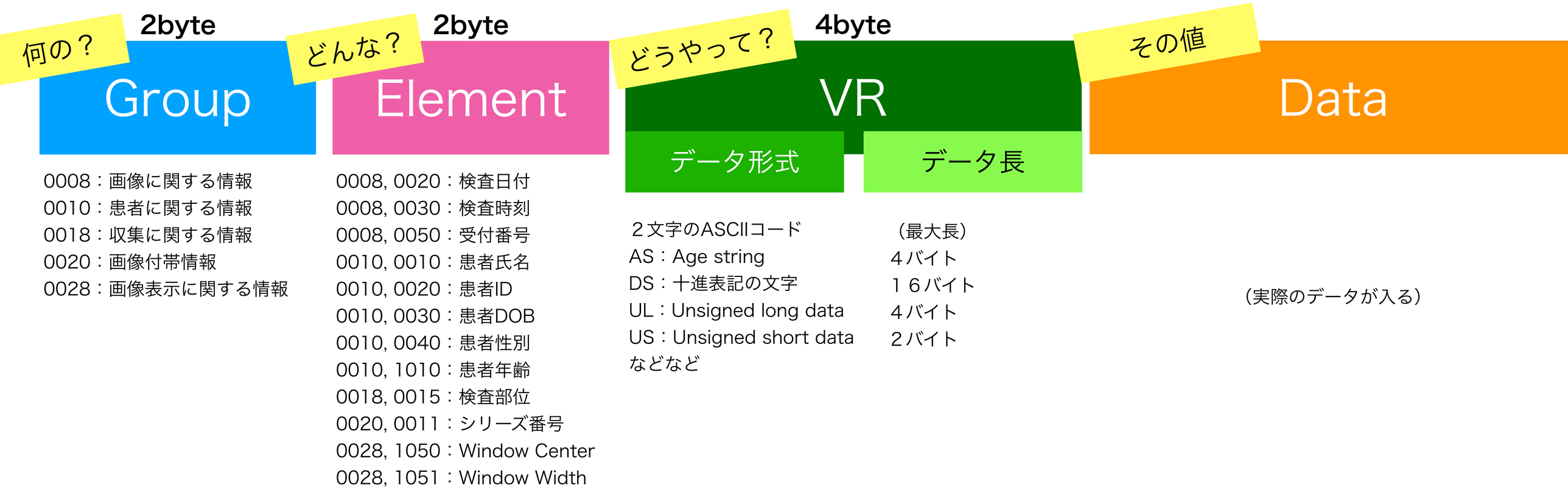
DICOM画像の処理

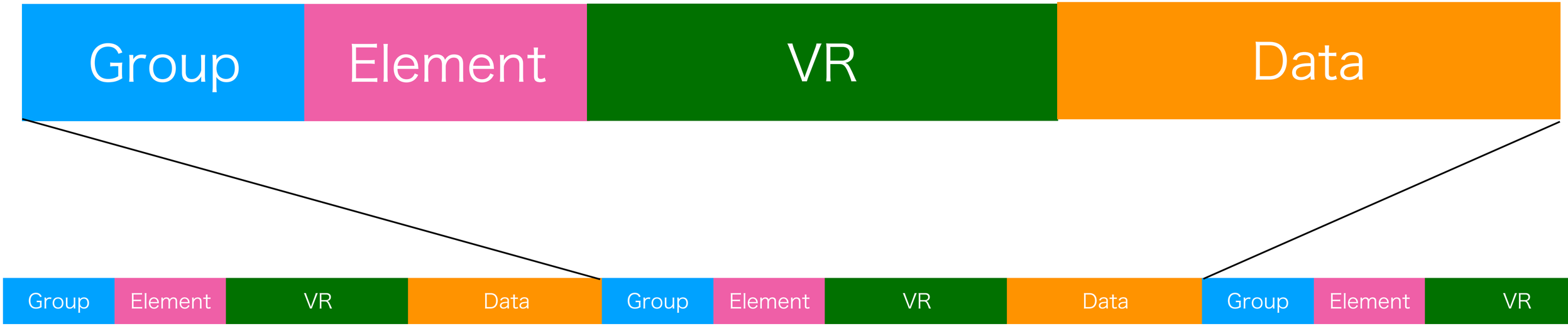
DICOM image using pydicom

DICOMの構造

- DICOMは，Group, Element, Value-Representation, Dataで構成
- GroupとElementは，それぞれ２バイト使う
- Groupの値，Elementの値で記述する情報が決まる



Data Element（データ要素）



Data Elementの繰り返しがDICOM

画像データもデータ要素だよ

Pydicomでは？

- DICOMファイルを開くと自動的に構造体に保存してくれる
- データ要素を自動的に解析してくれる

データを読む	mydicom = pydicom.read_file('./SPIECTBE001/000000.dcm')
読み込んだGroup, Elementを表示	dir(mydicom)
番号を指定してそのデータを表示	print(mydicom[0x0018, 0x0050])
タグの名前を指定して表示	mydicom.KVP
保存された画素値を表示	mydicom.pixel_array
空の配列：tmpを作って	**tmp = np.zeros((h, w), dtype="float32")**
画像データをtmp保存	**tmp = mydicom.pixel_array**

numpyの形式で書ければあとは自由！

.pixel_arrayに信号が保存される

しかしその信号の意味はタグから読む必要がある

```
#スライス枚数分だけファイル名のリストを順番に読んで
#リスケールスロープと切片を読みだして，
#インスタンス番号からスライス位置を指定して
#ボリュームデータに保存．tensorflowを意識して４次元配列にした

#スライス枚数＝画像枚数分だけ
for i in range(0, nslice):
    #ファイル名を決めて
    fname = DIR+'%s' %ctfiles[i]

    #そのDICOMファイルを一時的な領域に保存
    dcmtmp = pydicom.read_file(fname)

    #タグから，インターセプトと傾きを読み出して保存
    intercept = dcmtmp.RescaleIntercept
    slope = dcmtmp.RescaleSlope

    #タグから，インスタンスナンバーを保存
    slicenum = dcmtmp.InstanceNumber

    #保存されている画素値を抽出して，一時的な領域に保存
    tmp = dcmtmp.pixel_array

    #傾きと切片でCT値に変換
    tmp = slope*tmp + intercept

    #画素値に１チャンネルを与えて形式を変換
    tmp3d = np.reshape(tmp, (h, w, 1))

    #指定されたスライス位置に代入
    #スライス位置はゼロから始まるから-1してるよ！
    ctvoldata[slicenum-1] = tmp3d
```

CT値に変換するために
SlopeとInterceptを得て
Instance番号を得て
並べ替える

.pixel_arrayに信号が保存される

しかしその信号の意味はタグから読む必要がある

```
print(ds)
```

```
Dataset.file_meta -------------------------------
(0002, 0000) File Meta Information Group Length  UL: 186
(0002, 0001) File Meta Information Version       OB: b'¥x00¥x01'
(0002, 0002) Media Storage SOP Class UID         UI: MR Image Storage
(0002, 0003) Media Storage SOP Instance UID      UI: 1.3.46.670589.11.17169.5.0.9208.20121206101440734 87.1
(0002, 0010) Transfer Syntax UID                 UI: Explicit VR Little Endian
(0002, 0012) Implementation Class UID            UI: 1.2.40.0.13.1.1.1
(0002, 0013) Implementation Version Name         SH: 'dcm4che-1.4.34'
-------------------------------------------------
(0008, 0005) Specific Character Set              CS: 'ISO_IR 100'
(0008, 0008) Image Type                          CS: ['ORIGINAL', 'PRIMARY', 'M_FFE', 'M', 'FFE']
(0008, 0012) Instance Creation Date              DA: '20121206'
(0008, 0013) Instance Creation Time              TM: '153026'
(0008, 0014) Instance Creator UID                UI: 1.3.46.670589.11.17169.5
(0008, 0016) SOP Class UID                       UI: MR Image Storage
(0008, 0018) SOP Instance UID                    UI: 1.3.46.670589.11.17169.5.0.9208.20121206101440734 87.1
(0008, 0020) Study Date                          DA: '20121206'
(0008, 0021) Series Date                         DA: '20121206'
(0008, 0022) Acquisition Date                    DA: '20121206'
```

```
ds = dcmread(DIR+files[0])
slicenum = ds.InstanceNumber
print(slicenum)
```

```
1
```

```
ds = dcmread(DIR+files[0])
arr = ds.pixel_array
rescaled_arr = util.apply_modality_lut(arr, ds)
windowed_rescaled_arr = util.apply_voi_lut(rescaled_arr, ds, index=0)
```

MR信号値に変換するために
？？？なので
Modalityが保存するLUTを
適用してみた（正しくないかも）

.pixel_arrayに信号が保存される

しかしその信号の意味はタグから読む必要がある

```
#スライス枚数＝画像枚数分だけ
for i in range(0, nslice):
    #ファイル名を決めて
    fname = DIR+'%s' %files[i]

    #そのDICOMファイルを一時的な箱域に保存
    dcmtmp = pydicom.read_file(fname)

    #タグから，インターセプトと傾きを読み出して保存
    intercept = dcmtmp.RescaleIntercept
    slope = dcmtmp.RescaleSlope

    #タグから，インスタンスナンバーを保存
    slicenum = dcmtmp.InstanceNumber

    #保存されている画素値を抽出して，一時的な箱域に保存
    tmp = dcmtmp.pixel_array.astype(np.float32)*slope+intercept

    #http://kakugi-touhoku.kenkyuukai.jp/images/sys/information/20190115121034-C728
    #http://cnmt.umin.jp/public/kiso/formula/formula4.html
    decay = dcmtmp.DecayFactor
    ccf = dcmtmp.DoseCalibrationFactor #
    pweight = dcmtmp.PatientWeight # [kg]
    totaldose = dcmtmp[0x0054, 0x0016][0].RadionuclideTotalDose #[Bq]
    loc = dcmtmp.SliceLocation

    suvfact  = calculate_suv_factor(fname)

    #PET_SUV = double(PET) * double(SUV_factor)  ; %[g/ml] = [Bq/ml] * [g/Bq]

    tmp *= suvfact

    print(fname, slicenum, decay, ccf, totaldose, pweight, loc, suvfact, np.max(tmp)

    #画素値に１チャンネルを与えて形式を変換
    tmp3d = np.reshape(tmp, (h, w, 1))

    #指定されたスライス位置に代入
    #スライス位置はゼロから始まるから-１してるよ！
    voldata[slicenum-1] = tmp3d
```

PETのカウント値からSUVに変換

減弱，体重，投与量などを読み込み

SUVの原理に基づいて変換

DICOMを読むためには

DICOMのタグを読み込む

＋

画像記録を正しく理解

chatGPTに聞くとよいかも？！