iOS/Macアプリ開発のAutoLayoutプログラミングについて考えてみた

自己紹介

- 藤本尚邦 (@fhisa)
- https://github.com/fhisa
- フリーランスプログラマー
- RubyCocoaフレームワーク原作者
- Mac開発歴、薄く長く約25年
- iOS開発歴、1年弱

ビューのAutoLayoutを定義する方法

1. storyboard/xibファイルをXcodeのGUIで編集

2. プログラムでNSLayoutConstraintを生成

質問です

- AutoLayoutを、Xcode GUIではなくプログラムで定義することはありますか？
- どんなときにプログラムで定義しますか？

この発表ではプログラムで
のNSLayoutConstraint定義
に的を絞ります

NSLayoutConstraintの生成方法

1. ビジュアル言語で制約を記述し、文字列引数として与えて生成

2. 制約に関係する全ての値を引数に与えて生成

ビジュアル言語

- プログラム実行時に構文がチェックされる
  - つまり走らせてみなければわからない
- 覚えるのも面倒
- 実用に難ありなので、ここでは触れません

ならば全ての値を引数渡し

```
class NSLayoutConstraint: NSObject {
    convenience init(item view1: AnyObject,
                     attribute attr1: NSLayoutAttribute,
                     relatedBy relation: NSLayoutRelation,
                        toItem view2: AnyObject?,
                      attribute attr2: NSLayoutAttribute,
                    multiplier multiplier: CGFloat,
                      constant c: CGFloat)
}
```

引数多すぎワロたwww

実際に使うとこんな感じ...

NSLayoutConstraint

```
baseView.addConstraints([
    NSLayoutConstraint(
        item: mainView, attribute: .Height, relatedBy: .Equal,
        toItem: baseView, attribute: .Height, multiplier: 3.0 / 4.0, constant: 0),
    NSLayoutConstraint(
        item: mainView, attribute: .Leading, relatedBy: .Equal,
        toItem: baseView, attribute: .Leading, multiplier: 1, constant: 8),
    NSLayoutConstraint(
        item: mainView, attribute: .Top, relatedBy: .Equal,
        toItem: baseView, attribute: .Top, multiplier: 1, constant: 8),
NSLayoutConstraint(
    item: mainView, attribute: .Trailing, relatedBy: .Equal,
    toItem: baseView, attribute: .Trailing, multiplier: 1, constant: -8),
// 以下略
])
```

文字多すぎワロた

\(^o^)/

ワロえない

(´・ω・`)

NSLayoutConstraint なんて大っ嫌いなんだからねっ！

こんな風に書けたら読みやすいなぁ

```
baseView.addConstraints([
    mainView[.Height] * 4 == baseView[.Height] * 3,
    mainView[.Leading] == baseView[.Leading] + 8,
    mainView[.Top] == baseView[.Top] + 8,
    mainView[.Trailing] == baseView[.Trailing] - 8,
    // 以下略
])
```

そこで奥さん

FormulaStyleConstraint

ですよ

FormulaStyleConstraint

NSLayoutConstraintを、等式・不等式などの数式で定義できるようにするSwift用のフレームワーク

https://github.com/fhisa/FormulaStyleConstraint

FormulaStyleConstraintの使用例

```
baseView.addConstraints([
    mainView[.Height] * 4 == baseView[.Height] * 3,
    mainView[.Leading] == baseView[.Leading] + 8,
    mainView[.Top] == baseView[.Top] + 8,
    mainView[.Trailing] == baseView[.Trailing] - 8,
    // 以下略
])
```

FormulaStyleConstraintの特徴

- 目的をNSLayoutConstraintの数式による定義に絞ったシンプルな構成
- ソースコードトータル154行の極小サイズ(バージョン1.2)で、実装の理解が容易

FormulaStyleConstraintの課題

- UILayoutSupportプロトコルのサポート
  - 現バージョンではtopLayoutMarginとbottomLayoutMarginプロパティを使えない
    - Swift 2.0 の protocol extension でおそらく対応可能
- Mac OS X のサポート

FormulaStyleConstraintの競合品

制約を数式で定義するというアイディアは、誰かがすでに作ってる可能性大だと思ったけど、作る楽しみを味わいたかったので調べずに作りました。
ひとまず完成してから調べたところ、やっぱりありました(´・ω・｀)

Cartography

Using Cartography, you can set up your Auto Layout constraints in declarative code and without any stringly typing!

https://github.com/robb/Cartography

Cartographyの使用例

```
layout(baseView, mainView) {
    $1.height == $0.height * (3.0 / 4.0)
    $1.Leading == $0.leading + 8
    $1.top == $0.top + 8
    $1.trailing = $0.trailing - 8
    // 以下略
}
```

Cartographyの特徴
(FormulaStyleConstraintとの違い)

- 制約の定義をブロック内に記述するDSLタイプ
- 整列(align)などの拡張機能あり
- ソースコード約1400行(バージョン0.5)
- GitHubのスター数約2960。一方、FormulaStyleConstraintはスター数０、ゼロ、零！

その他

- 初めてTravis CIを使ってみた
- 初めてCarthageに対応してみた(これは便利)
- 初めてプルリクをもらった (ただしボットにw)
- ひとり焼き肉、ひとりディズニーランド、ひとり美ら海水族館、ひとりGitHub

まとめ

FormulaStyleConstraintにせよ、Cartographyにせよ、素でNSLayoutConstraintのコードを書くよりはるかに楽ちんなので、自動レイアウトをプログラムで書いている人にはたいへんオススメです。


Thank you!