報道各位

2020 年 1 月 27 日
株式会社インプレスR&D
https://nextpublishing.jp/

さらにレシピを「ちょい足し」！UI をもっとリッチにする表現集第二弾！

『iOS アプリ開発　UI 実装であると嬉しいレシピブック Vol.2』発行

技術の泉シリーズ、1月の新刊

インプレスグループで電子出版事業を手がける株式会社インプレス R&D は、『iOS アプリ開発　UI 実装であると嬉しいレシピブック Vol.2』（著者：酒井　文也）を発行いたしました。

最新の知見を発信する『技術の泉シリーズ』は、「技術書典」や「技術書同人誌博覧会」をはじめとした各種即売会や、勉強会・LT 会などで頒布された技術同人誌を底本とした商業書籍を刊行し、技術同人誌の普及と発展に貢献することを目指します。

『iOSアプリ開発　UI実装であると嬉しいレシピブックVol.2』
https://nextpublishing.jp/isbn/9784844378358

著者：酒井　文也
小売希望価格：電子書籍版　2000 円（税別）／印刷書籍版　2400 円（税別）
電子書籍版フォーマット:EPUB3／Kindle Format8
印刷書籍版仕様：B5 判／カラー／本文 222 ページ
ISBN：978-4-8443-7835-8
発行：インプレス R&D

＜＜発行主旨・内容紹介＞＞

本書は、筆者がこれまでサンプル開発や実務の中で培ったノウハウ等をもとに、UI 実装をいくつかのまとまったサンプル実装を例に、UI を構築する上で重要な実装ポイントやアイデアを紹介していきます。

第二弾の本書は、便利なライブラリを上手に活用した表現を解説します。

（本書は、次世代出版メソッド「NextPublishing」を使用し、出版されています。）

UI 設計の際に役立つ3つのレシピを紹介

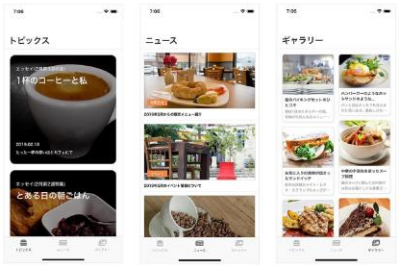
第1章

サンプル実装の難易度: ★★★★★★★☆☆☆

・サンプルプロジェクト名: TypicalAnimationContents

図1: 第1章のサンプル

<第1章：TabBar Animation & Layout Tips>



第2章

サンプル実装の難易度: ★★★★★★★★☆☆

・サンプルプロジェクト名: MediaStyleAnimationContents

図2: 第2章のサンプル

<第2章：API Connection & Layout Tips>

解説の際は細かく図示しながら説明

図 1.11: 視差効果を生み出すために UIImageView の上下方向の AutoLayout の制約値を更新できるような形にする

セルのクラス内に下記の制約値をOutlet接続する

① @IBOutlet weak private var topConstraint: NSLayoutConstraint!
② @IBOutlet weak private var bottomConstraint: NSLayoutConstraint!

実現のためのポイント

矩形用Viewとその中に配置したUIImageViewのclipToBoundsはtrueにする。
UITableViewのUIScrollViewDelegateを利用して、上記の①および②の制約値を更新する処理を実行することでスクロールに合わせた視差効果を実現する。

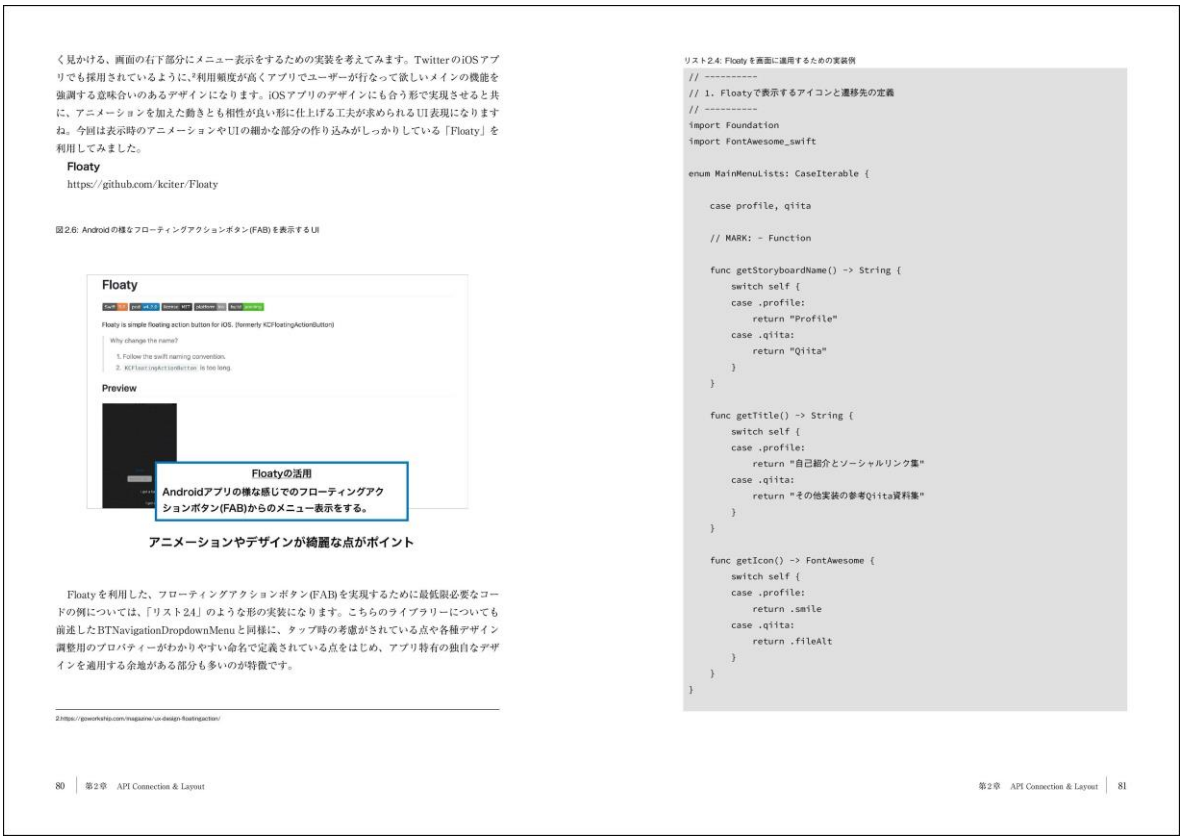
そして、Outlet接続をした上下方向のAutoLayoutの制約値を少しずつ変化させることで、サムネイル画像に動きをつけられるように処理を行います。画面のViewControllerのクラスから変更できるように別途メソッドとして定義します。また、このセルで表示するサムネイル画像は実際に配置するサイズよりも高さが大きいものを用意する点にご注意ください。具体的なコードについては、「リスト1.14」のような形の実装になります。

リスト 1.14: News表示部分のセルクラスにおける実装例

```
import UIKit

// MEMO: InterfaceBuilder内で下記のように設定する
// 1. 配置したUIImageViewのcontentModeをaspectFillとする
// 2. 配置したUIImageViewのclipToBoundsをtrueとする
// 3. 配置したUIImageViewの親のUIViewもclipToBoundsをtrueとする

final class NewsTableViewCell: UITableViewCell {

    // 視差効果のズレを生むための定数（大きいほど視差効果が大きい）
    private let parallaxFactor: CGFloat = 30.0
```

```
// 視差効果の計算用の変数
private var topInitialConstraint: CGFloat!
private var bottomInitialConstraint: CGFloat!

@IBOutlet weak private var categoryLabel: UILabel!
@IBOutlet weak private var titleLabel: UILabel!
@IBOutlet weak private var newsImageView: UIImageView!

// UIView内のUIImageViewの上下の制約
@IBOutlet weak private var topConstraint: NSLayoutConstraint!
@IBOutlet weak private var bottomConstraint: NSLayoutConstraint!

// MARK: - Initializer

override func awakeFromNib() {
    super.awakeFromNib()

    setupNewsTableViewCell()
}

// MARK: - Function

func setCell(_ news: NewsModel) {
    categoryLabel.text = news.category
    titleLabel.text = news.title
    newsImageView.image = news.image
}

// 画像にかけられているAutoLayoutの制約を再計算して制約をかけ直す
func setImageViewOffset(_ offset: CGFloat) {
    let boundOffset = max(0, min(1, offset))
    let pixelOffset = (1 - boundOffset) * 2 * parallaxFactor
    topConstraint.constant = topInitialConstraint - pixelOffset
    bottomConstraint.constant = bottomInitialConstraint + pixelOffset
}

// MARK: - Private Function

private func setupNewsTableViewCell() {

    // セルの装飾設定をする
```

ライブラリの紹介ではその特徴を丁寧に解説



く見かける、画面の右下部分にメニュー表示をするための実装を考えてみます。TwitterのiOSアプリでも採用されているように、[2]利用頻度が高くアプリでユーザーが行なって欲しいメインの機能を強調する意味合いのあるデザインになります。iOSアプリのデザインにも合う形で実現させると共に、アニメーションを加えた動きとも相性が良い形に仕上げる工夫が求められるUI表現になりますね。今回は表示時のアニメーションやUIの細かな部分の作り込みがしっかりしている「Floaty」を利用してみました。

Floaty
https://github.com/kciter/Floaty

図2.6: Androidの様なフローティングアクションボタン(FAB)を表示するUI

Floatyを利用した、フローティングアクションボタン(FAB)を実現するために最低限必要なコードの例については、「リスト2.4」のような形の実装になります。こちらのライブラリーについても前述したBTNavigationDropdownMenuと同様に、タップ時の考慮がされている点や各種デザイン調整用のプロパティーがわかりやすい命名で定義されている点をはじめ、アプリ特有の独自なデザインを適用する余地がある部分も多いのが特徴です。

2:https://goworkship.com/magazine/ux-design-floatingaction/

リスト2.4: Floatyを画面に適用するための実装例

```
// ----------
// 1. Floatyで表示するアイコンと遷移先の定義
// ----------
import Foundation
import FontAwesome_swift

enum MainMenuLists: CaseIterable {

    case profile, qiita

    // MARK: - Function

    func getStoryboardName() -> String {
        switch self {
        case .profile:
            return "Profile"
        case .qiita:
            return "Qiita"
        }
    }

    func getTitle() -> String {
        switch self {
        case .profile:
            return "自己紹介とソーシャルリンク集"
        case .qiita:
            return "その他実装の参考Qiita資料集"
        }
    }

    func getIcon() -> FontAwesome {
        switch self {
        case .profile:
            return .smile
        case .qiita:
            return .fileAlt
        }
    }
}
```


＜＜目次＞＞

3.2 このサンプル実装における概要
3.3 利用したライブラリーの紹介
3.4 このサンプルで利用しているアーキテクチャや画面構成について
3.5 各種画面表示をするViewControllerから見る実装ポイント
3.6 コーヒーブレイク

＜＜著者紹介＞＞

酒井 文也
アプリのUI実装が好きな元デザイナーからジョブチェンジをしたエンジニア。Qiitaや GitHubなどでもUI実装に関するサンプルや解説記事を投稿したり、Swift 愛好会をはじめとする勉強会等でもたまに登壇しています。アイデアを練ったり、設計のためのメモや図解を作る時はもっぱら手書き派です。見た目と違いお酒はまったく飲めませんが、お酒の席に参加するのは好きです。
Twitter: @fumiyasac
GitHub: https://github.com/fumiyasac
Qiita: http://qiita.com/fumiyasac@github
Slideshare: https://www.slideshare.net/fumiyasakai37

＜＜販売ストア＞＞

電子書籍:
Amazon Kindleストア、楽天koboイーブックストア、Apple Books、紀伊國屋書店 Kinoppy、Google Play Store、honto電子書籍ストア、Sony Reader Store、BookLive!、BOOK☆WALKER
印刷書籍:
Amazon.co.jp、三省堂書店オンデマンド、hontoネットストア、楽天ブックス
※ 各ストアでの販売は準備が整いしだい開始されます。
※ 全国の一般書店からもご注文いただけます。

【インプレスR&D】 https://nextpublishing.jp/
株式会社インプレスR&D(本社:東京都千代田区、代表取締役社長:井芹昌信)は、デジタルファーストの次世代型電子出版プラットフォーム「NextPublishing」を運営する企業です。また自らも、NextPublishing を使った「インターネット白書」の出版などIT関連メディア事業を展開しています。
※NextPublishingは、インプレスR&Dが開発した電子出版プラットフォーム(またはメソッド)の名称です。電子書籍と印刷書籍の同時制作、プリント・オンデマンド(POD)による品切れ解消などの伝統的出版の課題を解決しています。これにより、伝統的出版では経済的に困難な多品種少部数の出版を可能にし、優秀な個人や組織が持つ多様な知の流通を目指しています。

【インプレスグループ】 https://www.impressholdings.com/
株式会社インプレスホールディングス(本社:東京都千代田区、代表取締役:唐島夏生、証券コード:東証1部9479)を持株会社とするメディアグループ。「IT」「音楽」「デザイン」「山岳・自然」「旅・鉄道」「学術・理工学」を主要テーマに専門性の高いメデイア&サービスおよびソリューション事業を展開しています。さらに、コンテンツビジネスのプラットフォーム開発・運営も手がけています。

【お問い合わせ先】

株式会社インプレスR&D NextPublishingセンター
TEL 03-6837-4820
電子メール: np-info@impress.co.jp