

UICollectionViewLayoutで カバフローを作りたい！

沢 辰洋

@sawat1203

自己紹介

- アイティメディア株式会社勤務
- Twitter ID : @sawat1203



- 作ったアプリ

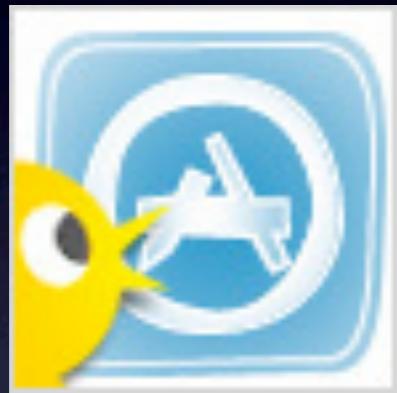


仕事

個人

自己紹介

- iOSアプリ開発 ONETOPI



あなたの選んだ情報は、きっと誰かのためになる。

ONETOPIは、おすすめ情報にコメントをつけて、
トピックごとに共有するサービスです。

ためる

お気に入りの情報を、新聞をスクラップする
ように保存。コメントをつけてトピック
を分類するから、検索も自由自在です。
*おすすめの投稿は [こちら](#)

つながる

トピックを通じてさまざまな人とつながる
ことで、同じ興味・趣味の人たちが発信する
情報や、新しい気付きが得られます。
*おすすめのトピックは [こちら](#)

伝える

Twitter/Facebookと連携させれば、
URL投稿ツールとしても使えます。あなた
のおすすめの情報を、友だちにどんどん
伝えましょう。

Login with Twitter

Login with Facebook

Login with Google+

* Twitter、Facebook、Google+のアカウントでユーザー登録できます。
* プライバシーポリシー、投稿ユーザー規約に同意の上、ご登録ください。
* いまユーザー登録すると先着で2000ポイントをプレゼント中！ [キャンペーンページへ](#)

宣伝

<http://1topi.jp>

今日の話

1. UICollectionViewって何？
2. UICollectionViewLayoutって何？
3. UICollectionViewLayoutでカバーフローを作りたい！

今日のサンプルコード

[https://github.com/sawat/
YidevCollectionViewLayoutSa
mple](https://github.com/sawat/YidevCollectionViewLayoutSample)

UICollectionView って何？

UICollectionViewとは

- iOS 6で追加された新しいUIKitのクラス
- UITableViewの思想を継承した設計
- 自由なセルサイズ&縦横、両方のスクロールをサポート
- Cellのレイアウト処理が差し替え可能！

例



公式情報

- WWDC 2012
 - “Introducing Collection View”
 - “Advanced Collection Views and Building Custom Layouts”
- Collection View Programming Guide for iOS
 - <http://developer.apple.com/library/ios/documentation/WindowsViews/>

UICollectionViewLayout って何？

UICollectionViewLayout とは

- UICollectionViewに含まれるの各セルの配置処理を提供するクラス
- UICollectionViewLayoutのサブクラスを実装して任意のレイアウトを実現可能
- UICollectionViewやCell, DataSourceの実装と完全に独立して定義できる → 再利用可能
- GoFのデザパタでいうStrategyパターン

UICollectionViewLayout とは

- デフォルトではUICollectionViewFlowLayoutが使用される
- 縦・横両方のスクロールを選択可能
- セル間の隙間などを細かく指定できる
- 不揃いのセルサイズも可能
- ぶっちゃけ、ほとんどの場合はFlowLayoutだけで十分

独自レイアウトの実装が必要になるケース

[Collection View Programming Guide for iOS](#)

P.38 Creating Custom Layouts

The only times to consider implementing a custom layout are in the following situations:

- The layout you want looks nothing like a grid or a line-based breaking layout.
- You want to change all of the cell positions frequently enough that it would be more work to modify the existing flow layout.

独自レイアウトの実装が必要になるケース

Collection View Programming Guide for iOS

P.38 Creating Custom Layouts

The only times to consider implementing a custom layout are the following situations:

- The layout you want looks **nothing like a grid or a line-based breaking layout.**
- You want to change all of the cell positions frequently enough that it would be more work to modify the existing flow layout.

グリッド、または行ベース
ではないレイアウト

独自レイアウトの実装が必要になるケース

Collection View Programming Guide for iOS

P.38 Creating Custom Layouts

The only times to consider creating a custom layout are the following situations:

- The layout you want looks better than the flow layout, but it's not breaking layout.
- You want to **change all of the cell positions frequently** enough that it would be more work to modify the existing flow layout.

すべてのセルの配置が頻繁に変わる

カバーフローをUICollectionViewLayout で実装するメリット

- DataSourceやCellと独立しているので、再利用しやすい
- 他のレイアウトとアニメーション付きで切り替え可能
- 他の実装方法より簡単！
 - 他1：UIView/UIScrollViewでごりごり
 - 他2：UITableViewハック

本題

UICollectionViewLayout

でカバーフローを実装

したい！

参考



では、始めます

UICollectionViewLayoutを サブクラス化する

```
#import <UIKit/UIKit.h>

@interface CVCLCoverFlowLayout : UICollectionViewLayout
```

...

```
@end
```

```
#import "CVCLCoverFlowLayout.h"

@implementation CVCLCoverFlowLayout
```

...

```
@end
```

最低限、オバーライドが 必要なメソッド

- - (CGSize)collectionViewContentSize
- - (NSArray *)layoutAttributesForElementsInRect:(CGRect)rect
- - (UICollectionViewLayoutAttributes *)layoutAttributesForItemAtIndexPath:(NSIndexPath *)indexPath
- - (BOOL)shouldInvalidateLayoutForBoundsChange:(CGRect)newBounds

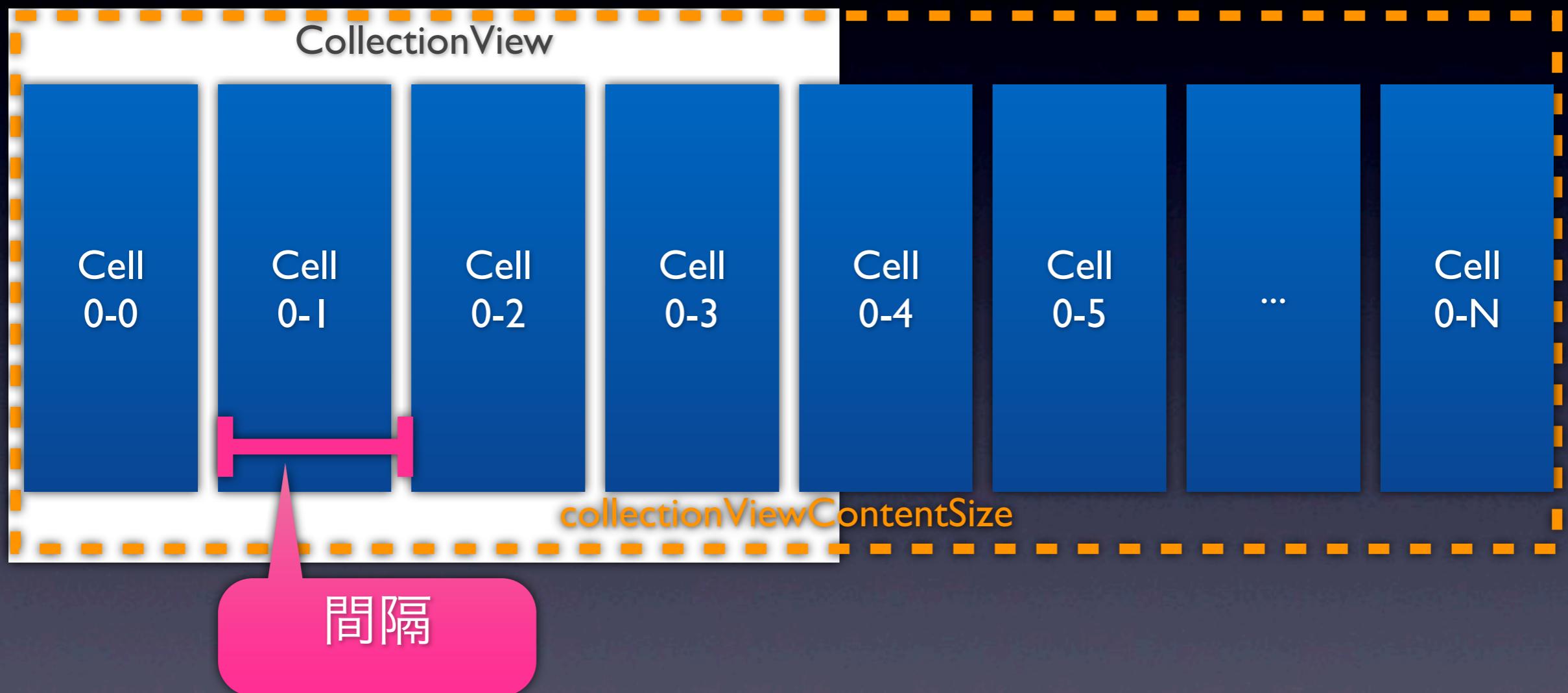
collectionViewContentSize:

- CollectionViewがスクロールできる範囲の大きさを返す
- UIScrollViewのcontentSizeと同じ
- これの戻り値で縦、横のスクロール有無が決まる

collectionViewContentSize:

- カバーフローなので、横スクロールにしたい
 - セルの数 × セルの配置間隔(pt) を横方向の大きさとする
 - 縦は、CollectionViewの高さに合わせる

collectionViewContentSize:



$\text{width} = \text{セル数} \times \text{セルの配置間隔}$

collectionViewContentSize:

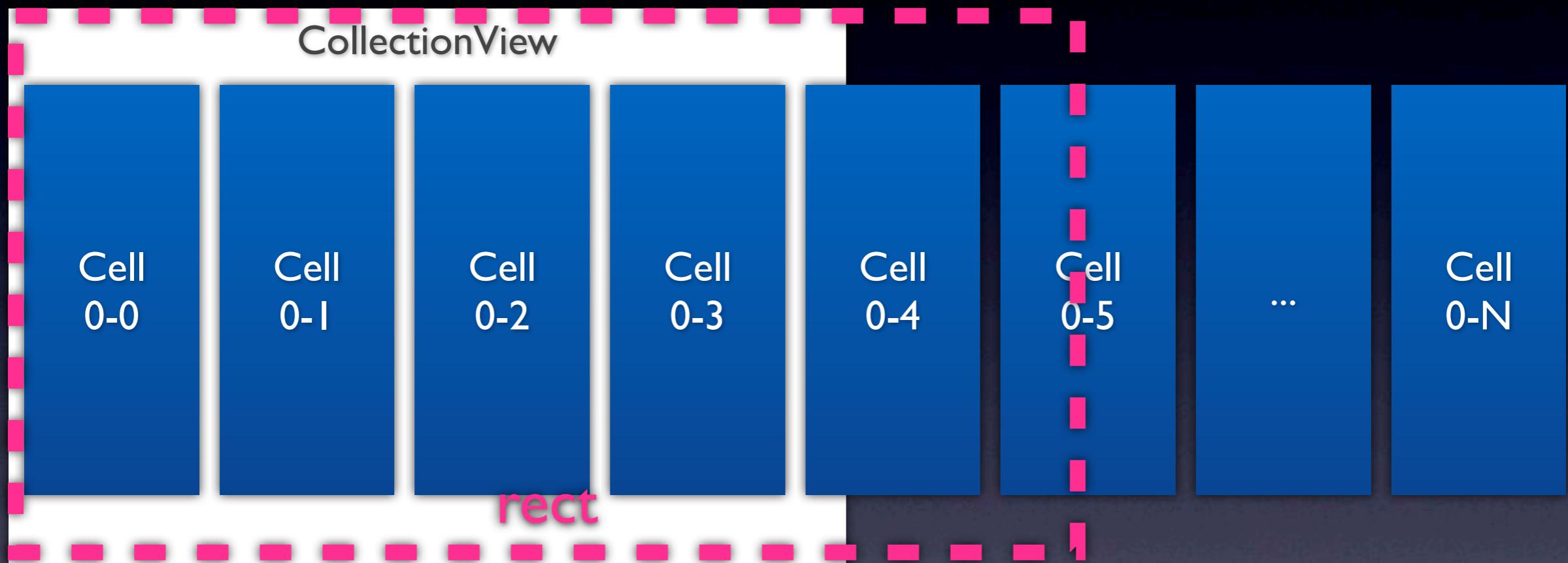
```
@interface CVCLCoverFlowLayout : UICollectionViewLayout  
...  
@property (nonatomic, assign) CGSize cellSize;  
@property (nonatomic, assign) CGFloat cellInterval;
```

```
@implement CVCLCoverFlowLayout  
...  
- (NSInteger)count {  
    return [self.collectionView  
            numberOfItemsInSection:0];  
}  
- (CGSize)collectionViewContentSize {  
    CGSize size = self.collectionView.bounds.size;  
    size.width = self.count * self.cellInterval;  
    return size;  
}
```

layoutAttributesForElementsInRect:

- 指定されたRectの中に含まれるCell, SupplementaryView, DecorationViewのレイアウト情報を返す
- 戻り値に含まれないCellなどは表示されない
- 戻り値は UICollectionViewLayoutAttributes の NSArray

layoutAttributesForElementsInRect:



Cell 0-0 から Cell 0-5 の `layoutAttribute` を
NSArrayに入れて返せばよい

layoutAttributesForElementsInRect:

- まず、指定されたRectの中に含まれるセルのIndexPathの配列を返すメソッドを実装する

```
- (NSArray *)indexPathsForItemsInRect:(CGRect)rect {
    CGFloat cw = self.cellInterval;
    int minRow = MAX(0, (int)floor(rect.origin.x / cw));

    NSMutableArray *array = [NSMutableArray array];
    for (int i=minRow; i < self.count && (i-1) * cw
        rect.origin.x + rect.size.width; i++) {
        [array addObject:
            [NSIndexPath indexPathForItem:i inSection:0]];
    }
    return array;
}
```

layoutAttributesForElementsInRect:

- さっきのindexPathsForItemsInRect: と layoutAttributesForItemAtIndexPath: を使って実装する

```
- (NSArray *)layoutAttributesForElementsInRect:(CGRect)rect {  
    NSArray *indices = [self indexPathsForItemsInRect:rect];  
    NSMutableArray *array = [NSMutableArray  
        arrayWithCapacity:indices.count];  
    for (NSIndexPath *indexPath in indices) {  
        [array addObject:[self  
            layoutAttributesForItemAtIndexPath:indexPath]];  
    }  
    return array;  
}
```

layoutAttributesForItem AtIndexPath:

- IndexPathで個別に指定されたItemについて、オンデマンドにレイアウト情報を返す
 - 主に表示されるItemの増減に伴って呼び出される
- 戻り値は UICollectionViewLayoutAttributes のオブジェクト

UICollectionViewLayout Attributesって何？

- UICollectionViewの中に表示する各種要素（セル、ヘッダーなど）の配置場所を指示するオブジェクト

UICollectionViewLayout Attributesって何？

- 以下の属性を持つ
 - frame (centerとsizeでも指定可能)
 - alpha
 - hidden
 - zIndex
 - transform3D

UICollectionViewLayout Attributesって何？

- 以下の属性を持つ
 - frame (centerとsizeでも指定可能)
 - セルに対して、
 - CATrnceform3Dが指定できる
 - zindex
 - trnceform3D



CATrnsform3DRotate

layoutAttributesForItem AtIndexPath:

```
- (UICollectionViewLayoutAttributes *)  
    layoutAttributesForItemAtIndexPath:(NSIndexPath *)indexPath {  
  
    UICollectionViewLayoutAttributes *attr =  
        [UICollectionViewLayoutAttributes  
            layoutAttributesForCellWithIndexPath:indexPath];  
  
    CGFloat offsetX = indexPath.item * self.cellInterval;  
  
    CGRect frame;  
    frame.origin.x = offsetX;  
    frame.origin.y = (self.collectionView.bounds.size.height -  
                    self.cellSize.height) / 2.0;  
    frame.size = self.cellSize;  
    attr.frame = frame;  
  
    attr.transform3D = [self transformWithCellOffsetX:offsetX];  
  
    return attr;  
}
```

layoutAttributesForItem AtIndexPath:

```
- (UICollectionViewLayoutAttributes *)  
    layoutAttributesForItemAtIndexPath:(NSIndexPath *)indexPath {  
  
    UICollectionViewLayoutAttributes *attr =  
        [UICollectionViewLayoutAttributes  
            layoutAttributesForCellWithIndexPath:indexPath];  
  
    CGFloat offset = ...  
    CGRect frame = ...  
    frame.origin = ...  
    frame.size = ...  
  
    attr.frame = frame;  
  
    attr.transform3D = [self transformWithCellOffsetX:offset];  
  
    return attr;  
}
```

セルの位置に合わせて、CATransform3D

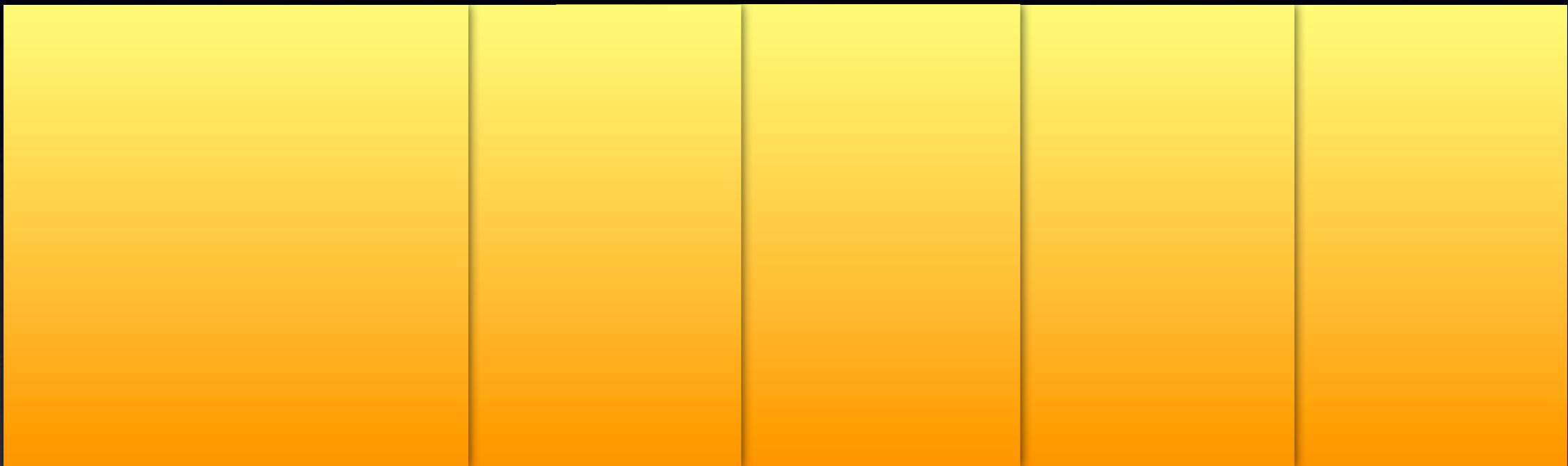
でセルの傾きをつける

```
frame.size = self.cellSize;  
attr.frame = frame;
```

```
attr.transform3D = [self transformWithCellOffsetX:offset];
```

```
return attr;
```

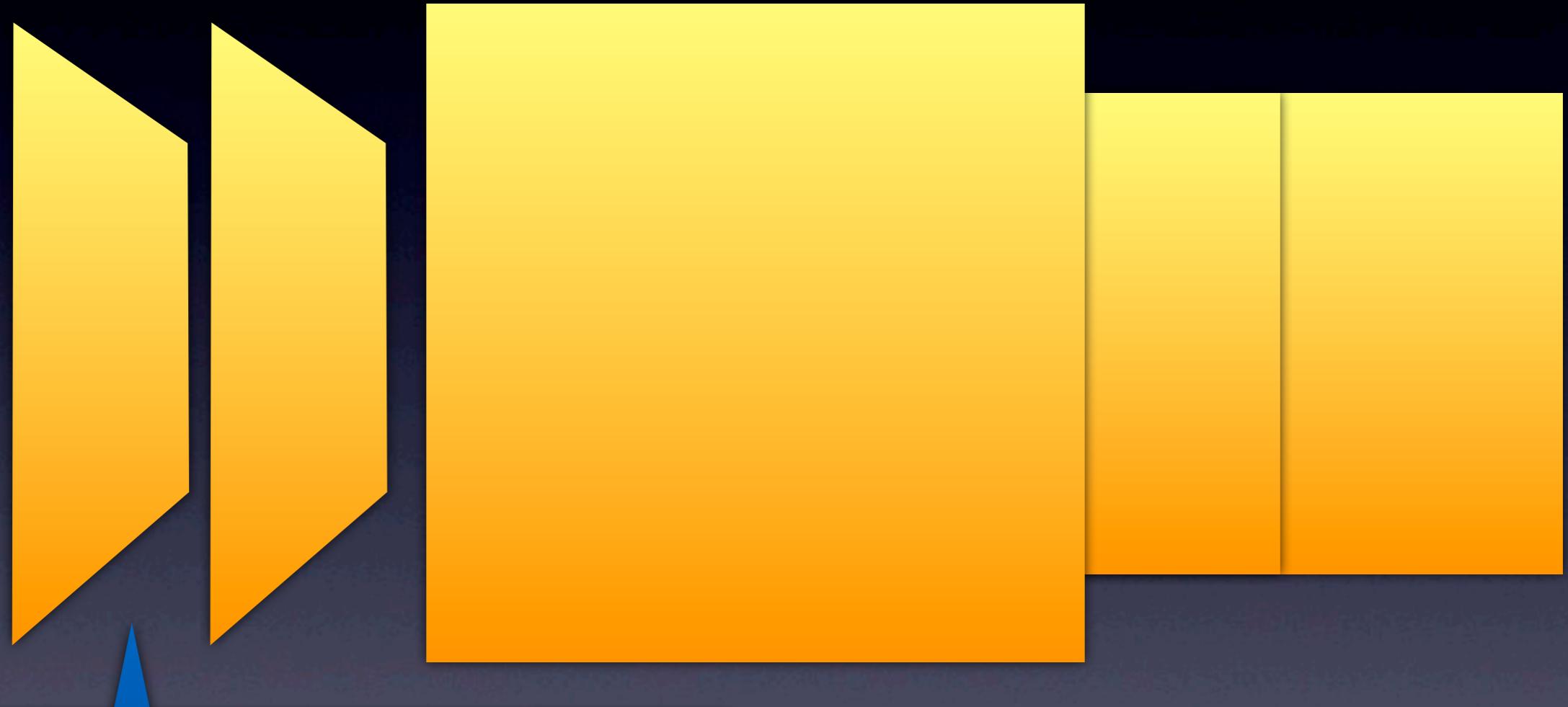
つまり





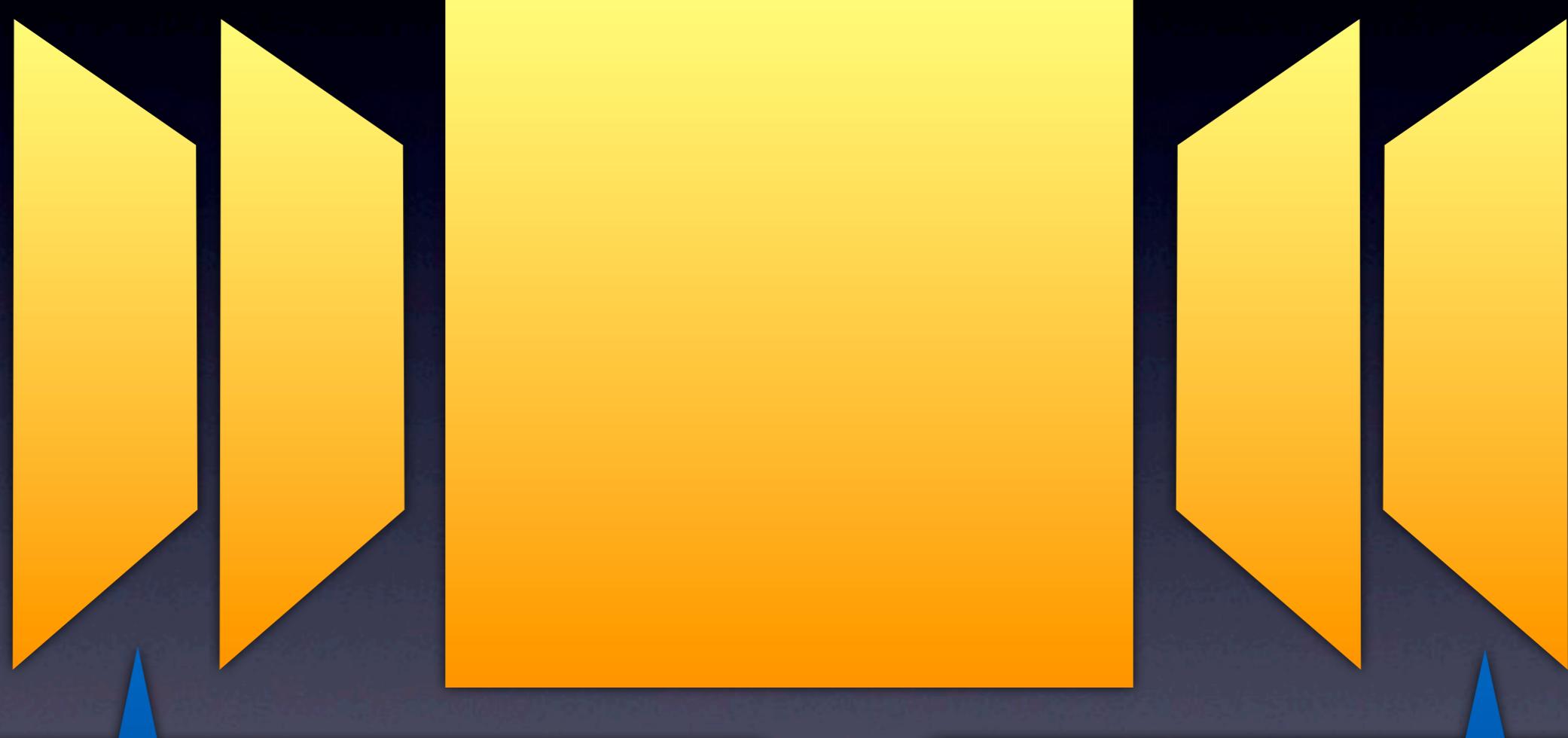
中央より左のセルは、右向きに回転&中央
のセルに重ならないように左に移動

中央のセルは手前に移動



中央より左のセルは、右向きに回転&中央
のセルに重ならないように左に移動

中央のセルは手前に移動



中央より左のセルは、右向きに回転&中央
のセルに重ならないように左に移動

中央より左のセルは、右向きに回転
&右に移動

この辺のあれこれは、

UITableViewハック

UITableViewでCoverFlowは作れるか？

@fladdict

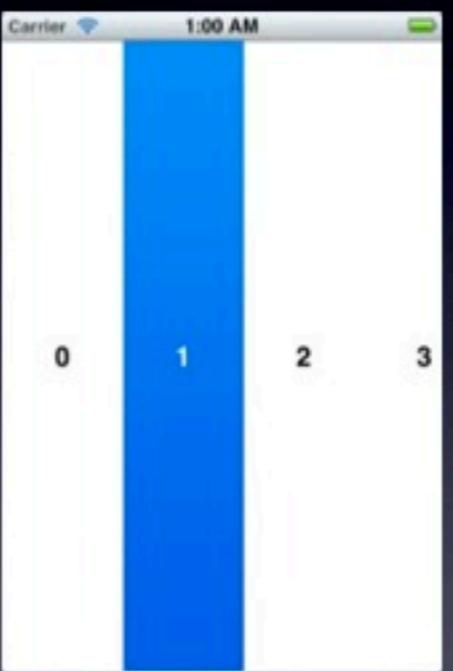


@fladdict 氏

12年5月26日土曜日

https://s3-ap-northeast-1.amazonaws.com/sliders.tl/1338005297280_5734dda2c903f5b359b209804920c189/UITableViewHack.pdf

「中央からセルまでの距離」を相対値で計算する



@fladdict 氏

12年5月26日土曜日

https://s3-ap-northeast-1.amazonaws.com/sliders.tl/1338005297280_5734dda2c903f5b359b209804920c189/UITableViewHack.pdf

Cellの位置、(-l～+l)



評価関数



角度、スケール、透明度、パース

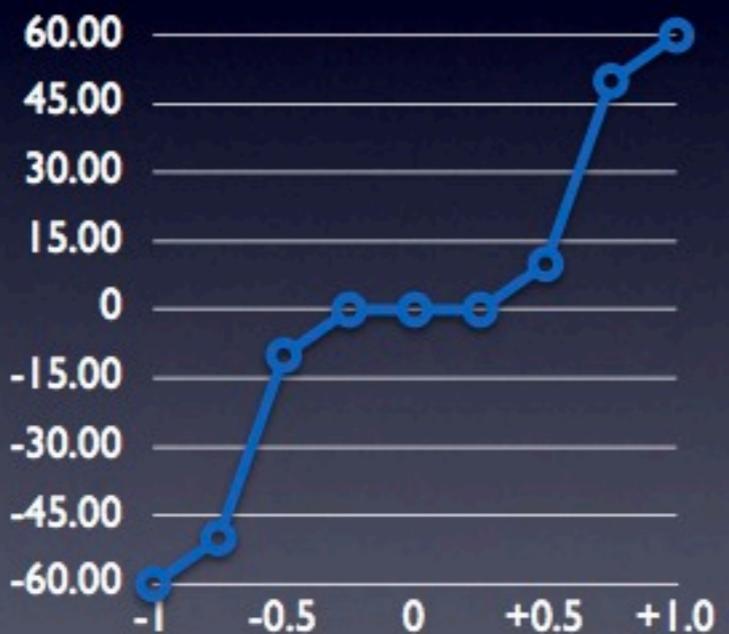
12年5月26日土曜日



@fladdict 氏

https://s3-ap-northeast-1.amazonaws.com/sliders.tl/1338005297280_5734dda2c903f5b359b209804920c189/UITableViewHack.pdf

CoverFlowの角度の評価関数



12年5月26日土曜日



@fladdict 氏

https://s3-ap-northeast-1.amazonaws.com/sliders.tl/1338005297280_5734dda2c903f5b359b209804920c189/UITableViewHack.pdf

参考にしました m(—)m

<[Objective-C][iphone]サーバか...|大学で臨時講義してきました>

2012-11-02

UITableViewでラウンドロビン的な無限スクロールCoverFlowを作る

iphone, Objective-C

「UITableViewでCoverFlowを作れるか？」というネタに興味が湧いたのでちょっとやってみました。

UITableViewハック

ちなみに、iOSのCoverFlowについて調べてみると下記のようなframeworkが見つかりました。

- FlowCover (<http://www.chaosinmotion.com/flowcover.html>)
- iCarousel (<https://github.com/nicklockwood/iCarousel>)
- OpenFlow (<http://apparantlogic.com/openflow/>)
- Tapku (<https://github.com/devinross/tapkulibrary>)

ソースを覗くと、TapkuがUIScrollView、その他はUIViewでゴリゴリ作っているみたいですね。

カレンダー						
<< 2012/11						
日	月	火	水	木	金	土
4	5	6	7			
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	1

カテゴリ

javascript

java

ruby

iphone

xcode

Objective-C

アプリ

Android

iOS



@h_mori
氏

http://d.hatena.ne.jp/h_mori/20121102/1351826242

```
- (CATransform3D)transformWithCellOffsetX:(CGFloat)cellOffsetX {
    static const CGFloat zDistance = 800.0f;

    // 「中央からセルまでの距離」を相対値で計算する
    CGFloat rate = [self rateForCellOffsetX:cellOffsetX];

    CATransform3D t = CATransform3DIdentity;
    // 視点の距離
    t.m34 = 1.0f / -zDistance;

    // 位置 (中央のセルを手前、それ以外のセルを左右にずらす)
    t = CATransform3DTranslate(t,
        [self translateXForDistanceRate:rate],
        0.0f,
        [self translateZForDistanceRate:rate]);

    // 角度 (Y軸に対して回転させる)
    t = CATransform3DRotate(t,
        [self angleForDistanceRate:rate],
        0.0f, 1.0f, 0.0f);

    return t;
}
```

セルの位置を -1.0 ~ 1.0 の指数に変換

```
- (CGFloat)rateForCellOffsetX:(CGFloat)cellOffsetX {
    CGFloat bw = self.collectionView.bounds.size.width;
    CGFloat offsetFromCenter = cellOffsetX +
        self.cellSize.width/2 -
        (self.collectionView.contentOffset.x + bw /2);
    CGFloat rate = offsetFromCenter / bw;
    return MIN(MAX(-1.0, rate), 1.0);
}
```

セルの位置の指数を回転の角度に変換

```
- (CGFloat)angleForDistanceRate:(CGFloat)rate {
    static const CGFloat baseAngle = - M_PI * 80 / 180;

    if (fabsf(rate) > _centerRateThreshold) {
        return copysignf(1.0f, rate) * baseAngle;
    }
    return (rate / _centerRateThreshold) * baseAngle;
}
```

セルの位置の指數をX,Zの移動量に変換

```
- (CGFloat)translateXForDistanceRate:(CGFloat)rate {  
  
    if (fabsf(rate) < _centerRateThreshold) {  
        return (rate / _centerRateThreshold) *  
            self.cellSize.width / 2;  
    }  
    return copysignf(1.0, rate) * self.cellSize.width / 2;  
}  
  
- (CGFloat)translateZForDistanceRate:(CGFloat)rate {  
  
    if (fabsf(rate) < _centerRateThreshold) {  
        return -1.0 - 2.0 * self.cellSize.width *  
            (1.0 - cos((rate / _centerRateThreshold) * M_PI_2));  
    }  
    return -1.0 - 2.0 * self.cellSize.width;  
}
```

(スケールの代わりにZ軸の移動を使ってます)

shouldInvalidateLayoutForBoundsChange:

- Collection Viewがスクロールする度に呼び出される。
- YES を返すと、
layoutAttributesForElementsInRect: が呼び出され
レイアウトが更新される
- デフォルトは常にNO

shouldInvalidateLayoutFor BoundsChange:

- どんなときにYESを返せばいいの？
- スクロール位置によって、表示済みの要素の位置や変形が動的に変化する
- カバーフローではスクロール位置によってセルの傾きを変えるので常にYESを返す
- UITableView (Plain Style) のようにスクロール端に張り付くヘッダーを表示したい時

shouldInvalidateLayoutFor BoundsChange:

```
- (BOOL)shouldInvalidateLayoutForBoundsChange:  
    (CGRect)newBounds {  
  
    return YES;  
}
```

prepareLayout

```
- (void)prepareLayout {
    [super prepareLayout];
    // セルの中央判定用しきい値を設定する
    _centerRateThreshold = _cellInterval /
        self.collectionView.bounds.size.width;
}
```

以上で、最低限のシ
ンプル実装CoverFlow
が完成

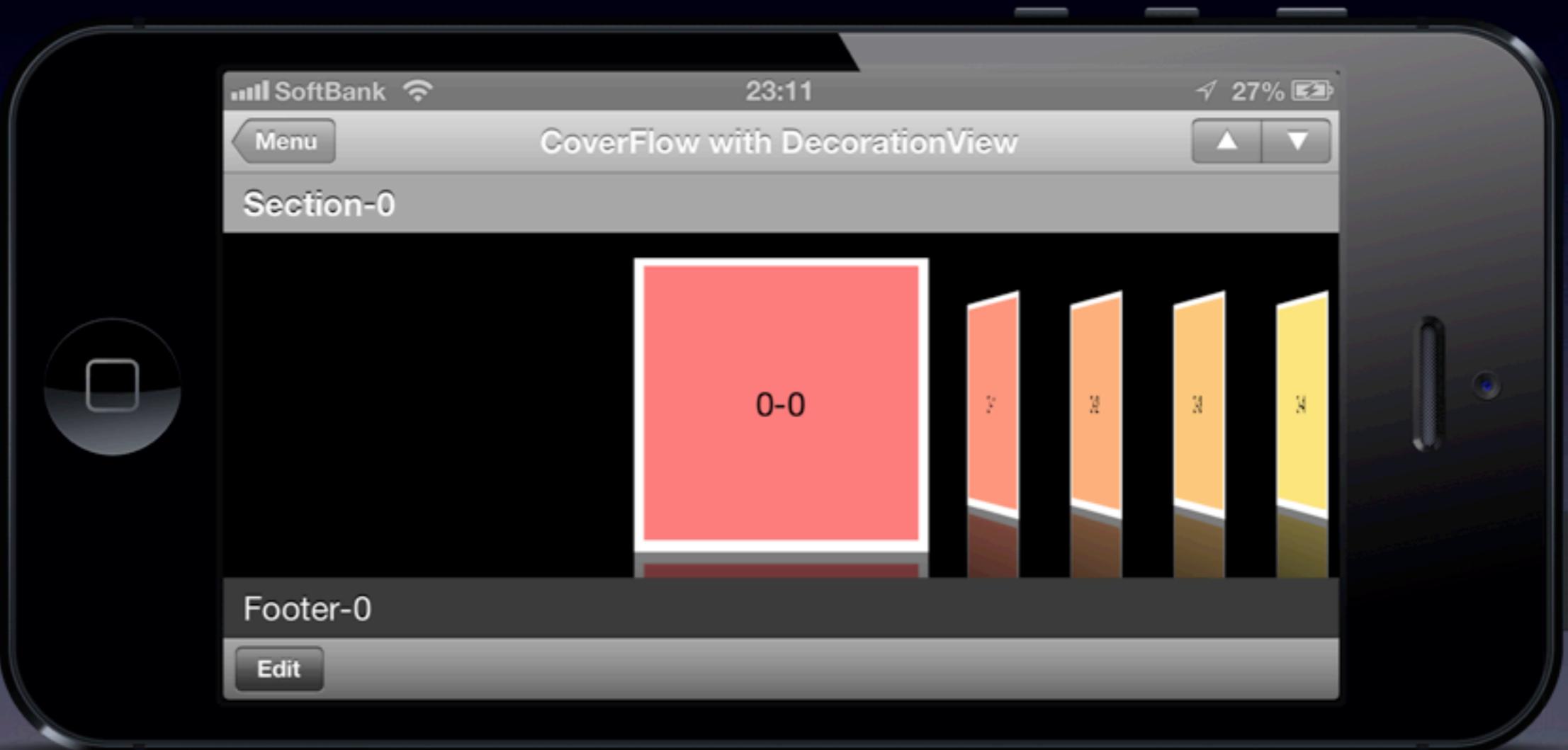
Demo

シンプルすぎるよね

- 端のセルが中央に表示できない
- 複数セクションにしたい
- ヘッダーやフッターも表示したい
- セルの追加や削除は？
- カバーフローなら反射の映り込みもないと...

- シルエットの端の上に余白をつけて調整すればOK
- 複数セクションの合計セル数を元にcontentSizeを計算
- ヘッダーのlayoutAttributesForSupplementaryViewOfKind:atIndexPath:を実装し、layoutAttributesForElementsInRect: の戻り値にもそれらを含めれば OK
- セルの実装はそのままでもそれなりに動くのでOK。
必要に応じて、prepareForCollectionViewUpdates,
initialLayoutAttributesForAppearingItemAtIndexPath:,
finalLayoutAttributesForDisappearingItemAtIndexPath: 等を実装
- カバーの上に余白をつけて調整すればOK
デコレーションビューを使って実現可能。
layoutAttributesForDecorationViewOfKind:atIndexPath: を実装し、
layoutAttributesForElementsInRect: の戻り値にもそれらを含めれば OK

CVCLCoverFlowLayout 完成版



- <https://github.com/sawat/CVCLCustomLayouts>

Demo

残念なお知らせ...

- 現在のUICollectionViewの実装にバグがあるようで、デコレーションビューが再利用されず、メモリリークが発生します...。
- さっきのデモの鏡像の部分です...。
- 参考) <http://qa.atmarkit.co.jp/q/2558>
- なので、デコレーションビューを本格的に使用するのはしばらく待った方が良さそうです。。。

ご清聴

ありがとうございました