里脊串的开发随笔

途有境而观无垠

2014-09-28 · 经验介绍

# Masonry介绍与使用实践(快速上手Autolayout)

## 前言

```
1  MagicNumber -> autoresizingMask -> autolayout
```

以上是纯手写代码所经历的关于页面布局的三个时期

- 在iphone1-iphone3gs时代 window的size固定为(320,480) 我们只需要简单计算一下相对位置就好了
- 在iphone4-iphone4s时代 苹果推出了retina屏 但是给了码农们非常大的福利:window的size不变
- 在iphone5-iphone5s时代 window的size变了(320,568) 这时`autoresizingMask`派上了用场(为啥这时候不用Autolayout? 因为还要支持ios5呗) 简单的适配一下即可
- 在iphone6+时代 window的width也发生了变化(相对5和5s的屏幕比例没有变化) 终于是时候抛弃`autoresizingMask`改用autolayout了(不用支持ios5了 相对于屏幕适配的多样性来说`autoresizingMask`也已经过时了)

那如何快速的上手autolayout呢? 说实话 当年ios6推出的同时新增了autolayout的特性 我看了一下官方文档和demo 就立马抛弃到一边了 因为实在过于的`繁琐和啰嗦`(有过经验的朋友肯定有同感)

直到iphone6发布之后 我知道使用autolayout势在必行了 这时想起了以前在浏览Github看到过的一个第三方库Masonry 在花了几个小时的研究使用后 我就将autolayout掌握了(`重点是我并没有学习任何的官方文档或者其他的关于autolayout的知识`) 这就是我为什么要写下这篇文章来推荐它的原因

# 介绍

Masonry 源码

Masonry是一个轻量级的布局框架 拥有自己的描述语法 采用更优雅的链式语法封装自动布局 简洁明了并具有高可读性 `而且同时支持 iOS 和 Max OS X`

我们先来看一段官方的sample code来认识一下Masonry

```
[view1 mas_makeConstraints:^(MASConstraintMaker *make) {
    make.edges.equalTo(superview).with.insets(padding);
}];
```

看到block里面的那句话: `make edges equalTo superview with insets`
通过链式的自然语言 就把view1给autolayout好了 是不是简单易懂?

# 使用

看一下Masonry支持哪一些属性

```
@property (nonatomic, strong, readonly) MASConstraint *left;
@property (nonatomic, strong, readonly) MASConstraint *top;
@property (nonatomic, strong, readonly) MASConstraint *right;
@property (nonatomic, strong, readonly) MASConstraint *bottom;
@property (nonatomic, strong, readonly) MASConstraint *leading;
@property (nonatomic, strong, readonly) MASConstraint *trailing;
@property (nonatomic, strong, readonly) MASConstraint *width;
@property (nonatomic, strong, readonly) MASConstraint *height;
@property (nonatomic, strong, readonly) MASConstraint *centerX;
```

```
10  @property (nonatomic, strong, readonly) MASConstraint *centerY;
11  @property (nonatomic, strong, readonly) MASConstraint *baseline;
```

这些属性与NSLayoutAttrubute的对照表如下

| Masonry | NSAutoLayout | 说明 |
| --- | --- | --- |
| left | NSLayoutAttributeLeft | 左侧 |
| top | NSLayoutAttributeTop | 上侧 |
| right | NSLayoutAttributeRight | 右侧 |
| bottom | NSLayoutAttributeBottom | 下侧 |
| leading | NSLayoutAttributeLeading | 首部 |
| trailing | NSLayoutAttributeTrailing | 尾部 |
| width | NSLayoutAttributeWidth | 宽 |
| height | NSLayoutAttributeHeight | 高 |
| centerX | NSLayoutAttributeCenterX | 横向中点 |
| centerY | NSLayoutAttributeCenterY | 纵向中点 |
| baseline | NSLayoutAttributeBaseline | 文本基线 |

其中leading与left trailing与right 在正常情况下是等价的 但是当一些布局是从右至左时(比如阿拉伯文?没有类似的经验) 则会对调 换句话说就是基本可以不理不用 用left和right就好了

在ios8发布后 又新增了一堆奇奇怪怪的属性(有兴趣的朋友可以去瞅瞅) Masonry暂时还不支持(不过你要支持ios6,ios7 就没必要去管那么多了)

下面进入正题(为了方便 我们测试的superView都是一个size为(300,300)的UIView)

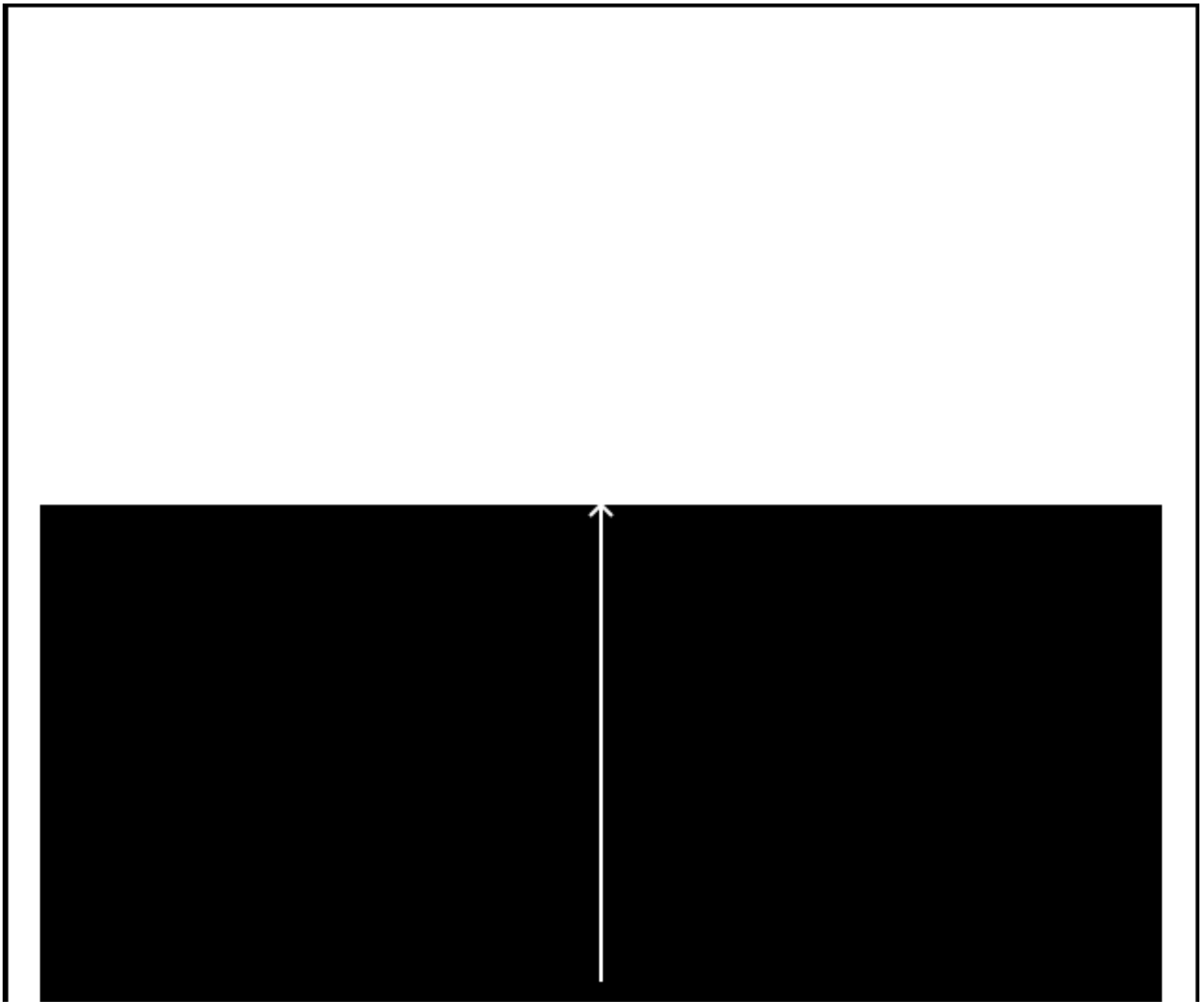
下面 通过一些简单的实例来简单介绍如何`轻松愉快`的使用Masonry:

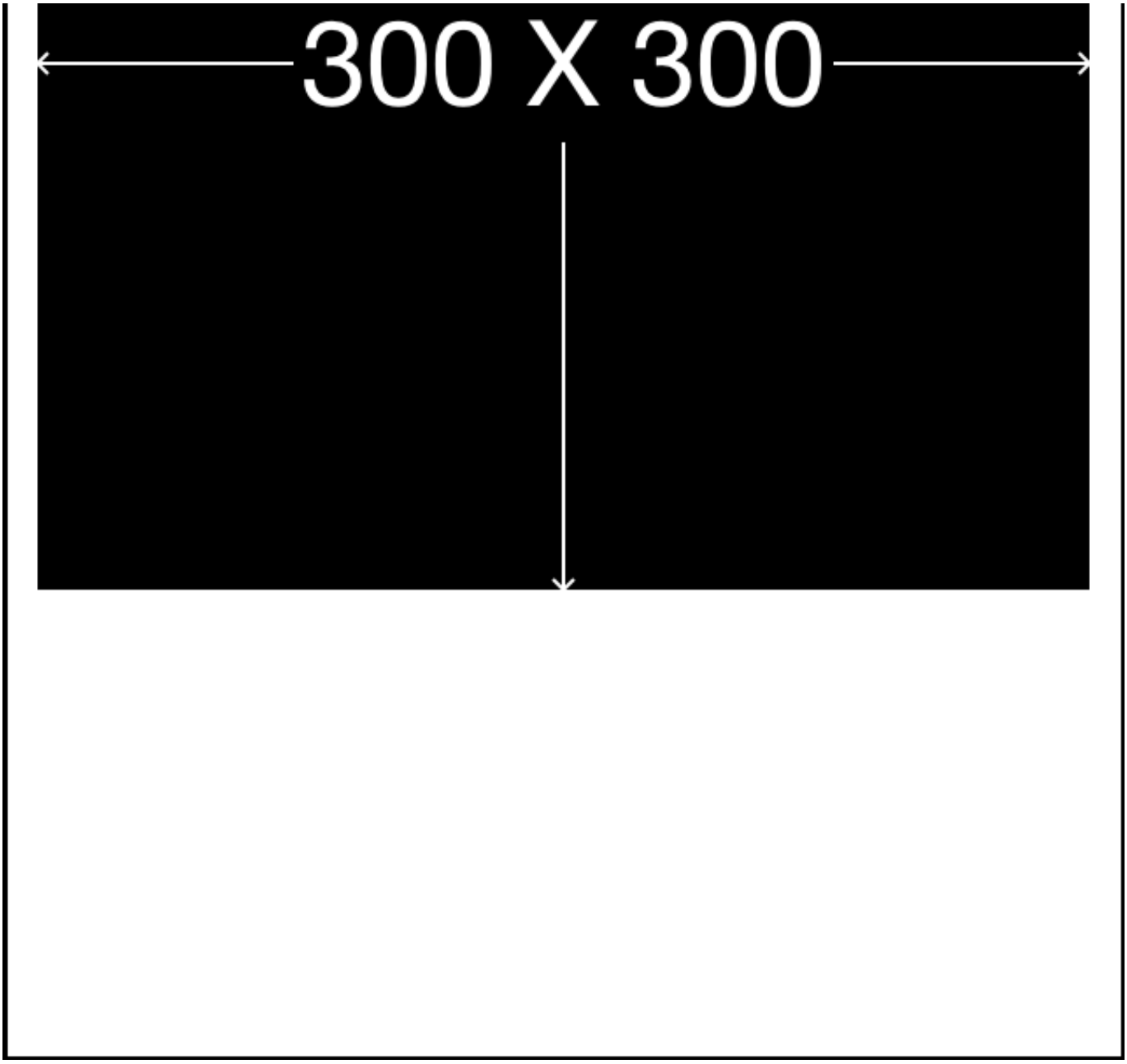
## 1. [基础] *居中显示一个view*

```objc
- (void)viewDidLoad
{
    [super viewDidLoad];
    // Do any additional setup after loading the view.

    UIView *sv = [UIView new];
    [sv showPlaceHolder];
    sv.backgroundColor = [UIColor blackColor];
    [self.view addSubview:sv];
    [sv mas_makeConstraints:^(MASConstraintMaker *make) {
        make.center.equalTo(self.view);
        make.size.mas_equalTo(CGSizeMake(300, 300));
    }];

}
```






代码效果

使用我之间写的MMPlaceHolder 可以看到superview已经按照我们预期居中并且设置成了适当的大小

那么先看看这几行代码

```
1
2  //从此以后基本可以抛弃CGRectMake了
3  UIView *sv = [UIView new];
4
5  //在做autoLayout之前 一定要先将view添加到superview上 否则会报错
6  [self.view addSubview:sv];
7
```

```
8  //mas_makeConstraints就是Masonry的autolayout添加函数 将所需的约束添加到block中行
9  [sv mas_makeConstraints:^(MASConstraintMaker *make) {
10
11     //将sv居中(很容易理解吧?)
12     make.center.equalTo(self.view);
13
14     //将size设置成(300,300)
15     make.size.mas_equalTo(CGSizeMake(300, 300));
16 }];
```

这里有两个问题要分解一下

- 首先在Masonry中能够添加autolayout约束有三个函数

```
1
2  - (NSArray *)mas_makeConstraints:(void(^)(MASConstraintMaker *make))block;
3  - (NSArray *)mas_updateConstraints:(void(^)(MASConstraintMaker *make))bloc
4  - (NSArray *)mas_remakeConstraints:(void(^)(MASConstraintMaker *make))bloc
5
6  /*
7      mas_makeConstraints 只负责新增约束 Autolayout不能同时存在两条针对于同一对象的约
8      mas_updateConstraints 针对上面的情况 会更新在block中出现的约束 不会导致出现两个
9      mas_remakeConstraints 则会清除之前的所有约束 仅保留最新的约束
10
11     三种函数善加利用 就可以应对各种情况了
12 */
```

- 其次 equalTo 和 mas_equalTo的区别在哪里呢? 其实 mas_equalTo是一个MACRO

```
1  #define mas_equalTo(...)                 equalTo(MASBoxValue((__VA_ARGS__))
2  #define mas_greaterThanOrEqualTo(...)    greaterThanOrEqualTo(MASBoxValue((
3  #define mas_lessThanOrEqualTo(...)       lessThanOrEqualTo(MASBoxValue((__V
4
5  #define mas_offset(...)                  valueOffset(MASBoxValue((__VA_ARGS
```

可以看到 mas_equalTo只是对其参数进行了一个BOX操作(装箱) MASBoxValue的定义具体可以看看源代码 太长就不贴出来了

所支持的类型 除了NSNumber支持的那些数值类型之外 就只支持`CGPoint` `CGSize` `UIEdgeInsets`

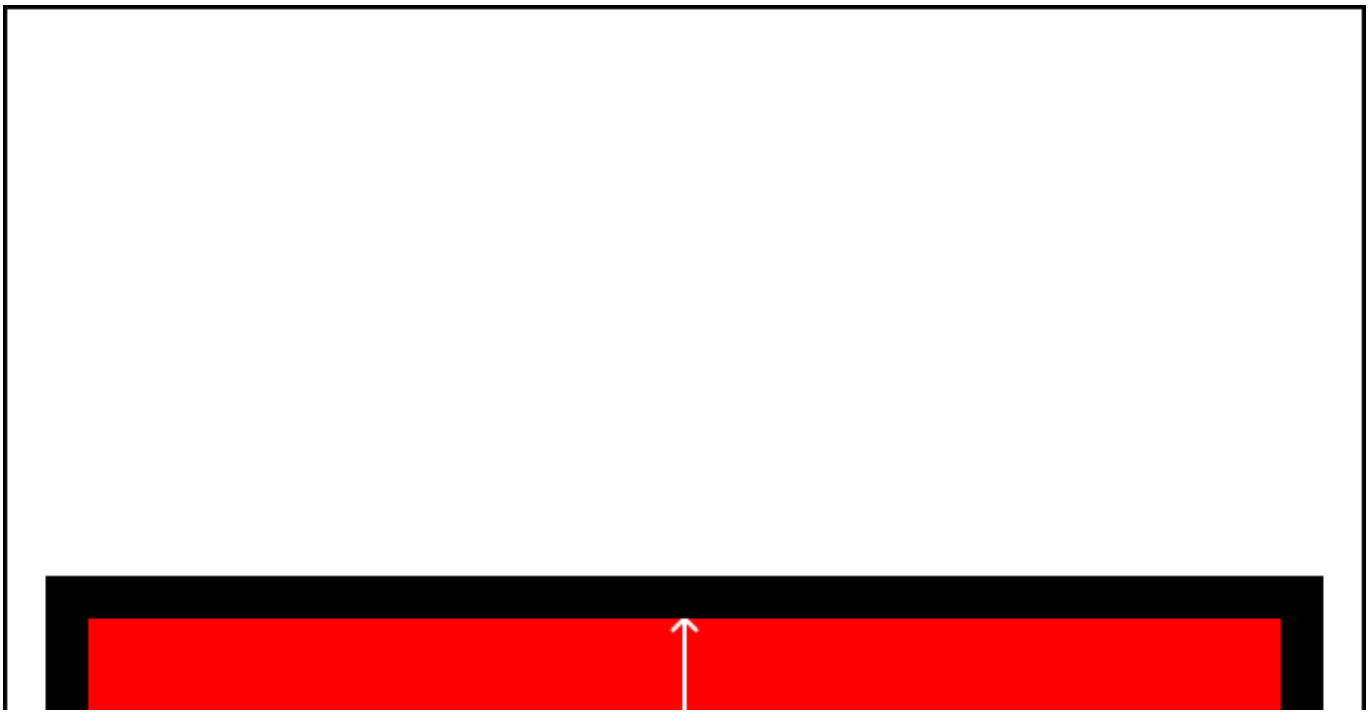
介绍完这几个问题 我们就继续往下了 PS:刚才定义的sv会成为我们接下来所有sample的superView

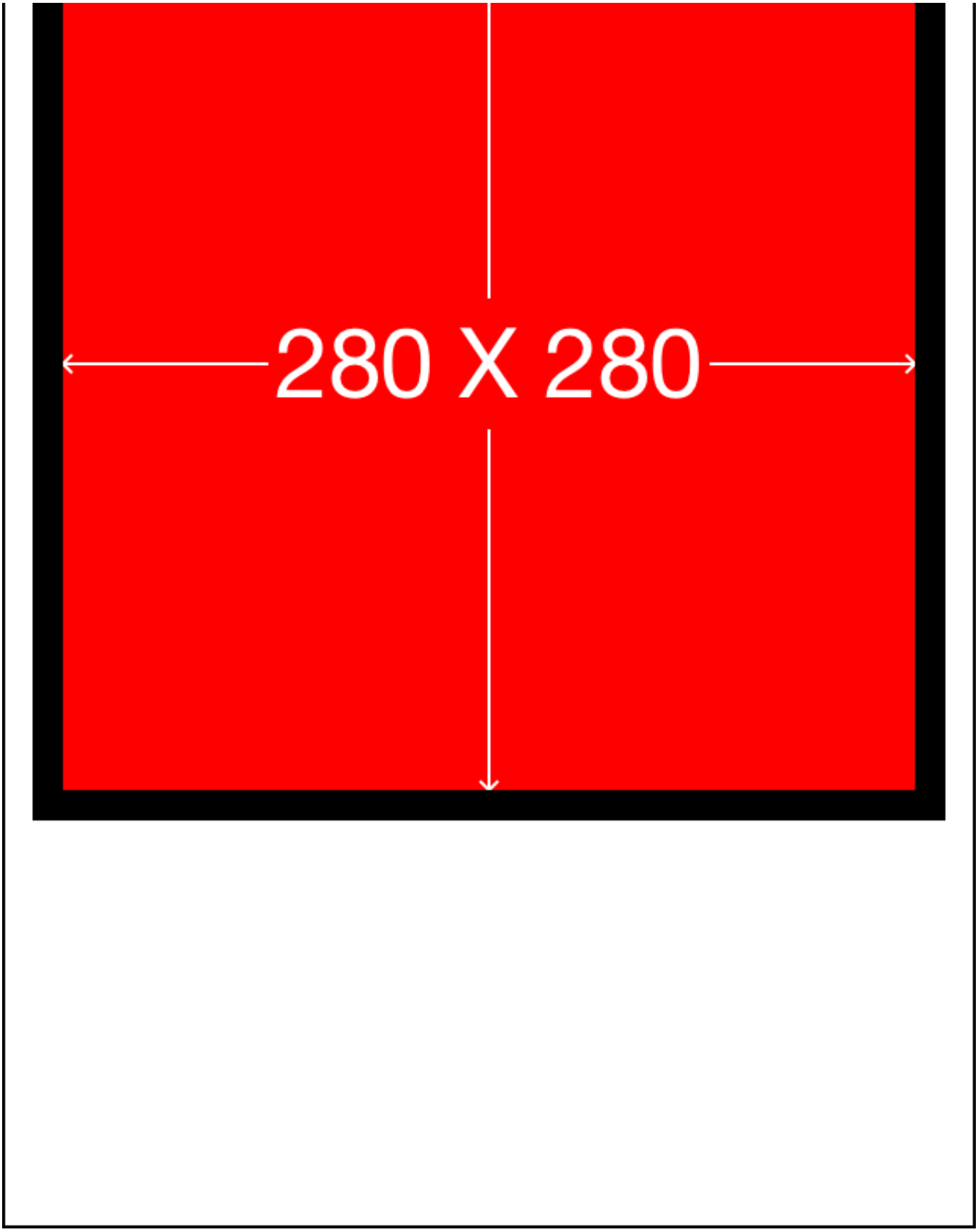
## 2. [初级] 让一个view略小于其superView(边距为10)

```
UIView *sv1 = [UIView new];
[sv1 showPlaceHolder];
sv1.backgroundColor = [UIColor redColor];
[sv addSubview:sv1];
[sv1 mas_makeConstraints:^(MASConstraintMaker *make) {
    make.edges.equalTo(sv).with.insets(UIEdgeInsetsMake(10, 10, 10, 10));

    /* 等价于
    make.top.equalTo(sv).with.offset(10);
    make.left.equalTo(sv).with.offset(10);
    make.bottom.equalTo(sv).with.offset(-10);
    make.right.equalTo(sv).with.offset(-10);
    */

    /* 也等价于
    make.top.left.bottom.and.right.equalTo(sv).with.insets(UIEdgeInsetsMak
    */
}];
```






代码效果

可以看到 edges 其实就是top,left,bottom,right的一个简化 分开写也可以 一句话更省事

那么为什么bottom和right里的offset是负数呢? 因为这里计算的是绝对的数值 计算的bottom需要小于sv的底部高度 所以要-10 同理用于right

这里有意思的地方是`and`和`with` 其实这两个函数什么事情都没做

```
- (MASConstraint *)with {
    return self;
}

- (MASConstraint *)and {
    return self;
}
```

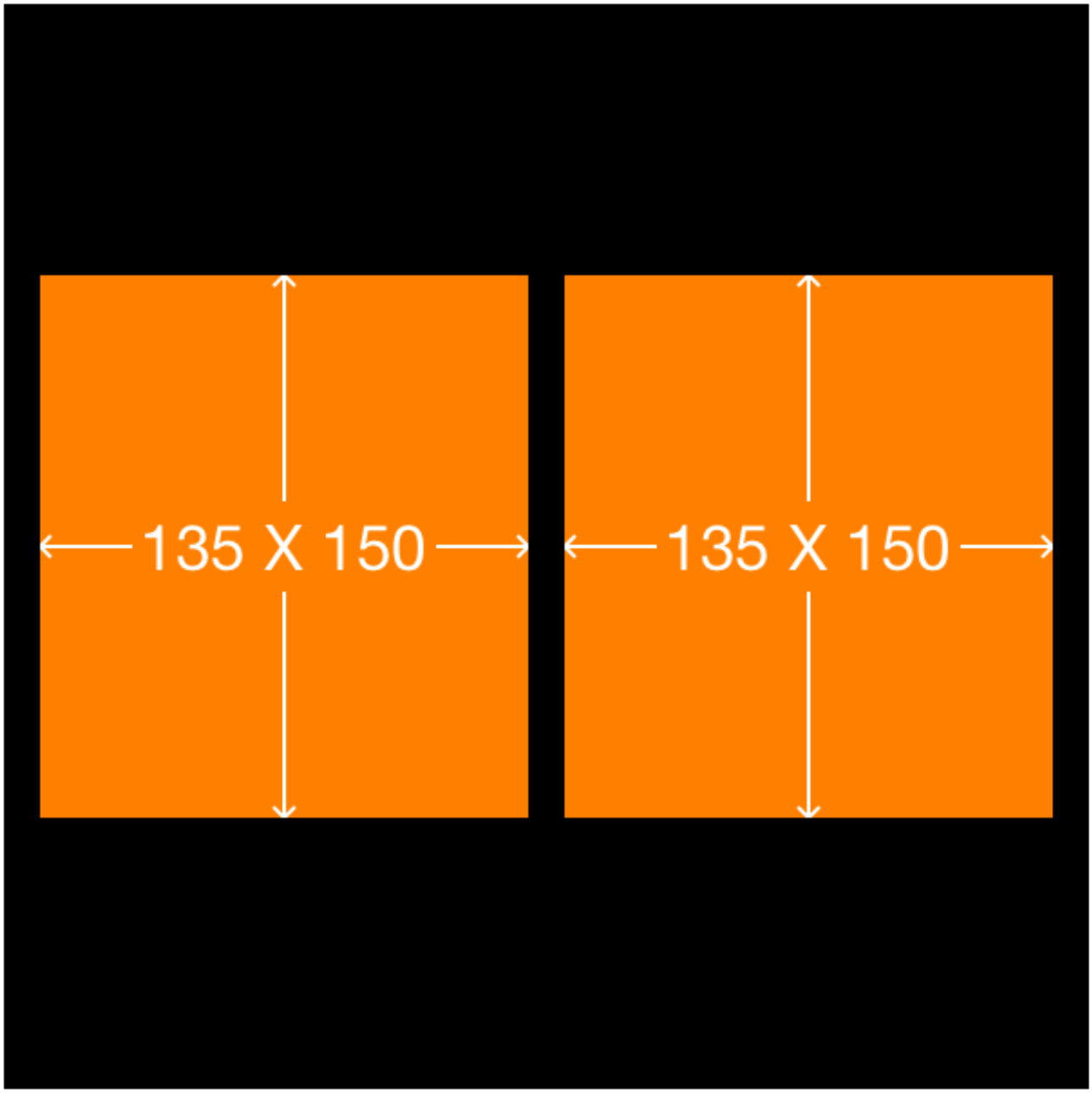
但是用在这种链式语法中 就非常的巧妙和易懂 不得不佩服作者的心思(虽然我现在基本都会省略)

## 3. [初级] 让两个高度为150的view垂直居中且等宽且等间隔排列 间隔为10(自动计算其宽度)

```
int padding1 = 10;

[sv2 mas_makeConstraints:^(MASConstraintMaker *make) {
    make.centerY.mas_equalTo(sv.mas_centerY);
    make.left.equalTo(sv.mas_left).with.offset(padding1);
    make.right.equalTo(sv3.mas_left).with.offset(-padding1);
    make.height.mas_equalTo(@150);
    make.width.equalTo(sv3);
}];

[sv3 mas_makeConstraints:^(MASConstraintMaker *make) {
    make.centerY.mas_equalTo(sv.mas_centerY);
    make.left.equalTo(sv2.mas_right).with.offset(padding1);
    make.right.equalTo(sv.mas_right).with.offset(-padding1);
    make.height.mas_equalTo(@150);
    make.width.equalTo(sv2);
}];
```


135 X 150
135 X 150

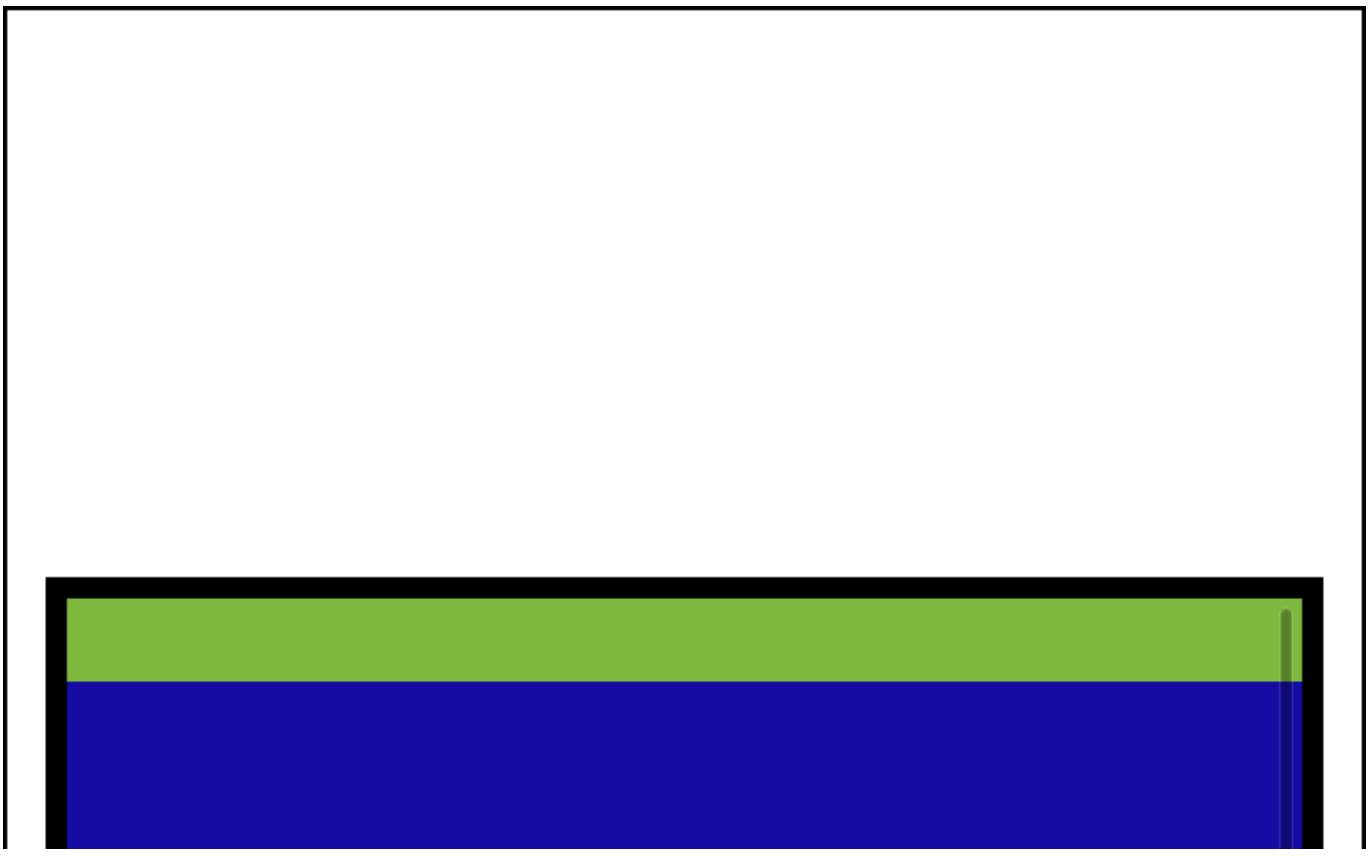
代码效果

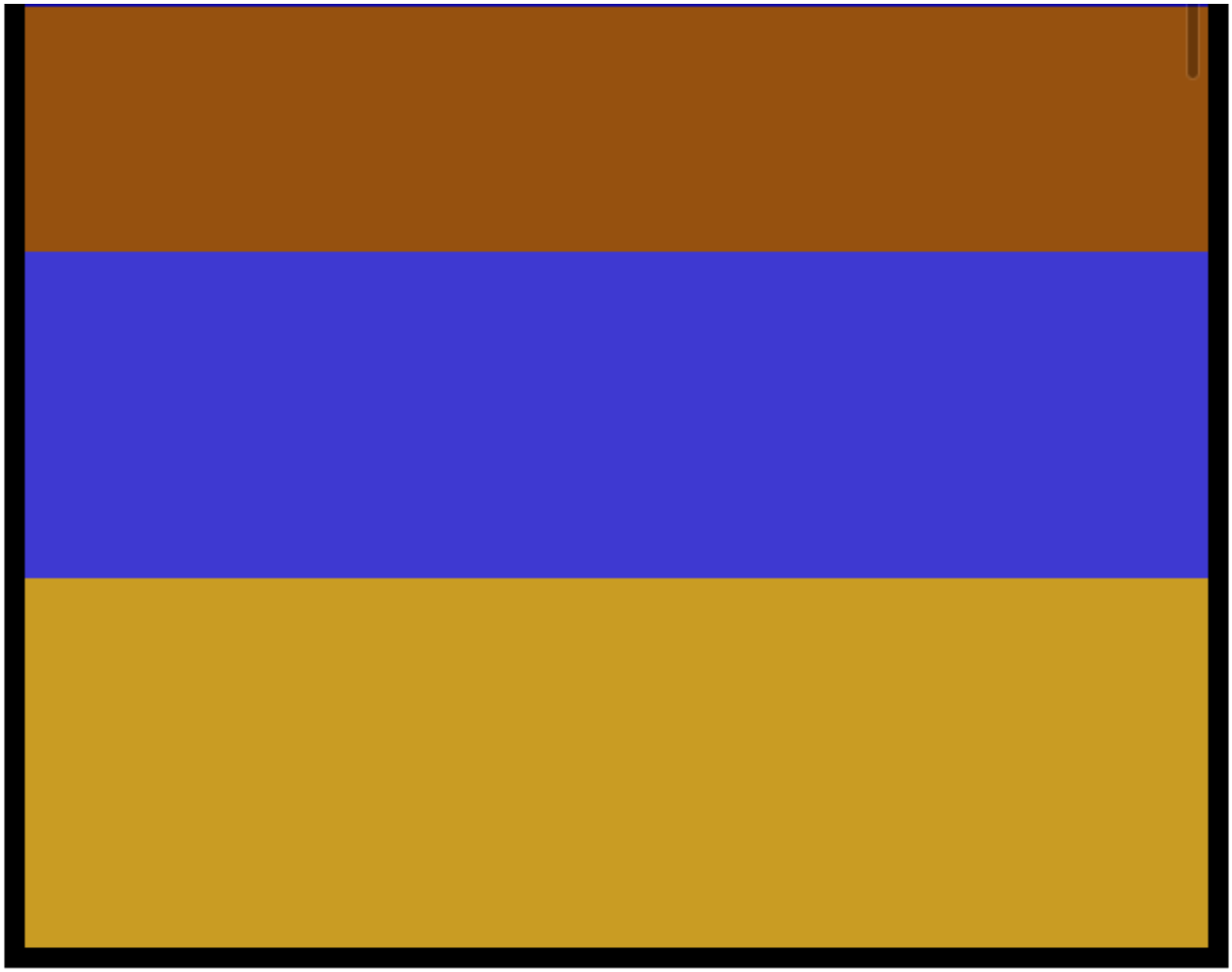
这里我们在两个子view之间互相设置的约束 可以看到他们的宽度在约束下自动的被计算出来了

## 4. [中级] *在UIScrollView顺序排列一些view并自动计算contentSize*

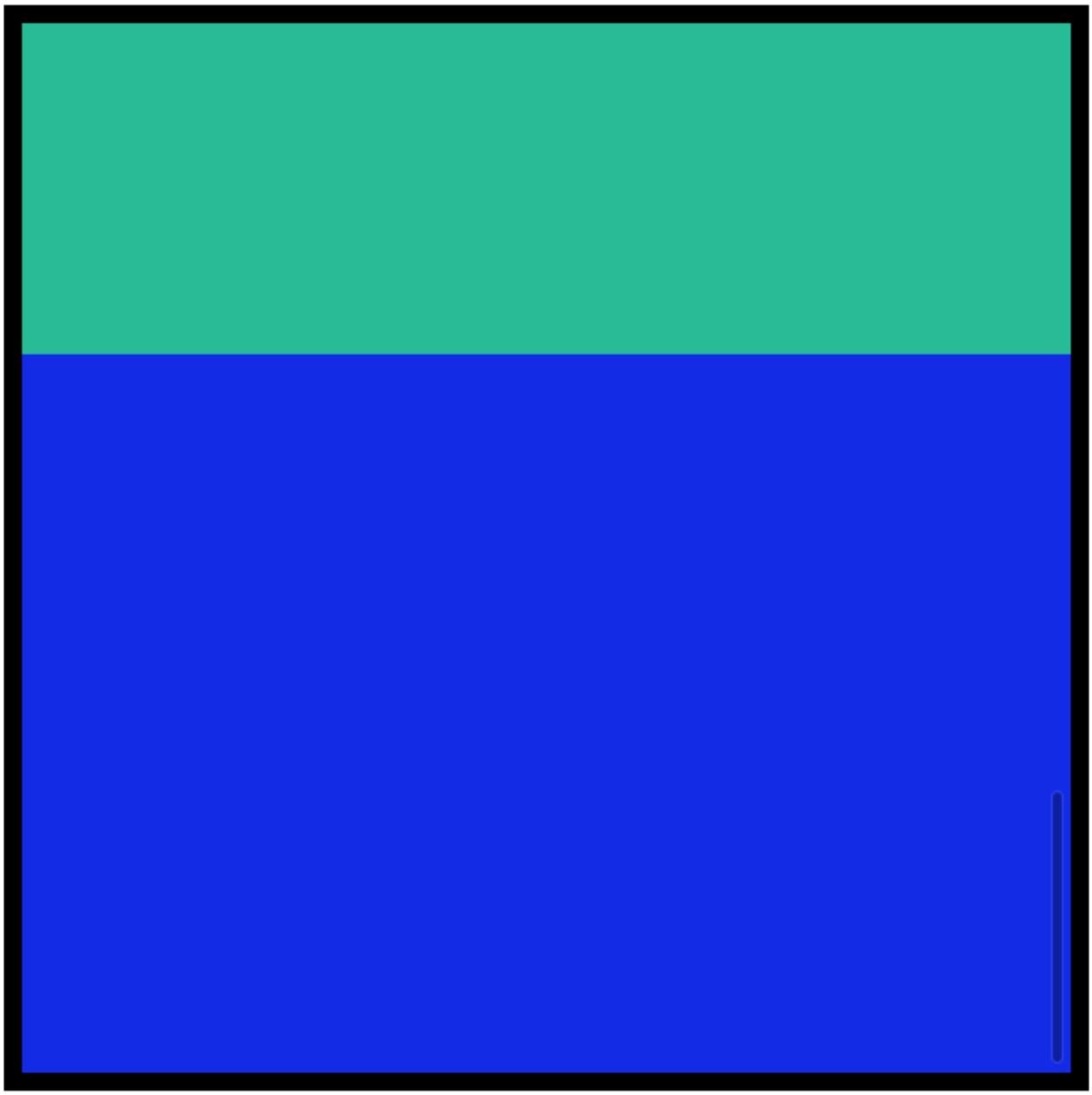
```
1  UIScrollView *scrollView = [UIScrollView new];
2  scrollView.backgroundColor = [UIColor whiteColor];
3  [sv addSubview:scrollView];
4  [scrollView mas_makeConstraints:^(MASConstraintMaker *make) {
5      make.edges.equalTo(sv).with.insets(UIEdgeInsetsMake(5,5,5,5));
6  }];
7
8  UIView *container = [UIView new];
9  [scrollView addSubview:container];
10 [container mas_makeConstraints:^(MASConstraintMaker *make) {
11     make.edges.equalTo(scrollView);
12     make.width.equalTo(scrollView);
13 }];
14
15 int count = 10;
16
17 UIView *lastView = nil;
18
19 for ( int i = 1 ; i <= count ; ++i )
20 {
21     UIView *subv = [UIView new];
22     [container addSubview:subv];
23     subv.backgroundColor = [UIColor colorWithHue:( arc4random() % 256 / 25
24                                       saturation:( arc4random() % 128 / 25
25                                       brightness:( arc4random() % 128 / 25
26                                            alpha:1];
```

```
27
28      [subv mas_makeConstraints:^(MASConstraintMaker *make) {
29          make.left.and.right.equalTo(container);
30          make.height.mas_equalTo(@(20*i));
31
32          if ( lastView )
33          {
34              make.top.mas_equalTo(lastView.mas_bottom);
35          }
36          else
37          {
38              make.top.mas_equalTo(container.mas_top);
39          }
40      }];
41
42      lastView = subv;
43  }
44
45
46  [container mas_makeConstraints:^(MASConstraintMaker *make) {
47      make.bottom.equalTo(lastView.mas_bottom);
48  }];
```





头部效果



尾部效果

从scrollView的scrollIndicator可以看出 scrollView的内部已如我们所想排列好了

这里的关键就在于container这个view起到了一个中间层的作用 能够自动的计算uiscrollView的contentSize

## 5. [高级] *横向或者纵向等间隙的排列一组view*

很遗憾 autoLayout并没有直接提供等间隙排列的方法(Masonry的官方demo中也没有对应的案例) 但是参考案例3 我们可以通过一个小技巧来实现这个目的 为此我写了一个Category

```
1  @implementation UIView(Masonry_LJC)
2
3  - (void) distributeSpacingHorizontallyWith:(NSArray*)views
4  {
5      NSMutableArray *spaces = [NSMutableArray arrayWithCapacity:views.count
6
7      for ( int i = 0 ; i < views.count+1 ; ++i )
8      {
9          UIView *v = [UIView new];
10         [spaces addObject:v];
11         [self addSubview:v];
12
13         [v mas_makeConstraints:^(MASConstraintMaker *make) {
14             make.width.equalTo(v.mas_height);
15         }];
16     }
17
18     UIView *v0 = spaces[0];
19
20     [v0 mas_makeConstraints:^(MASConstraintMaker *make) {
21         make.left.equalTo(self.mas_left);
22         make.centerY.equalTo(((UIView*)views[0]).mas_centerY);
23     }];
24
```

```
25      UIView *lastSpace = v0;
26      for ( int i = 0 ; i < views.count; ++i )
27      {
28          UIView *obj = views[i];
29          UIView *space = spaces[i+1];
30
31          [obj mas_makeConstraints:^(MASConstraintMaker *make) {
32              make.left.equalTo(lastSpace.mas_right);
33          }];
34
35          [space mas_makeConstraints:^(MASConstraintMaker *make) {
36              make.left.equalTo(obj.mas_right);
37              make.centerY.equalTo(obj.mas_centerY);
38              make.width.equalTo(v0);
39          }];
40
41          lastSpace = space;
42      }
43
44      [lastSpace mas_makeConstraints:^(MASConstraintMaker *make) {
45          make.right.equalTo(self.mas_right);
46      }];
47
48  }
49
50  - (void) distributeSpacingVerticallyWith:(NSArray*)views
51  {
52      NSMutableArray *spaces = [NSMutableArray arrayWithCapacity:views.count
53
54      for ( int i = 0 ; i < views.count+1 ; ++i )
55      {
56          UIView *v = [UIView new];
57          [spaces addObject:v];
58          [self addSubview:v];
59
60          [v mas_makeConstraints:^(MASConstraintMaker *make) {
61              make.width.equalTo(v.mas_height);
62          }];
63      }
64
65
```

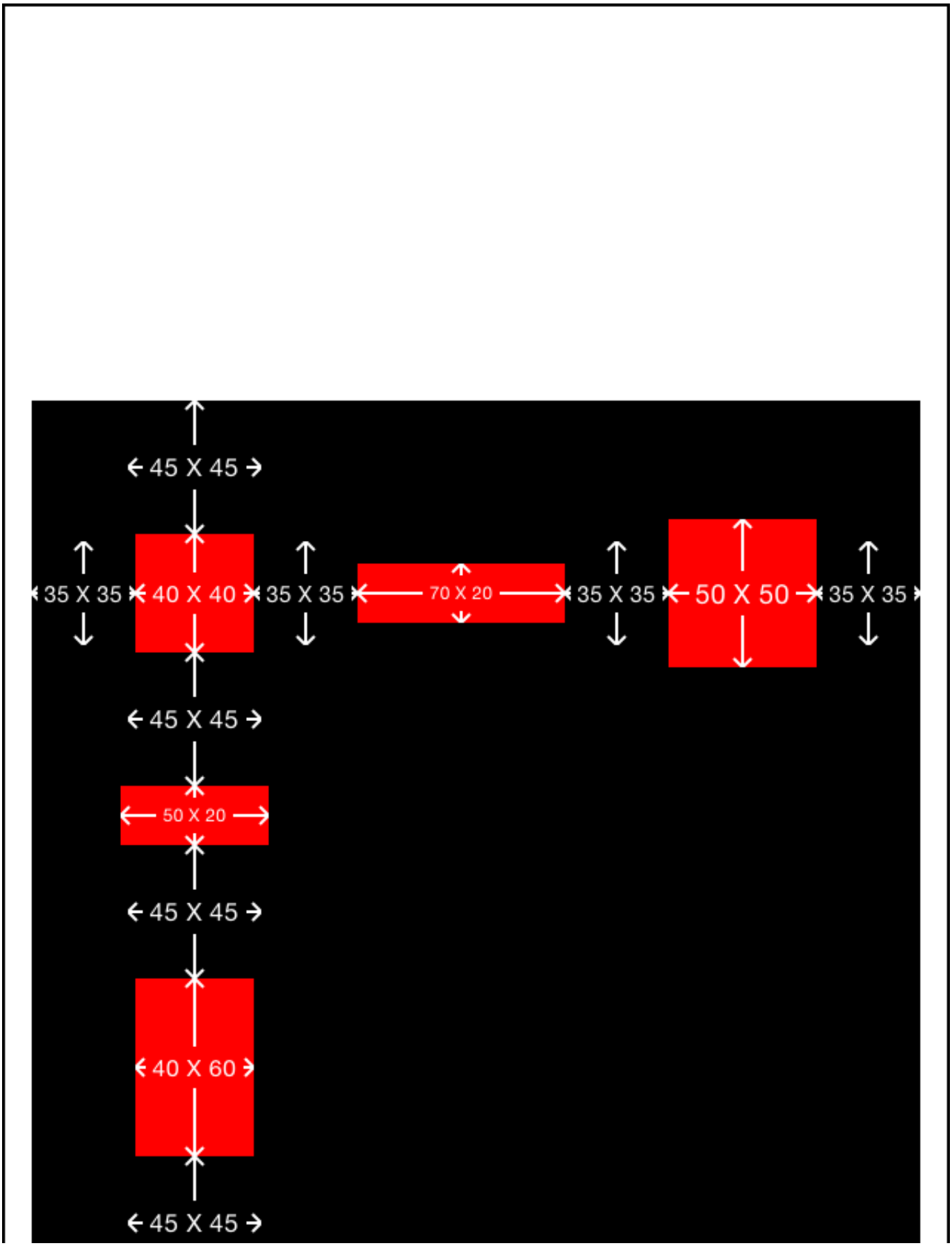
```
66      UIView *v0 = spaces[0];
67
68      [v0 mas_makeConstraints:^(MASConstraintMaker *make) {
69          make.top.equalTo(self.mas_top);
70          make.centerX.equalTo(((UIView*)views[0]).mas_centerX);
71      }];
72
73      UIView *lastSpace = v0;
74      for ( int i = 0 ; i < views.count; ++i )
75      {
76          UIView *obj = views[i];
77          UIView *space = spaces[i+1];
78
79          [obj mas_makeConstraints:^(MASConstraintMaker *make) {
80              make.top.equalTo(lastSpace.mas_bottom);
81          }];
82
83          [space mas_makeConstraints:^(MASConstraintMaker *make) {
84              make.top.equalTo(obj.mas_bottom);
85              make.centerX.equalTo(obj.mas_centerX);
86              make.height.equalTo(v0);
87          }];
88
89          lastSpace = space;
90      }
91
92      [lastSpace mas_makeConstraints:^(MASConstraintMaker *make) {
93          make.bottom.equalTo(self.mas_bottom);
94      }];
95
96  }
97
98  @end
```

简单的来测试一下

```
1  UIView *sv11 = [UIView new];
2  UIView *sv12 = [UIView new];
3  UIView *sv13 = [UIView new];
4  UIView *sv21 = [UIView new];
```

```
5  UIView *sv31 = [UIView new];
6
7  sv11.backgroundColor = [UIColor redColor];
8  sv12.backgroundColor = [UIColor redColor];
9  sv13.backgroundColor = [UIColor redColor];
10 sv21.backgroundColor = [UIColor redColor];
11 sv31.backgroundColor = [UIColor redColor];
12
13 [sv addSubview:sv11];
14 [sv addSubview:sv12];
15 [sv addSubview:sv13];
16 [sv addSubview:sv21];
17 [sv addSubview:sv31];
18
19 //给予不同的大小 测试效果
20
21 [sv11 mas_makeConstraints:^(MASConstraintMaker *make) {
22     make.centerY.equalTo(@[sv12,sv13]);
23     make.centerX.equalTo(@[sv21,sv31]);
24     make.size.mas_equalTo(CGSizeMake(40, 40));
25 }];
26
27 [sv12 mas_makeConstraints:^(MASConstraintMaker *make) {
28     make.size.mas_equalTo(CGSizeMake(70, 20));
29 }];
30
31 [sv13 mas_makeConstraints:^(MASConstraintMaker *make) {
32     make.size.mas_equalTo(CGSizeMake(50, 50));
33 }];
34
35 [sv21 mas_makeConstraints:^(MASConstraintMaker *make) {
36     make.size.mas_equalTo(CGSizeMake(50, 20));
37 }];
38
39 [sv31 mas_makeConstraints:^(MASConstraintMaker *make) {
40     make.size.mas_equalTo(CGSizeMake(40, 60));
41 }];
42
43 [sv distributeSpacingHorizontallyWith:@[sv11,sv12,sv13]];
44 [sv distributeSpacingVerticallyWith:@[sv11,sv21,sv31]];
45
```

```
46  [sv showPlaceHolderWithAllSubviews];
47  [sv hidePlaceHolder];
```



代码效果

perfect! 简洁明了的达到了我们所要的效果

这里所用的技巧就是 使用空白的占位view来填充我们目标view的旁边 这点通过图上的空白标注可以看出来

# 小结

通过以上5个案例 我觉得已经把Masonry的常用功能介绍得差不多了 以上五个例子的代码可以在这里找到 如果你觉得意犹未尽呢 请下载官方的demo来学习

总而言之 Masonry是一个非常优秀的autolayout库 能够节省大量的开发和学习时间 尤其适合我这种纯代码的iOSer 在iPhone6发布后引发的适配潮中 Masonry一定可以助你一臂之力 :)

---

#Autolayout #Masonry #教程 文章評論 分享到

NEWER

如何用纯代码构建一个Widget(today extension)

OLDER

如何强制旋转屏幕

123 条评论 23 条新浪微博 最新 最早 最热

流离寻岸007

是不错 ,但还是 习惯用UIView+layout

9月18日 回复 顶 转发

里脊串

回复 90后小金头: 是的 这个地方是笔误 不过上次经人提醒已经修正了 demo里的代码是没问题的 可以看下demo

8月28日 回复 顶 转发

90后小金头

2. [初级] 让一个view略小于其superView(边距为10) 的self不对吧，是不是应该改为sv,不然报错！

8月28日 回复 顶 转发

xx11dragon

回复 里脊串: 再次感谢~

8月19日 回复 顶 转发

里脊串

回复 xx11dragon: cell也是可以根据label的高度来确定的 你可以固定宽度 高度就自动计算了

具体的你可以看看 https://github.com/forkingdog/UITableView-FDTemplateLayoutCell 这个库 看看他是用什么方法来动态计算autolayout的cell高度的

8月19日 回复 顶 转发

xx11dragon

回复 里脊串: nice，nice。verynice。非常感谢您。
0，0 最近研究 cell里面 放了一个label
label是动态高度 label.numberOfLines = 0; lineBreakMode = NSLineBreakByCharWrapping;
我发现 用autolayout 只约束top，left，right 他会根据字数自动设置高度。

但是TableViewCell是无法根据动态的label高度来 动态定制cell自己高度的。

目前解决这个问题。我能想到的就是用 boundingRectWithSize 返回的CGsize的height属性 写死label的高度，再用您提供的.lessThanOrEqualTo 约束cell的高度，不知道这是不是已经是最优方案了，再次感谢。

8月19日 回复 顶 转发

里脊串

回复 xx11dragon: 你试试

[self.icon mas_makeConstraints:^(MASConstraintMaker *make) {

make.top.equalTo(self.contentView.mas_top).with.offset(10);
make.left.equalTo(self.contentView.mas_left).with.offset(10);
make.width.equalTo(@70);
make.height.equalTo(@70);
make.right.lessThanOrEqualTo(self.contentView.mas_right).with.offset(-10);
make.bottom.lessThanOrEqualTo(self.contentView.mas_bottom).with.offset(-10);
}];

我这里是可以的

8月19日 回复 顶 转发

xx11dragon

回复 xx11dragon: -(CGFloat)tableView:(UITableView *)tableView heightForRowAtIndexPath:(NSIndexPath *)indexPath {
CGSize size = [_cell.contentView systemLayoutSizeFittingSize:UILayoutFittingCompressedSize];
NSLog(@"%f",size.height);

return size.height + 1;
}

打印出来永远是0.0f;

8月19日 回复 顶 转发

xx11dragon

串兄有个问题请教一下，不知想用autolayout如何实现动态定制TableViewCell.例如我想在cell里添加一个image 70x70 top bottom各是10. 然而我怎么写都不太对...
[self.icon mas_makeConstraints:^(MASConstraintMaker *make) {

make.top.equalTo(self.contentView).with.offset(10);
make.left.equalTo(self.contentView).with.offset(10);
make.width.equalTo(@70);
make.height.equalTo(@70);

make.bottom.equalTo(self.contentView.mas_bottom).with.offset(10);
}];

8月19日 回复 顶 转发

里脊串

回复 48K纯摔: 是的是的 马上修改 不过demo的代码里应该是对的

8月13日 回复 顶 转发

48K纯摔

初级的那个例子有个小错误，make.edges.equalTo(self).with.insets(UIEdgeInsetsMake(10, 10, 10, 10));这句中的self应该是vs才对，不然会报错的，应该是相对于vs，或者写成sv1.superview .

8月13日 回复 顶 转发

里脊串

回复 maceesti: 我不用xib的 所以不太清楚你的初始化过程 可以看下代码吗

8月5日 回复 顶 转发

maceesti

博主你好，我想问一下为什么我从xib拉进来的控件用masonry写约束没有效果

8月5日 回复 顶 转发

陈鹏

mas 怎么可以让控件自适应屏幕呢?

7月31日 回复 顶 转发

里脊串

回复 苦瓜:

殊途同归

7月23日 回复 顶 转发

苦瓜

回复 张泽阳: 哈哈 这么巧，我也扩展了一个左右（上下）间距的方法。

7月23日 回复 顶 转发

苦瓜

再一次非常感谢楼主，PureLayout 就提供有等间距排列一组View的方法，一直在纠结Masonry没有类似的方法，看到你写的分类简直是帮了我大忙了。

7月23日 回复 顶 转发

Immort

学习学习

7月15日 回复 顶 转发

里脊串

回复 从今以后: 嗯 这是因为我们只需要竖向滑动 如果我们需要横向滑动或者both的话 则宽度也必须像高度那样设置 话说这种方法的好处就是可以根据内容的不同动态的调整contentSize 如果是固定大小或者已知大小的话 直接设置ContentSize就ok了

7月14日 回复 顶 转发

从今以后

回复 yellow: containerView 必须有个明确的尺寸,然后再与 scrollView 边界约束, scrollView 才能计算出滚动范围.我想这里 containerView 的高可以根据子控件确定,所以宽就需要额外指定了.- -

7月14日 回复 顶 转发

里脊串

回复 yellow: 这里我参考的是Masonry的官方例子
https://github.com/SnapKit/Masonry/blob/master/Examples/Masonry%20iOS%20Examples/MASExampleScrollView.m

你可以试试 去掉那句 是否还能正常运行

7月1日 回复 顶 转发

yellow

第4点中，既然已经 make.edges.equalTo(scrollView); 了，应该高宽边距都设置好了呀，为什么还要make.width.equalTo(scrollView); 呢?

7月1日 回复 顶 转发

里脊串

回复 xx11dragon: scrollview约束了宽的

make.edges.equalTo(sv).with.insets(UIEdgeInsetsMake(5,5,5,5));

6月25日 回复 顶 转发

xx11dragon

UIView *container = [UIView new];
[scrollView addSubview:container];
[container mas_makeConstraints:^(MASConstraintMaker *make) {
make.edges.equalTo(scrollView);
make.width.equalTo(scrollView);//为什么container 需要约束width，scrollview不用=、=指点一下谢谢
}];

6月25日 回复 顶 转发

王-炜圣

GOOD

6月2日 回复 顶 转发

黄书培

学习一下

5月29日 回复 顶 转发

李文豪

回复 李婧: 我们可以通过category来扩展方法，但是它有个很大的局限性，不能扩展属性。于是，就有了专门用来扩展属性的机制：associative。

5月27日 回复 顶 转发

李文豪

回复 李婧: Category 想扩展属性 需要objc_setAssociatedObject 和 objc_getAssociatedObject 这两个来协助 才可以

5月27日 回复 顶 转发

秋儿：q1653687340

回复 free_fa: 因为等间隙了。所以x由父视图的宽决定，y由父视图的高决定。。。....求楼主批评

5月26日 回复 顶 转发

相思树下说书人CG

博主你好,我做一个tableViewCell的适配,里面会有多图,cell的高度要变化5,6,6+,如何所有约束添加之后获取cell的最终高度,感谢解答

5月14日 回复 顶 转发

袁国文

回复 里脊串: 赞，是这个问题,博主威武

5月14日 回复 顶 转发

里脊串

回复 碧野MAX: equalTo 一般指向某个属性 比如mas_left mas_right 或者某个view mas_equalTo 一般是指定某个绝对数值 比如 30 50

mas_equalTo的定义是

#define mas_equalTo(...) equalTo(MASBoxValue((__VA_ARGS__)))

其实也就是equalTo的封装版本 把数值对象装箱了 这样才能给eqaulTo用

5月14日 回复 顶 转发

碧野MAX

还是不太明白max_equalTo和equalTo的区别,我现在都是用的mas_equalTo

5月14日 回复 顶 转发

里脊串

回复 袁国文: 我看那个黑色的高度好像是64 是不是automaticallyAdjustsScrollViewInsets的原因?

5月14日 回复 顶 转发

袁国文

又是问题多多的我来请教博主了，我在弄scrollView横向图片循环滚动时候遇到了问题，麻烦帮忙看看呗，有图有代码，http://segmentfault.com/q/1010000002764232

5月14日 回复 顶 转发

袁国文

回复 里脊串: 搞定，再次谢谢博主帮我打开新世界，不用再去故事版

5月13日 回复 顶 转发

里脊串

回复 袁国文: 横向滚动跟竖向滚动原理是一样的吧...

5月13日 回复 顶 转发

袁国文

回复 里脊串: 已经好了，我忘了在Podfile加上'7.1'，在研究怎么让scrollView横向滚动，博主有没什么demo可以参考的?

5月13日 回复 顶 转发

里脊串

回复 李婧: 这个我就不太清楚了 你可以按官方的安装指导一步一步的试一下

5月13日 回复 顶 转发

李婧

回复 里脊串: 项目里面之前有UIView 的其他分类 可以正常调用 这个就不行 是什么问题呢 

5月13日 回复 顶 转发

李婧

回复 里脊串: 我还发现调用 UIView的Category里所有方法都会这样

5月13日 回复 顶 转发

李婧

回复 里脊串: 编译是没问题的，但是一运行到那个方法就报错

5月13日 回复 顶 转发

里脊串

回复 李婧: 应该是没有打开xcworkspace吧?

5月13日 回复 顶 转发

李婧

您好,请问为什么我这边运行报错-[*** mas_makeConstraints:]: unrecognized selector sent to instance呢?

5月13日 回复 顶 转发

里脊串

回复 小黑: 不建议在layoutSubviews里布局

报错是因为layoutSubviews会被多次调用 使用make会重复添加多次导致冲突 使用remake就可以覆盖之前的布局

5月13日 回复 顶 转发

小黑

uitableviecell 里的layoutSubviews方法里布局子控件会报错

5月13日 回复 顶 转发

里脊串

回复 袁国文: 是不是你的项目的 deployment target 设置有问题? ios4也支持? 或者没开启ARC? sizeClass跟这个不冲突 masonry只是系统autolayout语法的封装而已

5月13日 回复 顶 转发

袁国文

博主博主，看了你的博文我跑去装了个cocoaPods，然后导入了Masnory框架，还没开始使用就发现报错，The current deployment target does not support automated __weak references，能跟我说下这个是什么原因导致的嘛？还有我想问下，sizeClasses和这个框架的约束有没相互作用?

5月12日 回复 顶 转发

围观群众头目

楼主写的很棒!

5月12日 回复 顶 转发

里脊串

回复 helloWorld: 推荐你用cocoapod来安装第三方库

5月11日 回复 顶 转发

## 分類

开源项目 (8)
技巧心得 (17)
生活琐碎 (2)
经验介绍 (8)

## 標簽

AlertView (1)
Autolayout (2)
BeyondCompare (1)
Git (1)
MKMapView (1)
Masonry (2)
NimbusKit (1)
Popup (1)
QRCode (1)
SheetView (1)
SourceTree (1)
Swift (1)
Today Extension (1)
UIScrollView (1)
UITableView (3)
UIViewController (1)
URI (1)
Widget (1)
Xcode (1)
ZXing (1)
appetize.io (1)

## 標簽雲

Extension UIScrollView UITableView UIViewController URI Widget Xcode ZXing appetize.io demo iCarousel iMessage iOS9 pop 任务管理器 优化 动画 区号 后台 图片 地图 多语言 定位 实践 家庭 导航 工具 开源 性能 手势 技巧 抗锯齿 插件 支付 支付宝 教程 旋转 旋转屏幕 最美创意 生活 电话 第三方库 脚本 获奖 表单 设计 资源 键盘

## 歸檔

九月 2015 (3)
八月 2015 (4)
七月 2015 (5)
六月 2015 (1)
五月 2015 (1)
四月 2015 (1)
三月 2015 (4)
十一月 2014 (1)
十月 2014 (1)
九月 2014 (2)
八月 2014 (4)
一月 2014 (1)
十二月 2013 (2)
十一月 2013 (1)
十月 2013 (1)
九月 2013 (1)
八月 2013 (1)
七月 2013 (1)

## 近期文章

再见ZXing 使用系统原生代码处理QRCode
使用appetize.io为你的demo创建在线预览
开源项目:MMPopupView
图片变形的抗锯齿处理方法
处理i18n国际电话区号的代码实践

## 友情鏈接

Github
StackOverflow
Weibo

Dropbox