autoLayout 自动布局

1. 简介：

   1) autolayout 是一种布局技术，专门用来布局 UI 界面的，用来取代 frame 布局在遇到多屏幕尺寸多种多样的不足。从 ios6 开始引入，由于 xcode4 不给力，当时没有得到很大推广。在 ios7(xcode5)开始，autolayout 的开发效率得到了很大的提升。苹果官方也推荐使用 autolayout 来解决屏幕适配的问题。

   2) autolayout 的实现很多种：苹果 api，之后的 vfl，storyboard，第三方 masonry 等，下面就分别举例介绍各种使用方法。

2. 苹果 API NSLayoutConstraint 注意参照关系，有时 constant 为负有时为正

   例：设置 redView 距离父视图左边距 20，底边距 20，宽 50，高 50

```
UIView *redView = [[UIView alloc] init];

[redView setBackgroundColor:[UIColor redColor]];

[redView setTranslatesAutoresizingMaskIntoConstraints:NO];//不以 autoresizingMask 进行计算(ios6 以前只有 3.5 寸屏幕)

//autoresizing 是一个相对父控件的布局解决方法，只有上下左右宽高可以调用

//由于 autolayout 和 autoresizing 是相对冲突的，所以此处禁用 autoresizing

[self.view addSubView:redView];

NSLayoutConstraint *redLeft = [NSLayoutConstraint constraintWithItem:redView attribute:NSLayoutAttributeLeft relateBy:NSLayoutRelationEqual toItem:self.view attribute NSLayoutAttributeLeft multiplier:1.0f constant:20.0];

NSLayoutConstraint *redBottom = [NSLayoutConstraint constraintWithItem:redView attribute:NSLayoutAttributeBottom relateBy:NSLayoutRelationEqual toItem:self.view attribute NSLayoutAttributeBottom multiplier:1.0f constant:20.0];

NSLayoutConstraint *redWidth = [NSLayoutConstraint constraintWithItem:redView attribute:NSLayoutAttributeWidth relateBy:NSLayoutRelationEqual toItem:nil attribute NSLayoutAttributeNotAnAttribute multiplier:1.0f constant:50.0f];
```

```
NSLayoutConstraint *redHeight = [NSLayoutConstraint
constraintWithItem:redView attribute:NSLayoutAttributeHeight
relateBy:NSLayoutRelationEqual toItem:nil attribute
NSLayoutAttributeNotAnAttribute multiplier:1.0f constant:50.0f];

[self.view addConstraint:redLeft];//有参照关系，约束加到父视图

[self.view addConstraint:redBottom];

[redView addConstraint:redWidth];//无参照关系，约束加到自身

[redView addConstraint:redHeight];
```

3. VFL 语法

(1) 简述:vfl 的出现是为了减轻原 api 带来的烦琐。Vfl 思想与其它实现方法不同，它更宏观化，它将约束分成了两块：水平方向 H:和竖直方向 V:，也就是说，建立约束时，得把水平与垂直方向的约束用字符串一并表达出。

(2) 例子：在 2 例子基础上水平放置 3 个 view，互相间隔 20

```
//redView,blueView,yellowView

//view 添加到 self.view

//开始建立约束

NSString *Hvfl =
@”H:|-margin-[redView(50)-margin-[blueView(==redView)]-margin-[yello
wView(==redView)]]”;

//设置替换的数字(用字典表示)

NSDictionary *metrics = @{@”margin”:@20};

//把药添加约束的 view 转成字典

NSDictionary *views =
NSDictionaryOfVariableBindings(redView,blueView,yellowView);

//设置对齐方式，顶部和底部均与 redView 对齐

NSLayoutFormatOptions ops =
NSLayoutFormateAllTop|NSLayoutFormatAllBottom;

//创建水平约束
```

```
NSArray *Hconstraints = [NSLayoutConstraint
constraintsWithVisualFormat:Hvfl options:ops metrics:metrics
views:views];
//创建竖直约束
NSString *Vvfl = @”V:[redView(50)]-margin-|”;
NSArray *Vconstraints = [NSLayoutConstraint
constraintsWithVisualFormat:Vvfl options:ops metrics:metrics
views:views];
//父视图添加约束
[self.view addConstraint:Hconstraints];
[self.view addConstraint:Vconstraints];
```

(3)总结：

1) VFL 实现的 autolayout 是有局限性的，如果想设置约束 优先级或是传入倍数关系必须调用苹果原 API [NSLayoutConstraint constraintWithItem:...]

2) VFL 仅供了解，最好用的莫过于第三方框架 masonry 了

4. Masonry 框架

1) 简介：这是一款公认的非常简洁优美的 autolayout 框架，这框架约束添加思维和原 API 添加约束是相同的，只是 Masonry 语法更简洁，代码更美观。

注：autolayout 目的是为了确定控件位置及尺寸，除此外不要添加多余约束，否则容易出现约束冲突，除非你想使用约束优先级

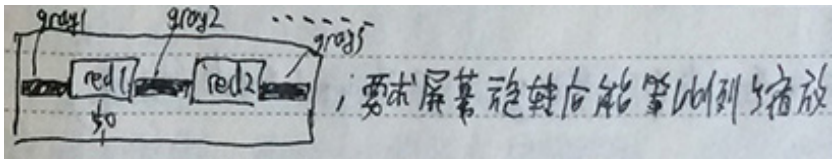
2) 例子讲解：

1) 将屏幕平分四块，redView blueView yellowView greenView

```
//添加四个 view 至父视图
//添加红色 view 约束
[redView mas_makeConstraints:^(MASConstraintMaker *make){
 make.left.equalTo(self.view.mas_left).offset(0);//使左边等于
self.view 的左边，间距为 0
 make.left.equalTo(self.view.mas_top).offset(0);
```

```
 make.width.equalTo(self.view.mas_width).multipliedBy(0.5);//宽是
self.view 的 0.5 倍
 make.height.equalTo(self.view.mas_height).multipliedBy(0.5);
}];//至此红色 view 约束添加完成，其它 view 以其确定自己位置和尺寸如：
[blueView mas_makConstraints:^(MASConstraintMaker *make){
 make.width.and.height.equalTo(redView);//使宽高等与 redView 的宽高
 make.top.equalTo(redView.mas_top);//与 redView 顶部对齐
 make.leading.equalTo(redView.mas_right);//与 redView 间距为 0
}];
```

2) 要求屏幕旋转后能等比例缩放，效果如下：




```
[gray1 mas_makeConstraints:^(MASConstraintMaker *maker){
 make.height.mas_equalTo(20);
 make.leading.equalTo(self.view.mas_leading).offset(0);
 make.bottom.equalTo(red1.mas_bottom);
}];
[red1 mas_makeConstraints:^(MASConstraintMaker *maker){
 make.width.mas_equalTo(100);
 make.height.mas_equalTo(50);
 make.left.equalTo(gray1.mas_right);
 make.bottom.equalTo(self.view.mas_bottom).offset(-50);
}];
[gray2 mas_makeConstraints:^(MASConstraintMaker *make){
 make.height.and.width.equalTo(gray1);
 make.leading.equalTo(red1.mas_right);
 make.bottom.equalTo(red1.mas_bottom);
}];
```

```
[red2 mas_makeConstraints:^(MASConstraintMaker *make){
  make.height.and.width.equalTo(red1);
  make.leading.equalTo(gray2.mas_right);
  make.bottom.equalTo(red1.mas_bottom);
}];
  [gray3 mas_makeConstraints:^(MASConstraintMaker *make){
     make.height.and.width.equalTo(gray1);
     make.leading.equalTo(red2.mas_right);
     make.trailing.equalTo(self.view.mas_right);
     make.bottom.equalTo(red1.mas_bottom);
  }]
```

5. storyboard 实现 autolayout

(1) 右键拖拽控件建立约束

(2) 右下角添加对齐和约束(可选参照是 view 还是 layoutGuide)

(3) 例子：略