Masonry源码解析

袁亚环

目录

- 常见布局方式介绍
- VFL布局简介
- Masonry与NSLayoutConstraint布局比较
- Masonry框架思想
- Masonry源码解析

常见的布局方式

- 1.自动布局AutoLayout
- 2.VFL（Visual Format Language）可视化格式语言
- 3.frame
- 4.Masonry
- 5.AutoresizingMasks

VFL（Visual Format Language）

- 语法：H: |-[button]-| <看代码>
- VFL是苹果推出的一门抽象化语言，用来自动布局的
- H:水平方向，V:垂直方向。
- | 表示父控件。-表示距离。

NSLayoutConstraint和Masonry调用方式对比

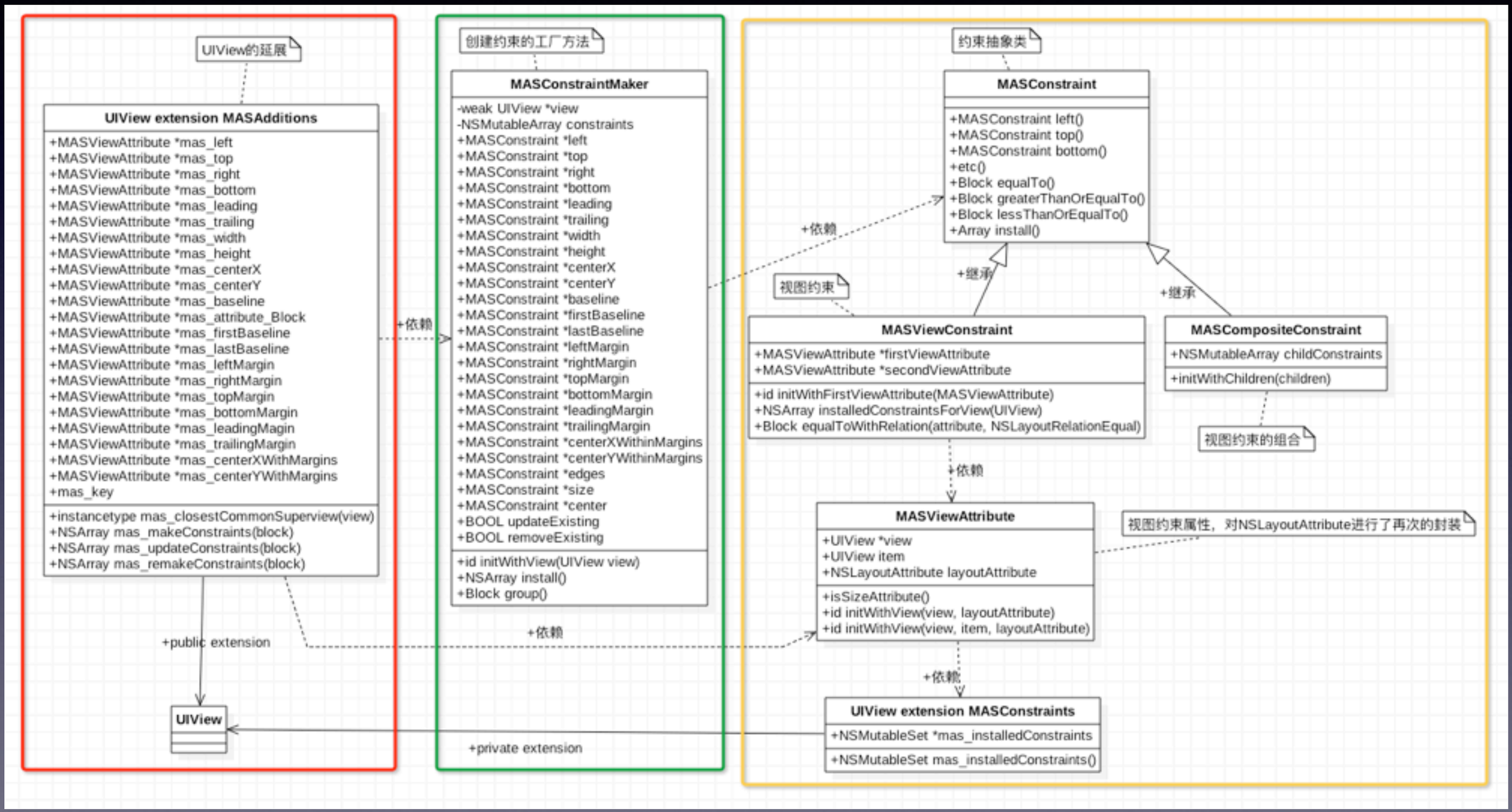
```objc
UIView *subView = [[UIView alloc]init];
subView.translatesAutoresizingMaskIntoConstraints = NO;
NSLayoutConstraint  *constraint = [NSLayoutConstraint
                                   constraintWithItem:subView
                                   attribute:NSLayoutAttributeTop
                                   relatedBy:NSLayoutRelationEqual
                                   toItem:subView
                                   attribute:NSLayoutAttributeTop
                                   multiplier:1.0 constant:10
];
[subView addConstraint:constraint];
```

```objc
[subView mas_makeConstraints:^(MASConstraintMaker *make) {
    make.top.equalTo(@10);
    make.top.equalTo(subView.mas_top).offset(10);
    make.top.equalTo(subView.mas_top).multipliedBy(1).offset(10);
}];
```

Masonry介绍

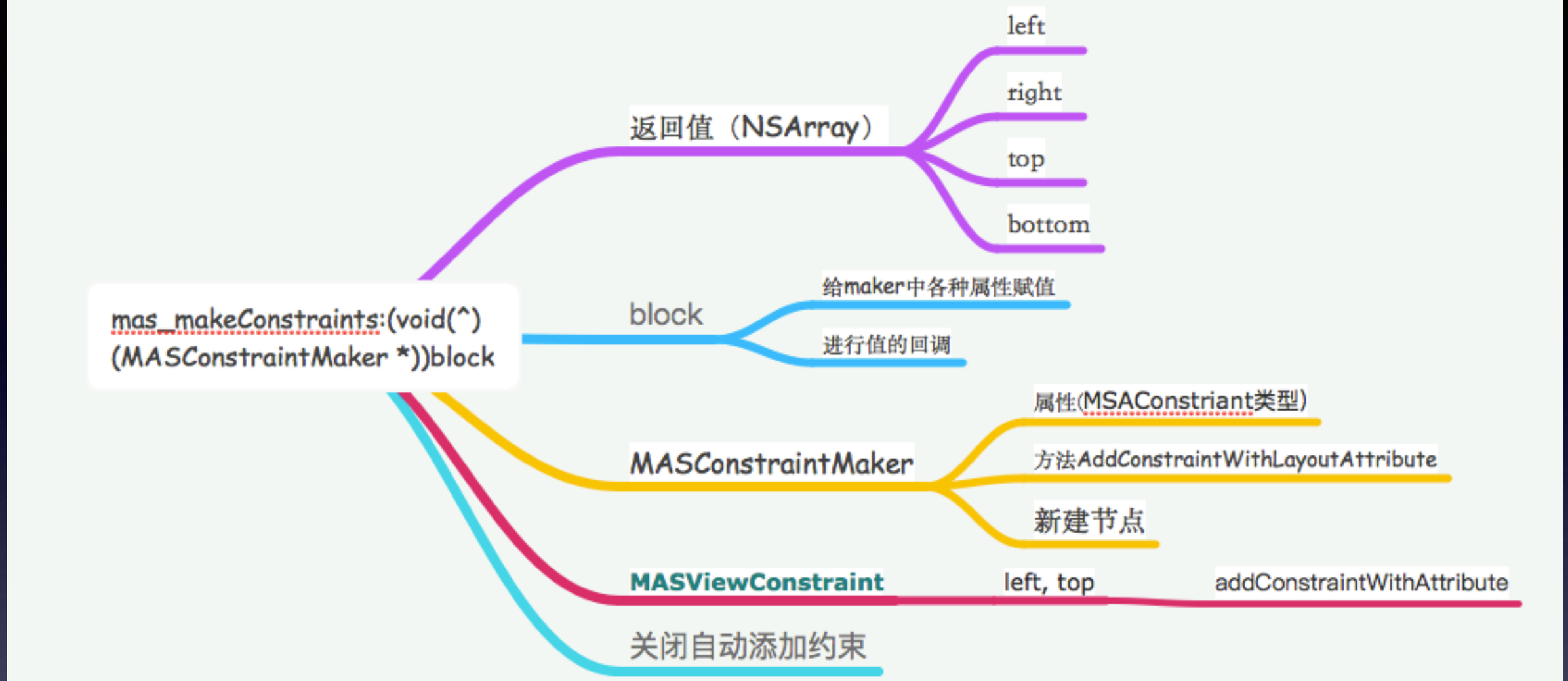
- Masonry是iOS 在控件布局中经常使用的一个轻量级框架，Masonry让NSLayoutConstraint使用起来更为简洁。

Masonry框架类结构图



Masonry框架思想

- 首先调用View+MASAdditions的对象方法，参数是Block，该Block根据实际调用地方的参数进行方法链式调用，View+MASAdditions里面先创建一个MASConstraintMaker对象，然后调用Block，把对象当做参数进行Block回调，在外部进行一套链式调用，链式的思想是，对象的方法调用没有参数返回值必须是Block，然后调用Block，Block的返回值就是对象，循环进行再次调用


mas_makeConstraints:(void(^)(MASConstraintMaker *))block
返回值（NSArray）
left
right
top
bottom
block
给maker中各种属性赋值
进行值的回调
MASConstraintMaker
属性(MSAConstriant类型)
方法AddConstraintWithLayoutAttribute
新建节点
MASViewConstraint
left, top
addConstraintWithAttribute
关闭自动添加约束

方法执行过程

View+MASAdditions

- View+MASAdditions主要成员属性
- mas_makeContrains方法解析
- mas_updateContrains
- mas_remakeContrains

View+MASAdditions

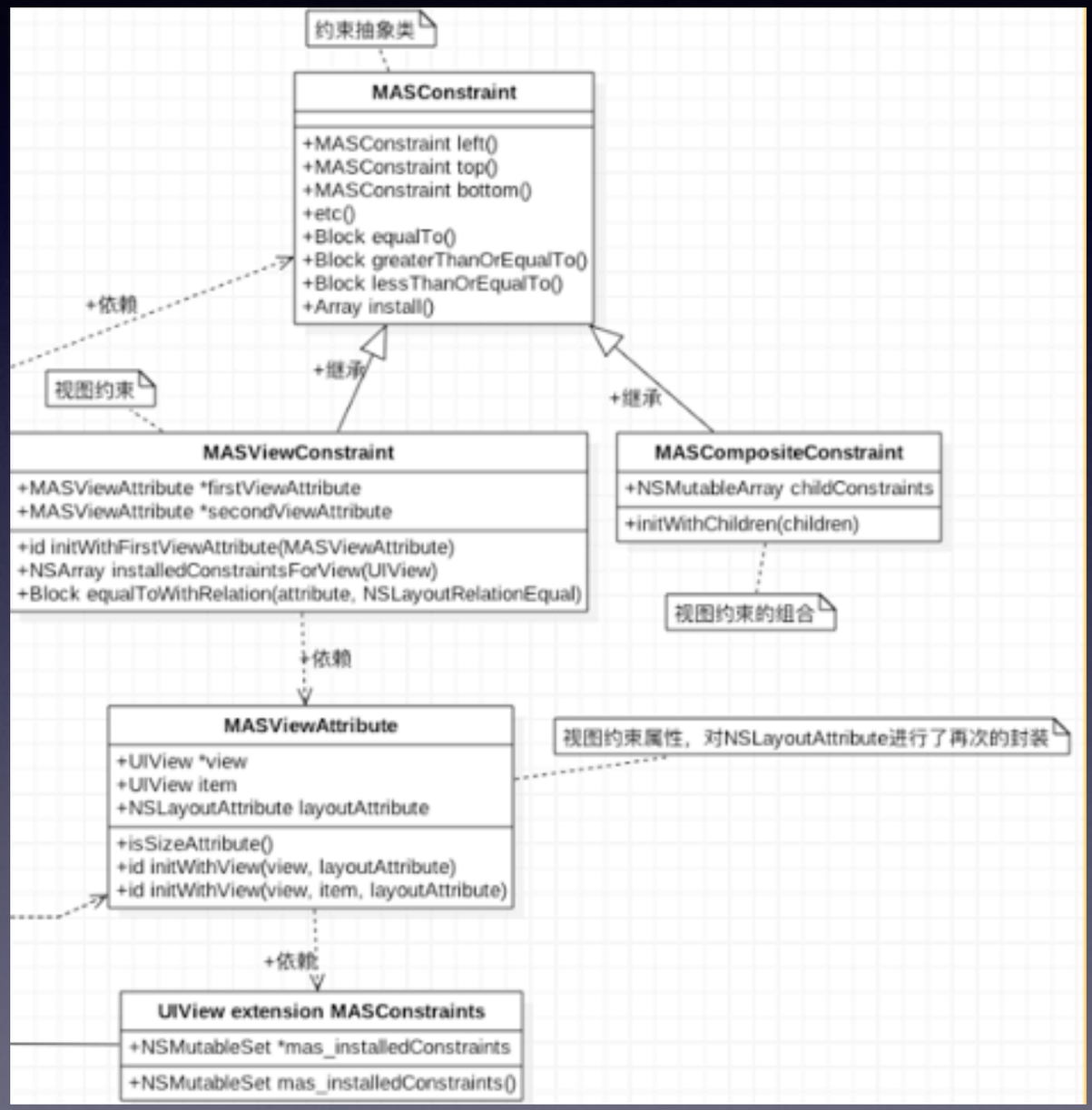
```objc
//新建添加约束
- (NSArray *)mas_makeConstraints:(void(^)(MASConstraintMaker *))block {

    //关闭自动添加约束，我们手动添加
    self.translatesAutoresizingMaskIntoConstraints = NO;
    //创建constraintMaker
    MASConstraintMaker *constraintMaker = [[MASConstraintMaker alloc]
        initWithView:self];
    //给maker中各种成员属性赋值，通过block进行值的回调，此处的block就是获取用户数据的钩子
    block(constraintMaker);
    //进行约束添加，并返回install的约束数组（Array<MASContraint>）
    return [constraintMaker install];
}
```

```objc
27 - (NSArray *)mas_updateConstraints:(void(^)(MASConstraintMaker *))block {
28     self.translatesAutoresizingMaskIntoConstraints = NO;
29     MASConstraintMaker *constraintMaker = [[MASConstraintMaker alloc]
           initWithView:self];
30     //打开更新的开关
31     constraintMaker.updateExisting = YES;
32     block(constraintMaker);
33     return [constraintMaker install];
34 }
35
36 - (NSArray *)mas_remakeConstraints:(void(^)(MASConstraintMaker *make))block {
37     self.translatesAutoresizingMaskIntoConstraints = NO;
38     MASConstraintMaker *constraintMaker = [[MASConstraintMaker alloc]
           initWithView:self];
39     //打开移除约束的开关
40     constraintMaker.removeExisting = YES;
41     block(constraintMaker);
```

MASViewAttribute

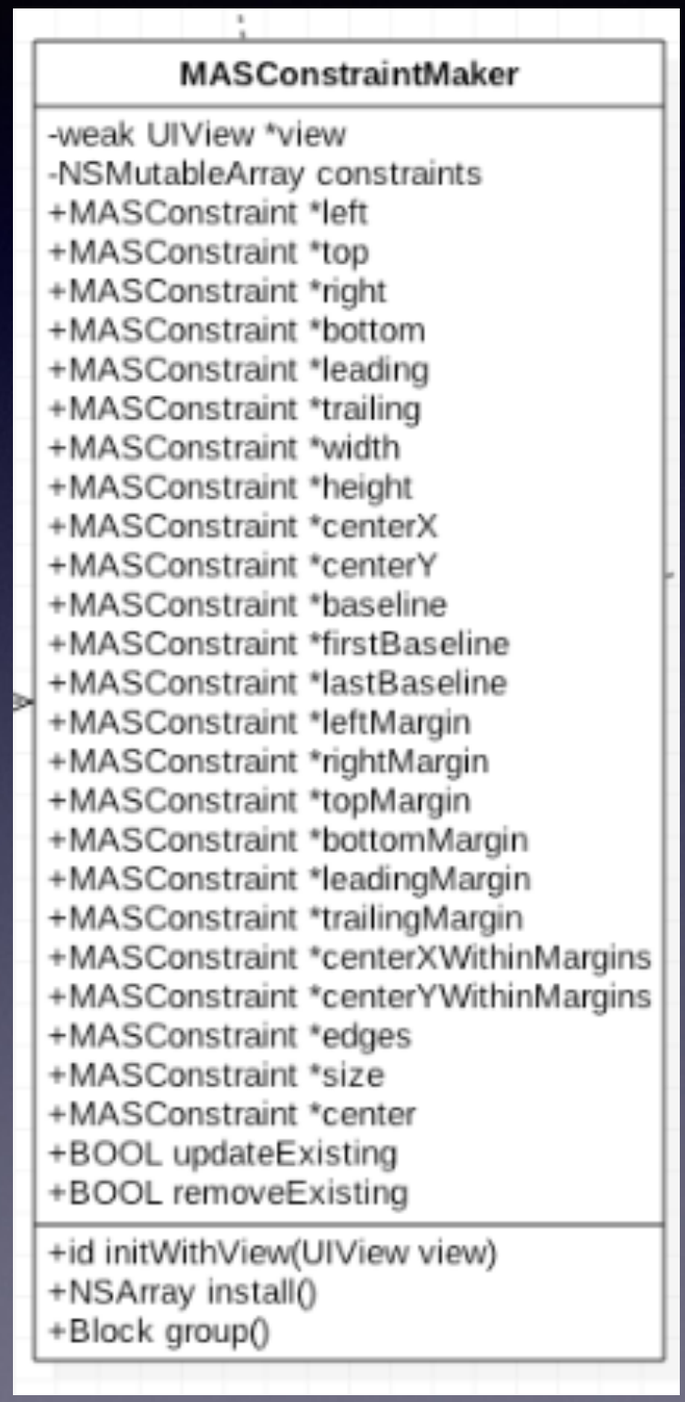
- MASViewAttribute主要包括三个属性，三个方法。
- 是对UIView的NSLayoutAttribute的封装。
- MASViewConstraint类。



MASConstraintMaker

- MASConstraintMaker是一个工厂类，负责创建MASConstraint类型的对象。在UIView的View+MASAdditions类目中就是调用了MASConstraintMaker类中的方法。

- mas_makeConstraints方法中的block的参数就是MASConstraintMaker的对象，用户通过该block回调回来的MASConstraintMaker对象给View指定要参加的约束以及该约束的值。



MASConstraintMaker

核心公有属性

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

每个MASConstraint类型的属性都对应一个getter方法，在getter中都会调用addConstraintWithLayoutAttribute方法，这个方法就是MASConstraintMaker工厂类的工厂方法

MASConstraintMaker工厂方法解析

- 在MASConstraintMaker中每个MASConstraint的属性都有一个对应的getter方法，在getter方法中会调用addConstraintWithLayoutAttribute方法。

- 在addConstraintWithLayoutAttribute方法中，根据提供的参数创建MSAViewConstraint对象，如果该函数的第一个参数不为空的话就会将新创建的MSAViewConstraint对象与参数进行合并组合成MASCompositeConstraint类（MASCompositeConstraint本质上是MSAViewConstraint对象的数组）的对象。

MASViewConstraint

- MASConstraintMaker工厂类所创建的对象实质上是MASViewConstraint类的对象。而MASViewConstraint类实质上是对MASLayoutConstraint的封装。

- 链式调用

链式调用的实现

实现链式编程的关键就是声明一个block的属性，而这个block返回值必须还是一个对象（根据业务需求不同，可以返回的是这个对象实例本身，也可以是这个类的另一个实例，更可以是另一个类的实例对象）。举个🌰说明。。。代码demo。