in 朋友页性能调优的探索

白菜

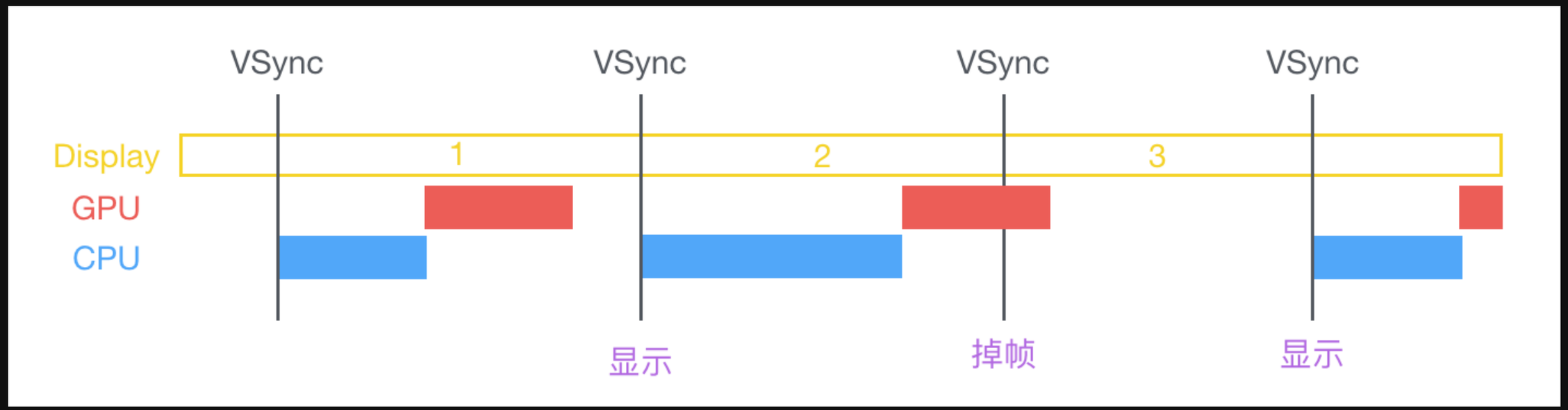
卡顿的原因？

60 FPS

V-Sync(垂直同步)

屏幕渲染过程中为了解决可能出现的屏幕撕裂问题,引入了一个叫垂直同步的机制. 那么产生的结果就是GPU会等待显示器的 V-Sync 信号发出后，才进行新的一帧渲染和缓冲区更新.

一个V-Sync周期内需要做的事情:
CPU和GPU会完成图片的处理和绘制. 然后提交到帧缓冲区,如果CPU或GPU的处理时间过长,会导致本次渲染的结果需要等到下一个周期才能显示. 这就是卡顿的原因.



*图片来自网络

CPU

- 布局计算
- autolayout 约束计算随着数量呈指数上涨,同时必须在主线程计算
- 文本计算渲染
- 图片解码
- 图形的绘制

GPU

- 离屏渲染
- 多视图的混合
- 纹理渲染

in 订阅

【inner优选】

看着这些隔着屏幕都能闻到香味的甜品，好想

约上做甜点的她一起吃个下午茶啊！👩

戳右侧昵称进入甜点大师的主页吧👉@董箫乐

in 订阅

看着这些隔着屏幕都能闻到香味的甜品，好想

约上做甜点的她一起吃个下午茶啊！👩

戳右侧昵称进入甜点大师的主页吧👉@董箫乐

#inner优选

754赞

听说逗逗很爱我 好想吃

展开所有12条评论

张起灵的辣条诶！ 报社😂😂😂

东戴河渔家院+代售承德山

4天前

渲染

- UILabel改为由CoreText后台渲染
- autolayout改为手动计算调整,并缓存计算结果
- 对于不需要触摸事件的控件,用CALayer来代替UIView和UIImageView等.
- 尽量减少Cell里面视图属性的变更

- 操作取消机制
- TableView等预先计算要显示的cell,优先渲染内容
- 线程控制
- 避免离屏渲染

AsyncDisplayKit

- 渲染,布局计算,UI操作转移到子线程
- 不需要触摸事件的node,可以设置为layerBacked
- 如果存在栅格化操作, 会将它的子视图与当前视图合并渲染成一个图层

将耗时代码搬离主线程

ASDK的渲染过程

Core Animation

- Runloop 注册Observer (BeforeWaiting,Exit)
- 代码修改的视图属性等操作,通过CATransaction保存到中间状态
- 通过Observer回调(休眠,退出)提交中间状态到GPU进行渲染

Network Zero KB/s

Thread 1 Queue: com.apple.main-thread (serial)

0 ASAsyncTransactionQueue::GroupImpl::schedule(long, NSObject<OS_dispa...

1 -[_ASAsyncTransaction addOperationWithBlock:priority:queue:completion:]

2 -[ASDisplayNode(AsyncDisplay) displayAsyncLayer:asynchronously:]

3 -[_ASDisplayLayer display:]

4 -[_ASDisplayLayer display]

5 recursivelyTriggerDisplayForLayer(CALayer*, bool)

6 recursivelyTriggerDisplayForLayer(CALayer*, bool)

7 -[ASDisplayNode _recursivelyTriggerDisplayAndBlock:]

8 __49+[ASDisplayNode scheduleNodeForRecursiveDisplay:]_block_invoke_2

9 -[ASRunLoopQueue processQueue]

10 __45-[ASRunLoopQueue initWithRunLoop:andHandler:]_block_invoke

11 __CFRUNLOOP_IS_CALLING_OUT_TO_AN_OBSERVER_CALLBACK_FUNCTIO...

12 __CFRunLoopDoObservers

13 __CFRunLoopRun

14 CFRunLoopRunSpecific

15 GSEventRunModal

Network Zero KB/s

Thread 1 Queue: com.apple.main-thread (serial)

0 __64-[ASDisplayNode(AsyncDisplay) displayAsyncLayer:asynchronously:]_b...
1 -[ASDisplayNodeAsyncTransactionOperation callAndReleaseCompletionBloc...
2 -[_ASAsyncTransaction completeTransaction]
3 __29-[_ASAsyncTransaction commit]_block_invoke
4 _dispatch_call_block_and_release
5 _dispatch_client_callout
6 _dispatch_main_queue_callback_4CF
7 __CFRUNLOOP_IS_SERVICING_THE_MAIN_DISPATCH_QUEUE__
8 __CFRunLoopRun
9 CFRunLoopRunSpecific
10 GSEventRunModal
11 UIApplicationMain
12 main
13 start
14 start

Enqueued from com.apple.main-thread (Thread 1)

0 _dispatch_barrier_async_f_slow
1 ASAsyncTransactionQueue::GroupImpl::notify(NSObject<OS_dispatch_queu...
2 -[_ASAsyncTransaction commit]
3 -[_ASAsyncTransactionGroup commit]
4 _transactionGroupRunLoopObserverCallback
5 __CFRUNLOOP_IS_CALLING_OUT_TO_AN_OBSERVER_CALLBACK_FUNCTION...

```
351     ASDisplayNodeCAssert
352     if (!canceled && !is
353       UIImage *image = (
354       BOOL stretchable =
355       if (stretchable) {
356         ASDisplayNodeSet
357       } else {
358         _layer.contentsS
359         _layer.contents
360       }
361       [self didDisplayAs
362     }
363   };
364
365   // Call willDisplay im
366   [self willDisplayAsync
367
368   if (asynchronously) {
369     // Async rendering o
370     // while synchronizi
371
372     // First, look to se
373     CALayer *containerLa
374
375     // In the case that
376     // this call will al
377     // It will automatic
378     _ASAsyncTransaction
379
```

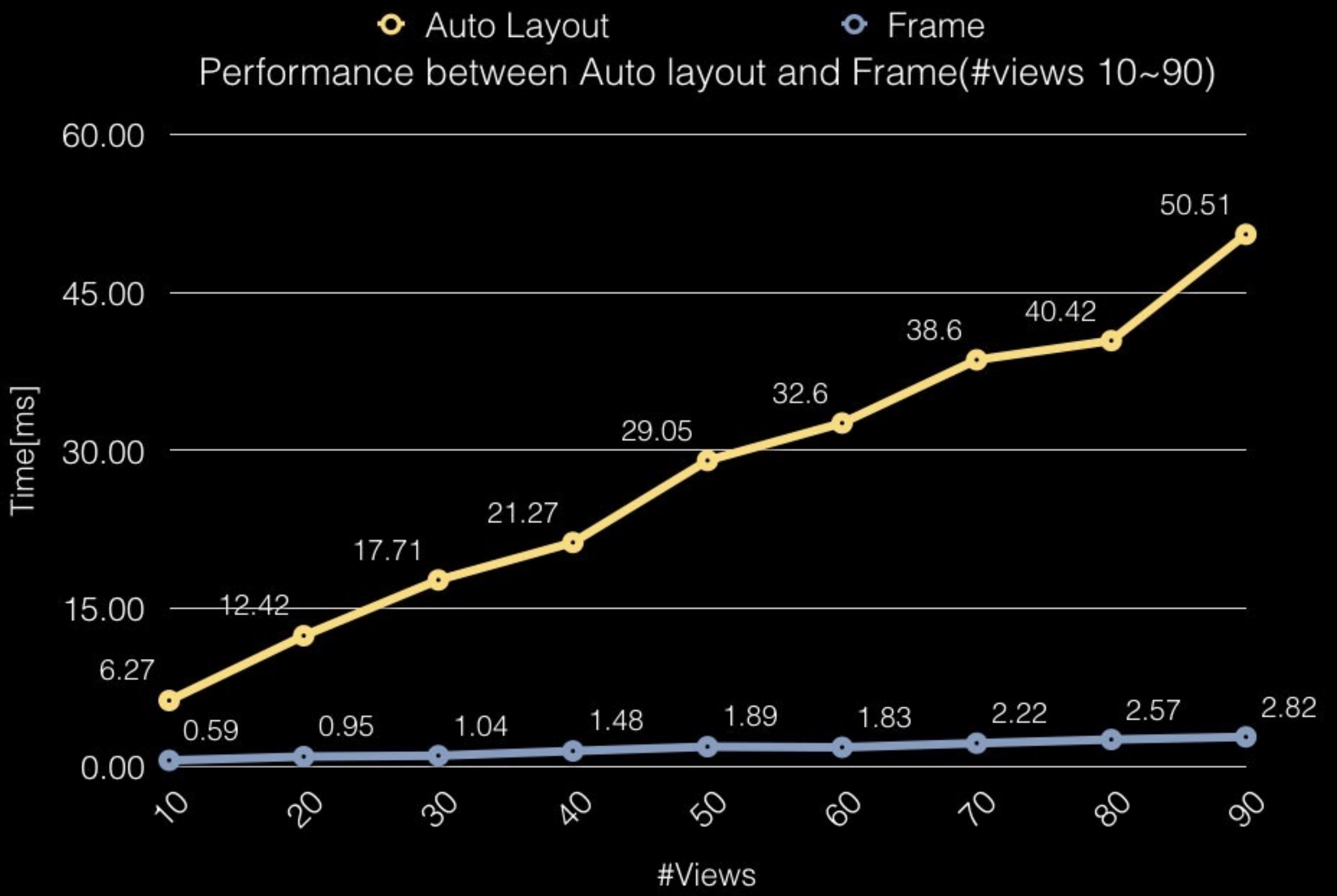
self = (ASTextNode *) 0x7f9e62
value = (UIImage *) 0x7f9e6144
image = (UIImage *) 0x7f9e614
stretchable = (BOOL) NO
_layer = (_ASDisplayLayer *) 0x7

ASAsyncTransaction

- 把渲染任务拆封成很多个小的事务,异步并发渲染
- 结合Runloop的机制,把渲染好的事务提交到GPU

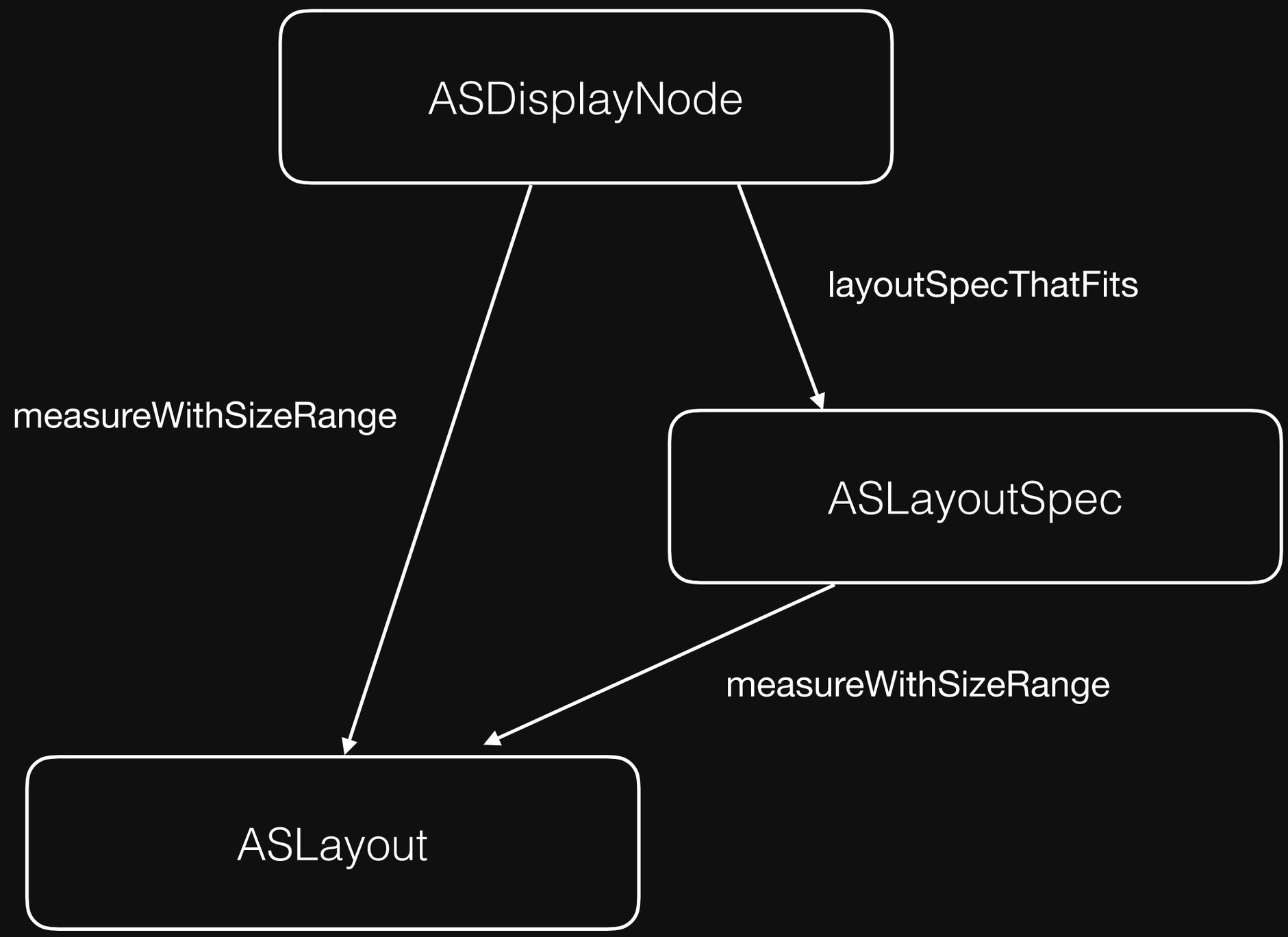
Autolayout

- 原生autolayout 易用性差
- 问题调试头疼
- 性能影响随着图层数量呈指数上涨


Auto Layout
Frame
Performance between Auto layout and Frame(#views 10~90)
60.00
45.00
30.00
15.00
0.00
Time[ms]
6.27
12.42
17.71
21.27
29.05
32.6
38.6
40.42
50.51
0.59
0.95
1.04
1.48
1.89
1.83
2.22
2.57
2.82
10
20
30
40
50
60
70
80
90
#Views

ASLayout

ASLayout is an automatic, asynchronous, purely Objective-C box model layout feature.
It is a simplified version of CSS flex box, loosely inspired by ComponentKit's Layout.
It is designed to make your layouts extensible and reusable.


ASDisplayNode
layoutSpecThatFits
measureWithSizeRange
ASLayoutSpec
measureWithSizeRange
ASLayout

Stack

Inset

Ratio

Static

Center

Overlay

Background


```
- (ASLayoutSpec *)layoutSpecThatFits:(ASSizeRange)constraint
{
  ASStackLayoutSpec *vStack = [[ASStackLayoutSpec alloc] init];

  [vStack setChildren:@[titleNode, bodyNode];

  ASStackLayoutSpec *hstack = [[ASStackLayoutSpec alloc] init];
  hStack.direction          = ASStackLayoutDirectionHorizontal;
  hStack.spacing            = 5.0;

  [hStack setChildren:@[imageNode, vStack]];

  ASInsetLayoutSpec *insetSpec = [ASInsetLayoutSpec
insetLayoutSpecWithInsets:UIEdgeInsetsMake(5,5,5,5) child:hStack];

  return insetSpec;
}
```

```
- (ASLayoutSpec *)layoutSpecThatFits:(ASSizeRange)constraint
{
  ASStackLayoutSpec *vStack = [[ASStackLayoutSpec alloc] init];

  [vStack setChildren:@[titleNode, bodyNode];

  ASStackLayoutSpec *hstack = [[ASStackLayoutSpec alloc] init];
  hStack.direction          = ASStackLayoutDirectionHorizontal;
  hStack.spacing            = 5.0;

  [hStack setChildren:@[imageNode, vStack]];

  ASInsetLayoutSpec *insetSpec = [ASInsetLayoutSpec
insetLayoutSpecWithInsets:UIEdgeInsetsMake(5,5,5,5) child:hStack];

  return insetSpec;
}
```

性能优化往往会导致代码的可维护性下降，
我们需要对问题的原因和优化时机有一个比较清楚的认识
才能避免过度优化带来的问题。

谢谢