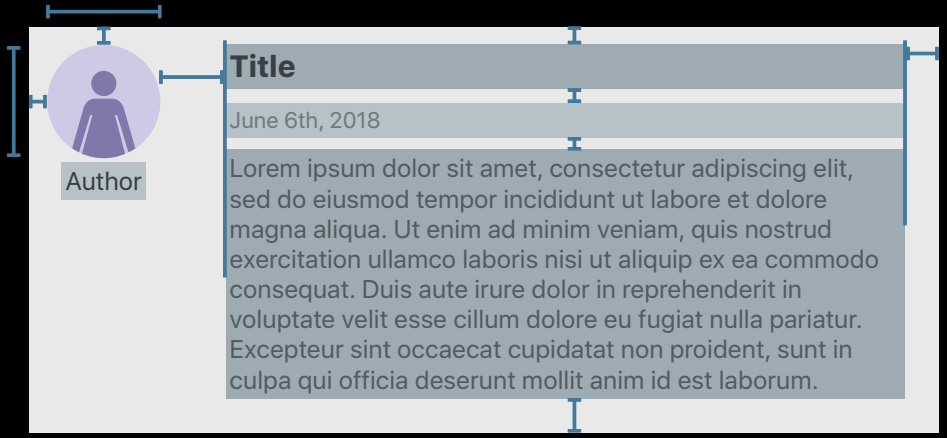
#WWDC18

High Performance Auto Layout

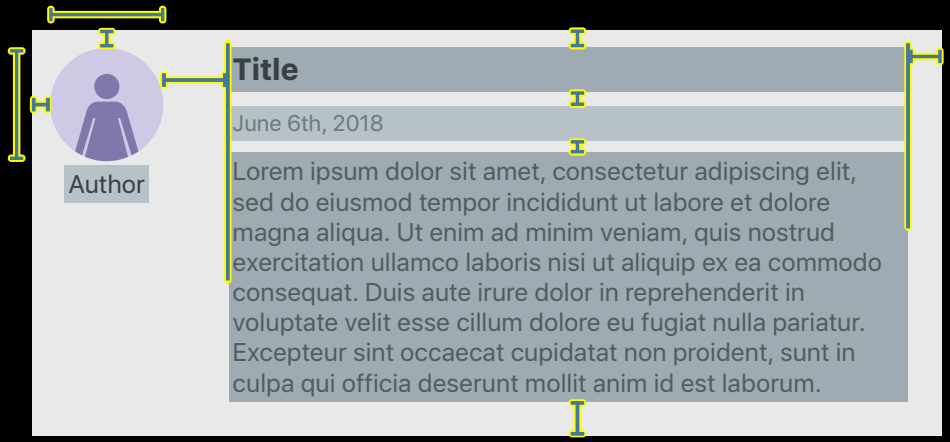
성능 고려하며 AutoLayout 작업하는법

Ken Ferry, iOS System Experience
Kasia Wawer, iOS Keyboards

Auto Layout


Author
Title
June 6th, 2018
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Auto Layout


Author
Title
June 6th, 2018
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

iOS 12 improvements

Internals and intuition

Building efficient layouts

iOS 12 improvements

Internals and intuition

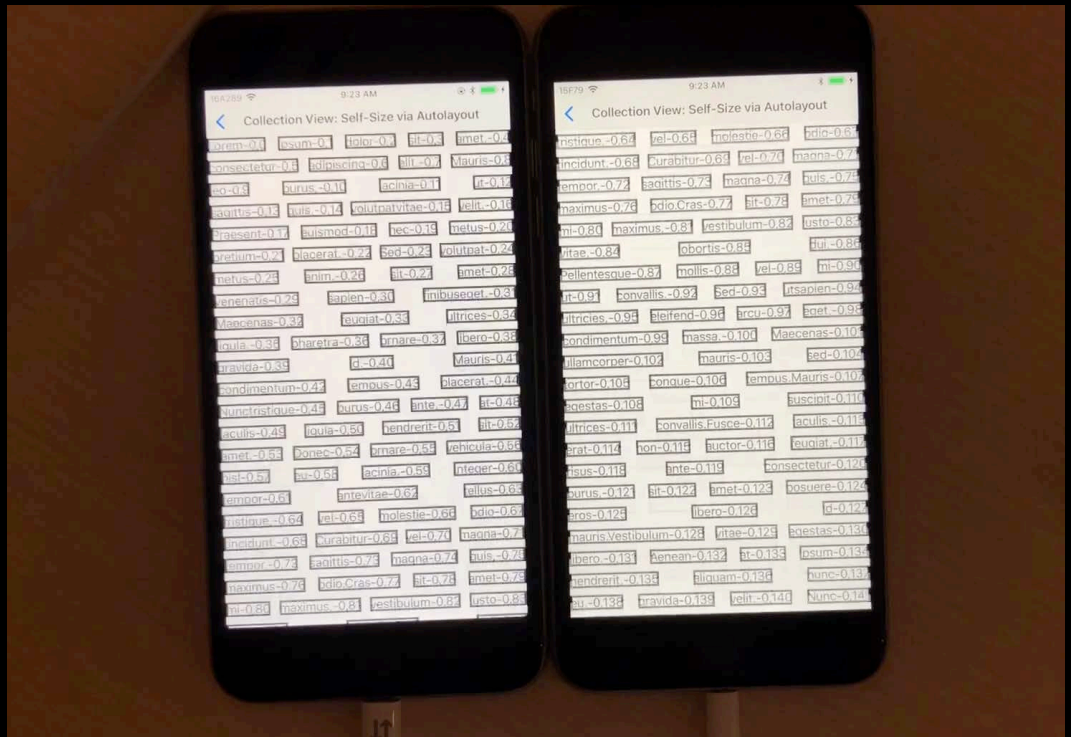
Building efficient layouts

Performance Improvements in iOS 12

You get it for free

벤치마크

UICollectionView
+
self sizing cell

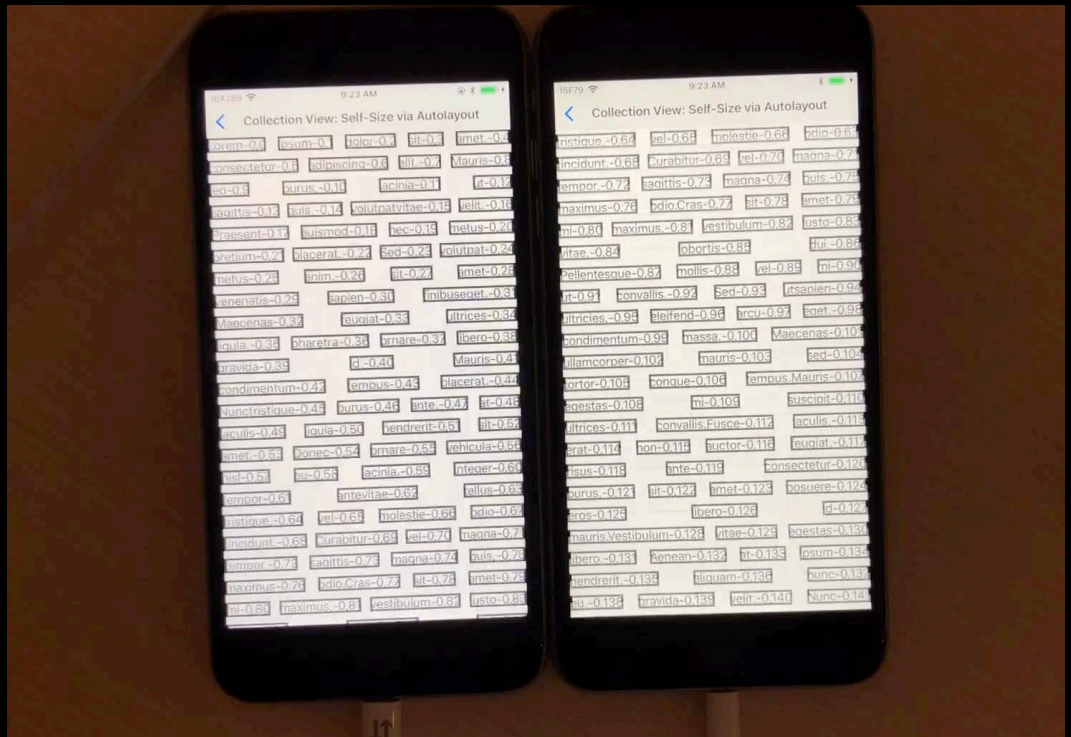


Performance Improvements in iOS 12

You get it for free

iOS 11 vs iOS12

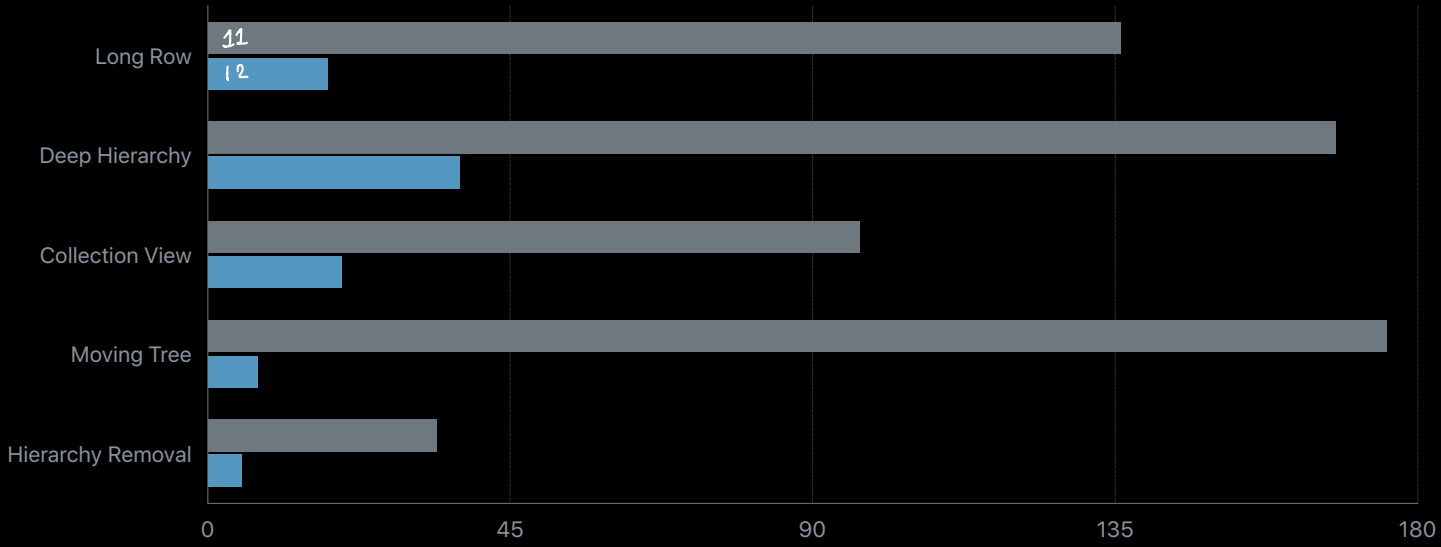
iOS12 에서는
frame 지연없이
부드럽다.



Performance Improvements in iOS 12

Shorter bars are better

CollectionView는 내부적으로 AutoLayout을 사용한다.



iOS 12 improvements

Internals and intuition

Building efficient layouts

iOS 12 improvements

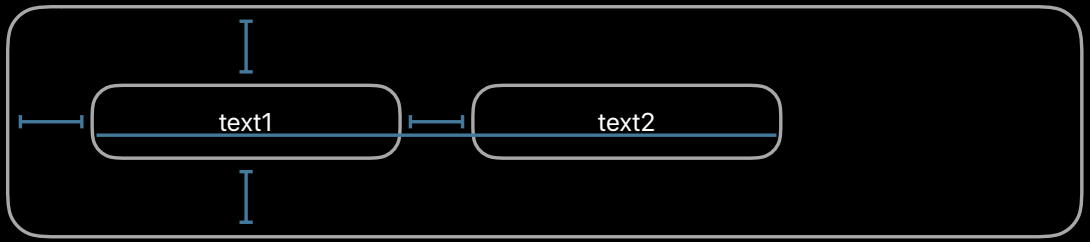
Internals and intuition

Building efficient layouts

Simple Layout

IB에서 이 화면을 만들었다면 고민할 게 없다

하지만 코드로 만들었다면?



인간 어떤 동작을 하는지 분석해봐

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

```
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

constraint를 모두 제거하고있다.

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

바꿀 부분만 배열로 Constraint 생성

```
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

Constraint activate

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

UIView level의 Constraint 갱신

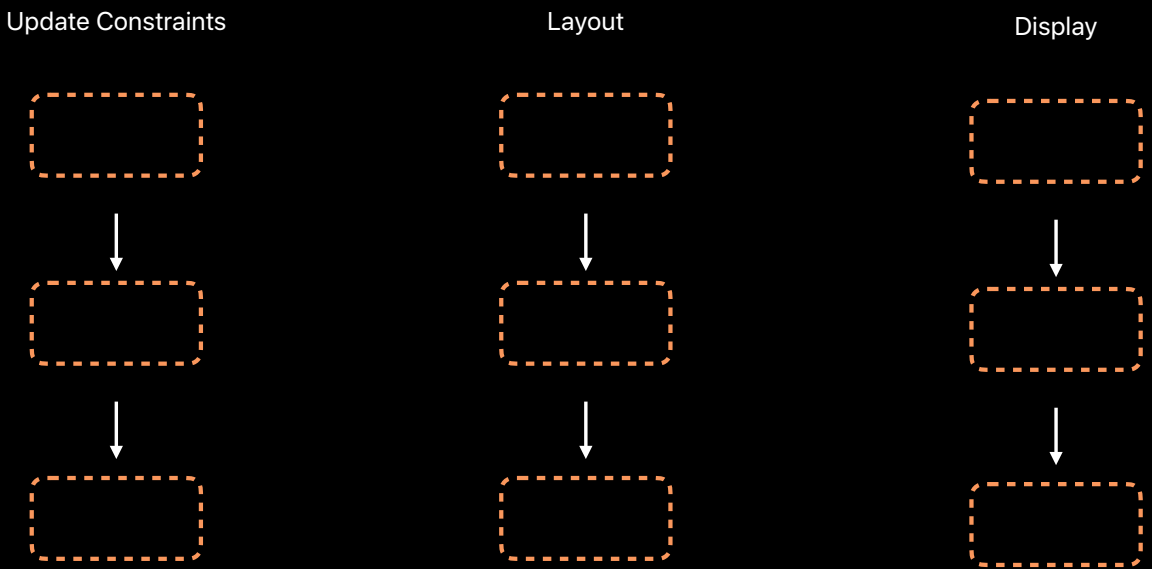
performance 관점에서 이 동작은 안좋아

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

The Render Loop

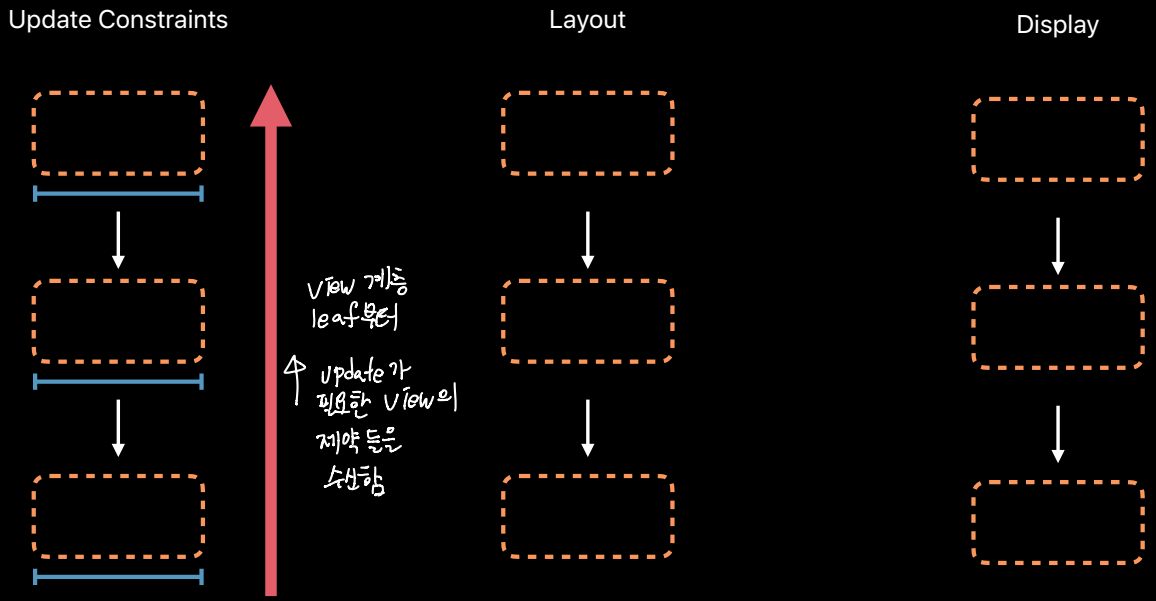
What is updateConstraints?

우선 렌더루프란? 잠재적으로 초당 120번 실행해야 콘텐츠가 올바르게 표시되는 주요



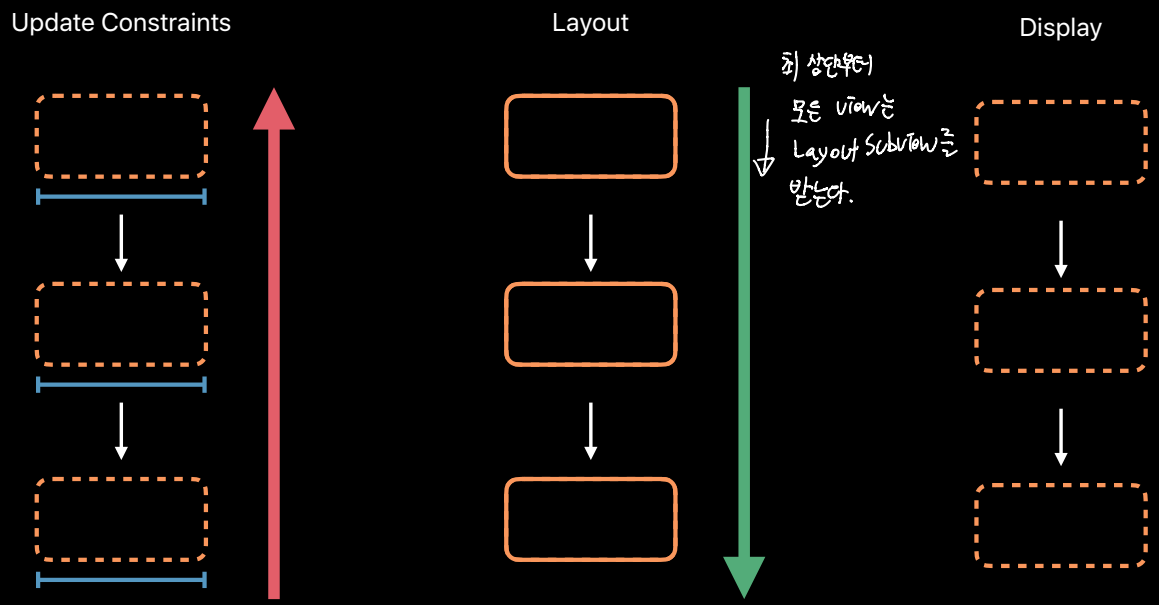
The Render Loop

What is updateConstraints?



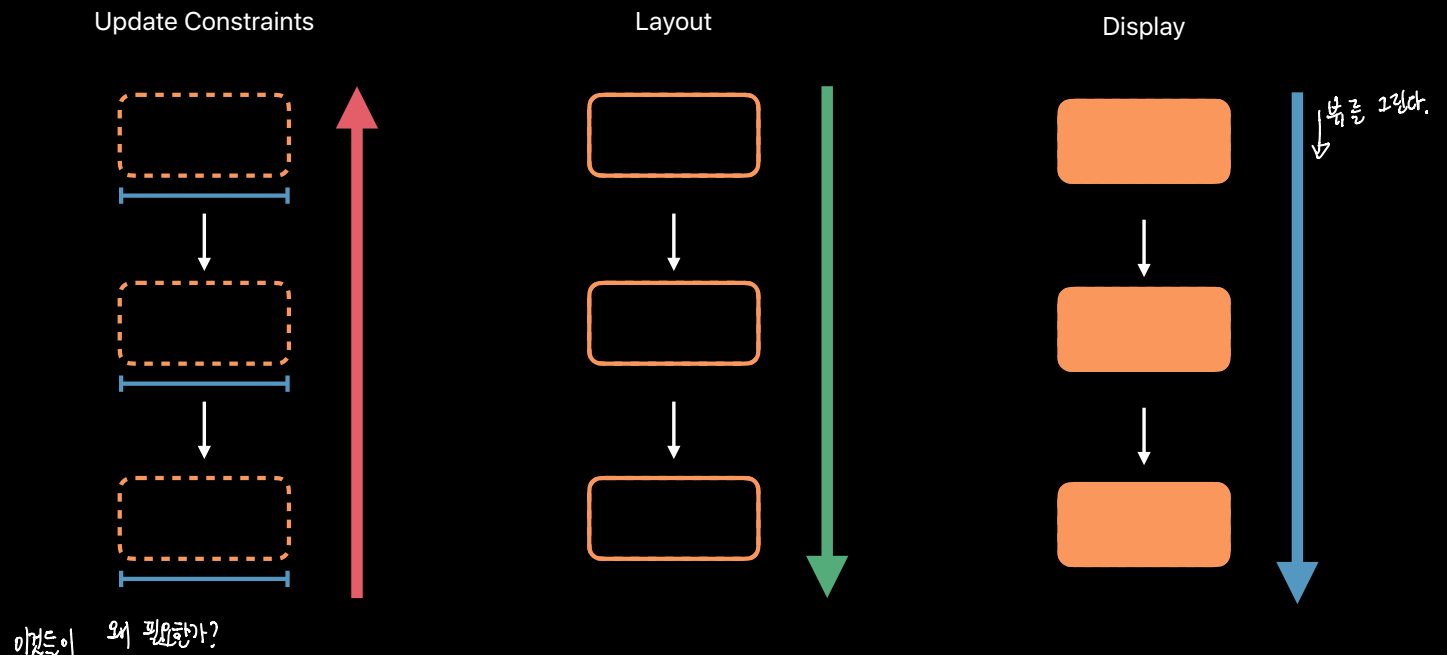
The Render Loop

What is updateConstraints?



The Render Loop

What is updateConstraints?



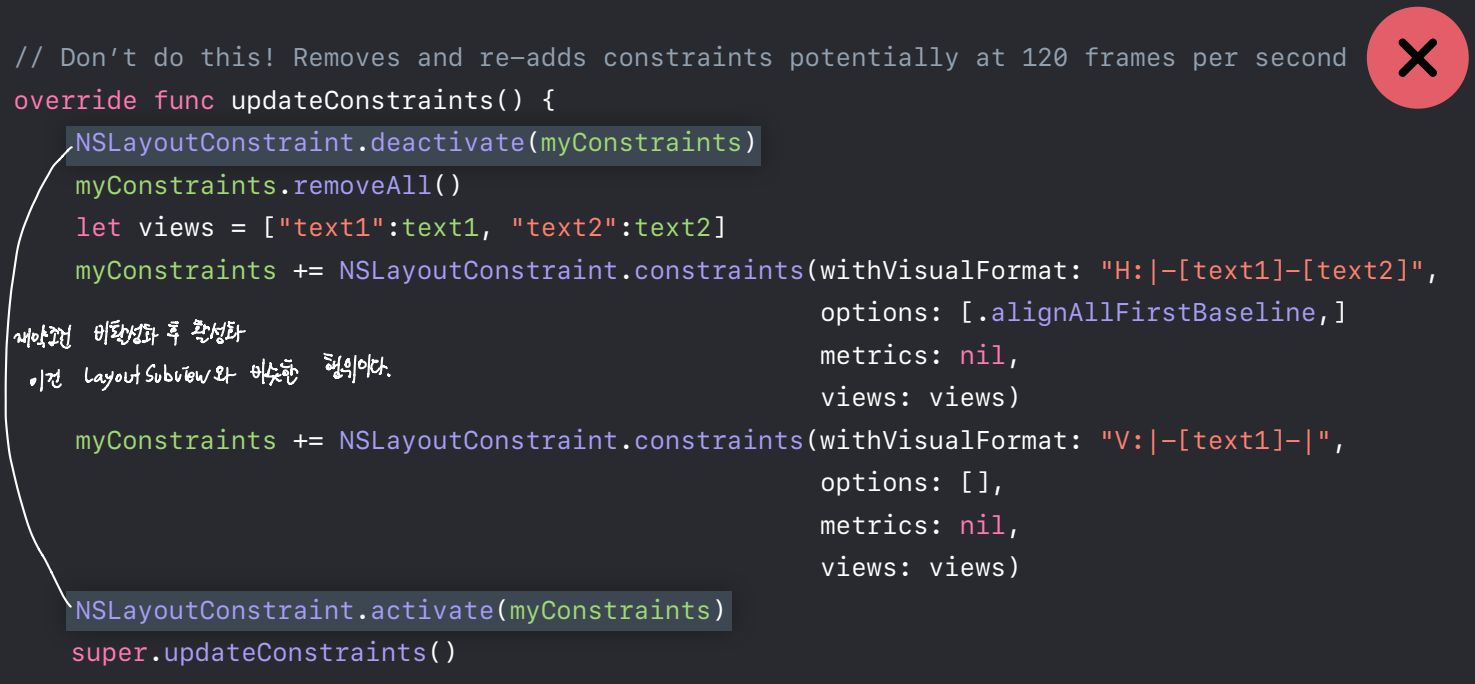
The Render Loop

What is updateConstraints?

쓸모없는 작업을 피하기위해 이런 구조를 가진다.
예를들어 UILabel의 size constraint가 있는 상황, 글씨만 font나 text size 와같이 크기와 관련된 프로퍼티는 아직 않았다
이런 모든 수정을 한번에 취합해서 적용해 주는게 render loop이다.

Update Constraints	Layout	Display
`updateConstraints()`	`layoutSubviews()`	`draw(_:)`
`setNeedsUpdateConstraints()`	`setNeedsLayout()`	`setNeedsDisplay()`
`updateConstraintsIfNeeded()`	`layoutIfNeeded()`	-

```
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline,]
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```


// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
NSLayoutConstraint.deactivate(myConstraints)
myConstraints.removeAll()
let views = ["text1":text1, "text2":text2]
myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
options: [.alignAllFirstBaseline,]
metrics: nil,
views: views)
myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
options: [],
metrics: nil,
views: views)
NSLayoutConstraint.activate(myConstraints)
super.updateConstraints()
}
제약조건 비활성화 후 활성화
이건 Layout Subview와 비슷한 형식이다.

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func layoutSubviews() {
    text1.removeFromSuperview()
    text1 = nil
    text1 = UILabel(frame: CGRect(x:  20, y: 20, width: 300, height: 30))
    self.addSubview(text1)

    text2.removeFromSuperview()
    text2 = nil
    text2 = UILabel(frame: CGRect(x: 340, y: 20, width: 300, height: 30))
    self.addSubview(text2)
    super.layoutSubviews()
}
```

매 프레임마다 모든것을 파괴하고 다시만드는것과
동일한 작업을 하고 있었다.

직관적으로 이건 잘못된 작업이란걸 느낄 수 있다.

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func layoutSubviews() {
    text1.removeFromSuperview()
    text1 = nil
    text1 = UILabel(frame: CGRect(x:  20, y: 20, width: 300, height: 30))
    self.addSubview(text1)

    text2.removeFromSuperview()
    text2 = nil
    text2 = UILabel(frame: CGRect(x: 340, y: 20, width: 300, height: 30))
    self.addSubview(text2)
    super.layoutSubviews()
}
```

그래서 어떻게 고쳤을까?

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

```
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
        let views = ["text1":text1, "text2":text2]
        myConstraints+=NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                      options: [.alignAllFirstBaseline],
                                                      metrics: nil,
                                                      views: views)
        myConstraints+=NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                      options: [],
                                                      metrics: nil,
                                                      views: views)
         NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

- Constraint는 단 한번만 설정

```swift
// This is ok! Doesn't do anything unless self.myConstraints has been nil'd out
override func updateConstraints() {
    if self.myConstraints == nil {
        var constraints = [NSLayoutConstraint]()
        let views = ["text1":text1, "text2":text2]
        constraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                       options: [.alignAllFirstBaseline],
                                                       metrics: nil,
                                                       views: views)
        constraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                       options: [],
                                                       metrics: nil,
                                                       views: views)
        NSLayoutConstraint.activate(constraints)
        self.myConstraints = constraints
    }
    super.updateConstraints()
}
```

```swift
// This is ok! Doesn't do anything unless self.myConstraints has been nil'd out
override func updateConstraints() {
    if self.myConstraints == nil {
        var constraints = [NSLayoutConstraint]()
        let views = ["text1":text1, "text2":text2]
        constraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                      options: [.alignAllFirstBaseline],
                                                      metrics: nil,
                                                      views: views)
        constraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                      options: [],
                                                      metrics: nil,
                                                      views: views)
        NSLayoutConstraint.activate(constraints)
        self.myConstraints = constraints
    }
    super.updateConstraints()
}
```

The Render Loop

Do you really need to be part of it?

Useful: Avoiding wasted work

Dangerous: Runs a lot

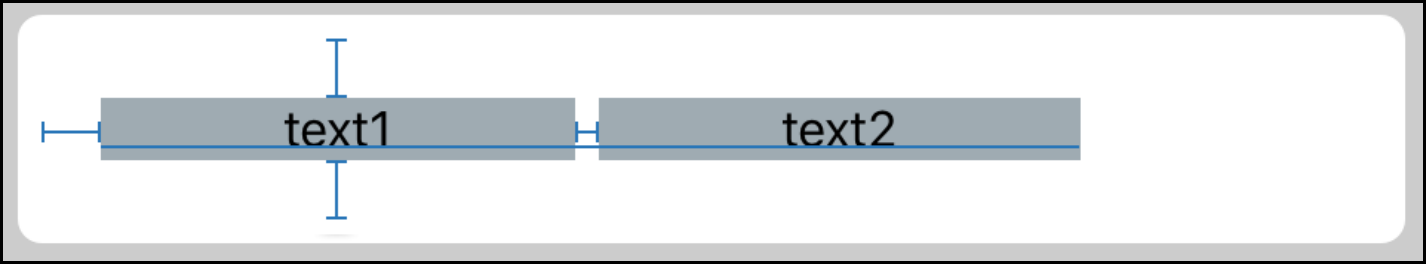
렌더루프에 대해 다시 생각해보자
이것은 최적화법을 피하는데 도움이 된다.

하지만 위험하다고 하다.
자주호출되기 때문에 매우 민감하다.

Interface Builder

제약은 한번만 넣어야한다. 가장 확실한 방법은 IB 를 사용하는 것 이다.

IB는 운영 다른 이유로 더 좋다고 한다...



왜 더 좋은지 이해해 보자

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

activate를 호출할때 무슨일이 일어날까?

Activating a Constraint

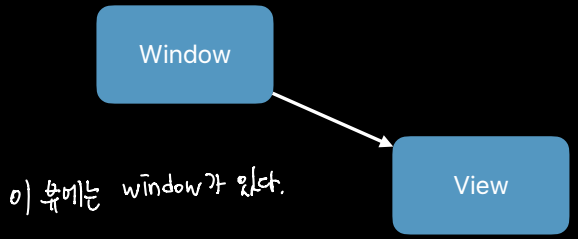
Internal structure

high level에서 도표를 그리며 이해 해 보자

이 뷰를 시작으로
constraint를 추가해 내려갔어.

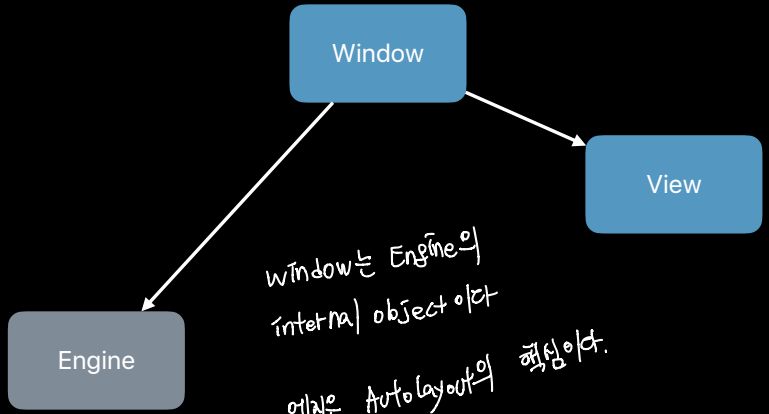
Activating a Constraint

Internal structure



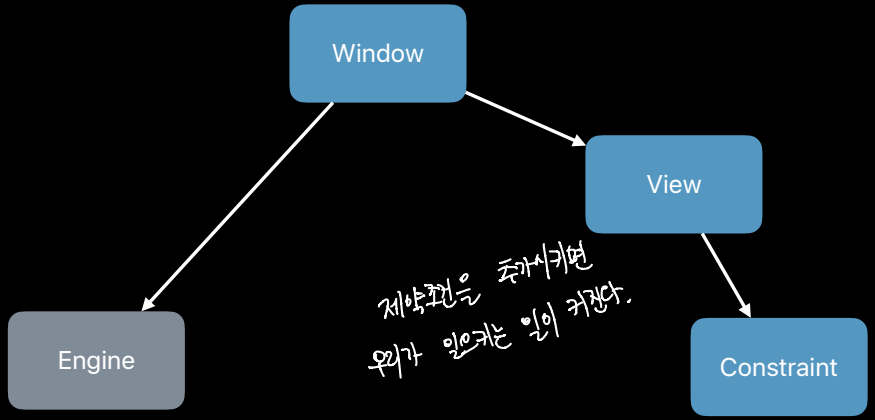
Activating a Constraint

Internal structure



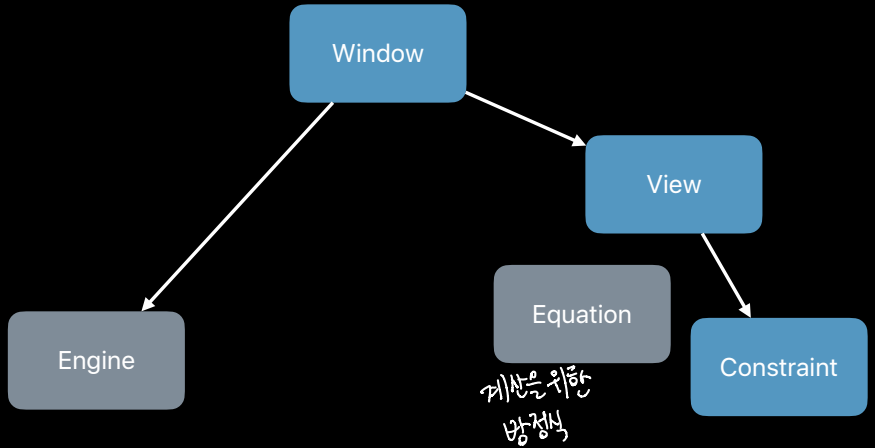
Activating a Constraint

Internal structure



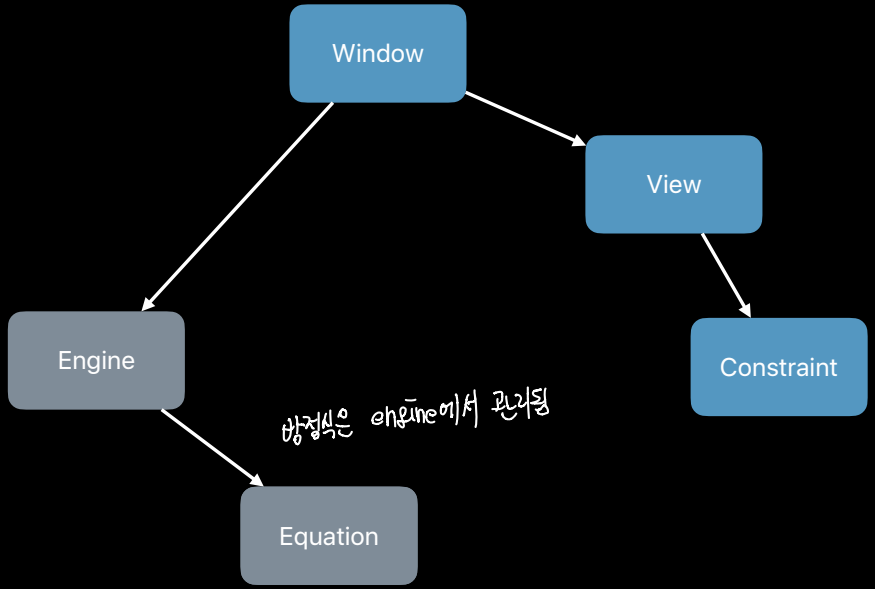
Activating a Constraint

Internal structure



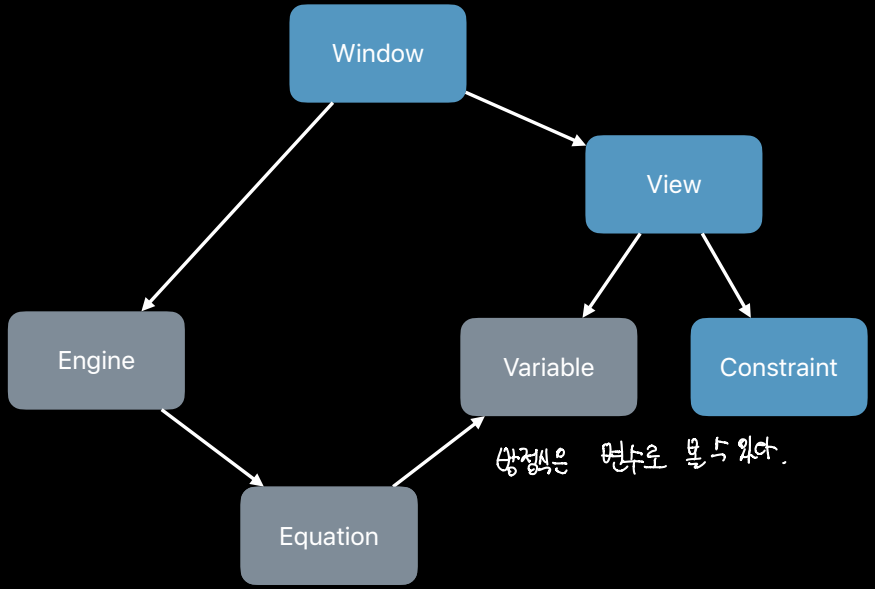
Activating a Constraint

Internal structure



Activating a Constraint

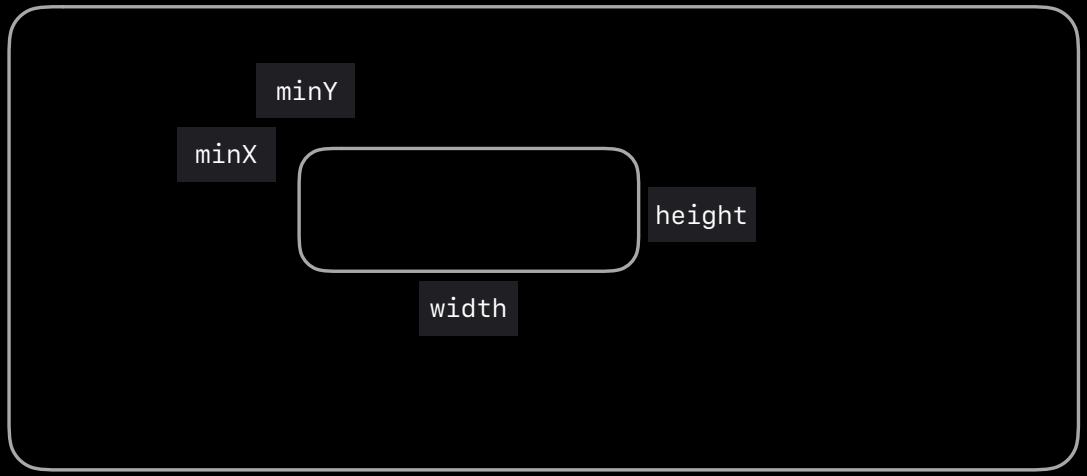
Internal structure



Variables

Unknowns layout must calculate

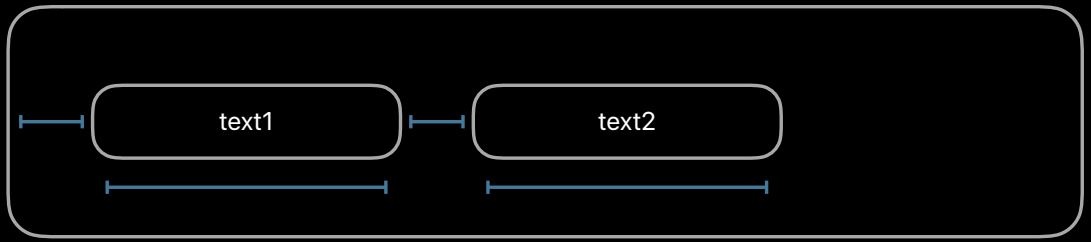
여기 변수가 4개 있다.



Activating a Constraint

process를 살펴보자 (수평에 대한 제약만 볼 것이다)

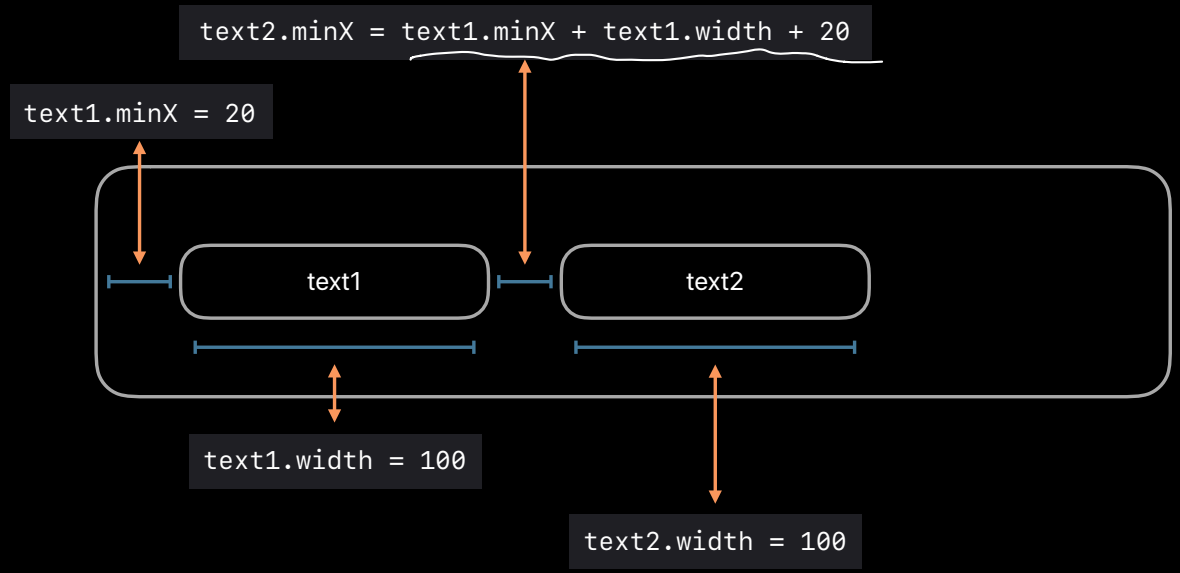
가장먼저, 우리는 이 화면을 만들었다. 이것은 방정식이다.



Activating a Constraint

이렇게 보이는 방정식야.

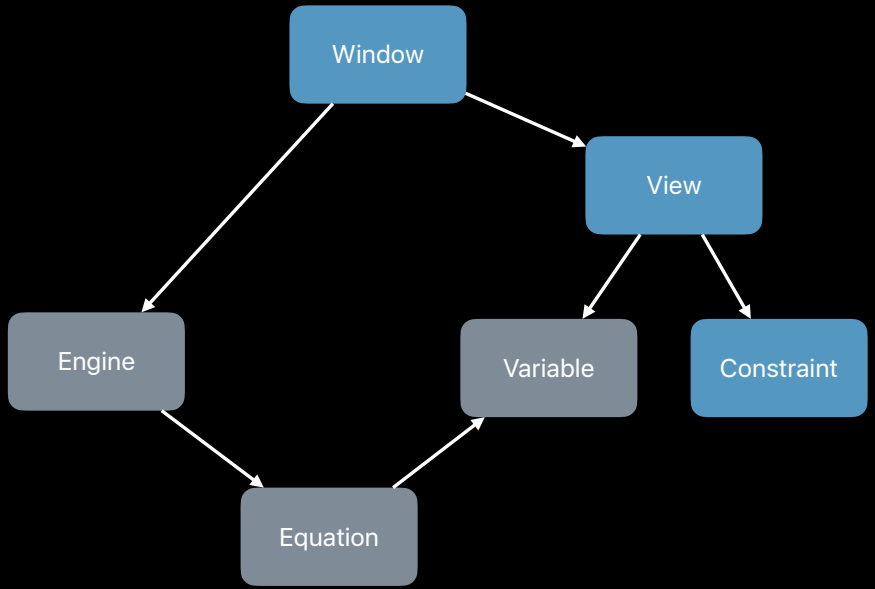
지금 가장 흥미로운건 뷰 사이에 공간야.

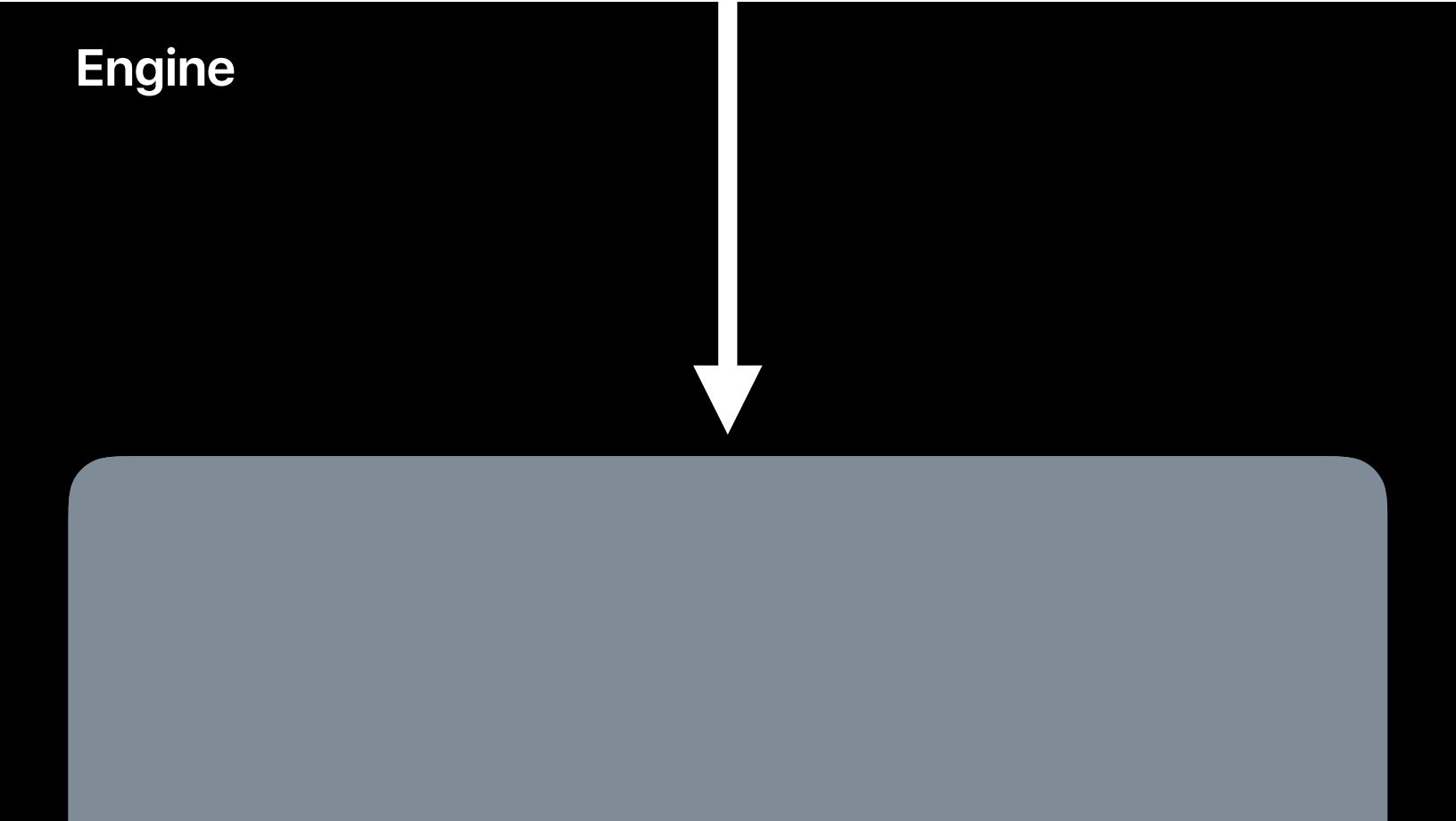


Activating a Constraint

Internal structure

engine 부터 process를 따라가보자
(성능을 고려하며)




Engine

Engine

엔진은 이 변수들을 해결하려 한다.
마치 수학 처럼

```
text1.minX  = 8
```

Engine

```
text1.minX  = 8
text1.width = 100
```

Engine

```
text1.minX  = 8
text1.width = 100
text2.minX = text1.minX + text1.width + 20
```

누가 이용제를 해줄 해 달라고 한다면?

Engine

```
text1.minX  = 8
text1.width = 100
text2.minX = text1.minX + text1.width + 20
```

Engine

```
text1.minX  = 8
text1.width = 100
text2.minX =     8       +   100        + 20
```

Engine

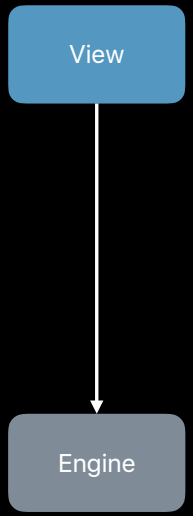
```
text1.minX  = 8
text1.width = 100
text2.minX = 128
```

Engine

```
text1.minX  = 8
text1.width = 100
text2.minX = 128
text2.width = 100
```

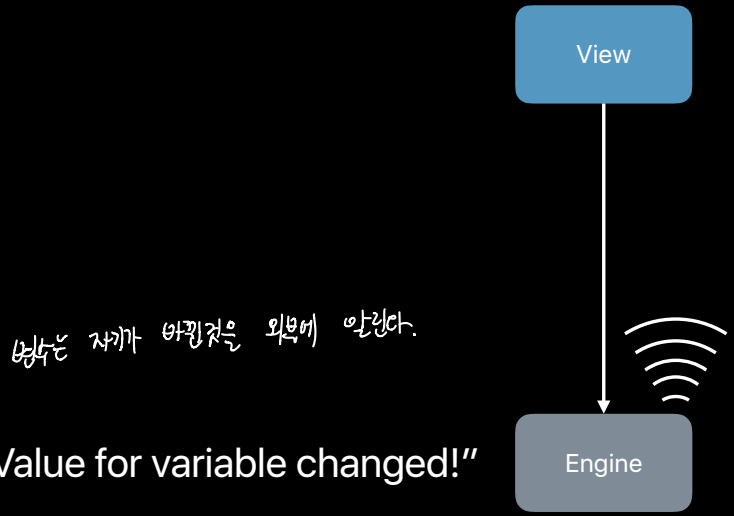
Activating a Constraint

Triggering the layout pass



Activating a Constraint

Triggering the layout pass

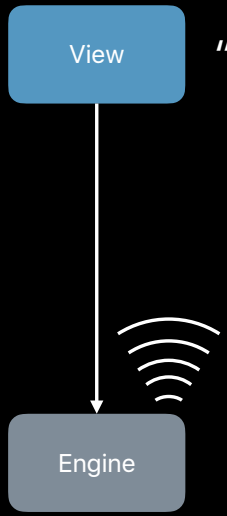


Activating a Constraint

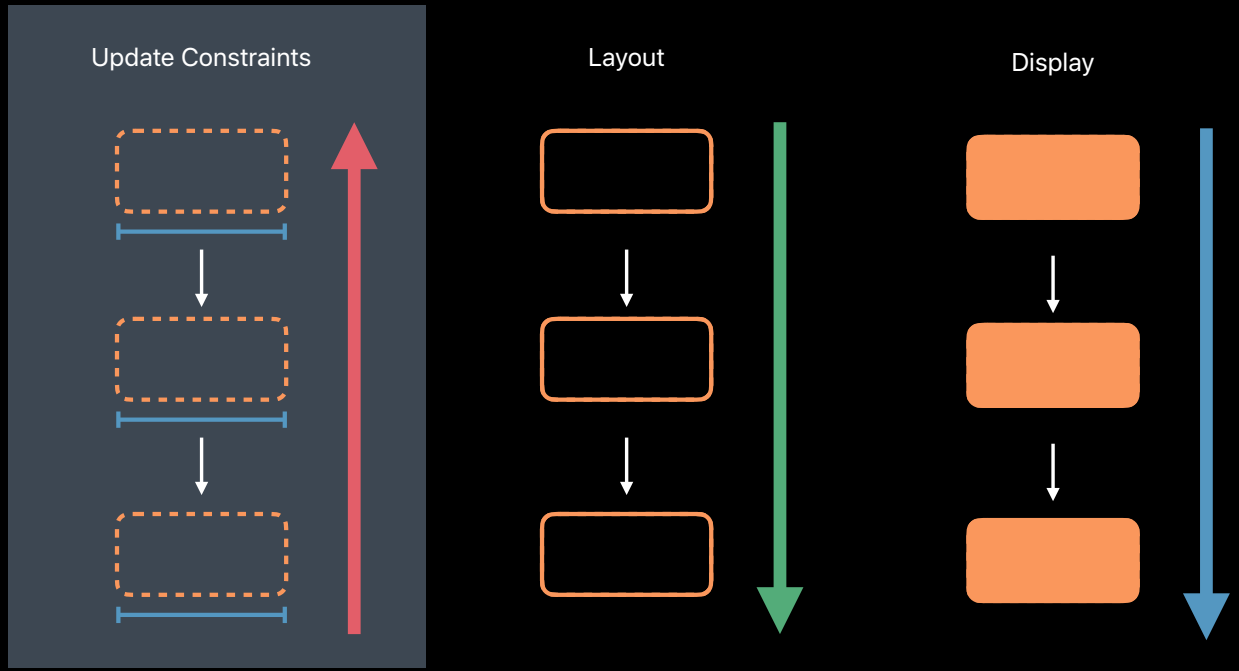
Triggering the layout pass

뷰가 그것에 반응할까?

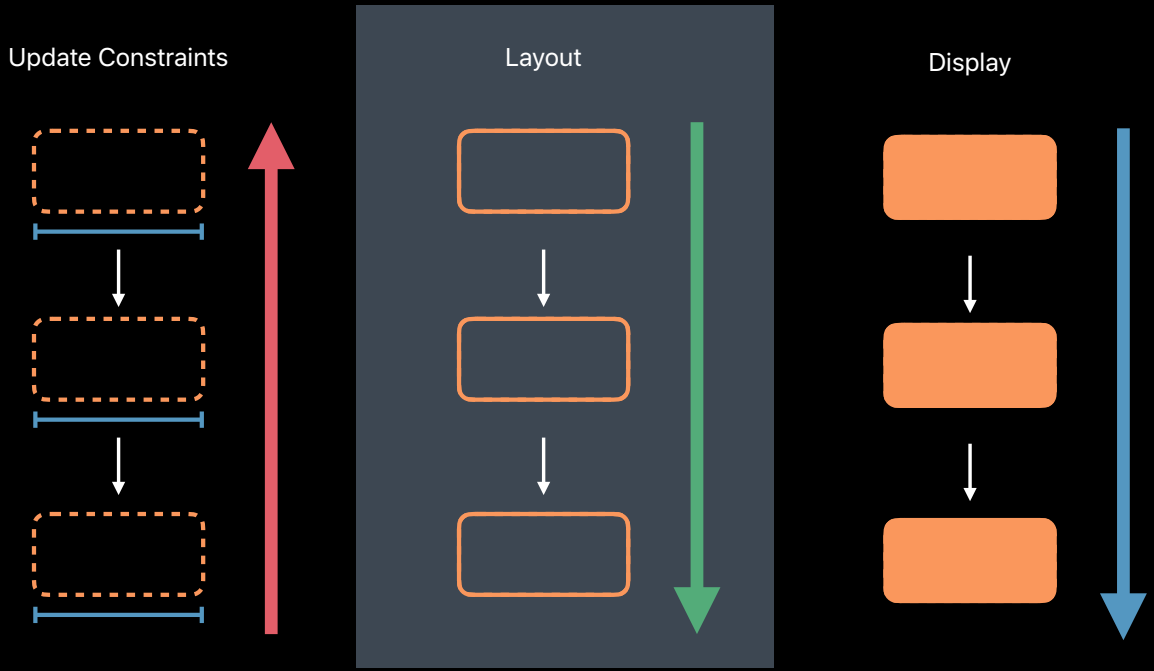
움직이기 위해 setNeedsLayout이 필요하므로 슈퍼뷰를 찾아가라고 한다.



The Render Loop



The Render Loop

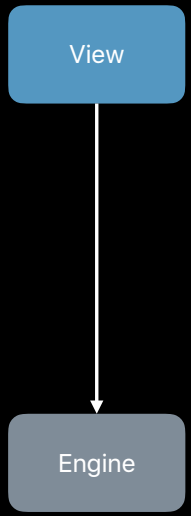


UIView.layoutSubviews()

Copying data from engine to subview

자 이제 layout subview 에서 일어나는일은

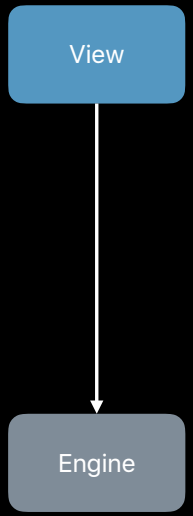
frame 을 그리려면 값들은 복사해와야함



UIView.layoutSubviews()

Copying data from engine to subview

"엔진아 값 내놔"

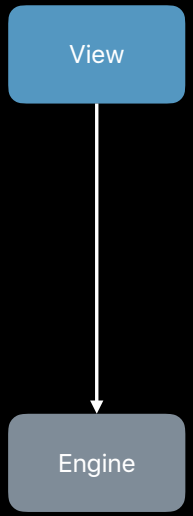


"Engine, what are
the values for the variables?"

UIView.layoutSubviews()

Copying data from engine to subview

그래서 세팅되었다.

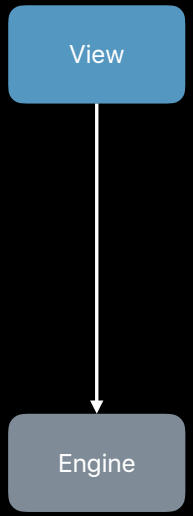


"Engine, what are
the values for the variables?"

"Subview setBounds(),
subview setCenter()"

UIView.layoutSubviews()

Copying data from engine to subview



“Engine, what are
the values for the variables?”

“Subview setBounds(),
subview setCenter()”

UIView.layoutSubviews()

Copying data from engine to subview

"Engine, what are
the values for the variables?"

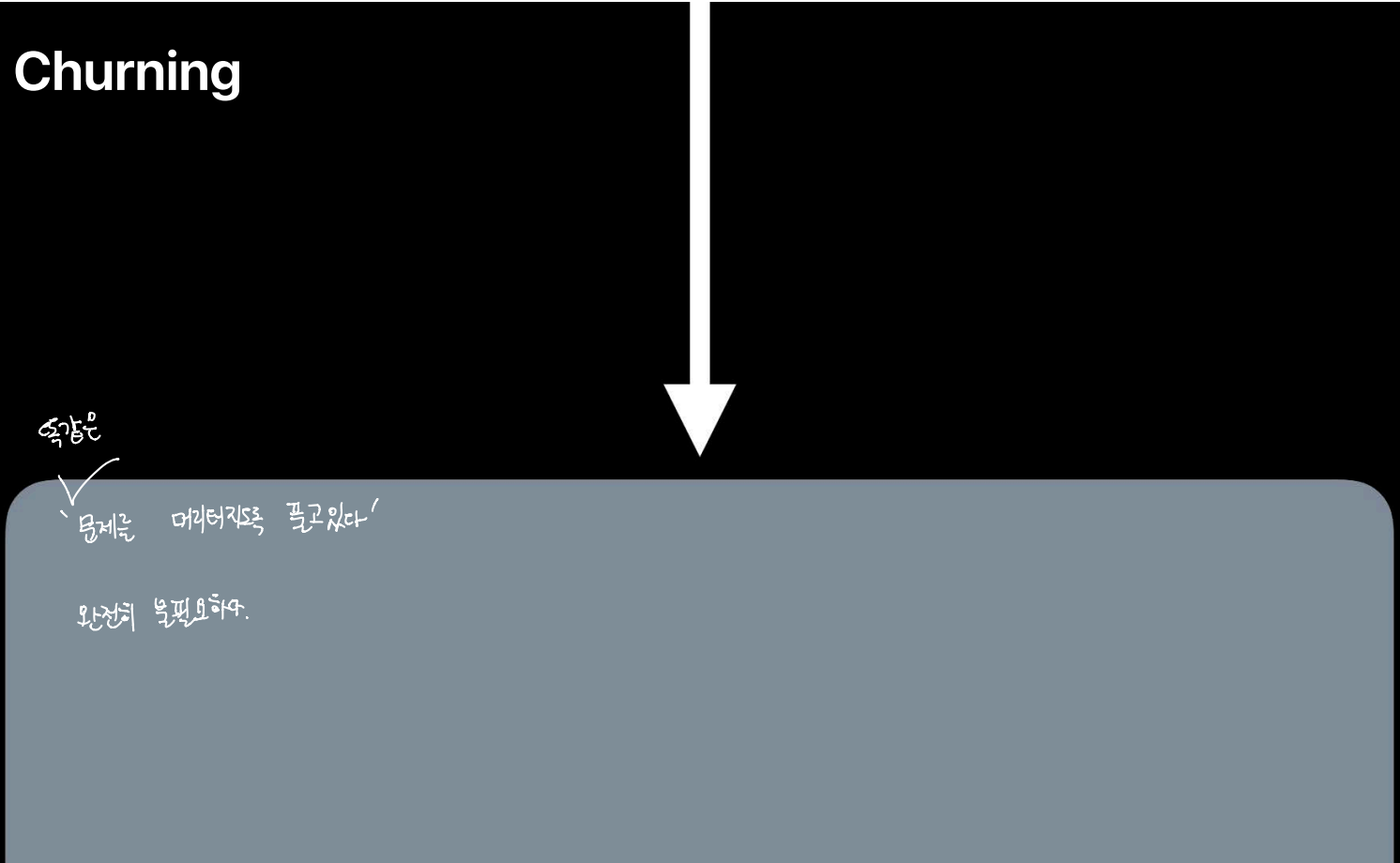
"Subview
s
Center()"

Done!

```swift
// Don't do this! Removes and re-adds constraints potentially at 120 frames per second
override func updateConstraints() {
    NSLayoutConstraint.deactivate(myConstraints)
    myConstraints.removeAll()
    let views = ["text1":text1, "text2":text2]
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "H:|-[text1]-[text2]",
                                                    options: [.alignAllFirstBaseline],
                                                    metrics: nil,
                                                    views: views)
    myConstraints += NSLayoutConstraint.constraints(withVisualFormat: "V:|-[text1]-|",
                                                    options: [],
                                                    metrics: nil,
                                                    views: views)
    NSLayoutConstraint.activate(myConstraints)
    super.updateConstraints()
}
```

이작업은 왜 하는가?

Churning

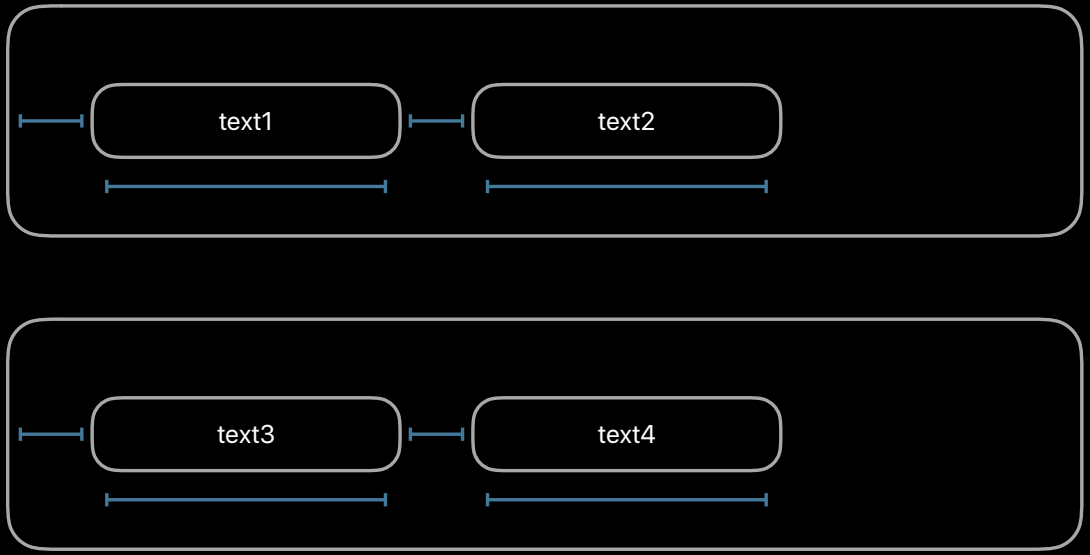


Churning

Local Versus Global Layout

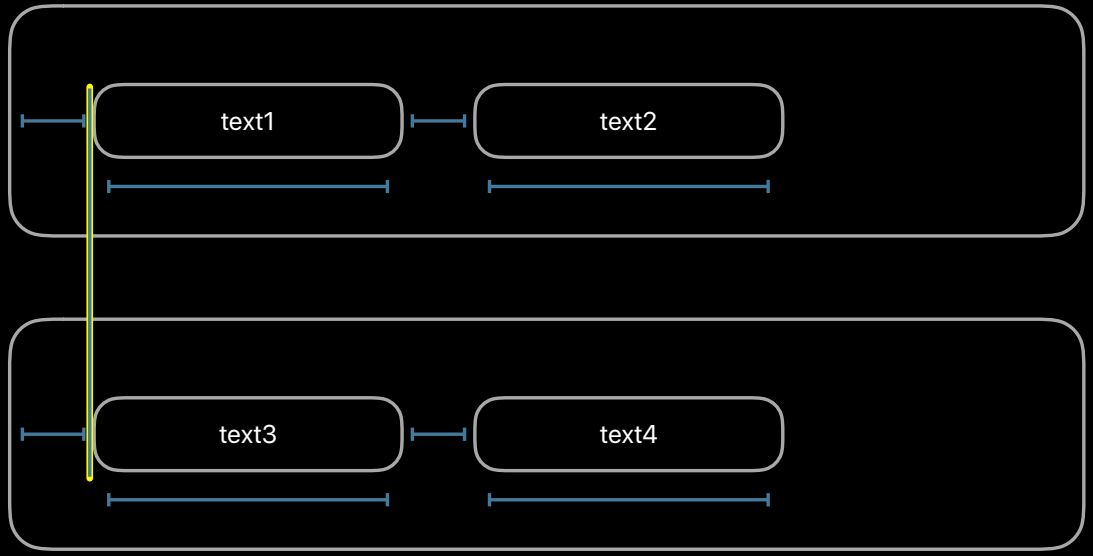
You don’t pay for what you don’t use

우리가 했던 일은 다음과 같다면



Local Versus Global Layout

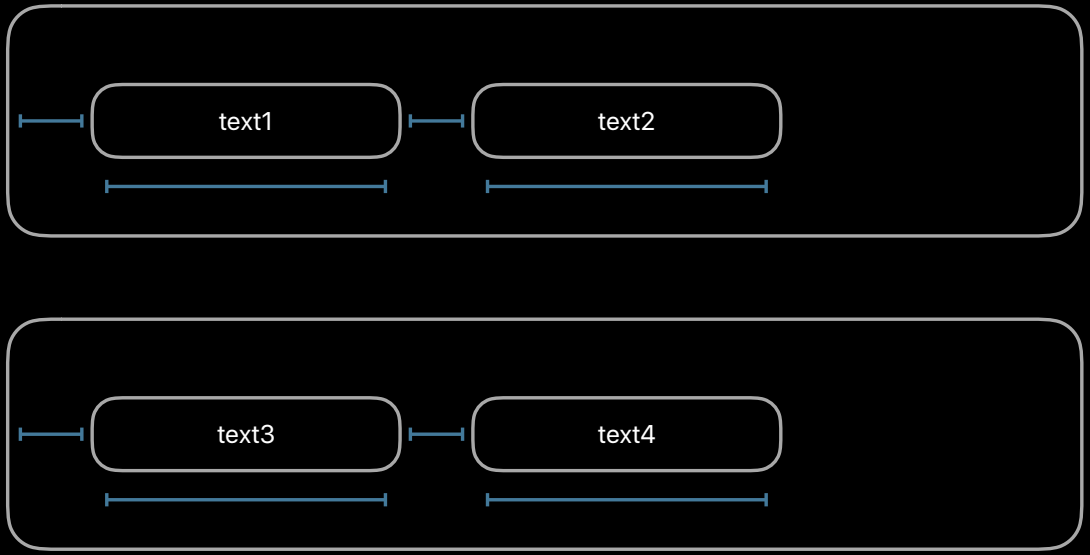
You don't pay for what you don't use



Local Versus Global Layout

You don't pay for what you don't use

보통은 과거 요소하고만 제약을 건다.



Unrelated Views Don’t Interact

Unrelated Views Don’t Interact

보통은 엔진은 두개의 독립 뷰를 사용한다.

```
text1.minX  = 8
text1.width = 100
text2.minX  = 20 + text1.minX + text1.width
```

```
text3.minX  = 8
text3.width = 100
text4.minX  = 20 + text3.minX + text3.width
```

Unrelated Views Don’t Interact

```
text1.minX  = 8
text1.width = 100
text2.minX  = 128
```

```
text3.minX  = 8
text3.width = 100
text4.minX  = 128
```

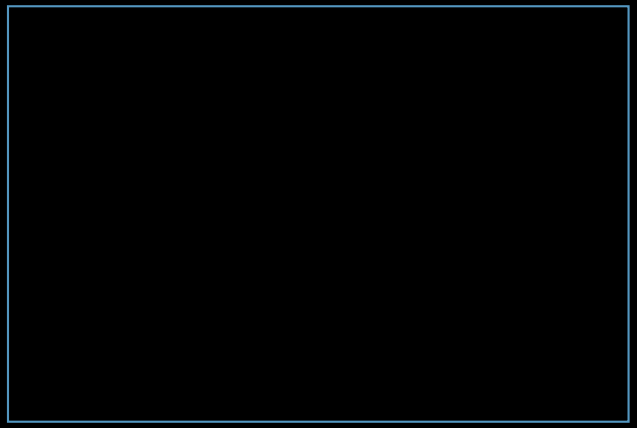
Unrelated Views Don’t Interact

```
text1.minX  = 8
text1.width = 100
text2.minX  = 128
text2.width = 100
```

```
text3.minX  = 8
text3.width = 100
text4.minX  = 128
text4.width = 100
```

Performance Scales Linearly When No Interaction

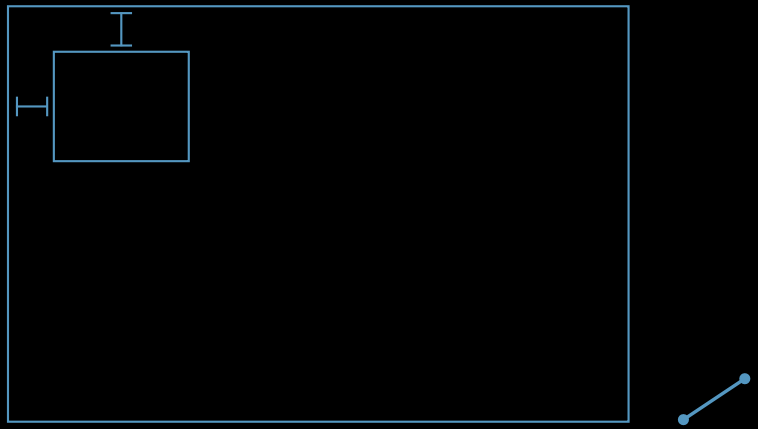
How much work would it take by hand?



Performance Scales Linearly When No Interaction

How much work would it take by hand?

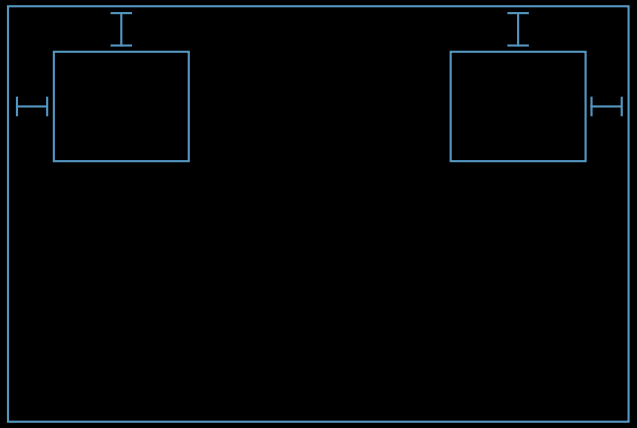
아무리 제약이 없어도

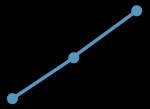


Performance Scales Linearly When No Interaction

How much work would it take by hand?

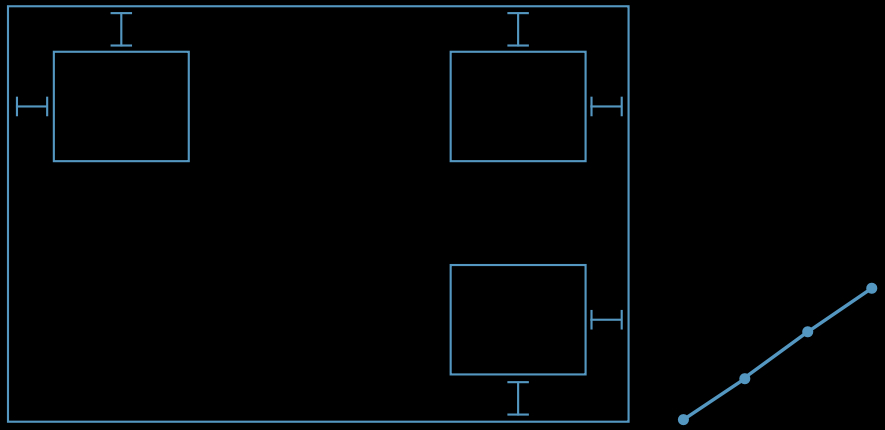
선형적으로 시간이 걸릴 뿐이다.





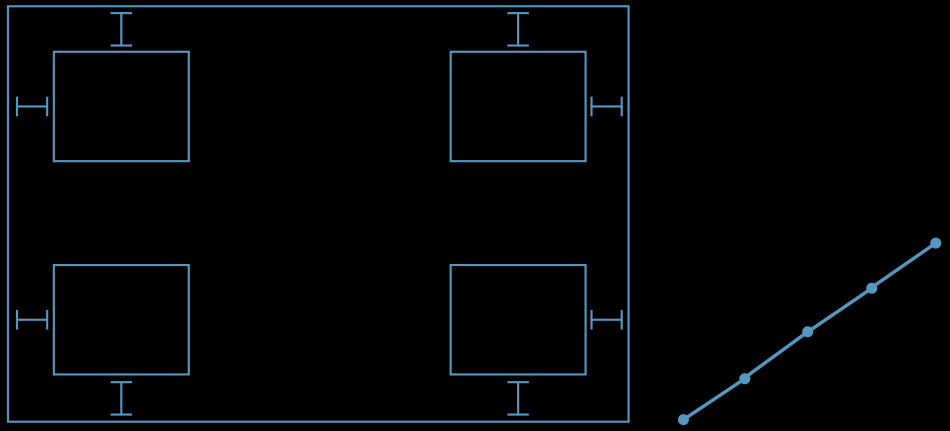
Performance Scales Linearly When No Interaction

How much work would it take by hand?



Performance Scales Linearly When No Interaction

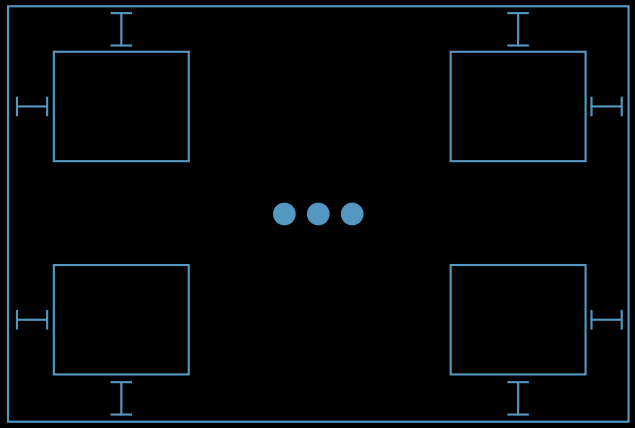
How much work would it take by hand?

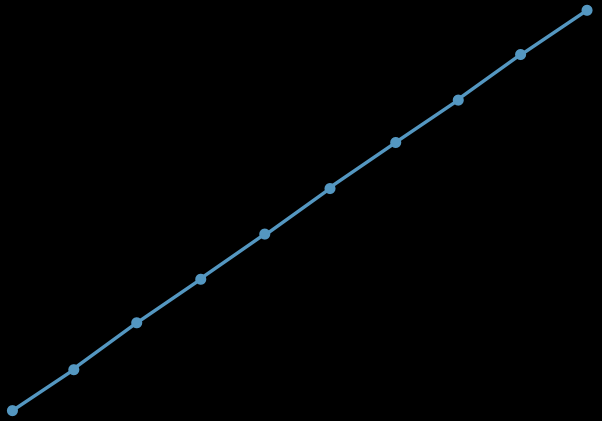


Performance Scales Linearly When No Interaction

How much work would it take by hand?

이게 선형인 이유는 서로 dependency가 없어서야.

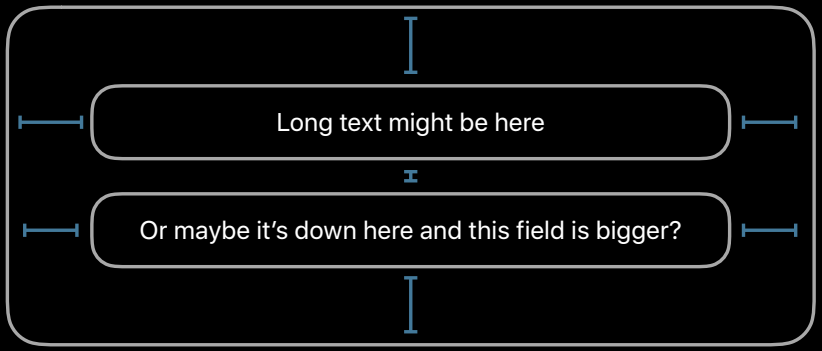




The engine is a layout cache
and dependency tracker.

Model The Problem Naturally

Don’t avoid using constraints



Model The Problem Naturally

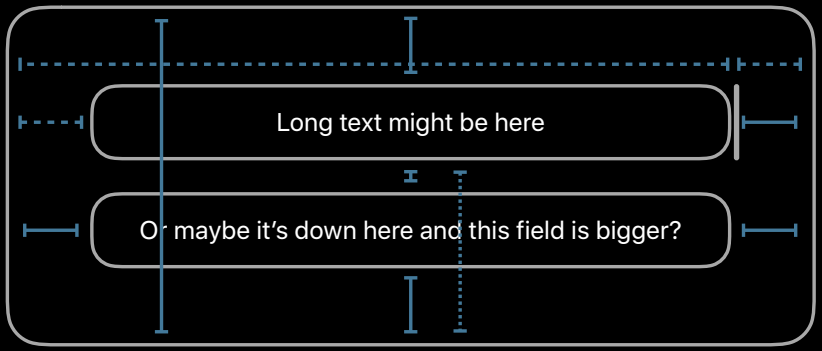
Don’t wedge two layouts into one set of constraints

가몸

많은 제약

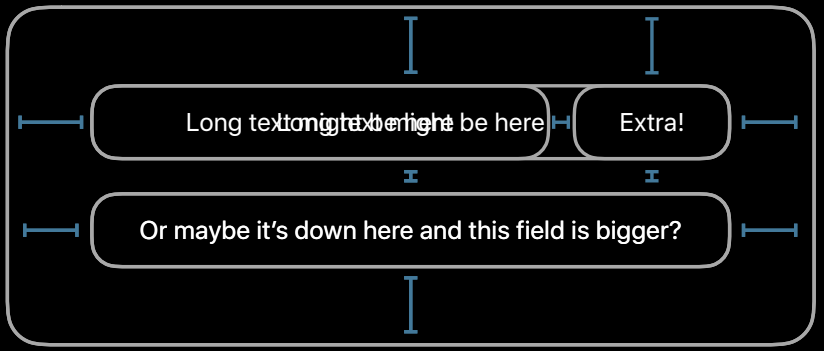
많은 우선순위

좋지 않은 상황



Model The Problem Naturally

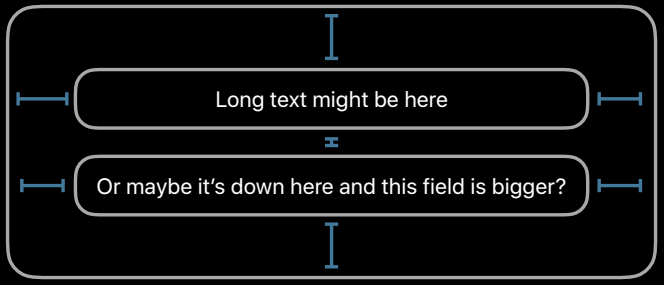
Don’t wedge two layouts into one set of constraints

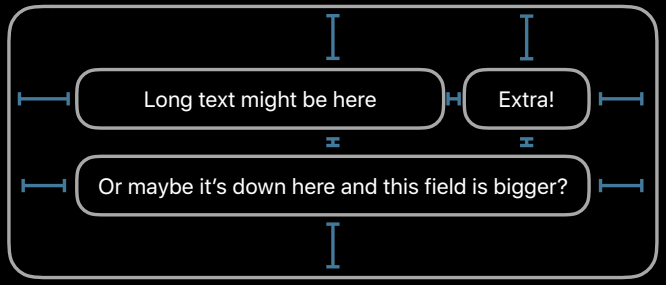


Model The Problem Naturally

Don’t wedge two layouts into one set of constraints

보통 누군가 완벽히 두개를 불리하고싶을때 발생
(여유)





정말 안좋음

디버그 거의 불가능 (만든사람 외에)

아예 불리하는게
성능이나 이해하기에도 좋다.

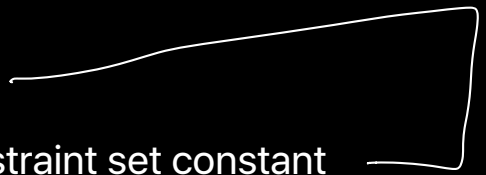
Is That Everything?

Costs of other auto layout features

Inequality

NSLayoutConstraint set constant

Priority



이 두개는 사실 빨리 작동한다.

Is That Everything?

Costs of other auto layout features

Inequality

NSLayoutConstraint set constant

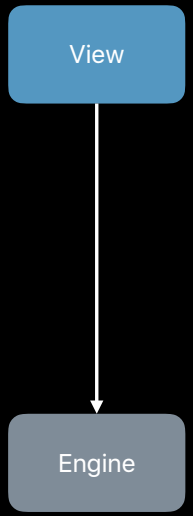
Priority

우선순위 짝이다.

중요한 건

Error Minimization

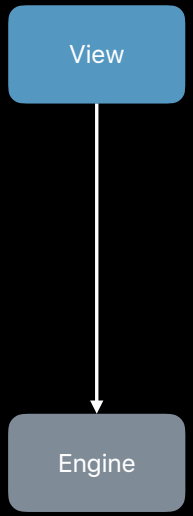
Constraints with non-required priority



Error Minimization

Constraints with non-required priority

"엔진, 이 변수들 값이 뭐니?"

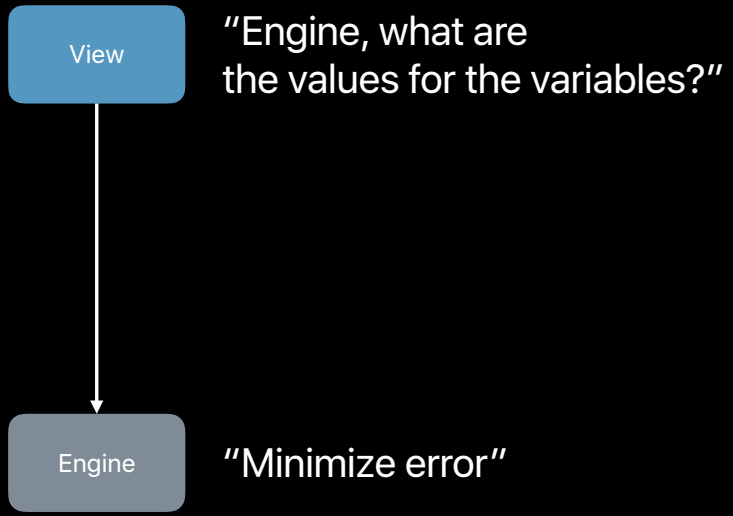


"Engine, what are
the values for the variables?"

Error Minimization

Constraints with non-required priority

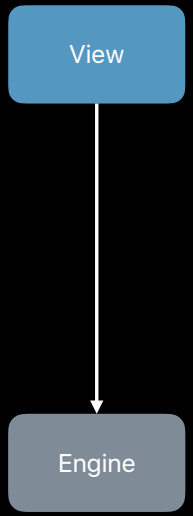
엔진은 모든 값들의 오차가 최소화 되길 보장해야함



Error Minimization

Constraints with non-required priority

신과 같은



"Engine, what are
the values for the variables?"

"Subview setBounds()"

"Minimize error"

Performance Intuition

직관에 따라

Don’t churn

Basic algebra

Engine is a layout cache and tracker

You pay for what you use

직관은 넘어 구체화해보자

iOS 12 improvements

Internals and intuition

Building efficient layouts

iOS 12 improvements

Internals and intuition

Building efficient layouts

#WWDC18

Building Efficient Layouts

Kasia Wawer, iOS Keyboards

Constraint Churn

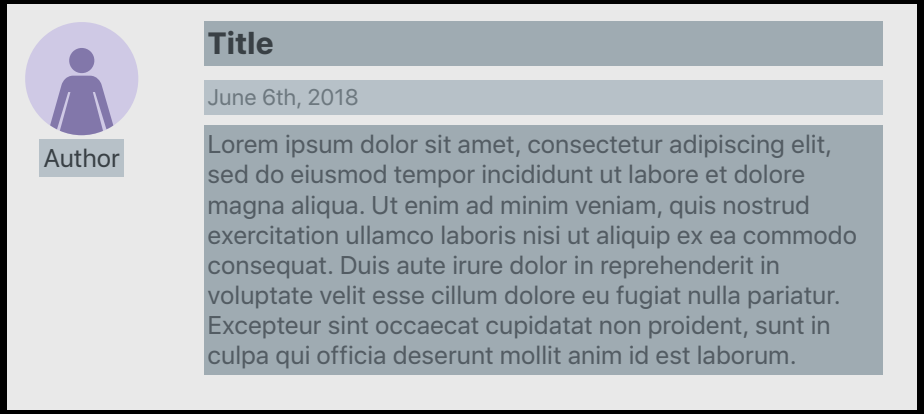
New solution, same layout

Extra work happens with no visible result

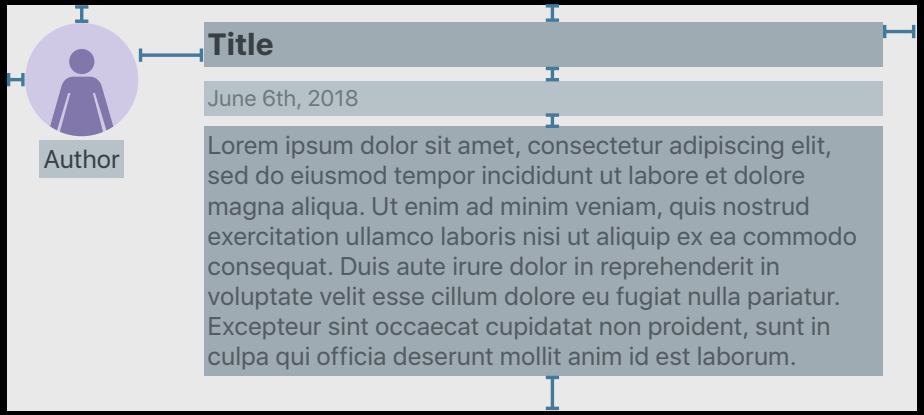
Enough churn can affect performance

Building a Layout

이런 레이아웃이 필요하다

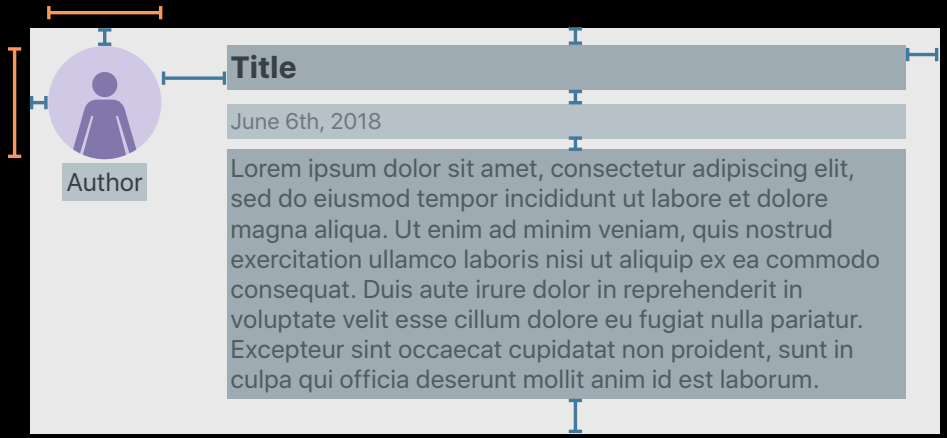


Building a Layout

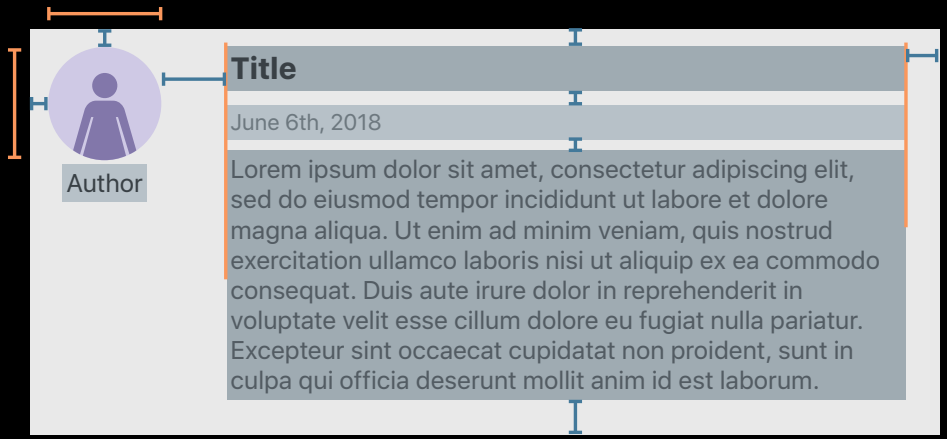


Building a Layout

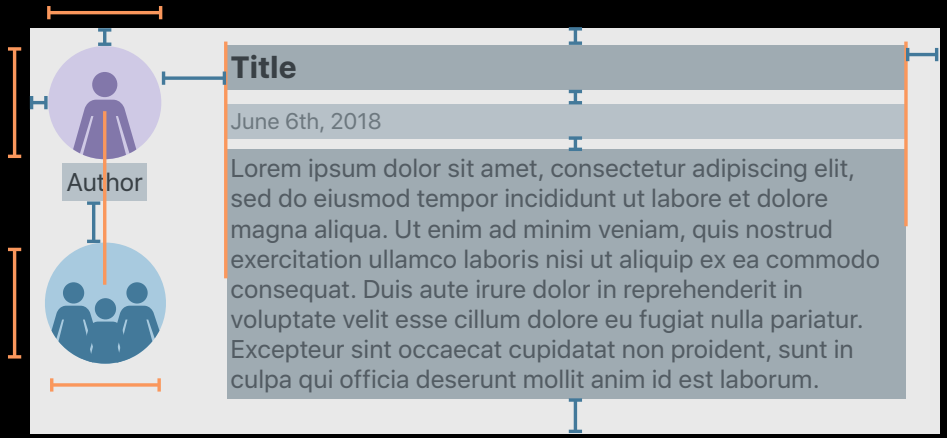
사이징



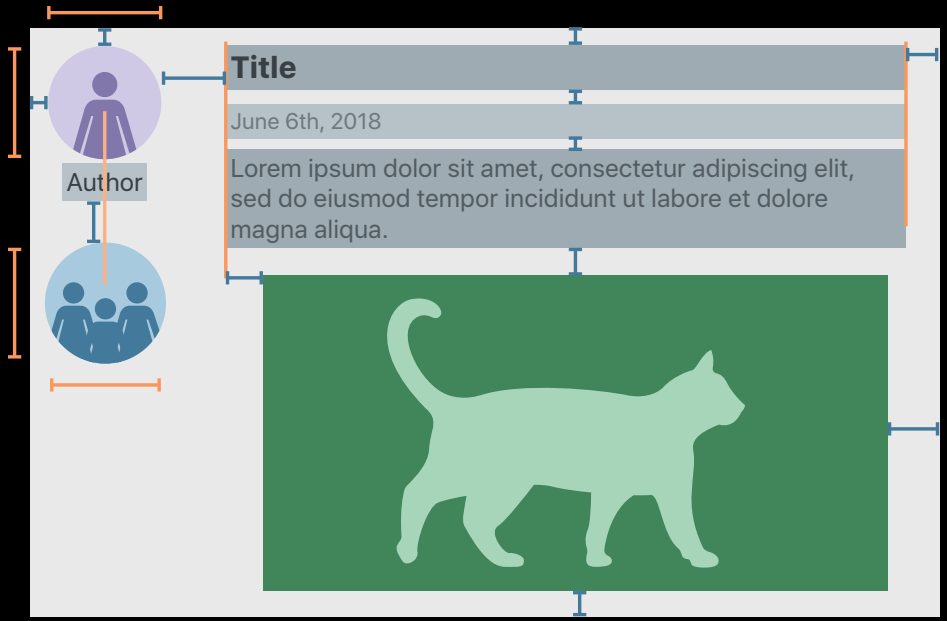
Building a Layout



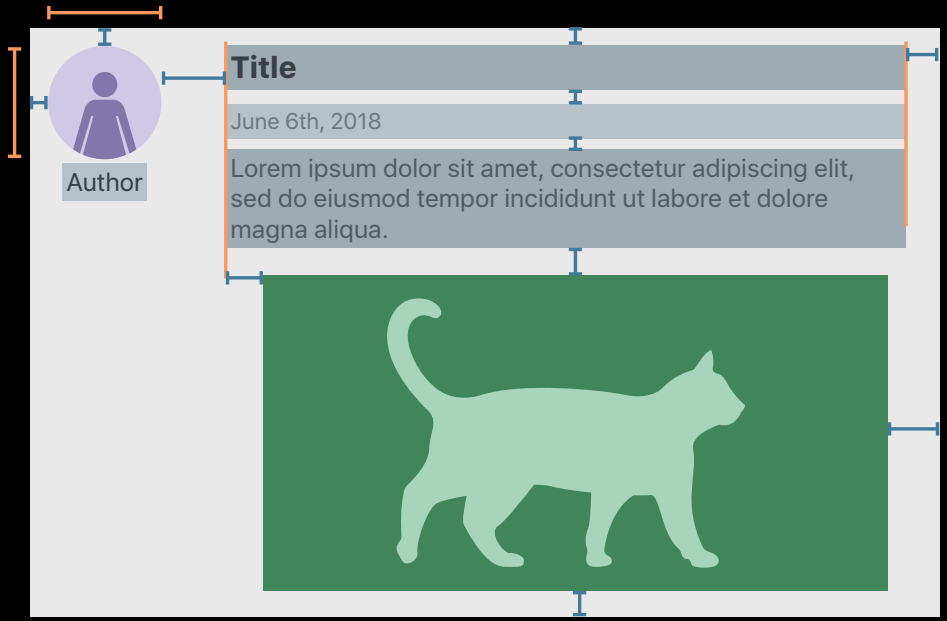
Building a Layout



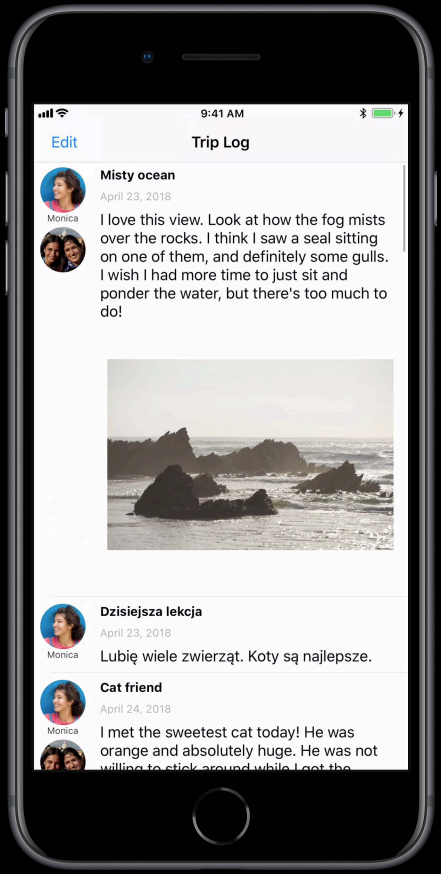
Building a Layout

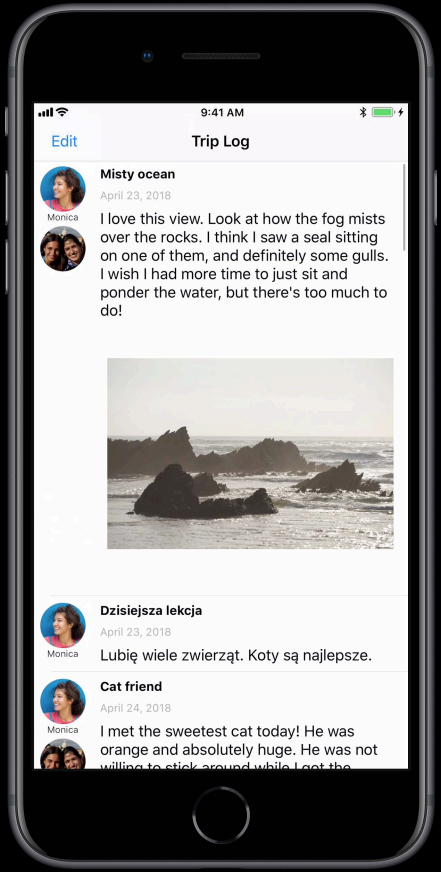


Building a Layout



스크롤해
자꾸
업됨


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

Finding the Problem

layout instrument

에뮬 iOS14되고 실제로 측시안됨!

Finding the Problem

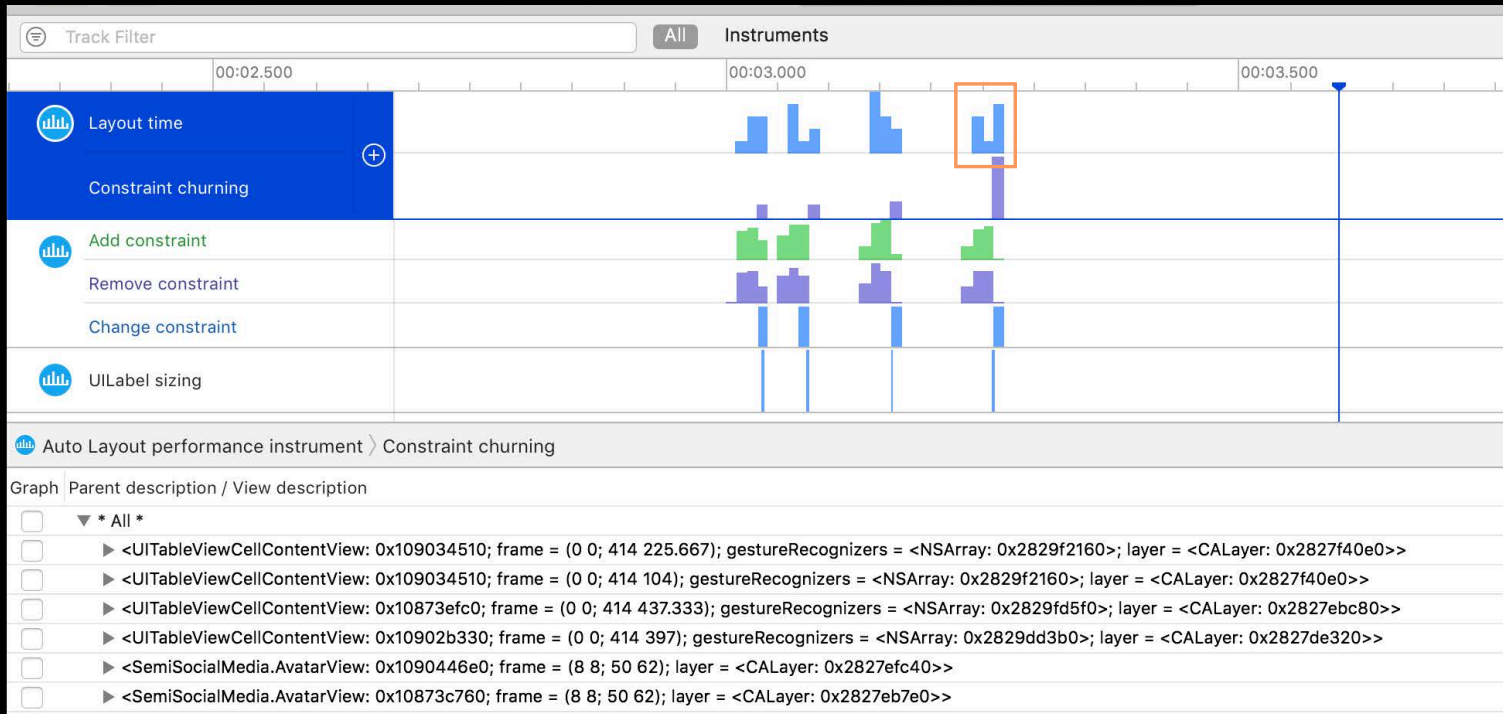
Finding the Problem

Finding the Problem

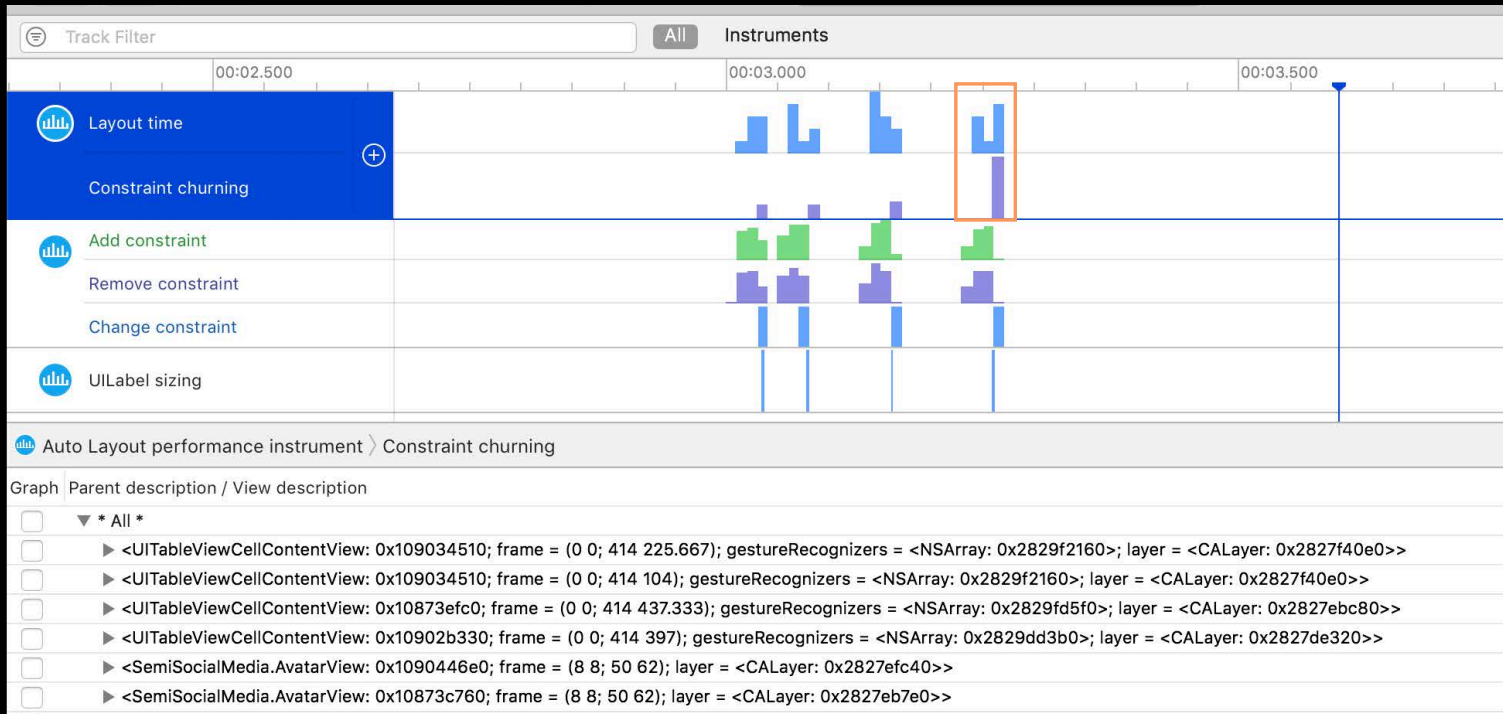
Finding the Problem

Finding the Problem

스크롤링 할때 어디서 문제일까?



Finding the Problem

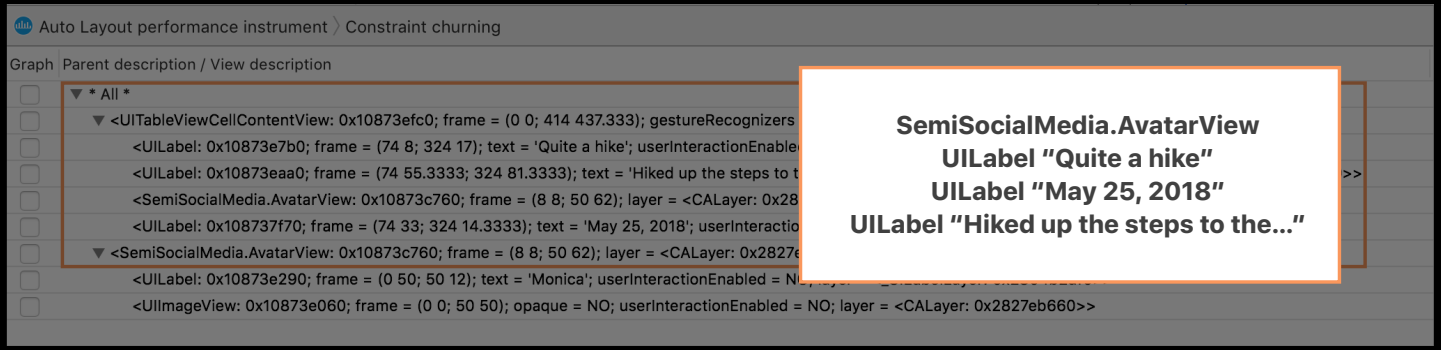


Finding the Problem

디테일뷰

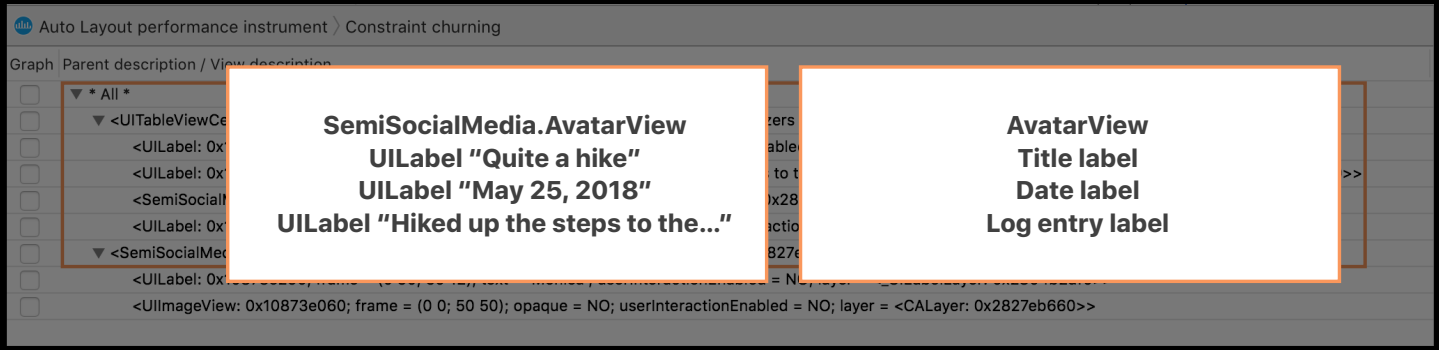
```
Auto Layout performance instrument 〉 Constraint churning

Graph | Parent description / View description

▼ * All *
  ▼ <UITableViewCellContentView: 0x10873efc0; frame = (0 0; 414 437.333); gestureRecognizers = <NSArray: 0x2829fd5f0>; layer = <CALayer: 0x2827ebc80>>
      <UILabel: 0x10873e7b0; frame = (74 8; 324 17); text = 'Quite a hike'; userInteractionEnabled = NO; layer = <_UILabelLayer: 0x2804b3e80>>
      <UILabel: 0x10873eaa0; frame = (74 55.3333; 324 81.3333); text = 'Hiked up the steps to the...'; userInteractionEnabled = NO; layer = <_UILabelLayer: 0x2804b8050>>
      <SemiSocialMedia.AvatarView: 0x10873c760; frame = (8 8; 50 62); layer = <CALayer: 0x2827eb7e0>>
      <UILabel: 0x108737f70; frame = (74 33; 324 14.3333); text = 'May 25, 2018'; userInteractionEnabled = NO; layer = <_UILabelLayer: 0x2804b3f70>>
  ▼ <SemiSocialMedia.AvatarView: 0x10873c760; frame = (8 8; 50 62); layer = <CALayer: 0x2827eb7e0>>
      <UILabel: 0x10873e290; frame = (0 50; 50 12); text = 'Monica'; userInteractionEnabled = NO; layer = <_UILabelLayer: 0x2804b2df0>>
      <UIImageView: 0x10873e060; frame = (0 0; 50 50); opaque = NO; userInteractionEnabled = NO; layer = <CALayer: 0x2827eb660>>
```

Finding the Problem

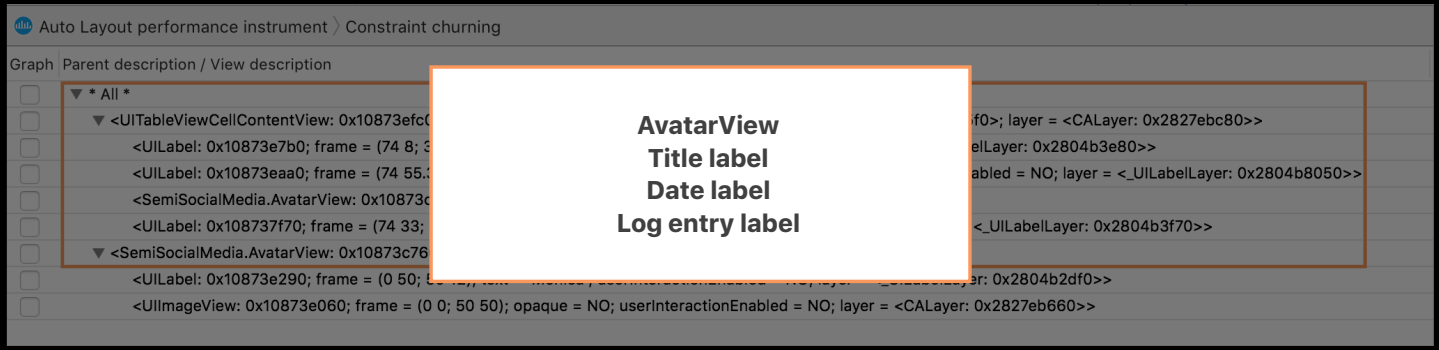


Finding the Problem

처음에 만든 뷰 들

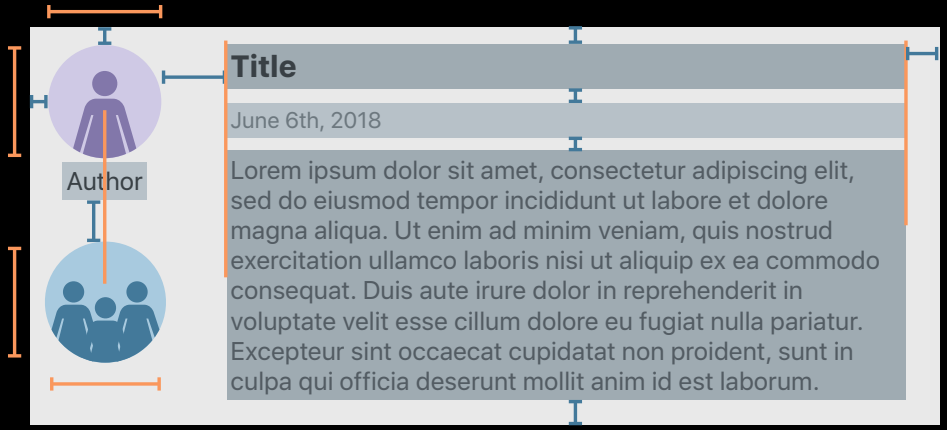


Finding the Problem



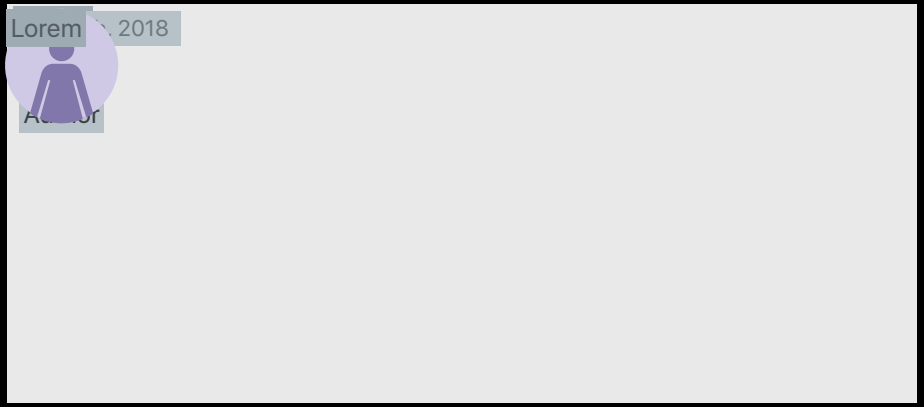
Building a More Performant Layout

대체 뭘이와



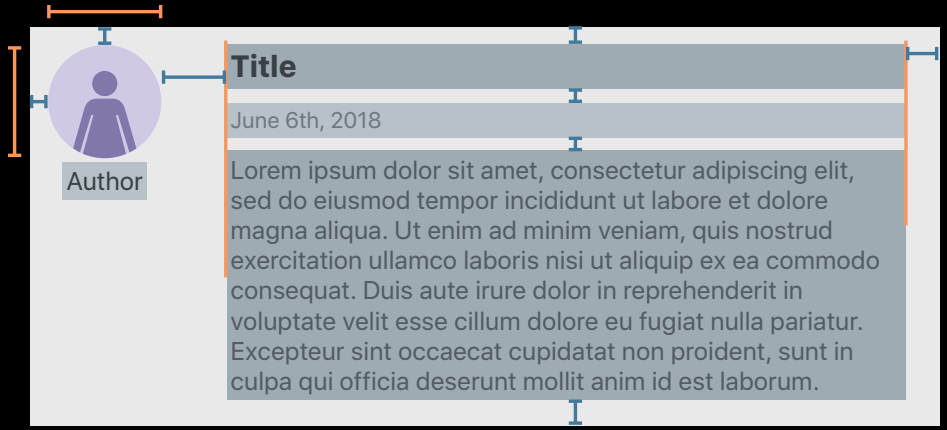
Building a More Performant Layout

제약조건이 없다면

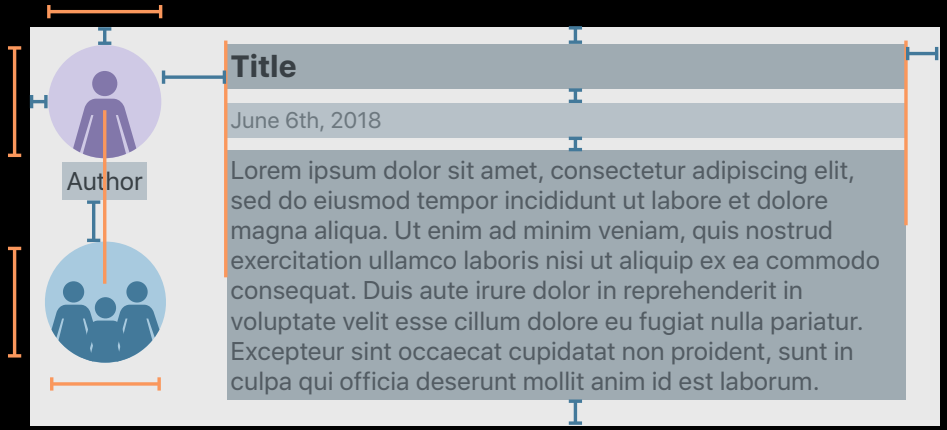


Building a More Performant Layout

그리고 예시 레이아웃

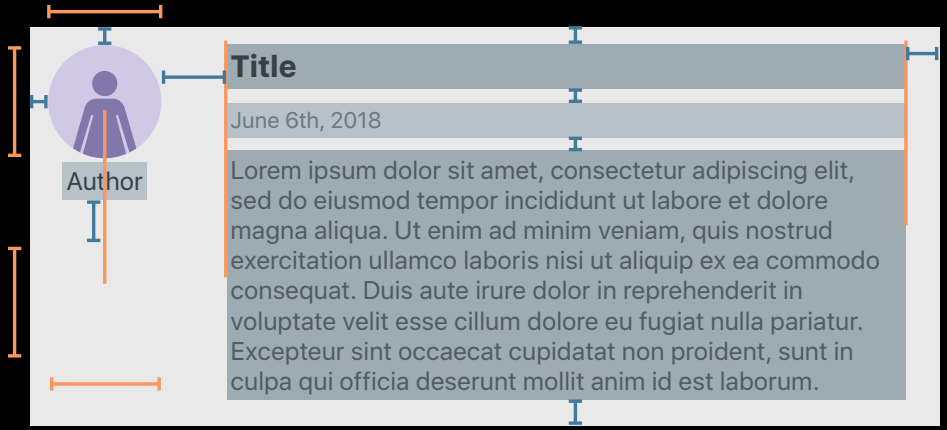


Building a More Performant Layout



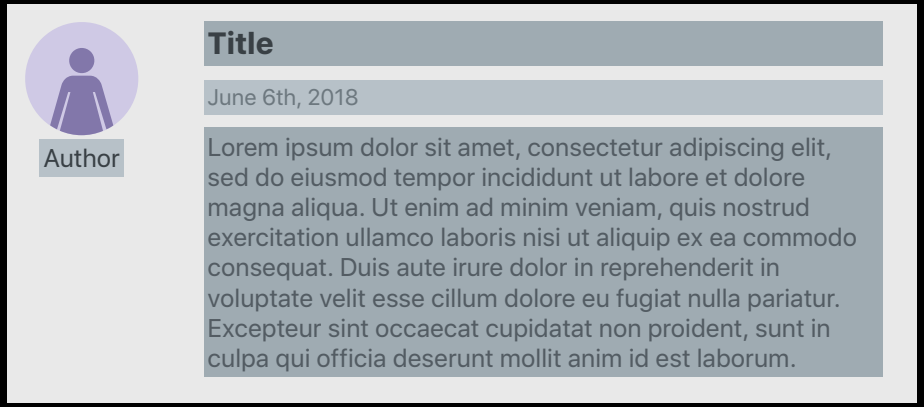
Building a More Performant Layout

그냥 setHidden을 써서



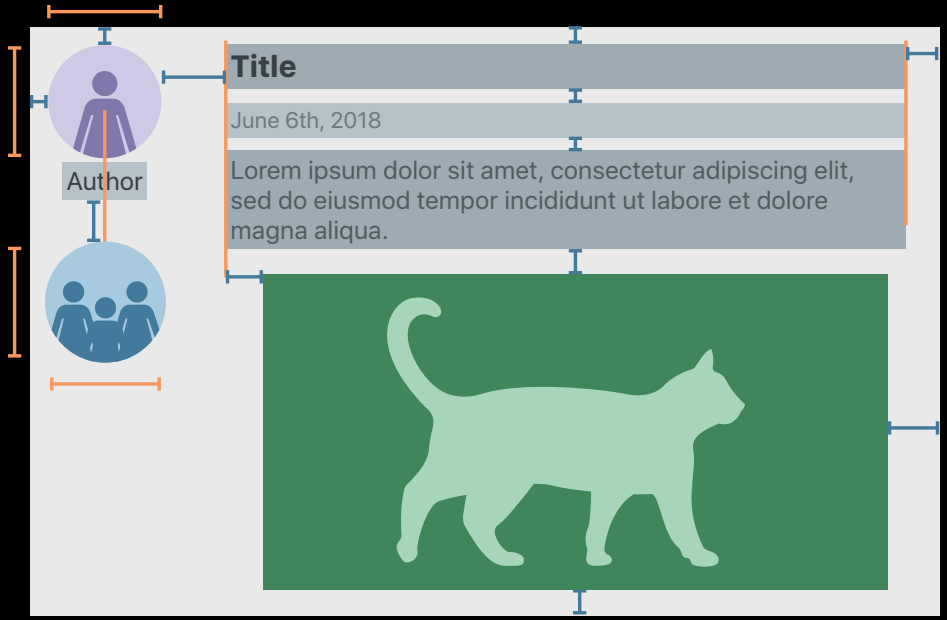
Building a More Performant Layout

모두 제각이

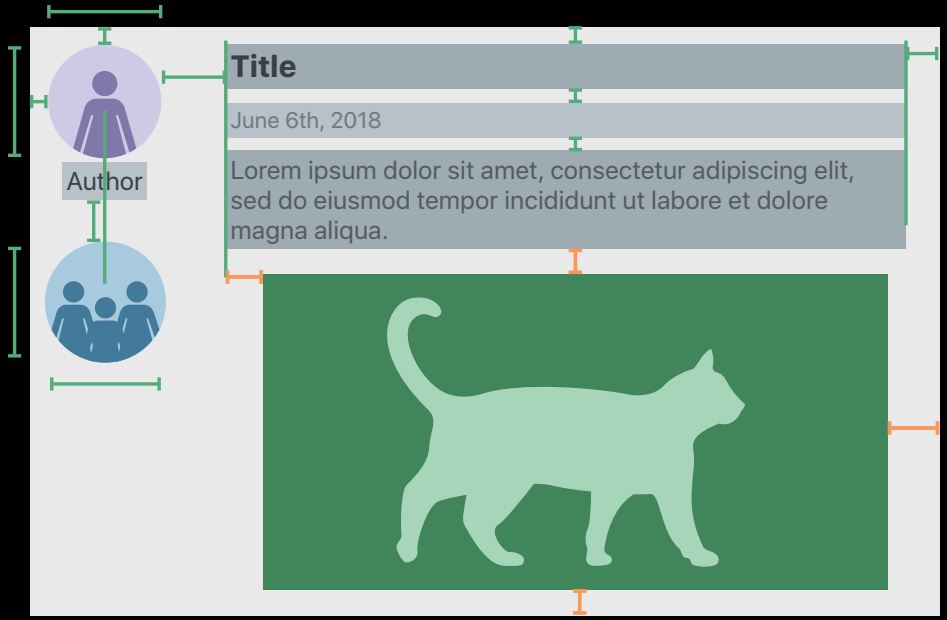


Building a More Performant Layout

그림이전?

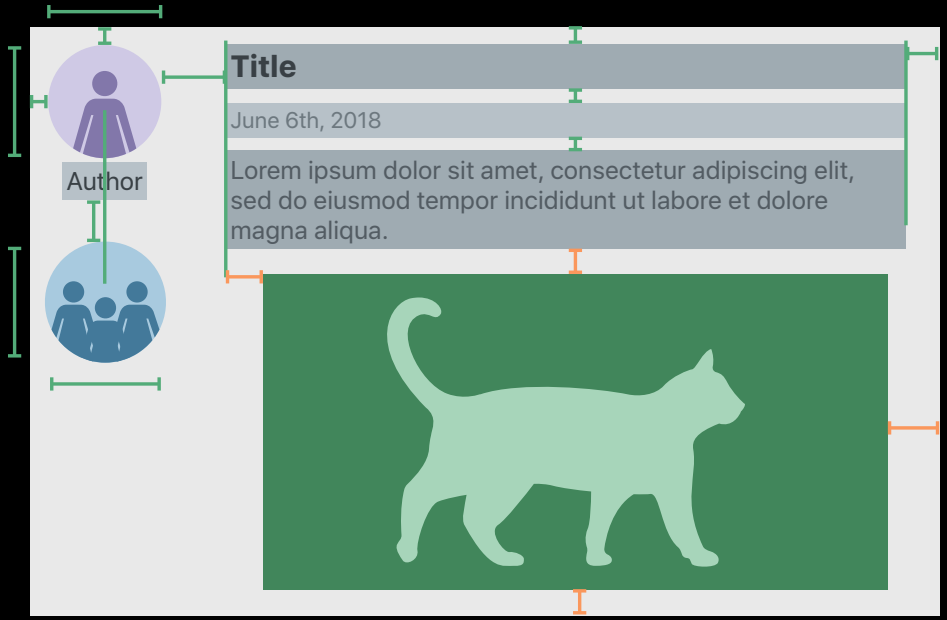


Building a More Performant Layout



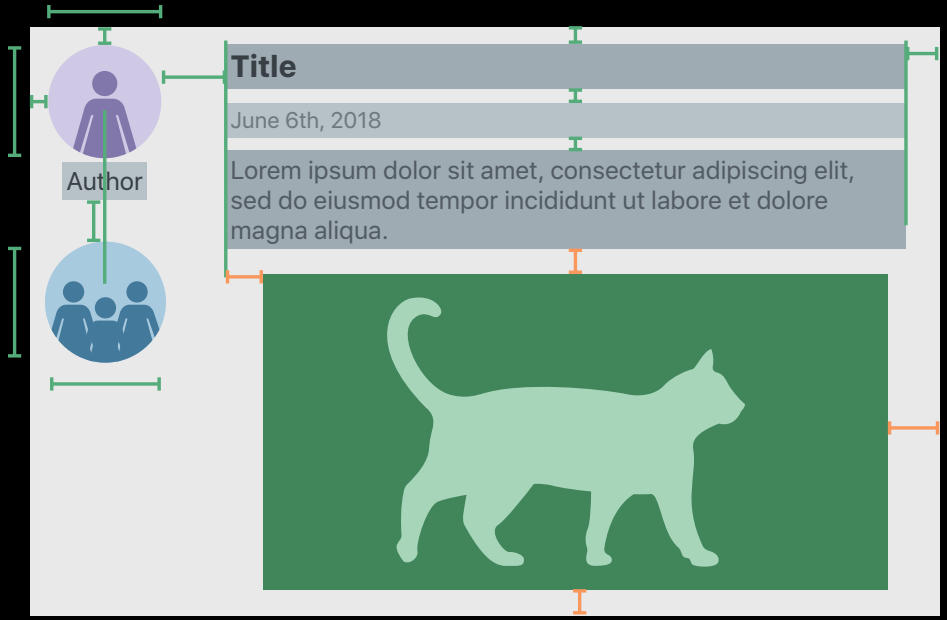
초록색은 건드릴 필요 없다

Building a More Performant Layout



imageConstraints =

Building a More Performant Layout

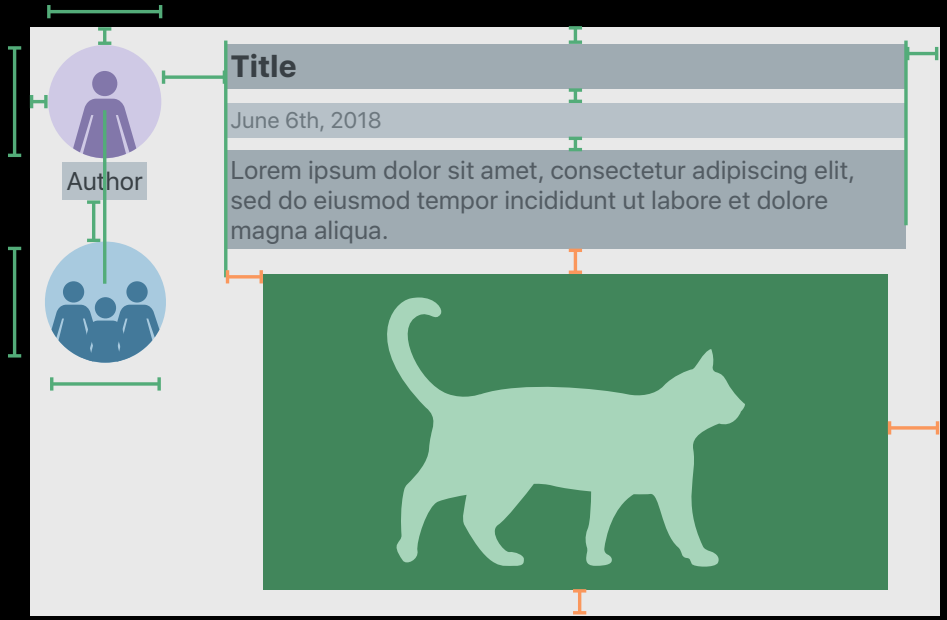


imageConstraints =

[⊢, I, ⊢, I]

이것들은 해뒀고

Building a More Performant Layout



imageConstraints =

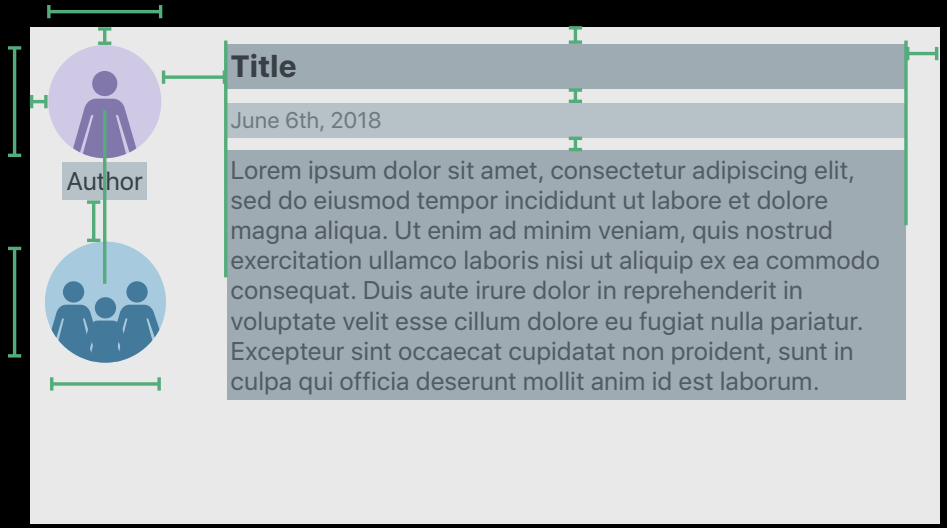
[⊢, I, ⊢, I]

noImageConstraints =

[I]

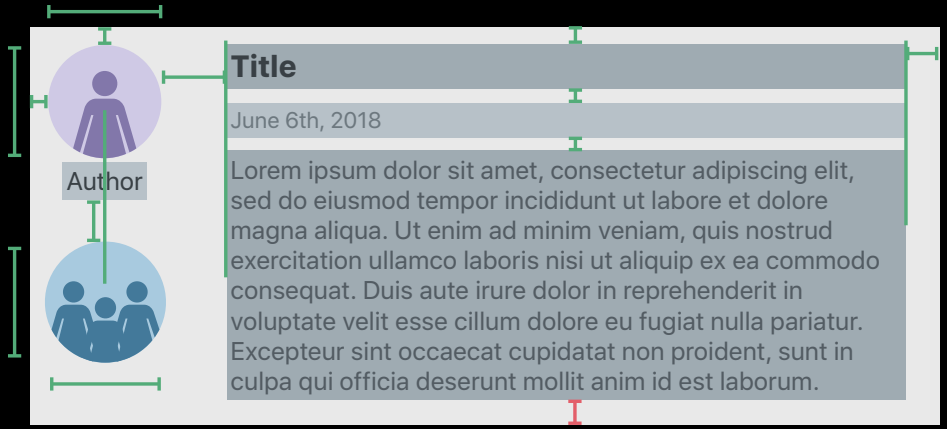
이미지 없을때 레이아웃 만들어줌

Building a More Performant Layout



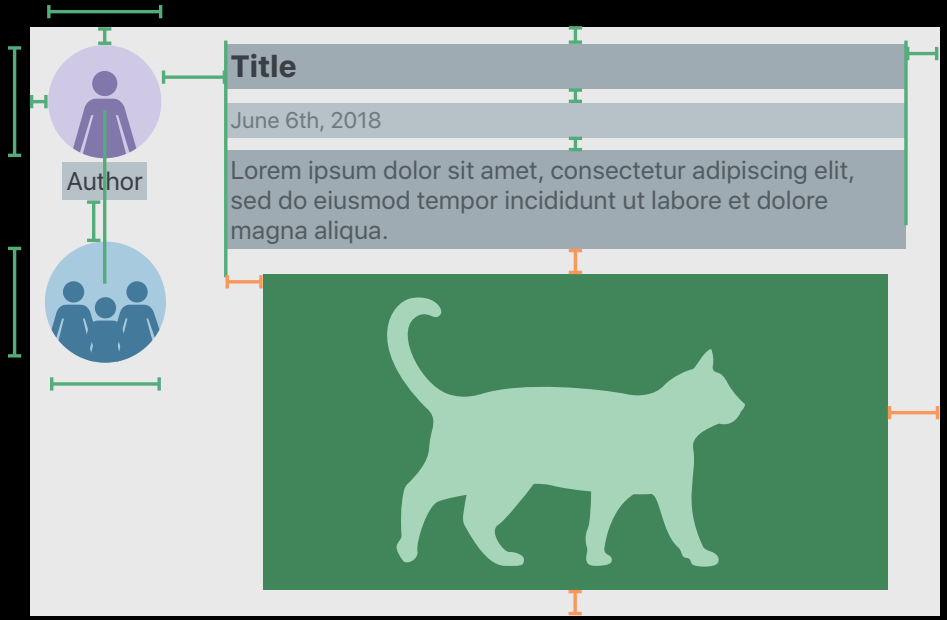

```
imageConstraints =
[ ⊢, I, ⊢, I ]

noImageConstraints =
[ I ]
```

Building a More Performant Layout




```
imageConstraints = [ |—|, I, |—|, I ]

noImageConstraints = [ I ]
```

Building a More Performant Layout



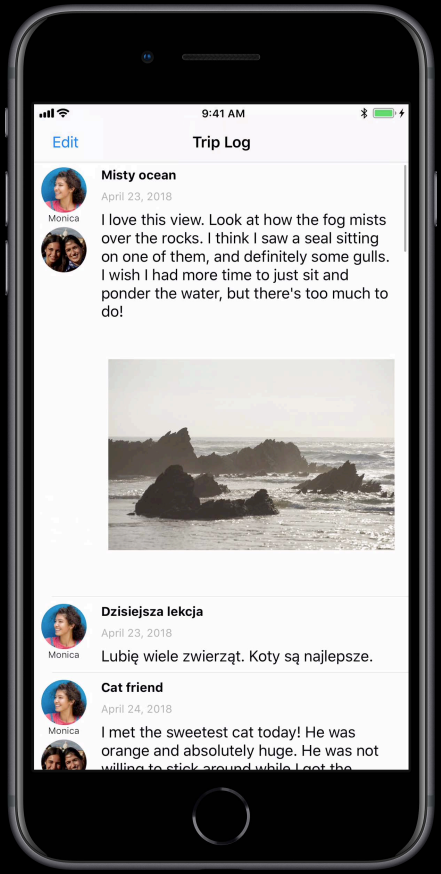
imageConstraints =

[⊢, I, ⊢, I]

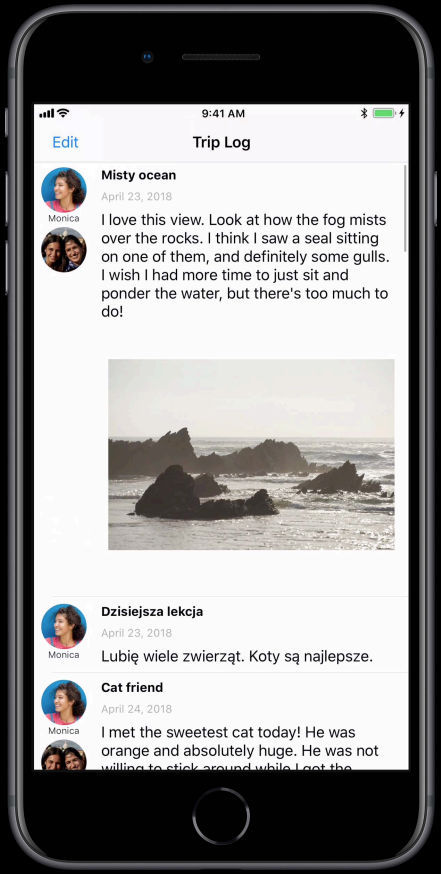
noImageConstraints =

[I]

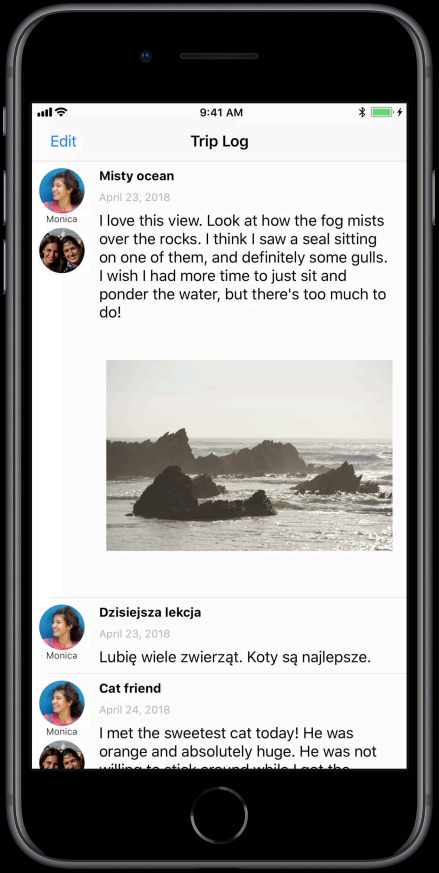
Before


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

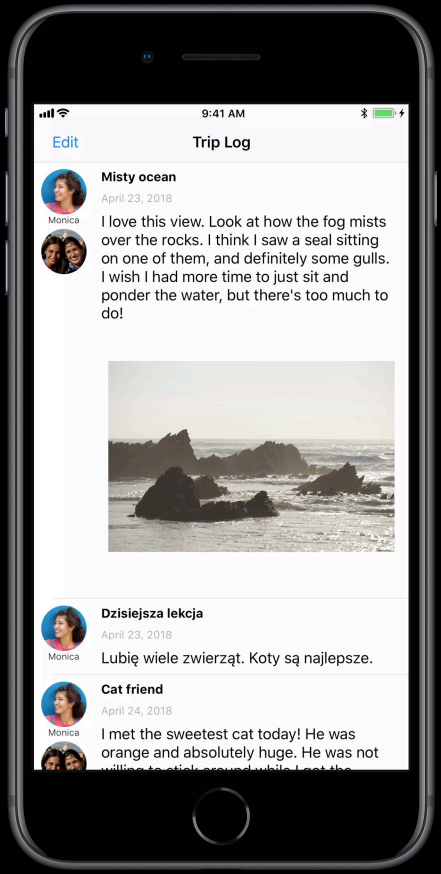
Before


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

After

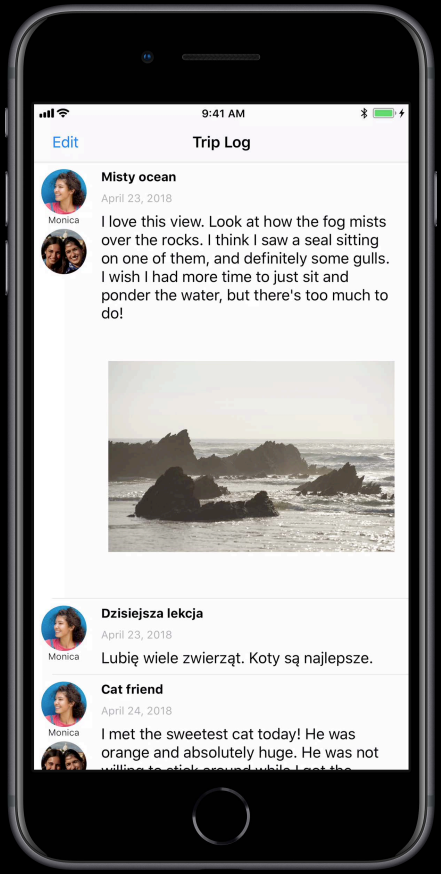

9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

After

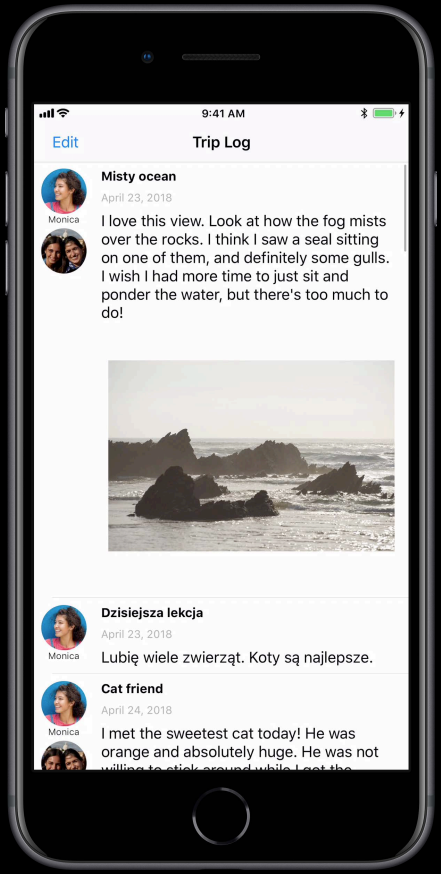

9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

더 좋다.

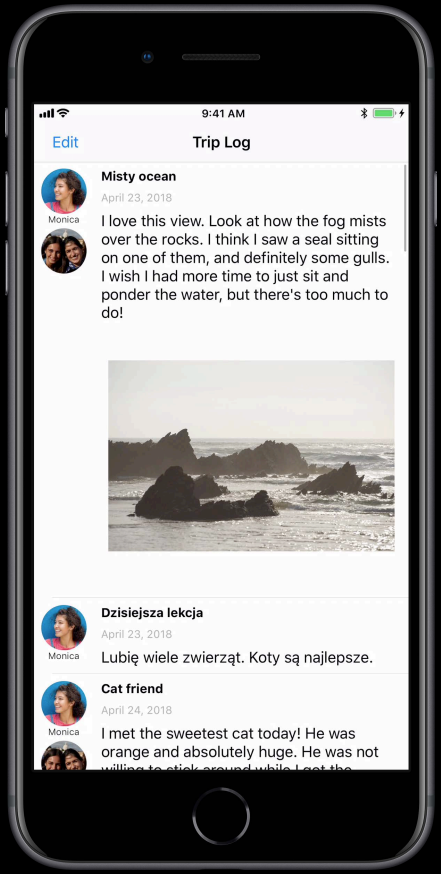
After (iOS 11)


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

After (iOS 12)


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

After (iOS 12)


9:41 AM
Edit
Trip Log
Misty ocean
April 23, 2018
Monica
I love this view. Look at how the fog mists over the rocks. I think I saw a seal sitting on one of them, and definitely some gulls. I wish I had more time to just sit and ponder the water, but there's too much to do!
Dzisiejsza lekcja
April 23, 2018
Monica
Lubię wiele zwierząt. Koty są najlepsze.
Cat friend
April 24, 2018
Monica
I met the sweetest cat today! He was orange and absolutely huge. He was not

Constraint Churn

Avoid removing all constraints

Add static constraints once

Only change the constraints that need changing

Hide views instead of removing them

Intrinsic Content Size

Not all views need it

Views with non view content:

- UIImageView returns its image size
- UILabel returns its text size

UIView uses it to make constraints

콘텐츠 가지고 있어서 제약생김

Overriding

text 내용으로 크기 계산은 비싼 작업이다.

Optionally override for text measurement

- Return size if known without text measurement
- Use `UIView.noIntrinsicMetric` and constraints

크기를 이미 안다면 내부적 계산을 안해야 할것

Overriding

Optionally override for text measurement

- Return size if known without text measurement
- Use `UIView.noIntrinsicMetric` and constraints

text content size 같이 측정하지마

```
override var intrinsicContentSize: CGSize {
    return CGSize(width: UIView.noIntrinsicMetric, height: UIView.noIntrinsicMetric)
}
```

System Layout Size Fitting Size

```
systemLayoutSizeFitting(_ targetSize:)
```

Used when combining with frames

사용이 많지는 않지만
이것을 응용해서 사용하기도 한다.

Getting Size from the Engine

Engine is created

Constraints are added

Layout is solved

Size of the top view is returned

Engine is discarded

Getting Size from the Engine

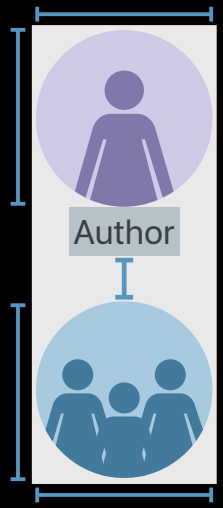
Engine is created

Constraints are added

Layout is solved

Size of the top view is returned

Engine is discarded



Getting Size from the Engine

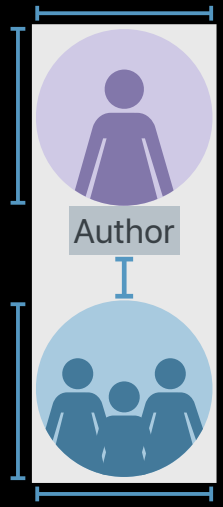
Engine is created

Constraints are added

Layout is solved

Size of the top view is returned

Engine is discarded



Getting Size from the Engine

Engine is created

Constraints are added

Layout is solved

Size of the top view is returned

Engine is discarded

```
{50,120}
```

Unsatisfiable Constraints

There is no solution for the specified constraints

Logging will help you to debug

Can mask other issues

See related session for debugging tips

Ready, Set, Go!

Think before you update constraints

Tune as needed

Enjoy your faster layouts

More Information

https://developer.apple.com/wwdc18/220

UIKit and Layout Lab	Technology Lab 2	Thursday 11:00AM


WWDC18