

NHN NEXT
iOS Advanced

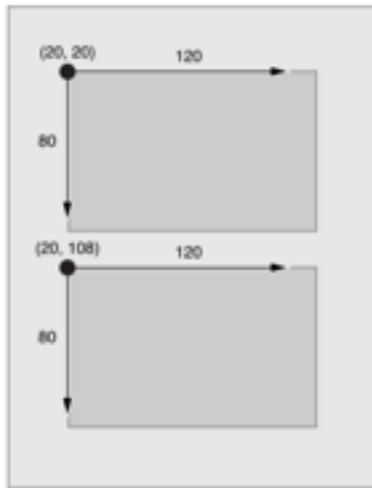
Auto Layout 이란?

- View hierarchy 안 모든 view의 크기와 위치를 **constraint**를 기반으로 **동적으로** 계산함
- constraint 기반 접근법 사용
 - 외부 변화: 슈퍼뷰의 사이즈나 모양 변화
 - iOS: split view, 기기 회전, 전화나 녹음 등 bar 변화
 - OSX: 윈도우 사이즈 조절
 - 내부 변화: 뷰의 크기나 UI 컨트롤 변화
 - 앱의 콘텐츠 화면 변화 e.g. 뉴스 앱의 기사 사이즈, 사진 앨범의 사진 사이즈
 - 여러 언어 지원 e.g. 영어 label -> 한국어 label, 통화 변경, 일본어 읽기 방식 변화
 - Dynamic type 지원 e.g. 사용자의 앱 실행 중 폰트 사이즈 변경

UI 레이아웃 설정 방법

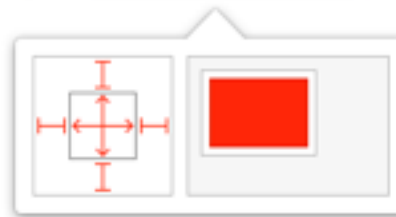
	하드코딩	Autoresizing Mask	Auto Layout
장점	프로그래머의 자율성이 높음, 원하는대로 설정이 가능	슈퍼뷰 프레임의 변화에 따라 뷰의 프레임을 정의해서 외부 변화에 적용 가능	프레임 대신 constraint를 통 한 관계 중심 으로 파악해서 내 부 외부 변화에 동적으로 적용
단점	모든 변화를 직접 파악해서 처 리해야 함	전체 레이아웃 중에서 지원하 는 범위가 작음	익숙해지기 어렵고 디버깅이 어려움

하드코딩

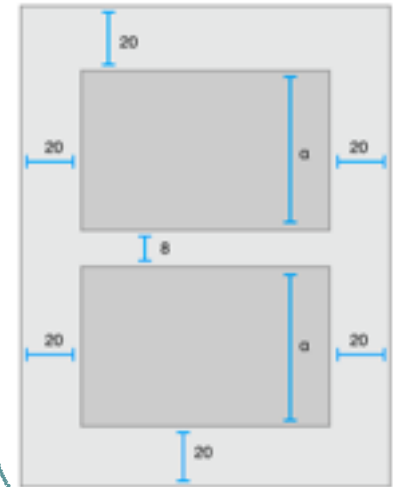


Autoresizing mask 사

```
[view setAutoresizingMask:(UIViewAutoresizing)];
```



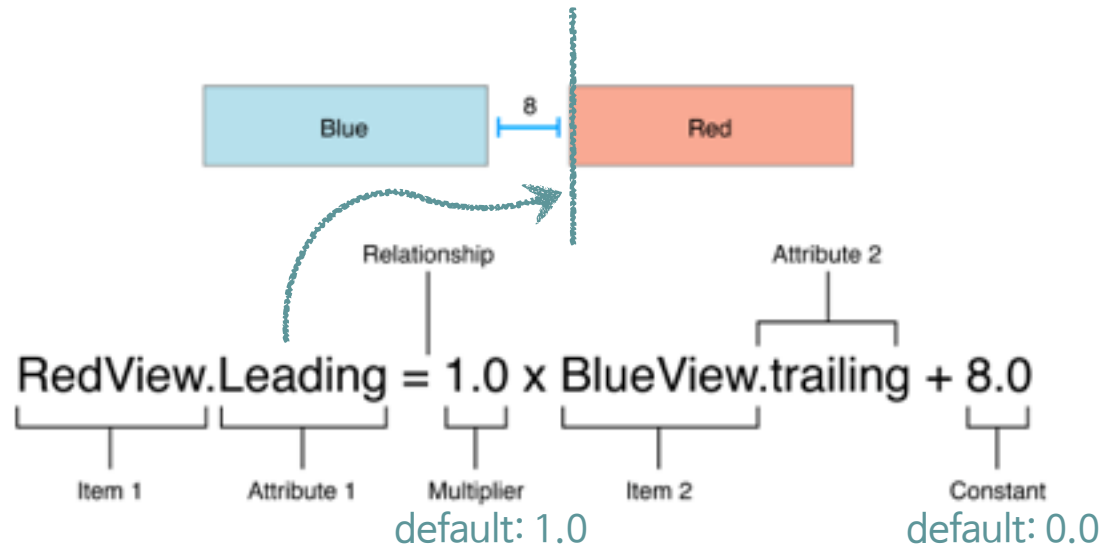
Auto Layout 사용



〈Frame-Based Layout〉

〈Auto Layout〉

Constraint



방정식처럼 계산

- Item 1: 뷰나 레이아웃 가이드로 계산하려는 아이템
- Attribute 1: Item 1의 Constraint로 계산하려는 목표값
- Relationship: 방정식 좌항과 우항의 관계 - 동등, 이하, 이상
- Multiplier: 좌항에 곱해지는 배수
- Item 2: 계산의 기준으로 삼은 아이템, blank일 수 있음
- Attribute 2: Item 2의 Constraint, Item 2가 blank일 경우 Attribute가 아니어야 함
- Constant: floating-point 오프셋 상수 값
- 하드코딩:
 - `myView.leadingAnchor.constraintEqualToAnchor(margins.leadingAnchor).active = true`

Auto Layout Attributes





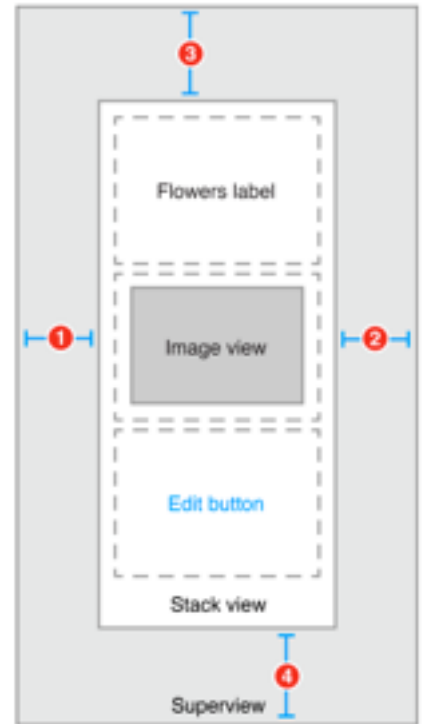
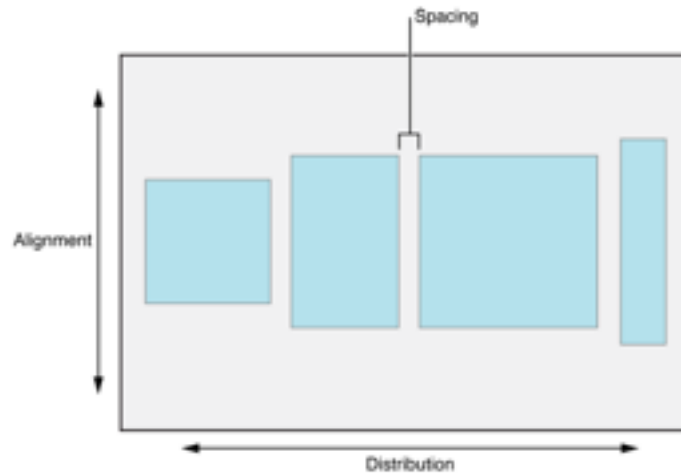
Attributes	value	설명
 Heights  Width	뷰의 크기	기본 설정. target task에 새로운 Activity 인스턴스를 생성하고 intent를 보냄, stack 어디든 놓일 수 있음
 Top  Bottom  Baseline	스크린 아래로 갈 수록 커짐	Center Y, Top, Bottom, Baseline과만 묶어서 사용 가능
 Leading  Trailing	화면 끝으로 갈 수록 커짐 (reading direction이 좌->우 방향일 경우 우측으로 갈 수록 커짐)	Leading, Trailing, Center X와만 묶어서 사용 가능
 Left  Right	우측으로 갈 수록 커짐	Left, Right, Center X과만 묶어서 사용 가능 view의 reading 방향이 나라별로 다를 수 있으므로 사용 비권장, Leading, Trailing 대체 사용 권장
 Center X  Center Y	방정식의 다른 Attribute에 좌우됨	Center X는 Center X, Leading, Trailing, Right, Left과, Center Y는 Center Y, Top, Bottom, Baseline과 함께 사용 가능

표 (영어원문) 출처:

https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/AutolayoutPG/AnatomyofaConstraint.html#//apple_ref/doc/uid/TP40010853-CH9-SW1

Constraint 없이 사용하기 : Stack View



- Stack View: constraints 없이 레이아웃 배치 가능
 - axis: 스택 뷰의 가로 혹은 세로 정렬 방향 결정 (UIStackView)
 - orientation: 스택 뷰의 가로 혹은 세로 정렬 방향 결정 (NSStackView)
 - distribution: 축에 맞춘 내부 뷰의 레이아웃 결정
 - alignment: 축과 수직 방향의 정렬 결정
 - spacing: 이웃 뷰와의 거리 결정

몹시 심플한 예제: <https://github.com/luvgaram/stackViewSimpleExample>

출처: https://developer.apple.com/library/ios/documentation/UIKit/Reference/UIStackView_Class_Reference/index.html#//apple_ref/doc/uid/TP40015256

Auto Layout Debugging (1)



- 미완성 레이아웃 (Unsatisfiable Layouts): constraint 세트 중 **정확한 값이 없을 경우**
 e.g. 동시에 충족될 수 없는 값이 두 개 이상 설정됨
 -> UIAlertViewForUnsatisfiableConstraints 브레이크 포인트 설정
- 모호한 레이아웃 (Ambiguous Layouts): **정확한 값이 두 개 이상** 있어서 결정할 수 없는 경우
 -> view hierarchy에 접근할 수 있는 곳에 브레이크 포인트를 설정하고 콘솔에 아래 메서드 사용
 - hasAmbiguousLayout
 - exerciseAmbiguityInLayout
 - constraintsAffectingLayoutForAxis: (iOS)
 - constraintsAffectingLayoutForOrientation: (OSX)
- 논리적 에러: 찾기 어려움

Auto Layout Debugging (2)

label의 leading edge가 슈퍼뷰의 leading margin과 같음

2015-08-26 14:27:54.790 Auto Layout Cookbook[10040:1906606] Unable to simultaneously satisfy constraints.

Probably at least one of the constraints in the following list is one you don't want. Try this: (1) look at each constraint and try to figure out which you don't expect; (2) find the code that added the unwanted constraint or constraints and fix it. (Note: If you're seeing `NSAutoresizingMaskLayoutConstraints` that you don't understand, refer to the documentation for the `UIView` property `translatesAutoresizingMaskIntoConstraints`)

label의 width가 400포인트 이상

```
(
  "<NSLayoutConstraint:0x7a87b000 H:[UILabel:0x7a8724b0'Name'(>=400)]>",
  "<NSLayoutConstraint:0x7a895e30 UILabel:0x7a8724b0'Name'.leading ==
  UIView:0x7a887ee0.leadingMargin>",
  "<NSLayoutConstraint:0x7a886d20 H:[UILabel:0x7a8724b0'Name']-
  (NSSpace(8))->[UITextField:0x7a88cff0]>",
  "<NSLayoutConstraint:0x7a87b2e0 UITextField:0x7a88cff0.trailing == UIView:
  0x7a887ee0.trailingMargin>",
  "<NSLayoutConstraint:0x7ac7c430 'UIView-Encapsulated-Layout-Width' H:
  [UIView:0x7a887ee0(320)]>"
)
```

슈퍼뷰의 width가 320포인트 (시스템이 설정)

Will attempt to recover by breaking constraint

```
<NSLayoutConstraint:0x7a87b000 H:[UILabel:0x7a8724b0'Name'(>=400)]>
```

Make a symbolic breakpoint at `UIViewAlertForUnsatisfiableConstraints` to catch this in the debugger.

The methods in the `UIConstraintBasedLayoutDebugging` category on `UIView` listed in `<UIKit/UIView.h>` may also be helpful.

시스템은 constraint 조건 충족을 위해 label의 width를 따르지 않을 것임

text field의 trailing edge가 슈퍼뷰의 trailing margin과 같음

Auto Layout Debugging (3)



- 로그에 Identifier 설정: 하드코딩하거나 Attribute Inspector에서 **identifier** 프로퍼티 설정
 - "`<NSLayoutConstraint:0x7b58bac0 'Label Leading' UILabel:0x7b58b040'Name'.leading == UIView:0x7b590790.leadingMargin>`", "`<NSLayoutConstraint:0x7b56d020 'Label Width' H:[UILabel:0x7b58b040'Name'](>=400)>`]"
- 뷰와 Constraint를 눈으로 확인하기
 - 시뮬레이터에서 앱 실행 후 XCode에서 Choose Debug > View Debugging > Show Alignment Rectangles
- View Debugger 사용
 - Xcode debug bar의 Debug View Hierarchy button  클릭
- private iOS method (release build에서는 사용 불가)
 - `_autolayoutTrace` 사용 (모호한 레이아웃 추적 시):
에러난 곳에서 lldb에 `po [[UIWindow keyWindow] _autolayoutTrace]`; 입력
 - `recursiveDescription: superview` (프로퍼티 포함) 추적
 - `_ivarDescription / _methodDescription`: 객체의 변수와 값 / 메서드를 모두 보여줌
 - `[UIViewController _printHierarchy] / [UIViewController _allDescriptions]`: 위계 질서 출력

출처: https://developer.apple.com/library/ios/documentation/UserExperience/Conceptual/AutolayoutPG/DebuggingTricksandTips.html#//apple_ref/doc/uid/TP40010853-CH21-SW1

<http://swiftiostutorials.com/using-private-undocumented-ios-methods-debugging/>

Auto Layout Guideline(1)

- 뷰의 프레임, 바운드나 센터 속성으로 뷰의 위치를 정하지 말라.
- constraint 로직을 줄여주는 stack view를 적극 활용하라.
- 뷰와 가장 가까운 이웃과 constraint를 설정하라.
- 이웃한 두 개의 버튼이 있다면 첫 버튼의 trailing edge로부터 두번째 버튼 leading edge의 constraint를 설정하라. 첫 버튼을 건너 뛰어서 뷰의 가장자리로부터 constraint를 만들지 마라.
- 가급적 뷰의 높이나 너비를 고정하는 것을 피하라.
- Auto Layout을 사용하는 이유는 변화에 동적으로 반응하기 위함이다. 고정 크기를 사용하면 뷰의 반응성을 낮춘다. 필요시에는 뷰의 최소 크기나 최대 크기를 설정하라.
- constraint 설정이 어렵다면 Pin과 Align 도구를 사용하라. control+드래그보다 느리지만 정확한 값과 관련 아이템을 설정하고 확인할 수 있다.
- 아이템의 프레임이 자동으로 갱신되는 경우를 주의하라. 아이템의 크기와 위치가 충족되는 constraint가 없다면 갱신이 되지 않을 수 있다. 화면 밖에 위치한 뷰의 높이나 너비가 0이 돼서 사라지는 경우도 발생할 수 있다.
- 자유롭게 아이템 프레임을 갱신할 수 있으며 갱신 내용을 취소할 수도 있다.
- 레이아웃 내의 모든 뷰가 의미 있는 이름을 갖도록 하라. 도구 사용시 뷰를 식별하는데 도움이 된다.
- 시스템은 자동으로 텍스트나 타이틀을 통해 라벨과 버튼의 이름을 만든다. 다른 뷰의 경우 Identity inspector에서 설정하거나 document outline 상의 뷰를 더블 클릭해서 이름을 만들 수 있다.
- right과 left 대신 leading과 trailing constraint를 사용하라.
- iOS의 semanticContentAttribute 프로퍼티나 OSX의 userInterfaceLayoutDirection 프로퍼티를 통해 뷰가 leading과 trailing edge를 계산하는 방법을 조정할 수 있다.

Auto Layout Guideline(2)

- iOS에서 뷰 컨트롤러의 루트 뷰 가장자리로부터 constraint를 만들 때 아래 constraint를 사용하라.
 - Horizontal constraints: layout margin으로부터 0 포인트 constraint를 사용하라. 시스템은 디바이스와, 앱이 뷰 컨트롤러를 보여주는 방식을 기반으로 자동으로 거리를 맞춘다.
margin을 포함해 루트 뷰를 채우는 텍스트 객체의 경우 layout margin 대신 가독성 있는 콘텐츠 가이드를 사용하라.
배경 이미지처럼 루트 뷰를 모두 채우는 아이템의 경우 뷰의 leading과 trailing edge를 사용하라.
 - Vertical constraints: 뷰가 bar 아래를 확장하는 경우 top과 bottom margin을 사용하라. 이는 스크롤 뷰에 흔히 사용되는 패턴이나, 콘텐츠의 초기 위치를 맞추려면 스크롤 뷰의 contentInset을 수정하고 scrollIndicatorInsets 프로퍼티를 수정해야 할 수 있다.
뷰가 bar 아래를 확장하지 않으면 레이아웃 가이드의 top과 bottom으로부터 constraint를 만들라.
- 코드에서 뷰를 생성할 경우, translatesAutoresizingMaskIntoConstraints 프로퍼티를 NO로 설정하라. 그렇지 않으면 기본 설정으로 뷰의 프레임과 autoresizing mask를 기반으로 시스템이 자동 constraint들을 만든다. 커스텀 constraint를 만들 경우 충돌이 발생하므로 미완성 레이아웃이 된다.
- OSX와 iOS가 레이아웃을 계산하는 방식이 다름에 유의하라.
- OSX에서는 Auto Layout으로 윈도우 콘텐츠와 윈도우 크기를 모두 수정할 수 있다.
- iOS에서는 시스템이 썬의 크기와 레이아웃을 결정하므로, Auto Layout으로 썬의 콘텐츠만을 수정할 수 있다.
- 이런 차이가 우선 순위 결정에 따라 레이아웃 디자인에 큰 영향을 미칠 수 있다.

1

Apple Developer: Auto Layout Guide

<https://developer.apple.com/library/prerelease/content/documentation/UserExperience/Conceptual/AutolayoutPG/>

2

iOS Tutorials

: Using private (undocumented) iOS methods for debugging

<http://swiftiostutorials.com/using-private-undocumented-ios-methods-debugging/>

3

Github : Masonry

<https://github.com/SnapKit/Masonry>



Thank
You