



IOS Autolayout

Lập trình giao diện thích ứng với nhiều kích thước màn hình
cuong@techmaster.vn

Đặt vấn đề

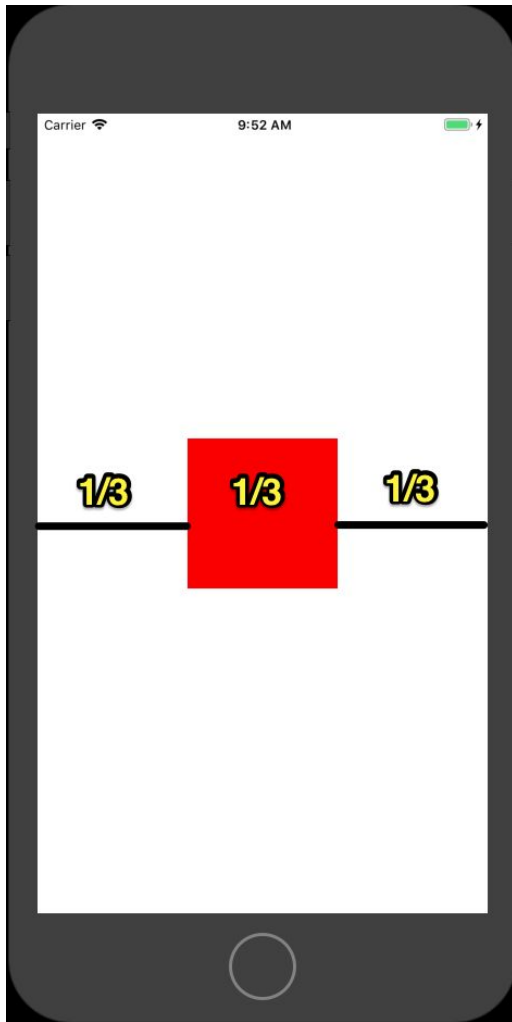
Tỷ lệ màn hình giữa đời iPhone khác nhau



```
import UIKit

class ViewController: UIViewController {
    override func viewDidLoad() {
        super.viewDidLoad()
        let screen_size = view.bounds.size
        print(
            "width = \(screen_size.width),
            height = \(screen_size.height),
            ratio = \(screen_size.height / screen_size.width)")
    }
}
```

	width	height	ratio
5S	320	568	1.775
7	375	667	1.778
8 Plus	414	736	1.777
X	375	812	2.165
XSMaX	414	896	2.164



Hãy vẽ một hình vuông,
nằm chính giữa tâm
màn hình iPhone, kích
thước bằng $\frac{1}{3}$ chiều rộng
của màn hình

```
var myView: UIView!
override func viewDidLoad() {
    super.viewDidLoad()
    let screen_size = view.bounds.size
    myView = UIView(frame: CGRect(x: 0, y: 0,
                                   width: screen_size.width * 0.3333,
                                   height: screen_size.width * 0.3333))

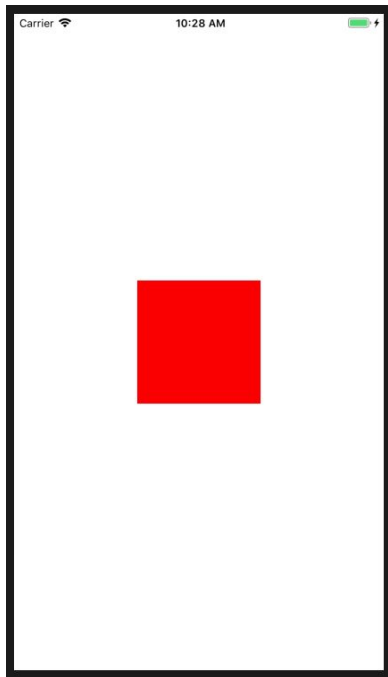
    myView.center = CGPoint(x: screen_size.width * 0.5,
                             y: screen_size.height * 0.5)

    myView.backgroundColor = UIColor.red
    view.addSubview(myView)
}
```

Hãy thử chuyển màn hình portrait sang landscape và ngược lại



Xem ô vuông màu đỏ có nằm chính giữa tâm màn hình iPhone không?

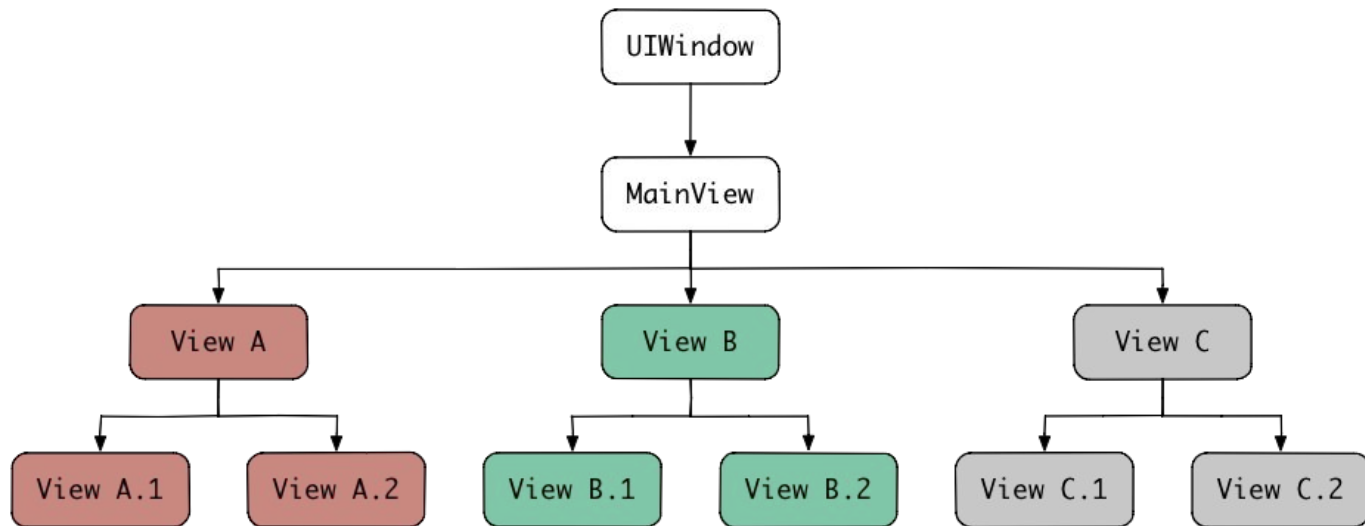
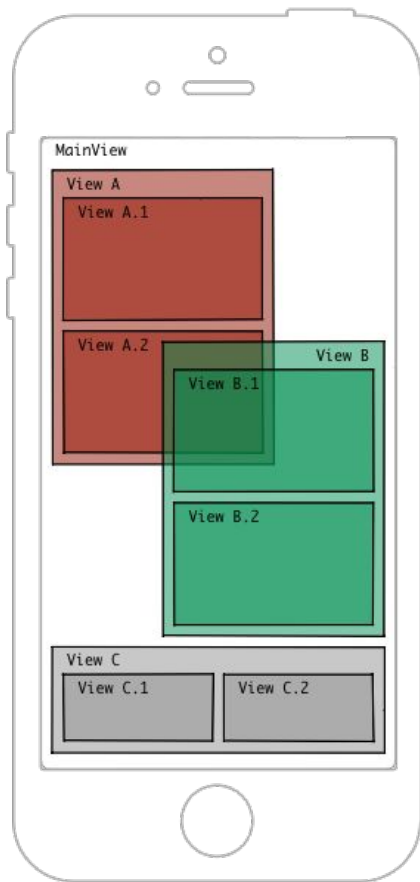


Layout bằng lập trình pixel coordinate



- Lấy kích thước width, height của màn hình iPhone thực tế
- Lập trình để tính toán tọa độ và kích thước từng phần tử
- Cố gắng tránh sử dụng các giá trị cố định, mà tính theo tỷ lệ $\text{ratio} = \text{height} / \text{width}$
- Sử dụng 2 sự kiện `viewWillLayoutSubviews` và `viewDidLayoutSubviews` để tính toán lại tọa độ, kích thước từng view con

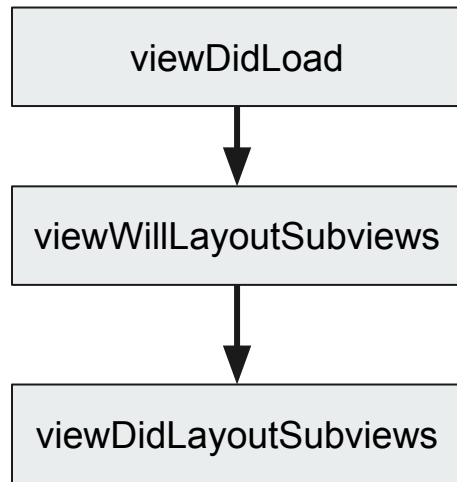
Khi có sự thay đổi kích thước màn hình (chuyển chế độ portrait sang landscape) cần phải tính toán lại tọa độ từng view con bên trong



viewWillLayoutSubviews, viewDidLayoutSubviews

```
import UIKit

class ViewController: UIViewController {
    var myView: UIView!
    override func viewDidLoad() {
        super.viewDidLoad()
        print("viewDidLoad")
    }
    override func viewWillLayoutSubviews() {
        print("viewWillLayoutSubviews")
    }
    override func viewDidLayoutSubviews() {
        print("viewDidLayoutSubviews")
    }
}
```



#1 chú ý hàm relayout

```
import UIKit

class ViewController: UIViewController {
    var myView = UIView(frame: CGRect.zero)
    override func viewDidLoad() {
        super.viewDidLoad()
        print("viewDidLoad")
        myView.backgroundColor = UIColor.red
        view.addSubview(myView)
    }

    func relayout() {
        let screen_size = view.bounds.size
        let min_size = CGFloat.minimum(screen_size.width, screen_size.height)
        myView.frame = CGRect(x: 0, y: 0, width: min_size * 0.3333, height: min_size * 0.3333)
        myView.center = CGPoint(x: screen_size.width * 0.5, y: screen_size.height * 0.5)
    }
}
```

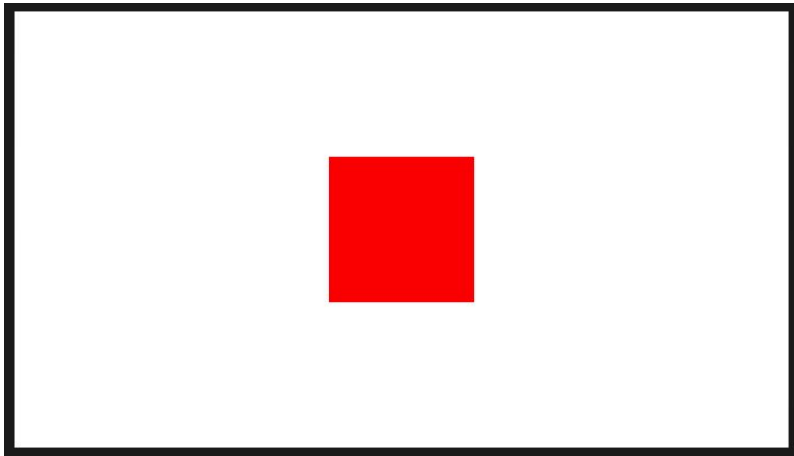
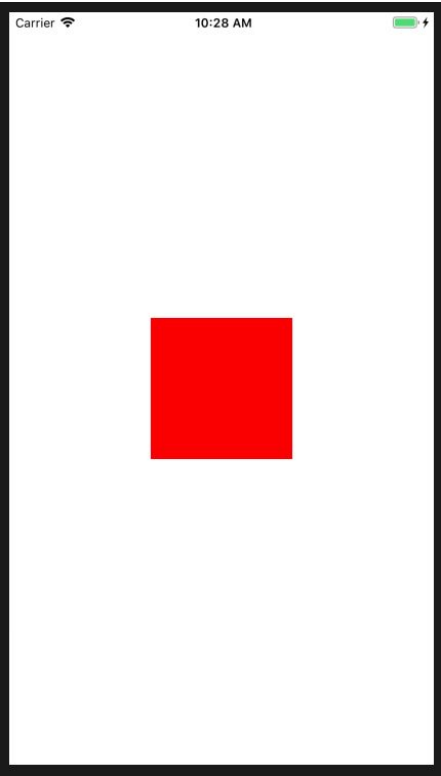
#2 relaylayout được gọi trong hàm ...LayoutSubviews

```
import UIKit

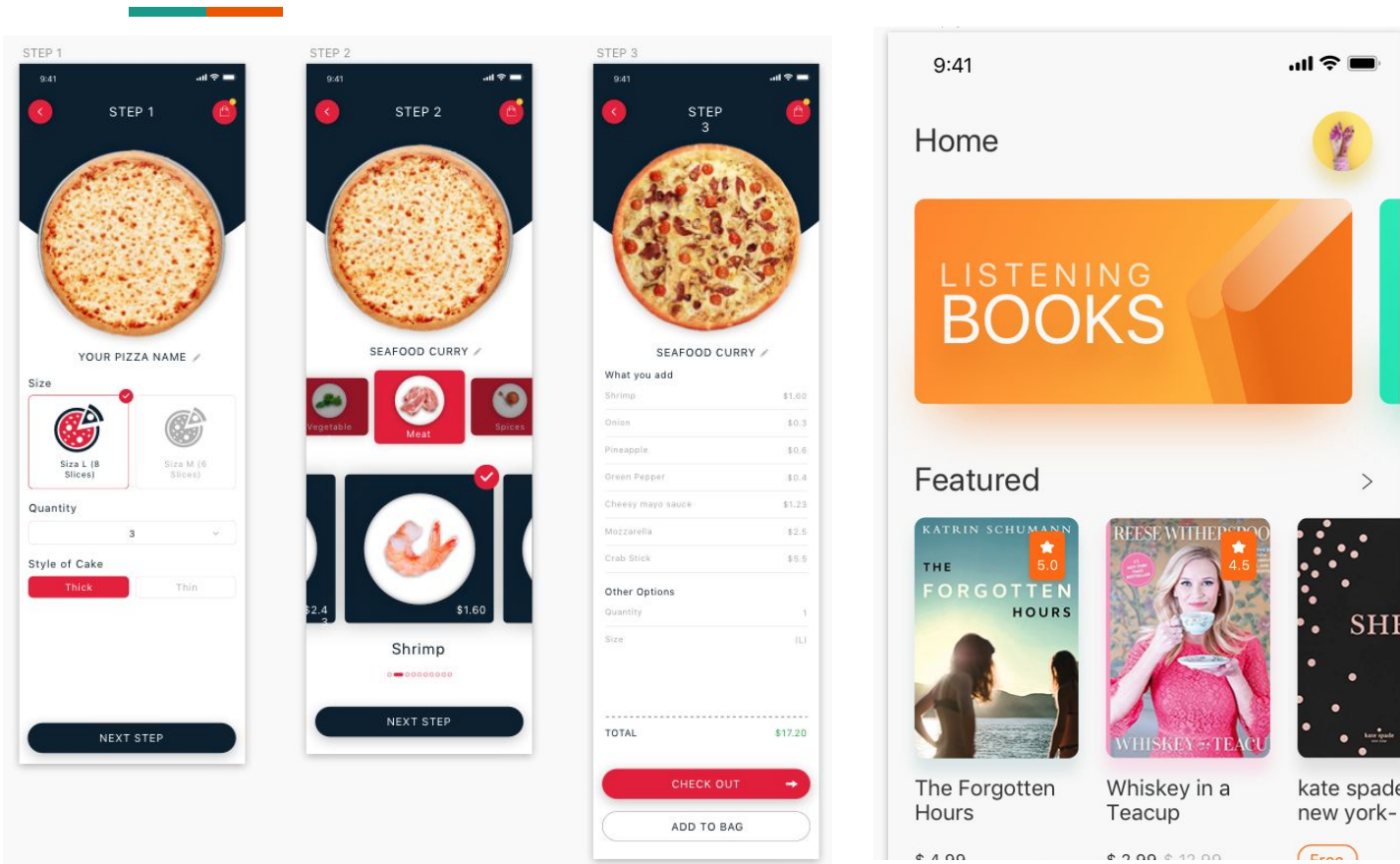
class ViewController: UIViewController {
    ...

    override func viewWillLayoutSubviews() {
        print("viewWillLayoutSubviews")
        relaylayout()
    }

    override func viewDidLayoutSubviews() {
        print("viewDidLayoutSubviews")
        relaylayout()
    }
}
```



Nếu giao diện phức tạp thì code tay không khả thi

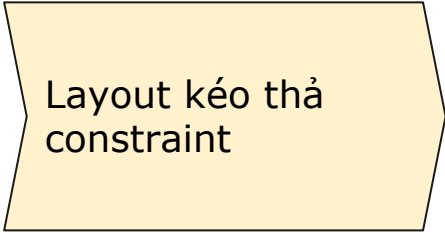


XCode Autolayout

Auto Layout



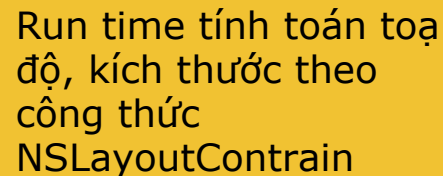
1. Giúp dev có thể kéo thả thành phần giao diện, đặt những ràng buộc (constraint) trực quan
2. Constraint trên giao diện khi biên dịch sẽ chuyển hoá thành tập các NSLayoutConstraint



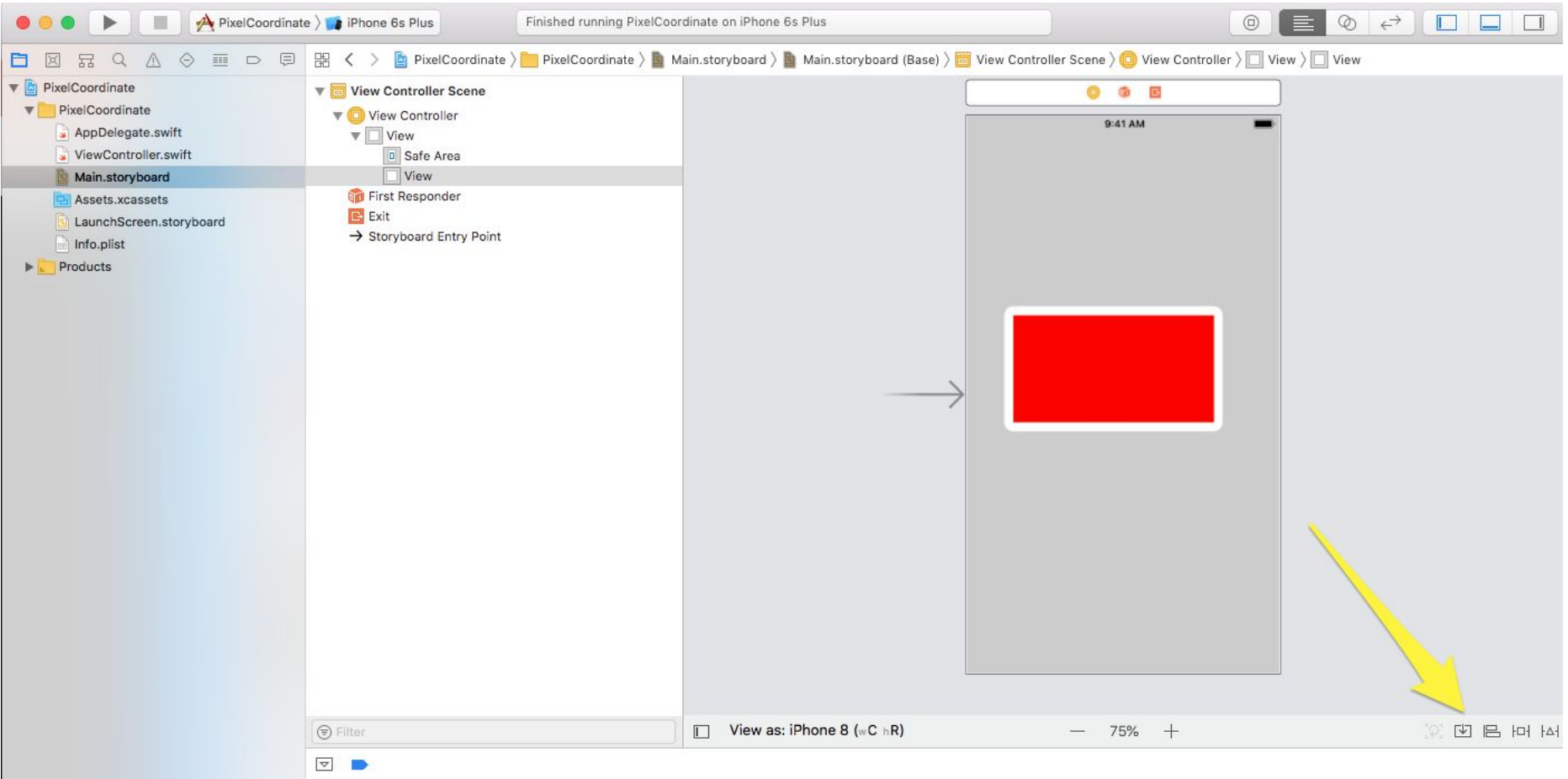
Layout kéo thả
constraint



Biên dịch sinh ra mã
NSLayoutConstraint



Run time tính toán toạ
độ, kích thước theo
công thức
NSLayoutConstraint

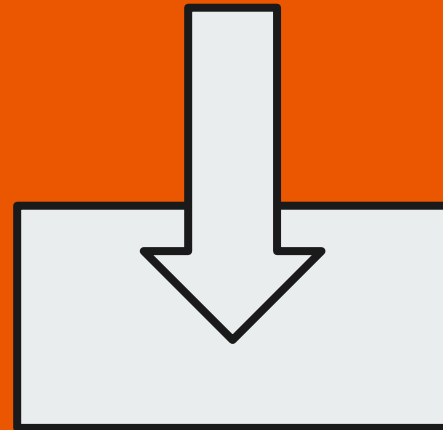


4 nhóm chức năng AutoLayout



1. Embed ~ Nhúng
 - a. Embed in UIView ~ Nhúng vào UIView
 - b. Embed in UIViewController ~ Nhúng vào UIViewController
2. Align
 - a. Edge ~ Cạnh (Trên - Dưới - Trái - Phải)
 - b. Center ~ Chính giữa (Ngang - Dọc)
3. Add New Constraint
 - a. Margin
 - b. Width, Height, Ratio
4. Resolve Autolayout Issues
 - a. Update
 - b. Add
 - c. Remove
 - d. Clear

Embed in \sim Nhúng

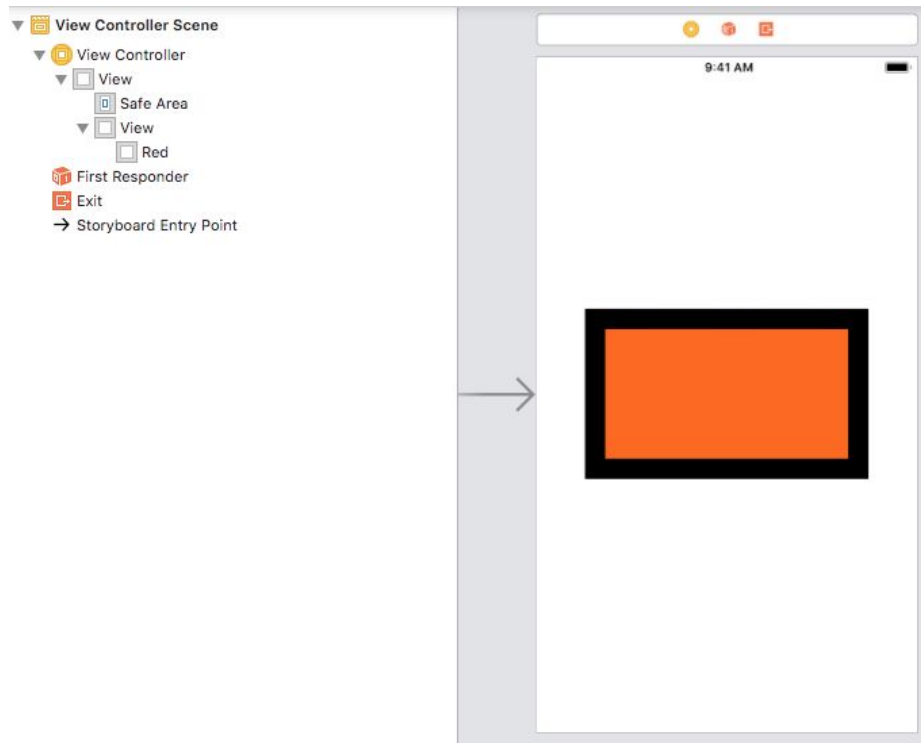


—

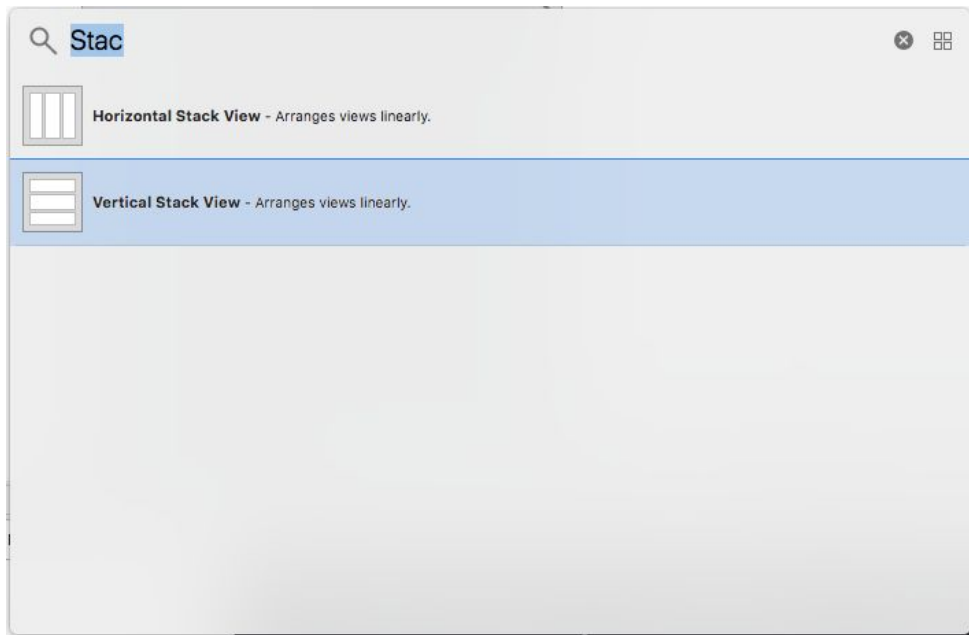
Embed in UIView, Embed in UIView without inset

Đơn giản chỉ là tạo một UIView mới bao lấy UIView đang được chọn. Tính năng này không có gì quá đặc biệt.

Dev có thể tự drag and drop

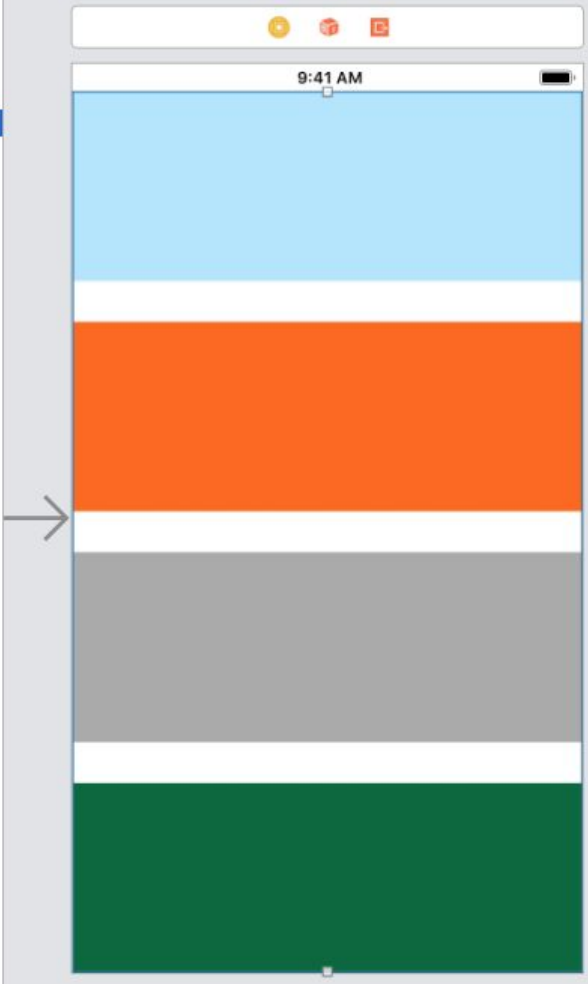


StackView



1. StackView để giúp dev làm gì?
2. Có mấy loại
3. Thuộc tính alignment

- View Controller Scene
 - View Controller
 - View
 - Safe Area
 - Stack View**
 - View
 - View
 - View
 - View
 - Constraints
 - First Responder
 - Exit
 - Storyboard Entry Point



Stack View

- + Axis: Vertical
- + Alignment: Fill
- + Distribution: Fill Equally
- + Spacing: 30
- + ☐ Baseline Relative

Fill Equally

View

Content Mode: Scale To Fill

Semantic: Unspecified

Tag: 0

Interaction: ☒ User Interaction Enabled, ☐ Multiple Touch

Alpha: 1

+ Background: Default

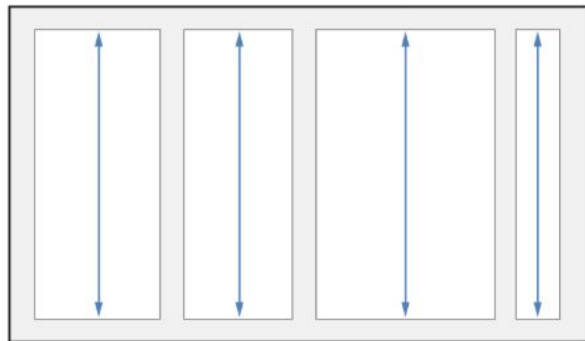
+ Tint: Default

Drawing: ☐ Opaque, ☐ Hidden, ☒ Clears Graphics Context, ☐ Clip to Bounds, ☒ Autoresize Subviews

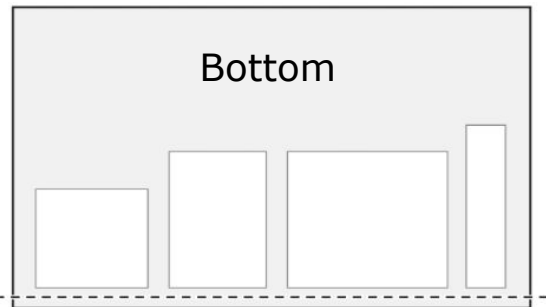
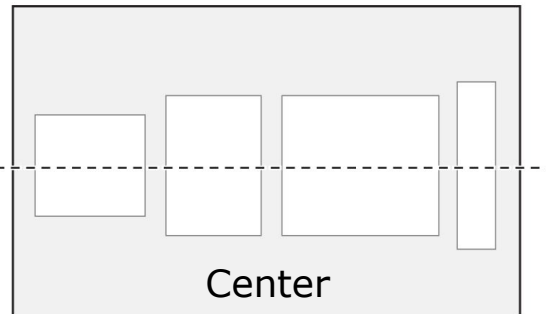
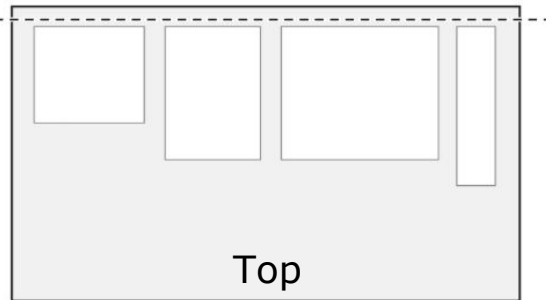
Stretching: X: 0, Y: 0, Width: 1, Height: 1

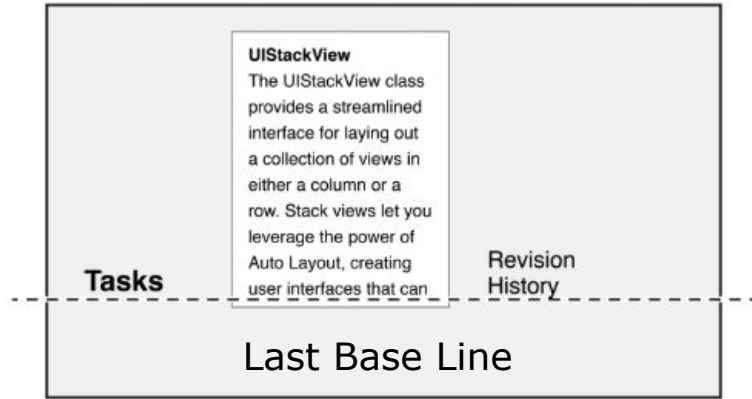
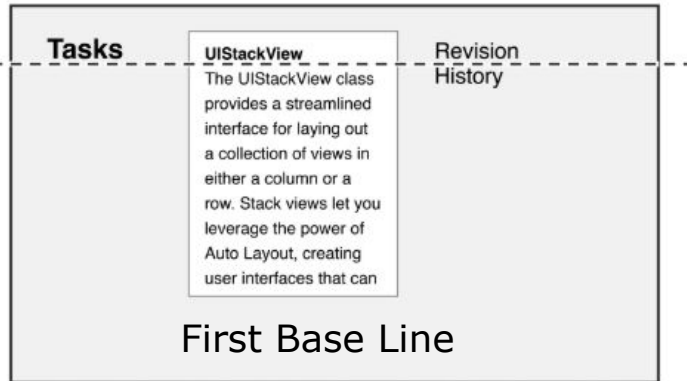
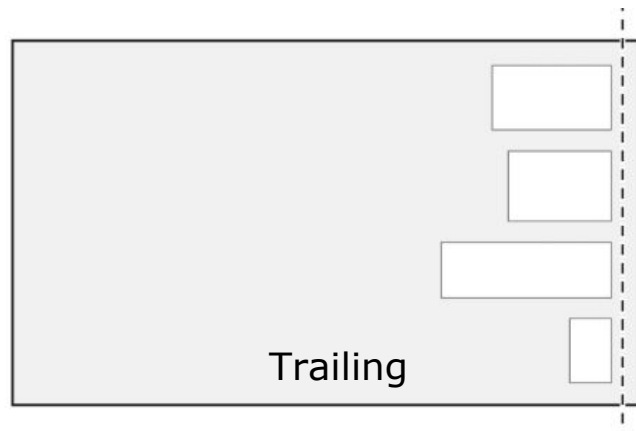
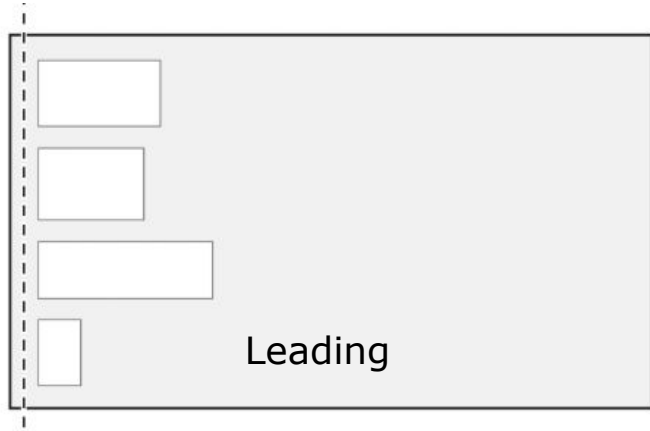
+ ☒ Installed

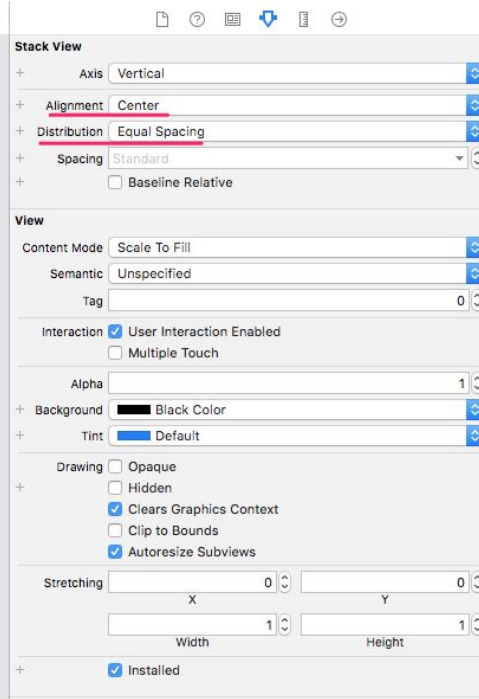
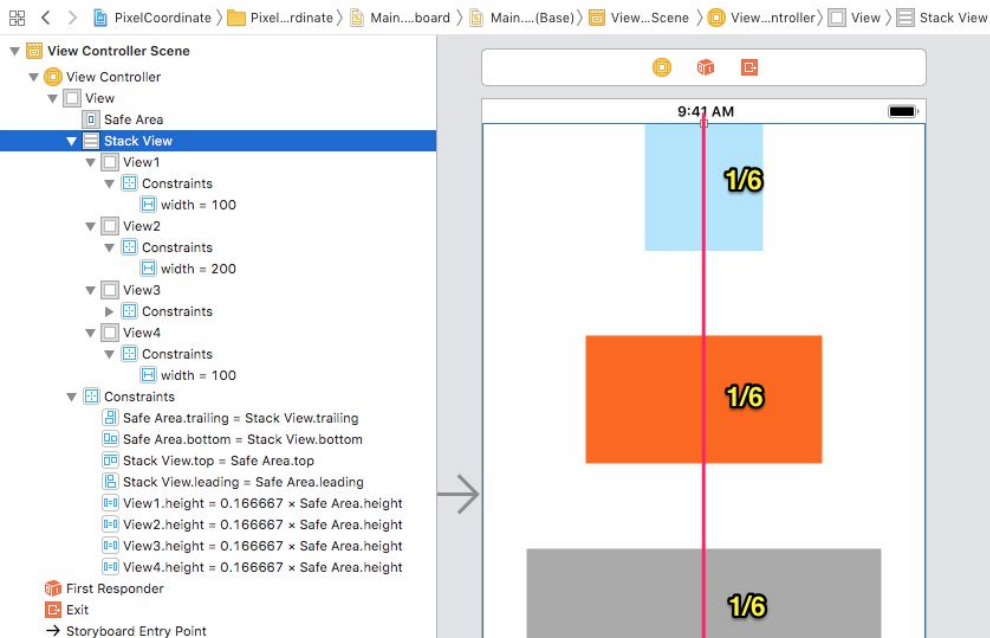
StackView.alignment



Fill





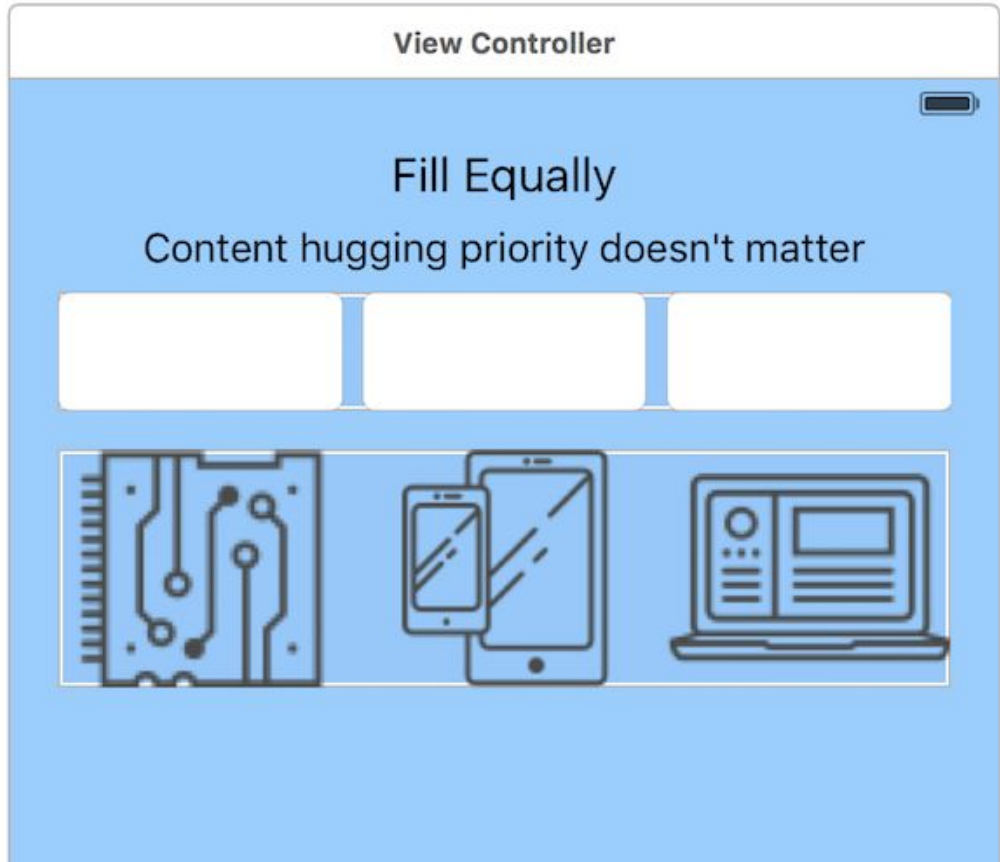
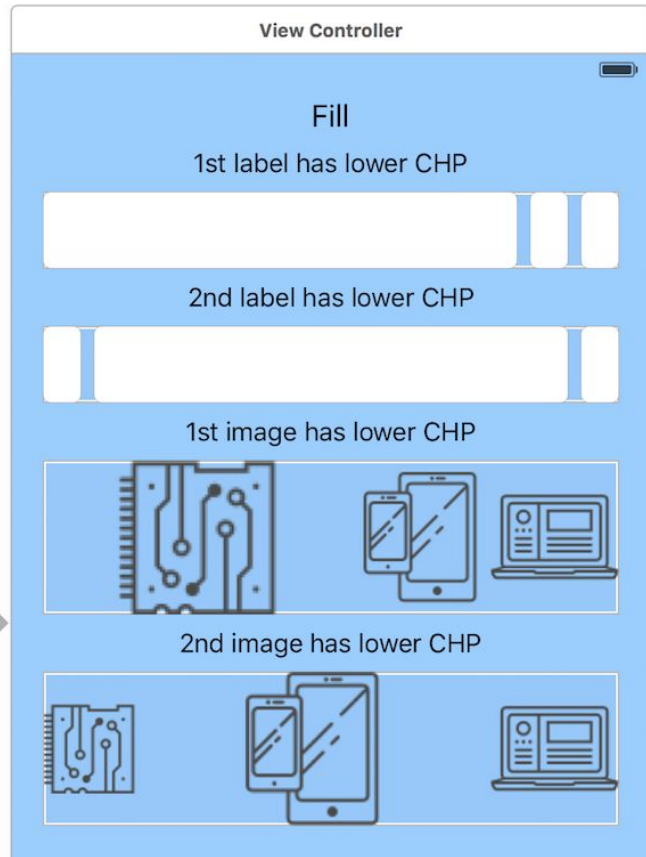


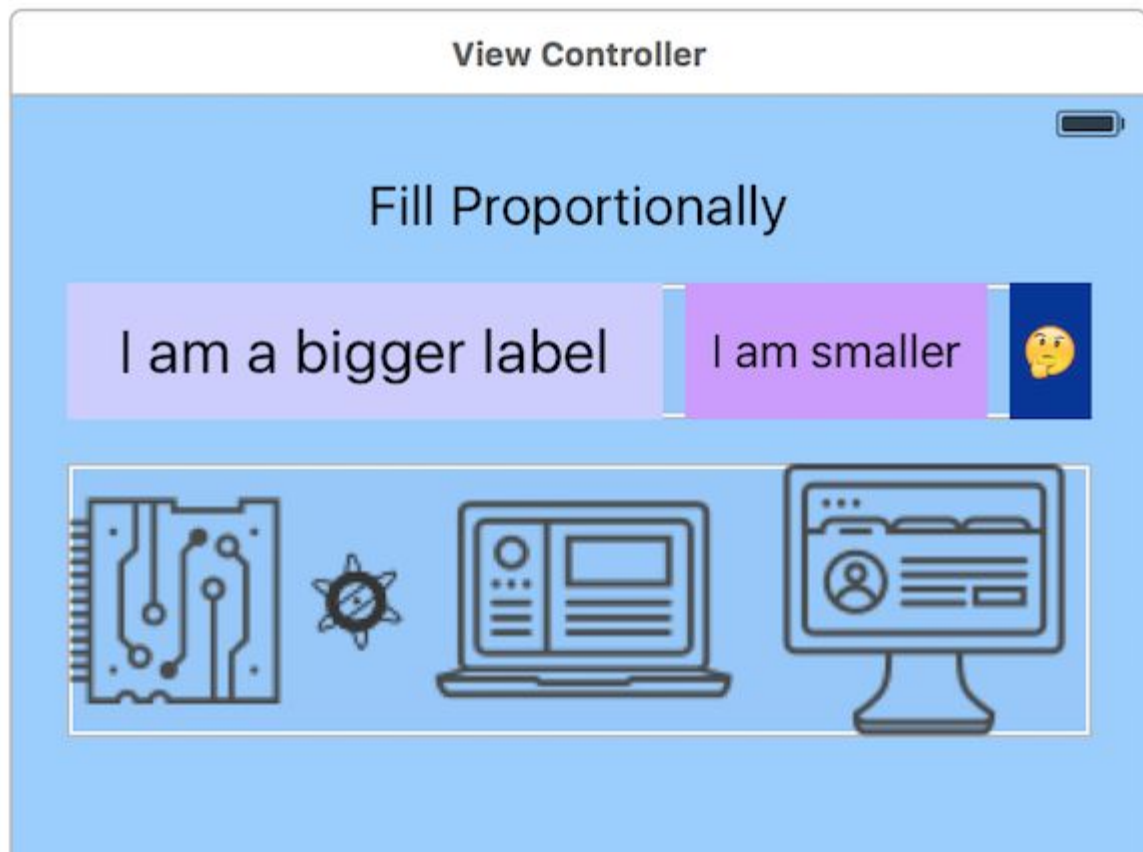
Căn các view trong Vertical StackView theo trục đứng giữa tâm

Chiều cao các view con bằng $\frac{1}{6}$ chiều cao Safe Area

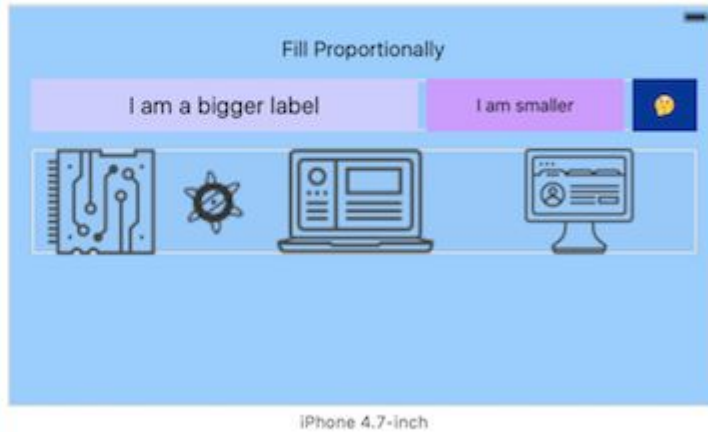
Chiều rộng tùy ý

UIStackView.distribution

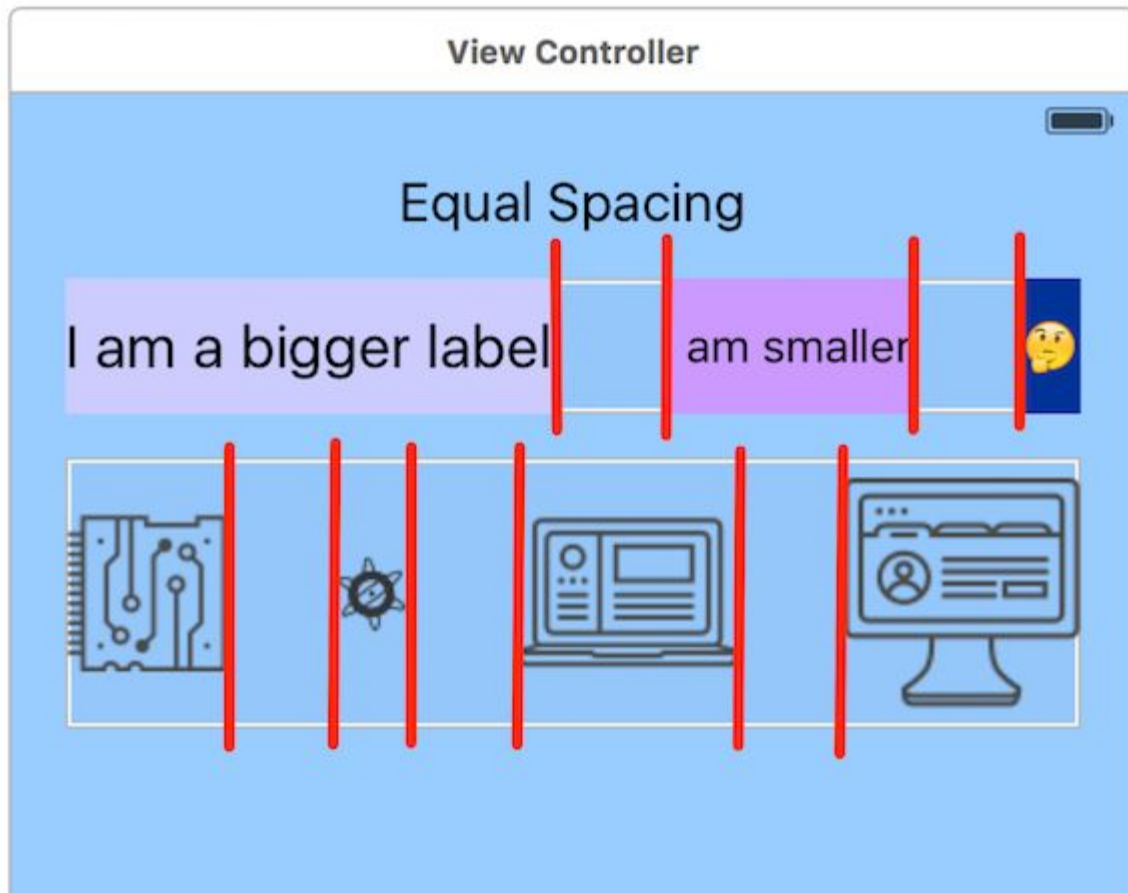




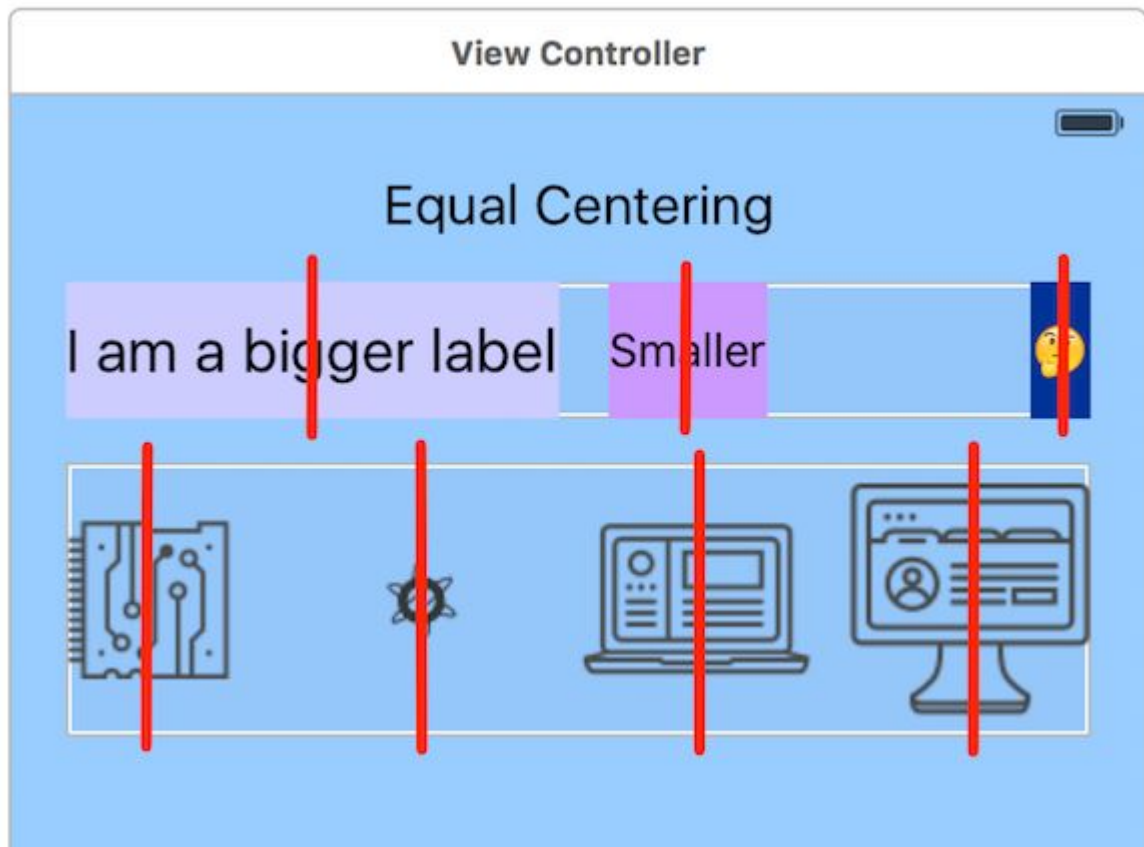
Chú ý đến label có
chiều rộng
điều chỉnh theo nội
dung



Portrait và Landscape
đều có dẫn theo nội
dung text



Các view con cách đều nhau



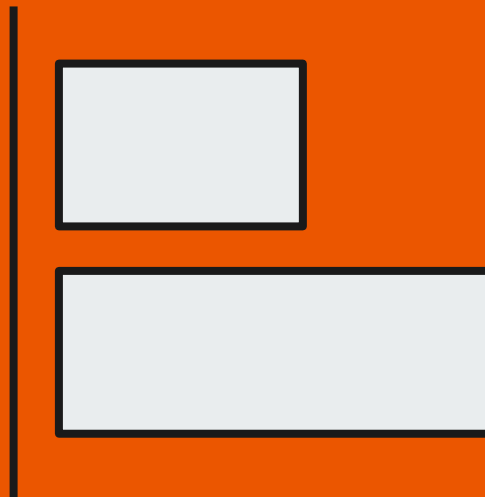
Các view con căn giữa tâm

Safe Area là gì?



1. Định nghĩa Safe Area
2. Nó giải quyết vấn đề gì?
3. Kích thước nó bằng bao nhiêu?

Align



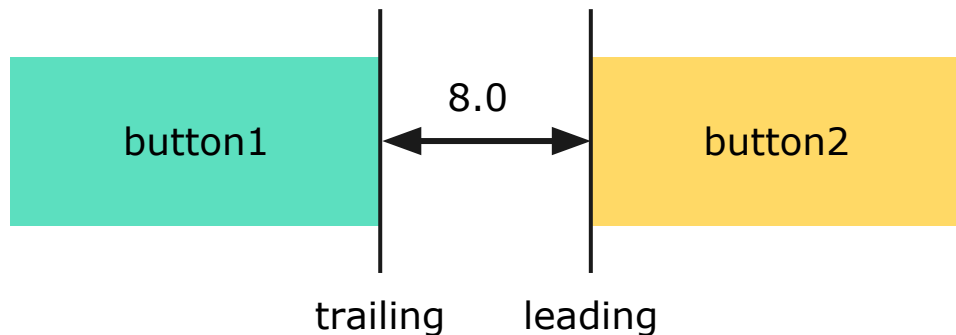
Công thức quan hệ kích thước, tọa độ giữa các view

```
item1.attribute1 = multiplier × item2.attribute2 + constant
```

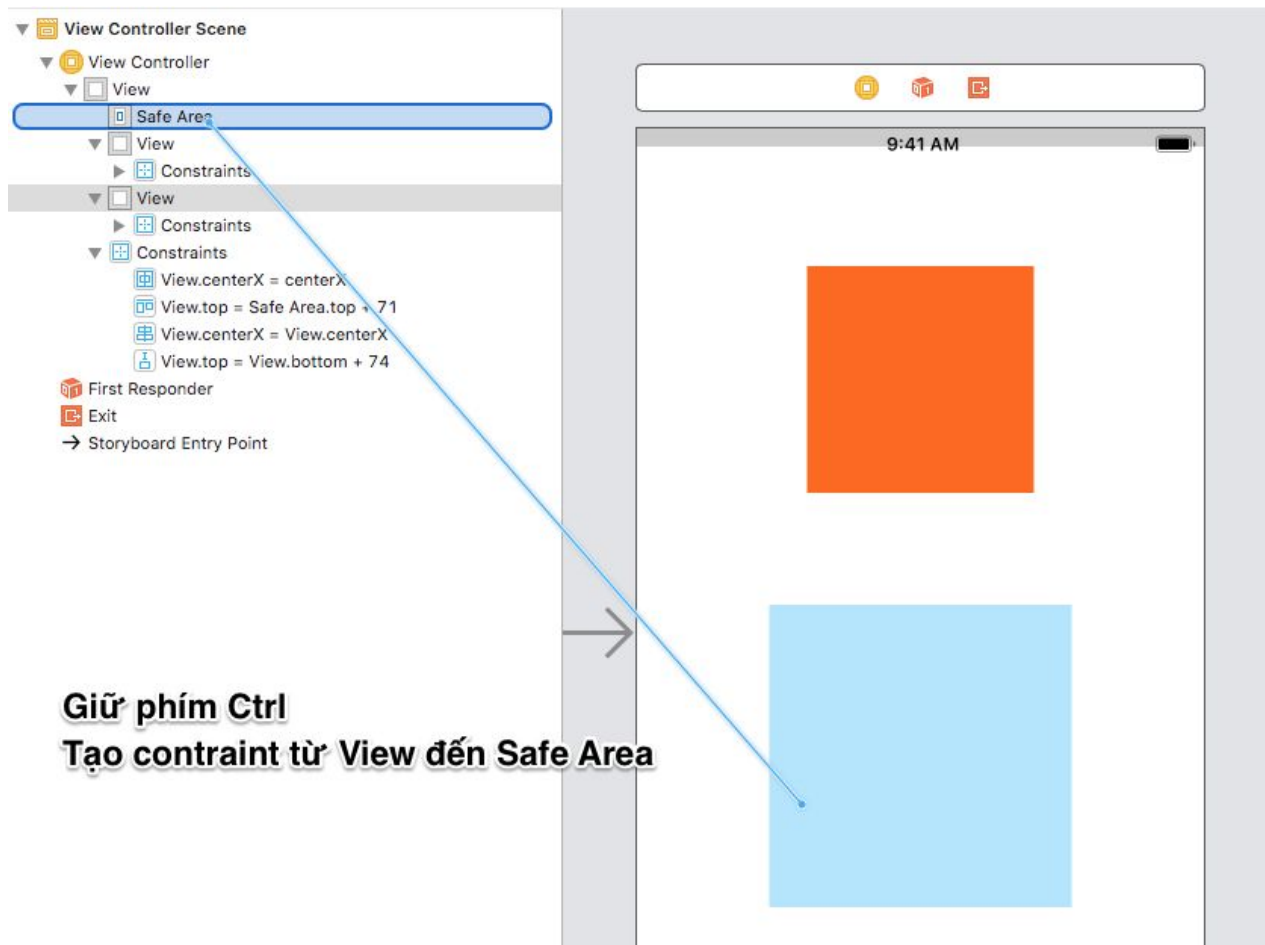
```
// These equations produce identical constraints
```

```
button2.leading = 1.0 × button1.trailing + 8.0
```

```
button1.trailing = 1.0 × button2.leading - 8.0
```



Để tạo ràng buộc tương quan giữa hai đối tượng, giữ phím Ctrl kéo từ view này sang view kia



Ý nghĩa của constant để bổ xung sai khác vào tỷ lệ liên quan

PixelCoordinate > Pi...te > M...rd > M...e) > Vi...e > Vi...er > View > Constraints > View.leading = View.leading + 50

View Controller Scene

View Controller

View

Safe Area

View

Constraints

View

Constraints

Constraints

View.centerX = centerX

View.top = Safe Area.top + 71

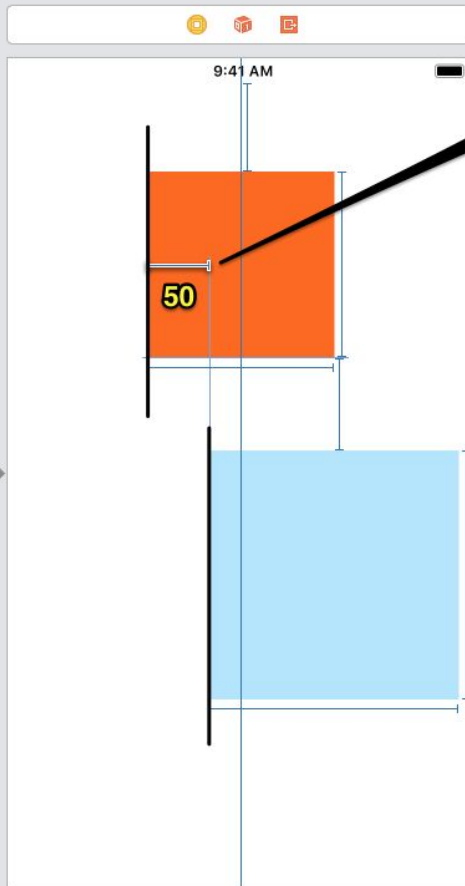
View.leading = View.leading + 50

View.top = View.bottom + 74

First Responder

Exit

Storyboard Entry Point



Leading Alignment Constraint

First Item View.Leading

Relation Equal

Second Item View.Leading

Constant 50

Priority 1000

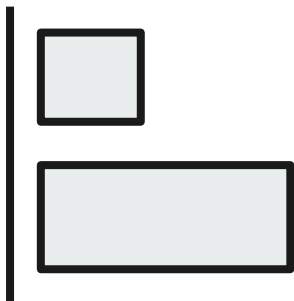
Multiplier 1

Identifier Identifier

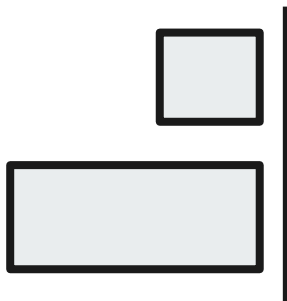
Placeholder ☐ Remove at build time

☒ Installed

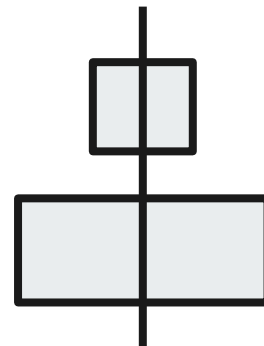
Bật tắt constraint, dùng khi bị lỗi xung đột



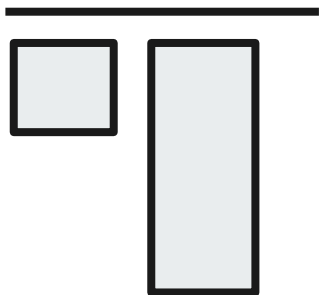
Leading



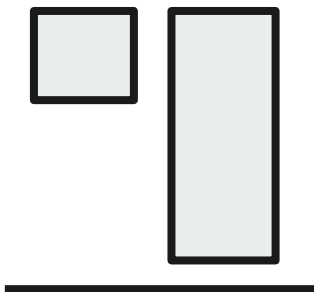
Trailing



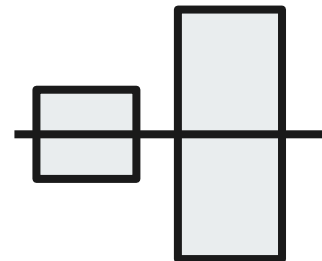
Vertical Center



Top

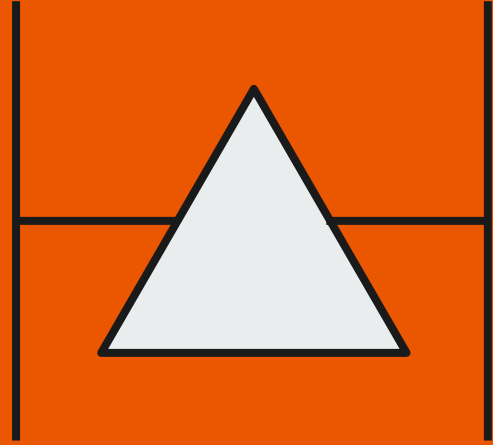


Bottom



Horizontal Center

Resolve Auto Layout Issues



Kéo thả layout
constraint



Under
constraint



Enough
constraint



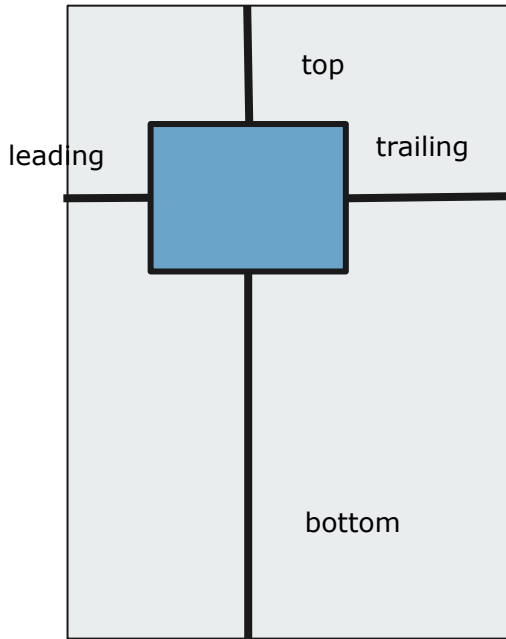
Over
constraint

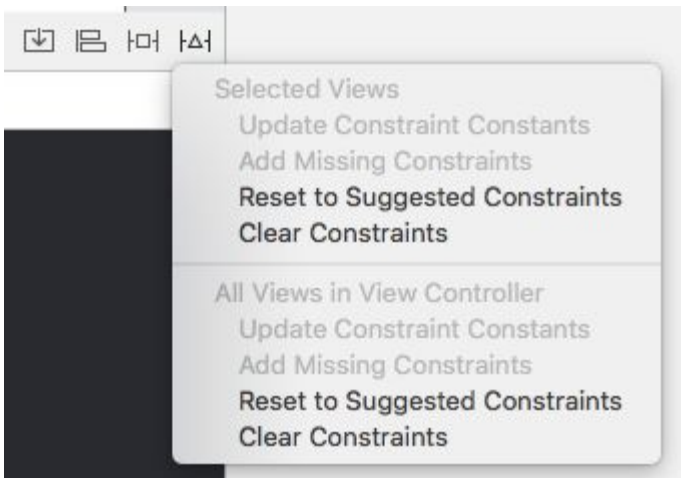


Under Constrain là gì?

Khi lập trình viên không tạo đủ ràng buộc, ứng dụng lúc run time không thể tính toán chính xác tọa độ (x, y) kích thước (width, height) của đối tượng. Lúc runtime, đối tượng hoặc là nằm ngoài màn hình, hoặc có chiều rộng hoặc dài hoặc cả hai bằng 0 và bạn sẽ không thể nhìn thấy nó trên màn hình.

- Nếu đã có ràng buộc top và bottom thì không cần height nữa
- Nếu đã có ràng buộc leading và width thì không cần trailing nữa
- Nếu box đã căn chính tâm màn hình, thì chỉ cần ràng buộc width, height là đủ





1. Update Constraint Constants: bổ xung tham số để hợp lý hoá
2. Add Missing Constraints: bổ xung constraint cần thiết để giao diện đủ ràng buộc
3. Reset to Suggested Constrains: bỏ bớt constraints vừa thêm gây nên xung đột
4. Clear Constraints: xóa tất cả constraints

Hạn chế của Autolayout kéo thả



1. Không tạo được biến trung gian phục vụ cho công thức ràng buộc
2. Ban đầu dễ học, nhưng sau đó khi có quá nhiều ràng buộc việc xử lý xung đột rất khó khăn

Hướng giải quyết

1. Sử dụng SteviaLayout
2. Chờ XCode ra mắt 9/2019 trên Catalina, có SwiftUI !
3. Chuyển qua code bằng Flutter lớp thầy Cường

Kinh nghiệm bí truyền

Layout giao diện như chuyên gia



1. Phải vẽ giao diện chi tiết bằng SketchApp, Figma trước khi layout
2. Xác nhận giao diện chỉ ở chế độ Portrait hay có cả Landscape
3. Có phải hỗ trợ cả iPad hay chỉ có iPhone
4. Tuân thủ nguyên tắc:
 - a. Trên xuống dưới
 - b. Trái sang phải
 - c. Cha đến con
5. Nếu có những thành phần giao diện phức tạp cần nhắc đóng gói thành một Custom UIView

Xử lý tình huống ràng buộc bị xung đột



Khi layout giao diện, bất kỳ lúc nào các layout constraint cũng có thể xung đột lẫn báo đỏ lòe.

1. Ghi nhớ step trước đó, bạn vừa thêm constraint nào để quay lui lại
2. Bạn đã thực hiện quy tắc Trên -> Dưới, Trái -> Phải chưa?
- 3.

Bài tập

