







**UNIVERSITY**

## ЛЕКЦІЯ 14

# “AutoLayout”



# **AutoLayout**

Що таке AutoLayout?

Трошку історії

Особливості iPhone дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



# AutoLayout

## Що таке AutoLayout?

Трошку історії

Особливості iPhone дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



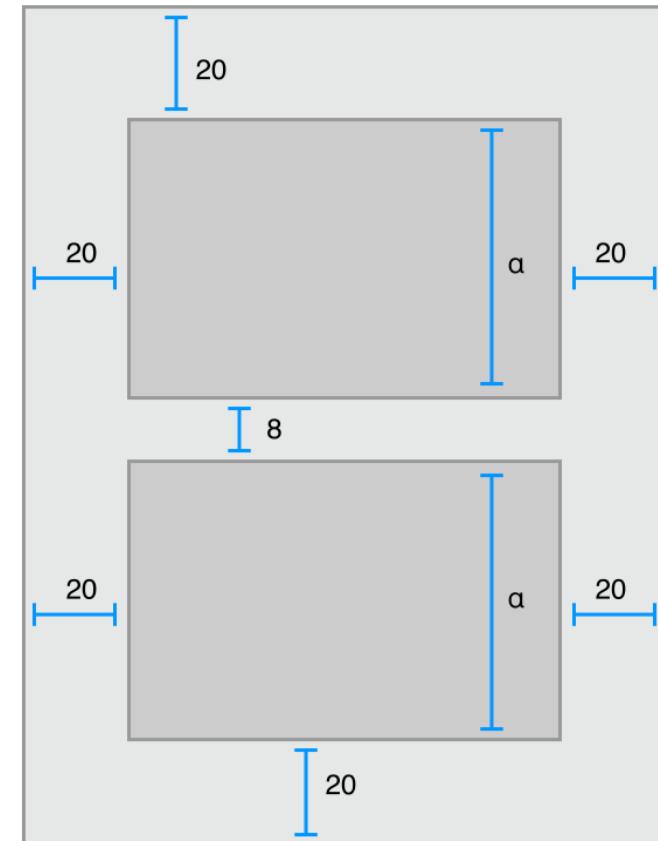
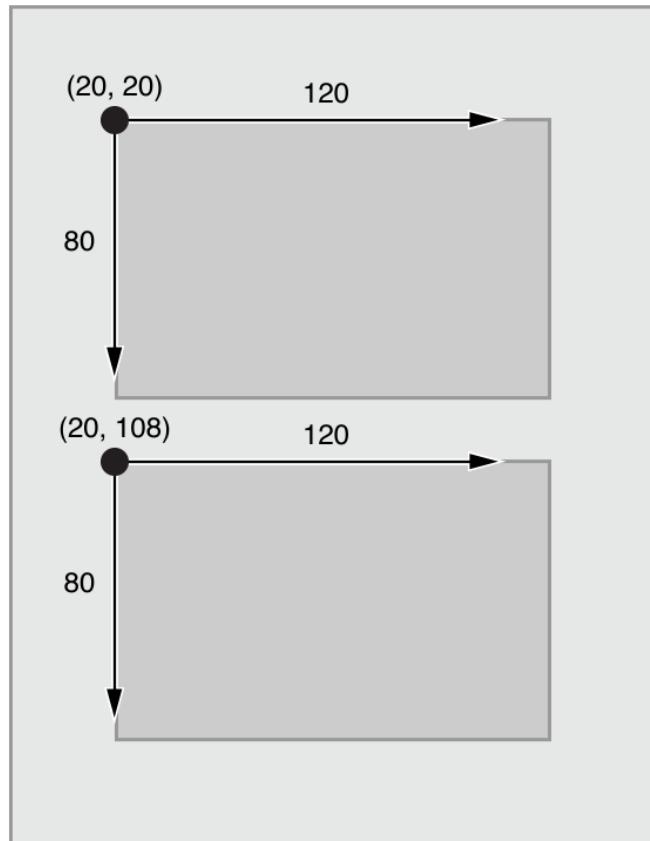
# Що таке AutoLayout?

AutoLayout - це механізм який динамічно обраховує розмір і позицію всіх елементів у вашій ієрархії виглядів, базуючися на констрайнтах які розміщені на виглядах

Можна обмежити позицію об'єкту базуючись на властивостях іншого об'єкту - як результат, вигляд буде змінюватися в залежності від зовнішніх та внутрішніх чинників



# Що таке AutoLayout?



# AutoLayout

## Що таке AutoLayout?

Трошки історії

Особливості iPhones дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



# AutoLayout

Що таке AutoLayout?

## Трошку історії

Особливості iPhones дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



# Трошкі історії



# Трошки історії



wiki

# Трошкі історії

новий розмір екрану (320x568 pt)

iOS 6 (2012)

AutoLayout (Cassowary constraint-solving toolkit)



# AutoLayout

Що таке AutoLayout?

## Трошку історії

Особливості iPhones дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



# AutoLayout

Що таке AutoLayout?

Трошку історії

## Особливості iPhone дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout

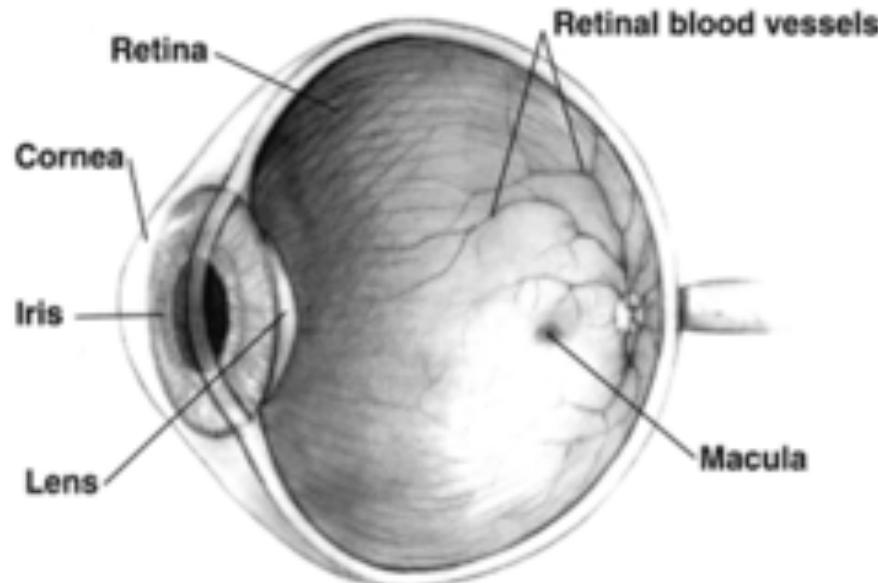


# Особливості iPhones дисплеїв

	iPhone 4	iPhone 5	iPhone 6/7/8	iPhone 6/7/8 Plus	iPhone X
<b>Display Size</b>	3.5 in	4 in	4.7 in	5.5 in	5.8 in
<b>Screen Size</b>	320 x 480 points	320 x 568 points	375 x 667 points	414 x 736 points	375 x 812 points
<b>Rendered Pixels</b>	640 x 960 (@2x)	640 x 1136 (@2x)	750 x 1334 (@2x)	1242 x 2208 (@3x)	1125 x 2436 (@3x)
<b>Physical Pixels</b>	640 x 960	640 x 1136	750 x 1334	1080 x 1920	946 x 2048
<b>Pixels Per Inch (PPI)</b>	326	326	326	401	458
<b>Browser Size Portrait</b>	320 x 372 px (320 x 440* / 320 x 460**)	320 x 460 px (320 x 528* / 320 x 548**)	375 x 559 px (375 x 627* / 375 x 647**)	414 x 628 px (414 x 696* / 414 x 716**)	375 x 633 px
<b>Browser Size Landscape</b>	480 x 212 px (480 x 280* / 480 x 300**)	568 x 212 px (568 x 280* / 568 x 300**)	667 x 267 px (667 x 335* / 667 x 355**)	736 x 306 px (736 x 374* / 736 x 394**)	812 x 280 px

# Retina

« The **retina** is the third and inner coat of the eye which is a light-sensitive layer of tissue.»



# Retina

«Retina Display is a brand name used by Apple for its series of IPS panel and OLED displays that have a higher pixel density than traditional Apple displays.»



# Фактор збільшення

1x - old iPads (e.g. image size 40x40px)

2x - iPads, all iPhones (besides 6+/6s+) (e.g. image size 80x80px)

3x - iPhone 6+/6s+/7+/8+, X (e.g. image size 120x120px)



# Interface Orientation

Portrait

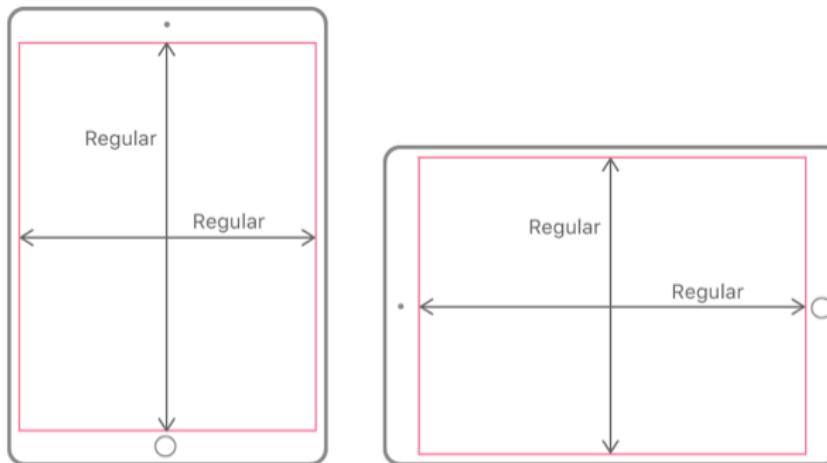
UpsideDown

LandscapeLeft

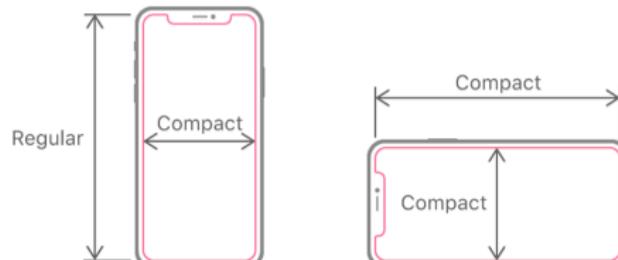
LandscapeRight



# UITraitCollection



10.5" iPad size class



iPhone X size class

# AutoLayout

Що таке AutoLayout?

Трошку історії

## Особливості iPhone дисплеїв

Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



# AutoLayout

Що таке AutoLayout?

Трошку історії

Особливості iPhones дисплеїв

## Layout Constraints

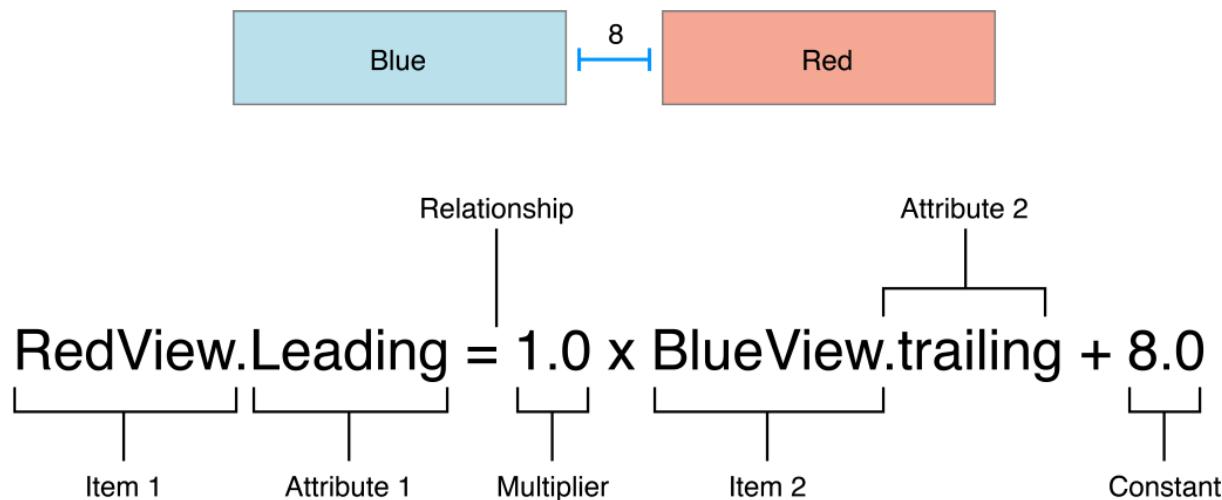
Auto Layout Attributes

Adaptivity and Layout



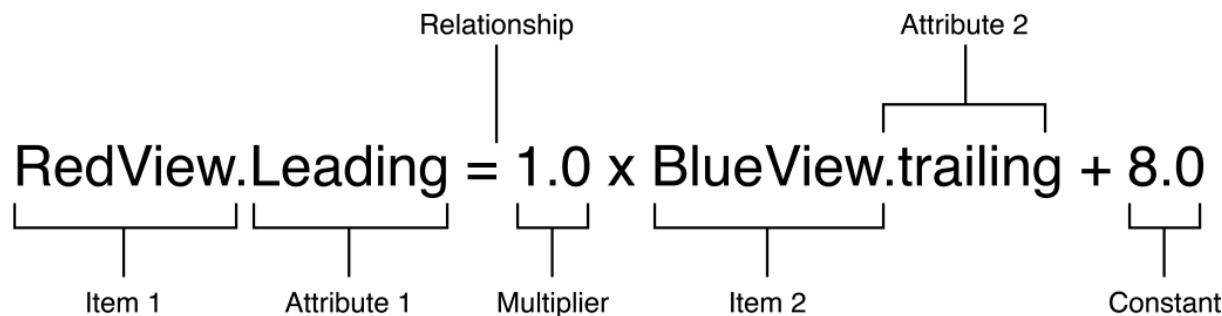
# Layout Constraints

Розмітка екрану вашого інтерфейсу визначена серією лінійних рівнянь. Кожна констрайнта представляє собою одне рівняння. Ваша ціль - визначити необхідний набір таких рівнянь для отримання необхідного інтерфесу. Такий набір може бути лише один.



Apple

# Layout Constraints



**Item 1.** The first item in the equation—in this case, the red view. The item must be either a view or a layout guide.

**Attribute 1.** The attribute to be constrained on the first item—in this case, the red view's leading edge.

**Relationship.** The relationship between the left and right sides. The relationship can have one of three values: equal, greater than or equal, or less than or equal. In this case, the left and right side are equal.

**Multiplier.** The value of attribute 2 is multiplied by this floating point number. In this case, the multiplier is 1.0.

**Item 2.** The second item in the equation—in this case, the blue view. Unlike the first item, this can be left blank.

**Attribute 2.** The attribute to be constrained on the second item—in this case, the blue view's trailing edge.

If the second item is left blank, this must be Not an Attribute.

**Constant.** A constant, floating-point offset—in this case, 8.0. This value is added to the value of attribute 2.

# Layout Constraints

Створити за допомогою InterfaceBuilder

Створити за допомогою NSLayoutConstraint

Створити за допомогою Layout Anchors

Створити за допомогою Visual Format Language



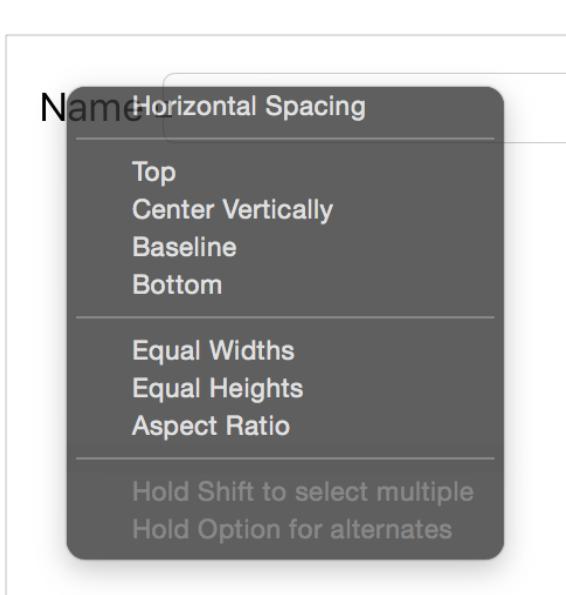
# Layout Constraints

## Створити за допомогою InterfaceBuilder

Створити за допомогою NSLayoutConstraint

Створити за допомогою Layout Anchors

Створити за допомогою Visual Format Language



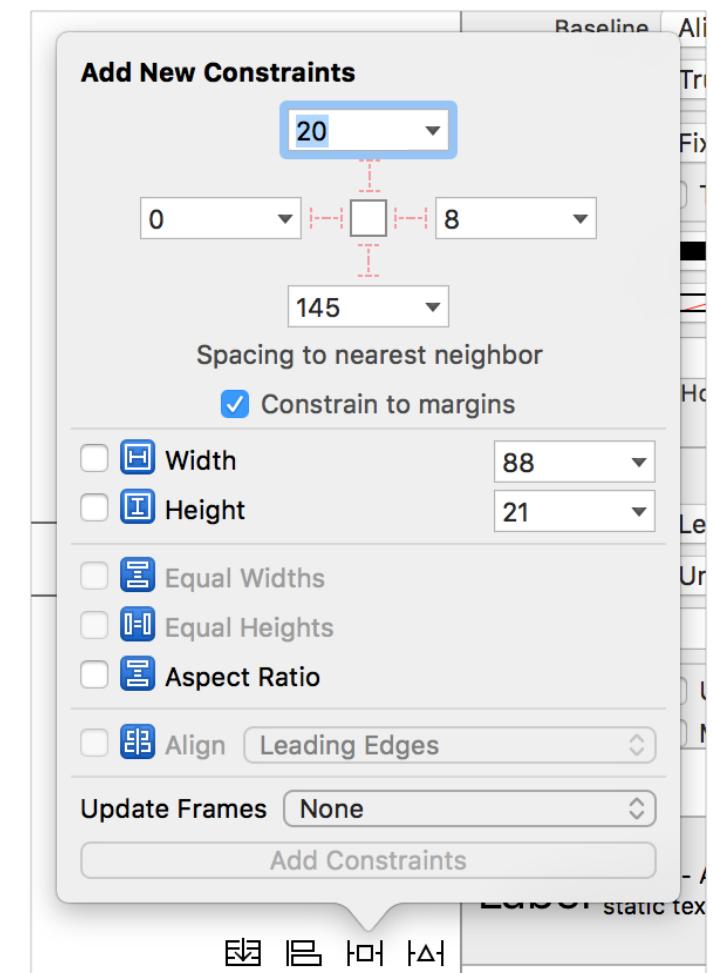
# Layout Constraints

## Створити за допомогою InterfaceBuilder

Створити за допомогою NSLayoutConstraint

Створити за допомогою Layout Anchors

Створити за допомогою Visual Format Language



# Layout Constraints

## Створити за допомогою NSLayoutConstraint

Створити за допомогою Layout Anchors

Створити за допомогою Visual Format Language

```
NSLayoutConstraint(item: myView, attribute: .Leading, relatedBy: .Equal, toItem:  
view, attribute: .LeadingMargin, multiplier: 1.0, constant: 0.0)  
  
NSLayoutConstraint(item: myView, attribute: .Height, relatedBy: .Equal, toItem:  
myView, attribute:.Width, multiplier: 2.0, constant:0.0).active = true  
  
NSLayoutConstraint(item: myView, attribute: .Height, relatedBy: .Equal, toItem:  
nil, attribute:.NotAnAttribute, multiplier: 1.0, constant:200.0).active = true
```



# Layout Constraints

## Створити за допомогою Layout Anchors

Створити за допомогою Visual Format Language

```
1 // Get the superview's layout
2 let margins = view.layoutMarginsGuide
3
4 // Pin the leading edge of myView to the margin's leading edge
5 myView.leadingAnchor.constraint(equalTo: margins.leadingAnchor).isActive = true
6
7 // Pin the trailing edge of myView to the margin's trailing edge
8 myView.trailingAnchor.constraint(equalTo: margins.trailingAnchor).isActive = true
9
10 // Give myView a 1:2 aspect ratio
11 myView.heightAnchor.constraint(equalTo: myView.widthAnchor, multiplier:
    2.0).isActive = true
```

# Layout Constraints

## Створити за допомогою Visual Format Language

```
1 let views = ["myView" : myView]
2 let formatString = "|-[myView]-|"
3
4 let constraints = NSLayoutConstraint.constraints(withVisualFormat: formatString,
      options: .alignAllTop, metrics: nil, views: views)
5
6 NSLayoutConstraint.activate(constraints)
```

# AutoLayout

Що таке AutoLayout?

Трошку історії

Особливості iPhones дисплеїв

## Layout Constraints

Auto Layout Attributes

Adaptivity and Layout



# AutoLayout

Що таке AutoLayout?

Трошку історії

Особливості iPhone дисплеїв

Layout Constraints

## Auto Layout Attributes

Adaptivity and Layout



# Auto Layout Attributes

Атрибути - властивості об'єктів які можуть бути використання для визначення лінійних рівнянь необхідних для створення констрайнт

NSLayoutAttribute



# Auto Layout Attributes

top

width

height

center X / center Y

left/leading and right/trailing

baseline

bottom



# Auto Layout Attributes

Велика різноманітість атрибутів доволяє створювати рівняння багатьма способами  
- визначити відстань, вирівнювання, позицію, співвідношення об'єктів

Не всі атрибути можуть бути використані в парі

Є 2 типи атрибутів - атрибути розміру та атрибути позиції



# Auto Layout Attributes

Є 2 типи атрибутів - атрибути розміру та атрибути позиції

Правила:

Не можна мішати атрибути розміру та атрибути позиції

Не можна задати константне значення до атрибути позиції

Не можна задати multiplier value відмінне від 1 до атрибути позиції

Не можна задати вертикальне/горизонтальне значення константи до атрибути позиції

Не можна задати leading/trailing до Left/Right атрибутів

# Auto Layout Attributes

The size of view

- Height
- Width

The values increase as you move down the screen.

- Top
- Bottom
- Baseline

The interpretation is based on the other attribute in the equation.

- Center X
- Center Y

# Auto Layout Attributes

The values increase as you move towards the trailing edge.

For a left-to-right layout directions, the values increase as you move to the right. For a right-to-left layout direction, the values increase as you move left.

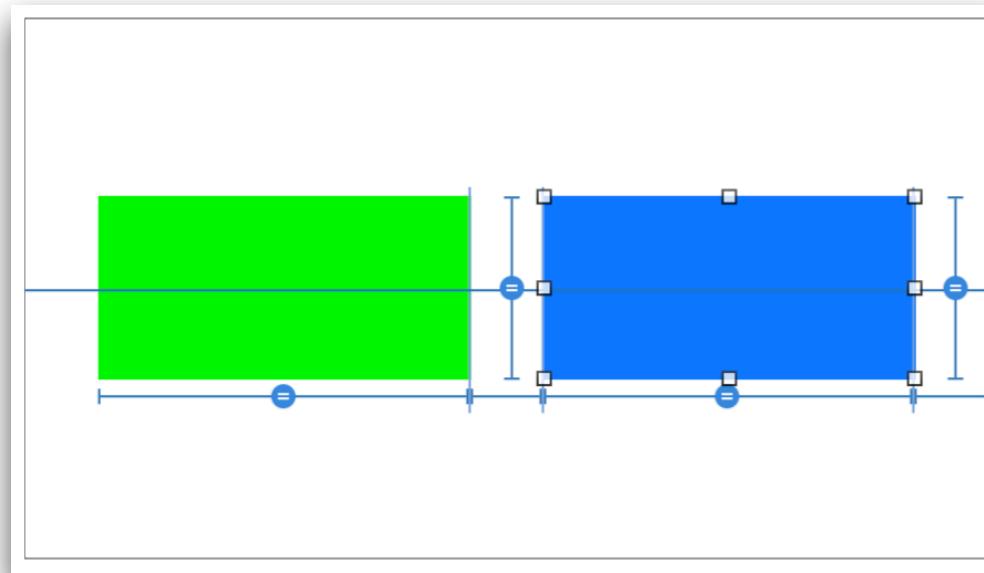
- Leading
- Trailing

The values increase as you move to the right.

- Left
- Right

# Satisfiable Layout

Layout який описується рівняннями що мають лише 1-не правильне рішення



# AutoLayout

Що таке AutoLayout?

Трошку історії

Особливості iPhone дисплеїв

Layout Constraints

## Auto Layout Attributes

Adaptivity and Layout



# **AutoLayout**

Що таке AutoLayout?

Трошку історії

Особливості iPhone дисплеїв

Layout Constraints

Auto Layout Attributes

**Adaptivity and Layout**



# Adaptivity and Layout

People generally want to use their favourite apps on all their devices and in multiple contexts, such as different device orientations and in Split View on iPad.

**Size classes** and **Auto Layout** help you meet this expectation by defining how the layout of screens, view controllers, and views should adapt when the display environment changes.



# Size Classes

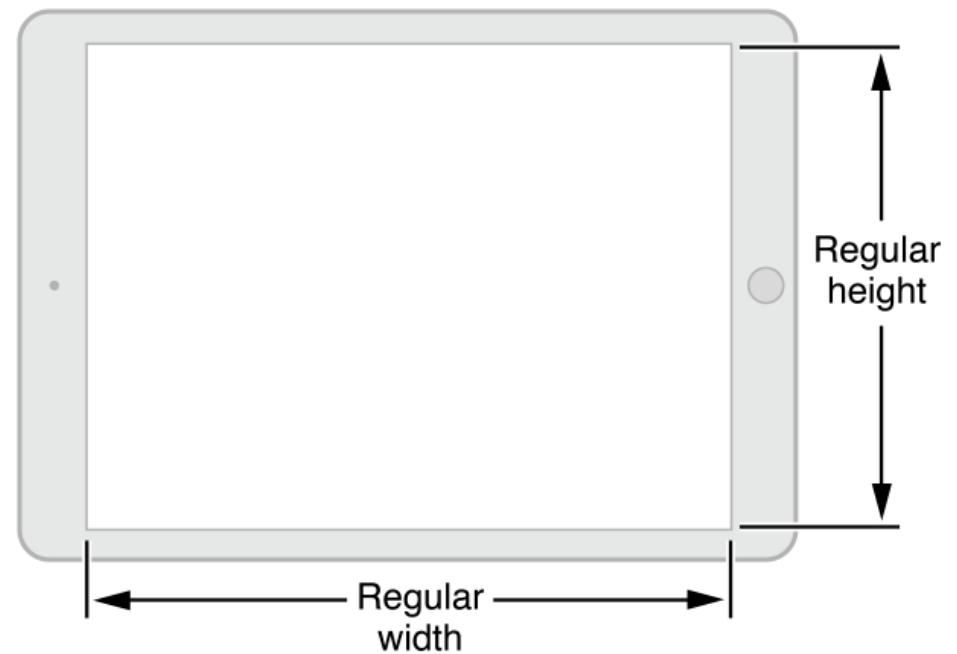
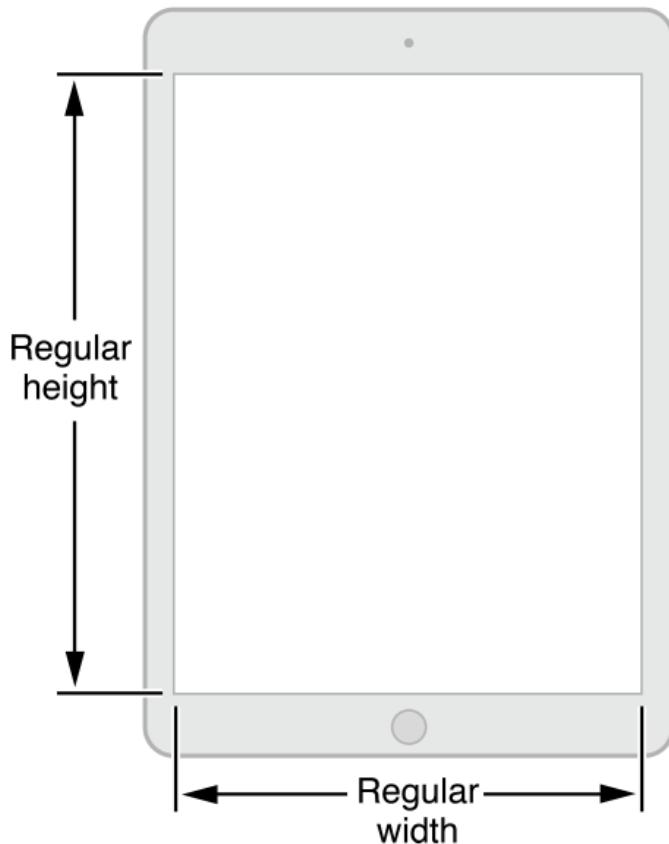
iOS defines two size classes: regular and compact.

The *regular* size class is associated with expansive space and the *compact* size class is associated with constrained space.

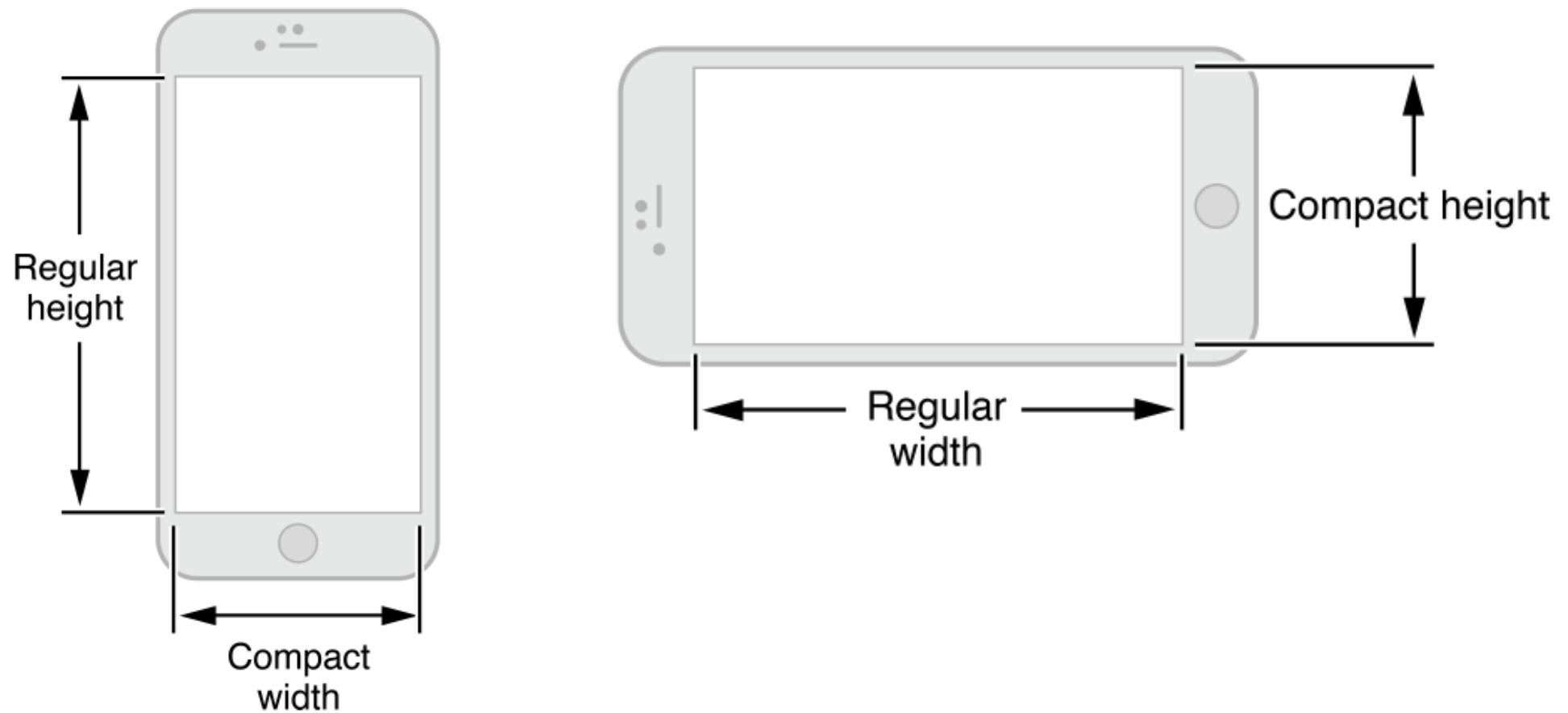
To characterize a display environment, you specify a horizontal size class and a vertical size class.



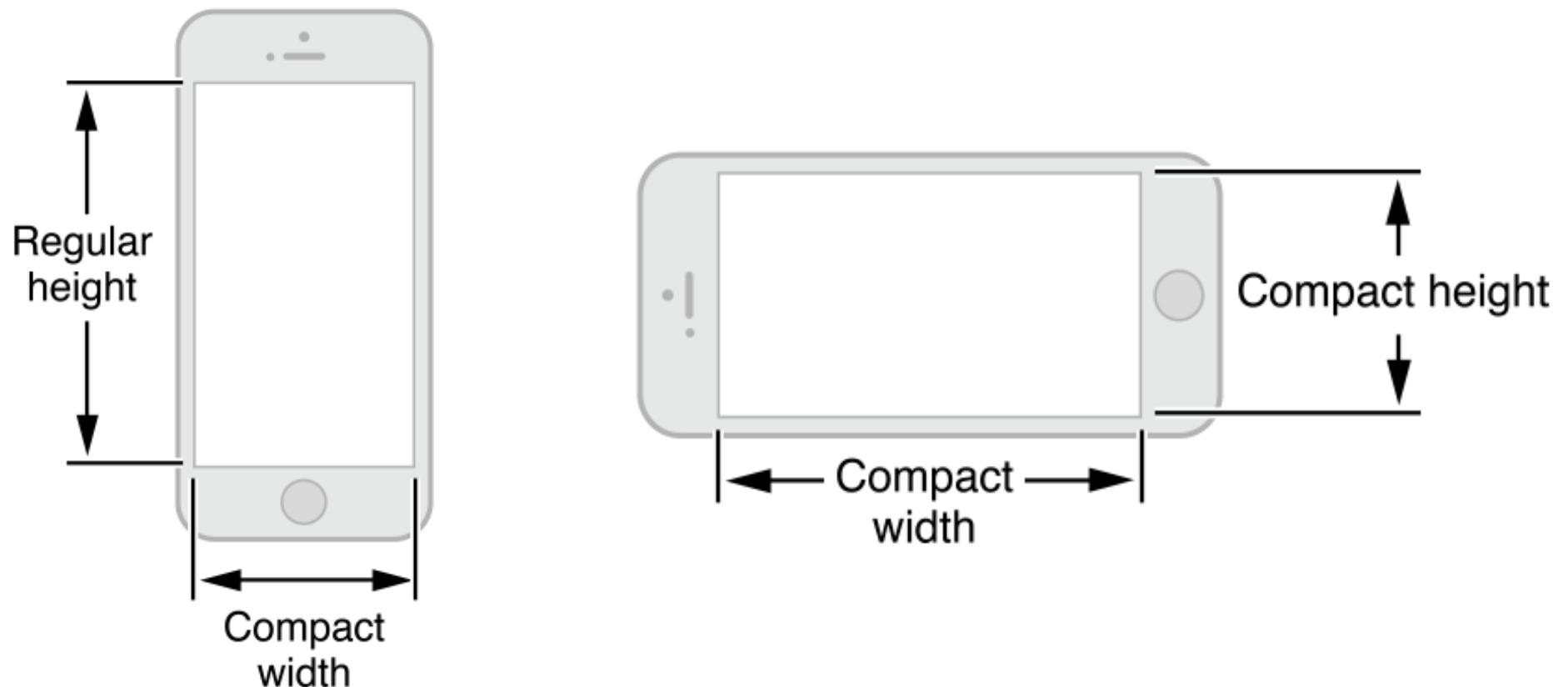
# Size Classes of iPad



# Size Classes of iPhone +Series



# Size Classes of iPhone 8 / 7 / 6 / 6s / 5 / 5s / 5c / 4s







**SIGMA**  
Software

**UNIVERSITY**