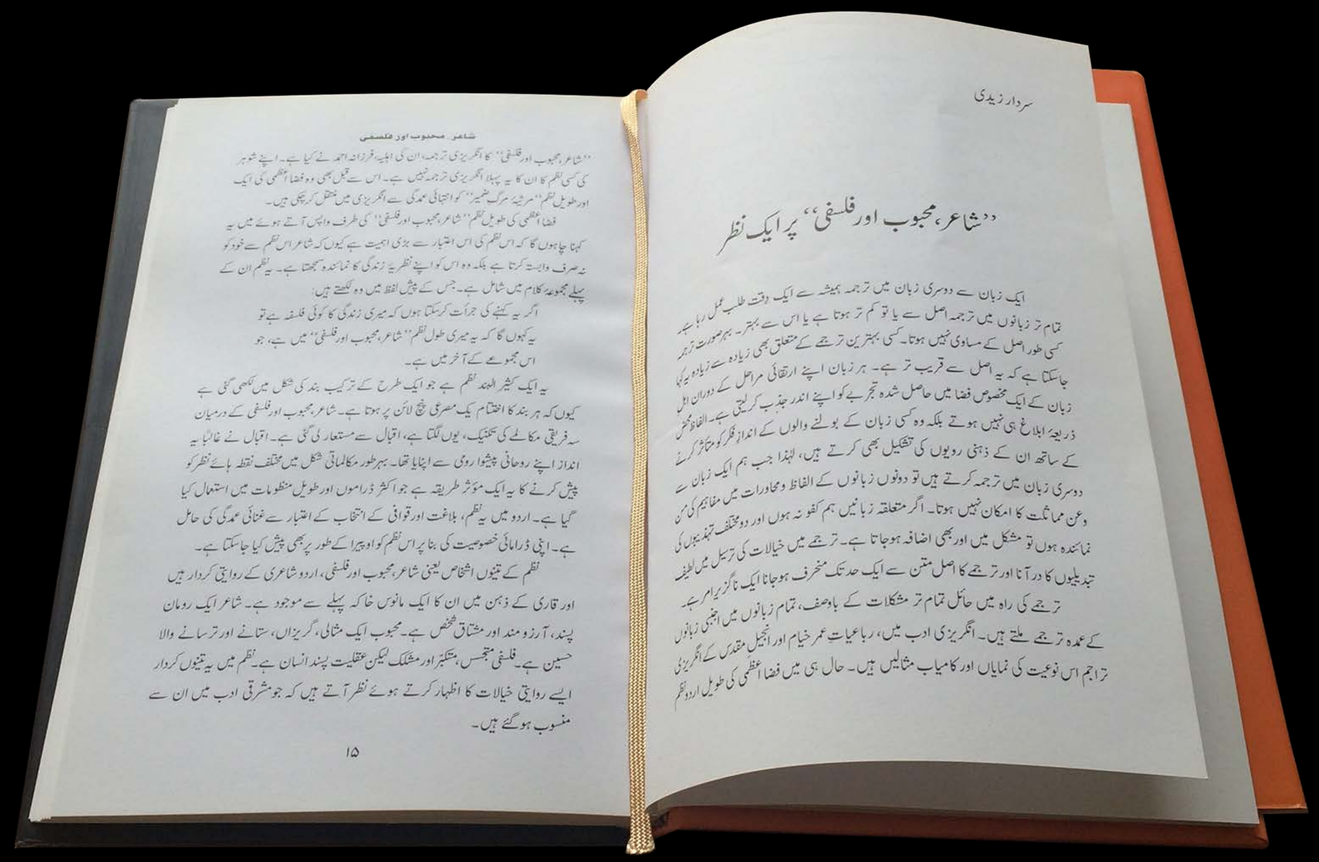
New UIKit Support for International User Interfaces

Session 222

Sara Radi Internationalization Software Engineer
Aaltan Ahmad Internationalization Software Engineer
Paul Borokhov Internationalization Software Engineer

Designing UI for RTL Languages



سردار زیدی

''شاعر، محبوب اور فلسفی'' پر ایک نظر

ایک زبان سے دوسری زبان میں ترجمہ ہمیشہ سے ایک دقت طلب عمل رہا ہے۔ تمام تر زبانوں میں ترجمہ اصل سے یا تو کم تر ہوتا ہے یا اس سے بہتر۔ بہرصورت ترجمہ کسی طور اصل کے مساوی نہیں ہوتا۔ کسی بہترین ترجمے کے متعلق بھی زیادہ سے زیادہ یہ کہا جاسکتا ہے کہ یہ اصل سے قریب تر ہے۔ ہر زبان اپنے ارتقائی مراحل کے دوران اپنی زبان کے ایک مخصوص فضا میں حاصل شدہ تجربے کو اپنے اندر جذب کرلیتی ہے۔ الفاظ محض ذریعۂ ابلاغ ہی نہیں ہوتے بلکہ وہ کسی زبان کے بولنے والوں کے اندازِ فکر کو متاثر کرنے کے ساتھ ان کے ذہنی رویوں کی تشکیل بھی کرتے ہیں، لہٰذا جب ہم ایک زبان سے دوسری زبان میں ترجمہ کرتے ہیں تو دونوں زبانوں کے الفاظ و محاورات میں مفاہیم کی مکمل و عین مماثلت کا امکان نہیں ہوتا۔ اگر متعلقہ زبانیں ہم کفو نہ ہوں اور دو مختلف تہذیبوں کی نمائندہ ہوں تو مشکل میں اور بھی اضافہ ہوجاتا ہے۔ ترجمے میں خیالات کی ترسیل میں لطیف تبدیلیوں کا در آنا اور ترجمے کا اصل متن سے ایک حد تک منحرف ہوجانا ایک ناگزیر امر ہے۔ ترجمے کی راہ میں حائل تمام تر مشکلات کے باوصف، تمام زبانوں میں اجنبی زبانوں کے عمدہ ترجمے ملتے ہیں۔ انگریزی ادب میں، رباعیاتِ عمر خیام اور انجیل مقدس کے انگریزی تراجم اس نوعیت کی نمایاں اور کامیاب مثالیں ہیں۔ حال ہی میں فضا اعظمی کی طویل اردو نظم

شاعر، محبوب اور فلسفی

''شاعر، محبوب اور فلسفی'' کا انگریزی ترجمہ، ان کی اہلیہ فرزانہ احمد نے کیا ہے۔ اپنے شوہر کی کسی نظم کا ان کا یہ پہلا انگریزی ترجمہ نہیں ہے۔ اس سے قبل بھی وہ فضا اعظمی کی ایک اور طویل نظم ''مرثیۂ مرگِ ضمیر'' کو ابتدائی عمومی سے انگریزی میں منتقل کرچکی ہیں۔

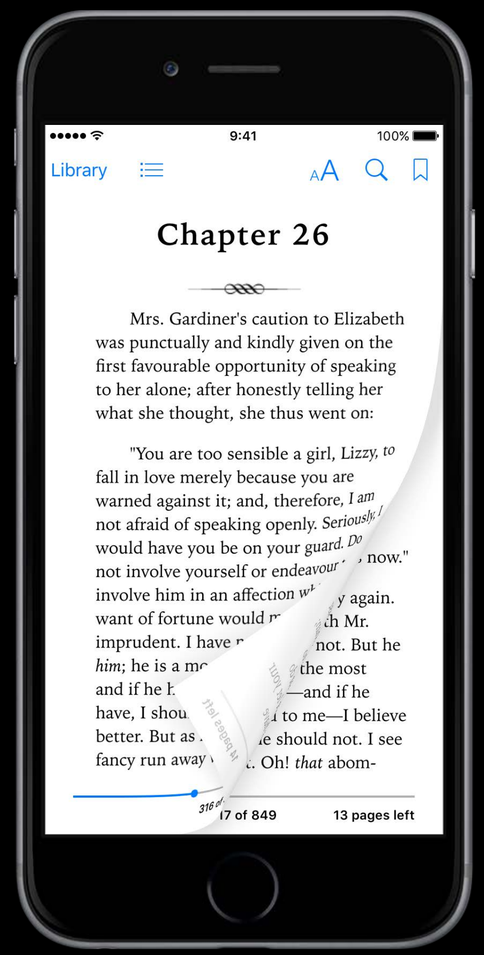
فضا اعظمی کی طویل نظم ''شاعر، محبوب اور فلسفی'' کی طرف واپس آتے ہوئے میں یہ کہنا چاہوں گا کہ اس نظم کی اس اعتبار سے بڑی اہمیت ہے کیوں کہ شاعر اس نظم سے خود کو نہ صرف وابستہ کرتا ہے بلکہ وہ اس کو اپنے نظریۂ زندگی کا نمائندہ سمجھتا ہے۔ یہ نظم ان کے پہلے مجموعۂ کلام میں شامل ہے۔ جس کے پیش لفظ میں وہ لکھتے ہیں:

> اگر یہ کہنے کی جرأت کرسکتا ہوں کہ میری زندگی کا کوئی فلسفہ ہے تو یہ کہوں گا کہ یہ میری طول نظم ''شاعر، محبوب اور فلسفی'' میں ہے، جو اس مجموعے کے آخر میں ہے۔

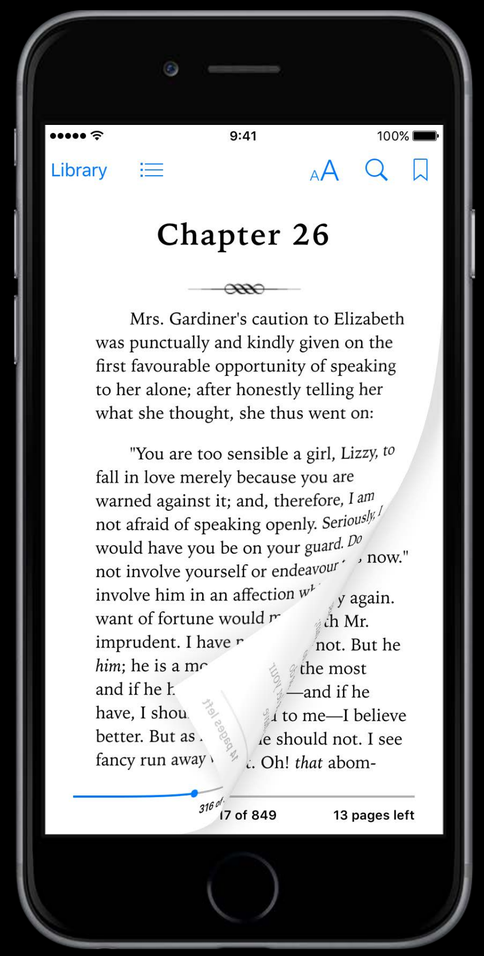
یہ ایک کثیر الجہت نظم ہے جو ایک طرح کے ترکیب بند کی شکل میں لکھی گئی ہے کیوں کہ ہر بند کا اختتام ایک مصرعی بیچ لائن پر ہوتا ہے۔ شاعر، محبوب اور فلسفی کے درمیان سہ فریقی مکالمے کی تکنیک، یوں لگتا ہے، اقبال سے مستعار لی گئی ہے۔ اقبال نے غالباً یہ انداز اپنے روحانی پیشوا رومی سے اپنایا تھا۔ بہرطور مکالماتی شکل میں مختلف نقطہ ہائے نظر کو پیش کرنے کا یہ ایک مؤثر طریقہ ہے جو اکثر ڈراموں اور طویل منظومات میں استعمال کیا گیا ہے۔ اردو میں یہ نظم، بلاغت اور قوافی کے انتخاب کے اعتبار سے غنائی عمدگی کی حامل ہے۔ اپنی ڈرامائی خصوصیت کی بنا پر اس نظم کو اوپیرا کے طور پر بھی پیش کیا جاسکتا ہے۔

نظم کے تینوں اشخاص یعنی شاعر، محبوب اور فلسفی، اردو شاعری کے روایتی کردار ہیں اور قاری کے ذہن میں ان کا ایک مانوس خاکہ پہلے سے موجود ہے۔ شاعر ایک رومان پسند، آرزو مند اور مشتاق شخص ہے۔ محبوب ایک مثالی، گریزاں، ستانے اور ترسانے والا حسین ہے۔ فلسفی متجسس، متکبر اور مشکک لیکن عقلیت پسند انسان ہے۔ نظم میں یہ تینوں کردار ایسے روایتی خیالات کا اظہار کرتے ہوئے نظر آتے ہیں کہ جو مشرقی ادب میں ان سے منسوب ہوگئے ہیں۔

۱۵



Left-to-Right



Left-to-Right

Right-to-Left


Over 500 million
native speakers

Agenda

Right-to-Left (RTL) User Interface Challenges

Agenda

Right-to-Left (RTL) User Interface Challenges

Supporting RTL UI with UIKit Controls

Agenda

Right-to-Left (RTL) User Interface Challenges

Supporting RTL UI with UIKit Controls

Custom Layout

Agenda

Right-to-Left (RTL) User Interface Challenges

Supporting RTL UI with UIKit Controls

Custom Layout

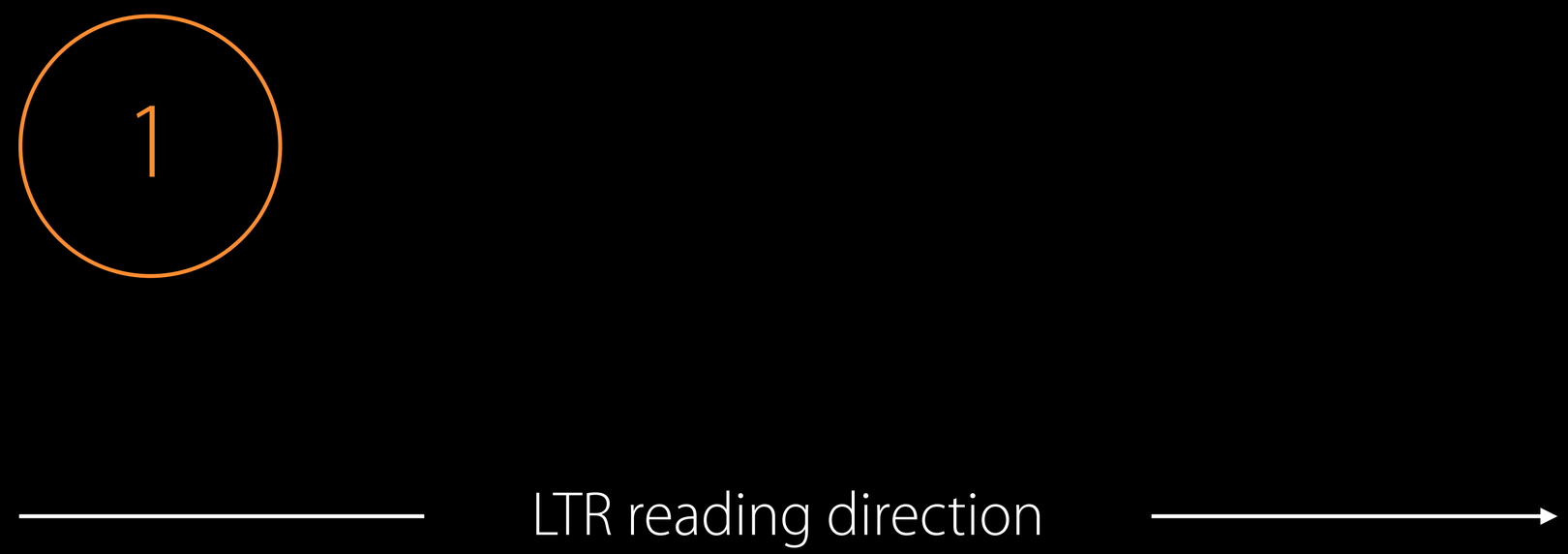
Exceptions

Overview

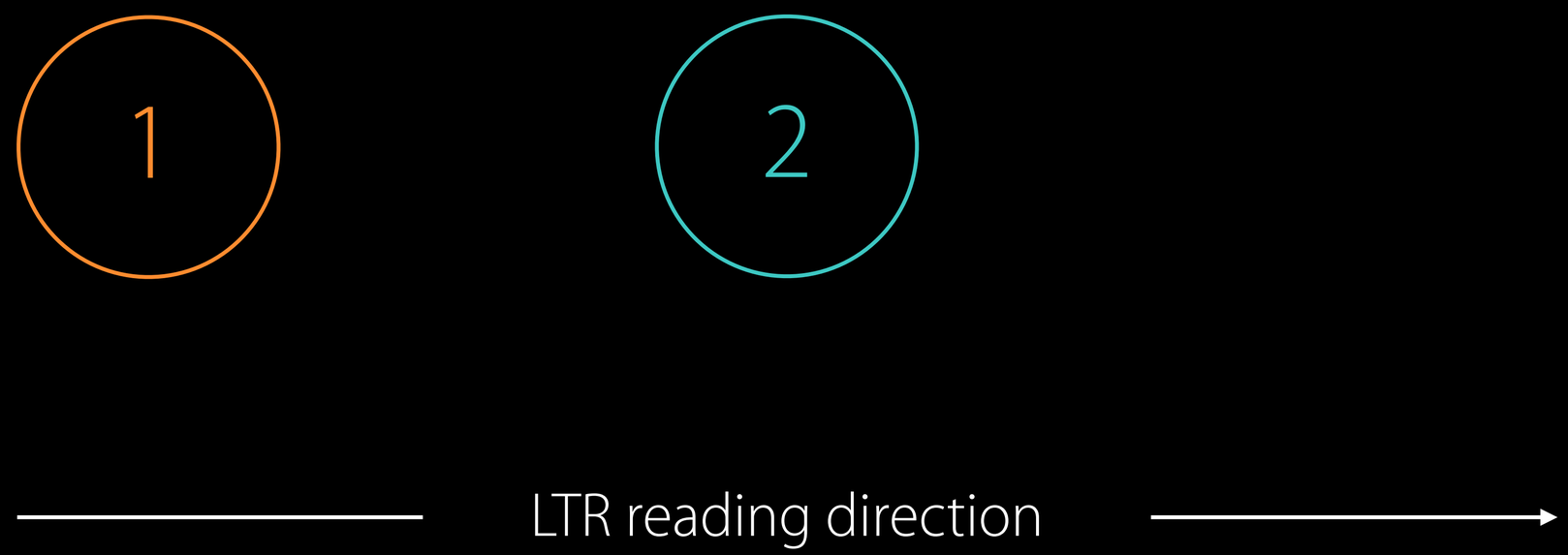
Order

LTR reading direction

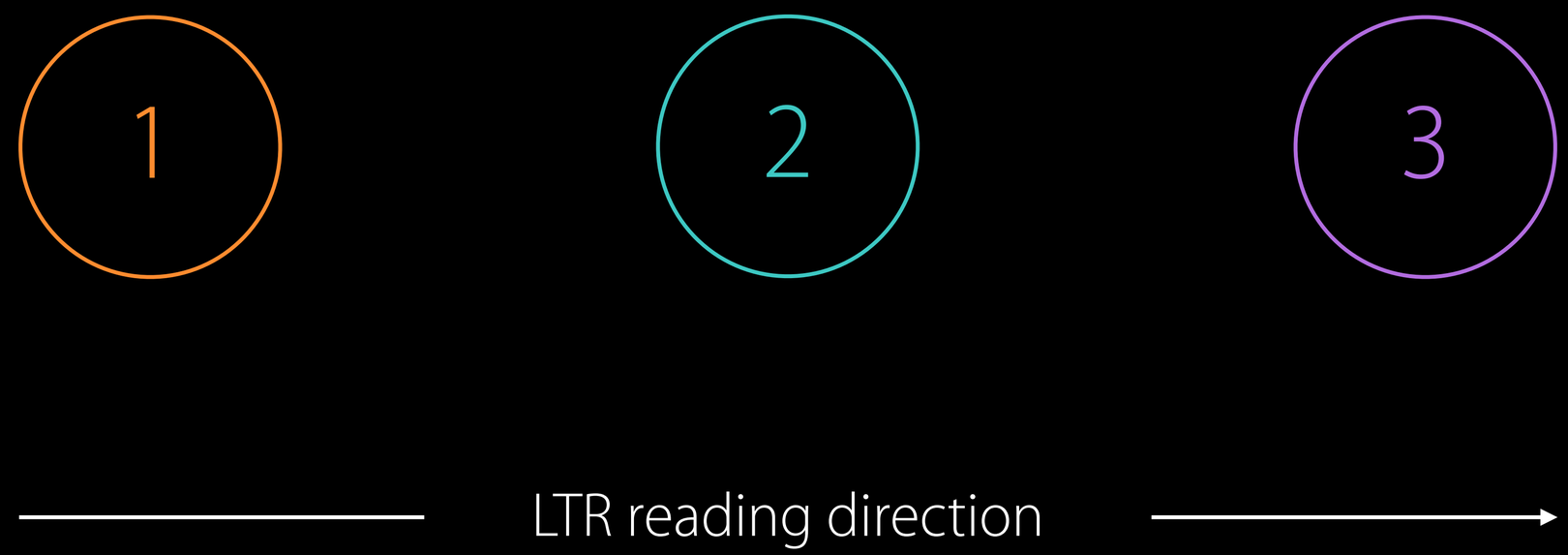
Order



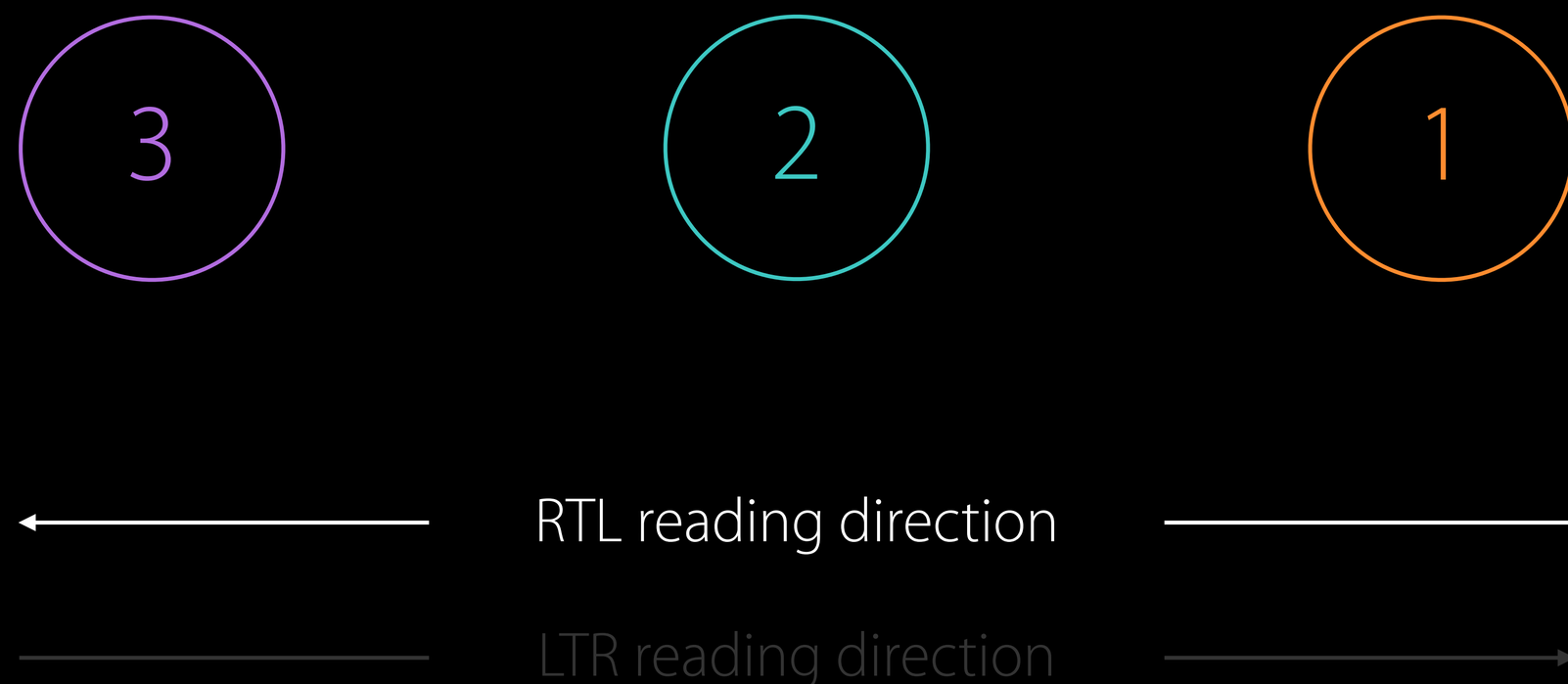
Order



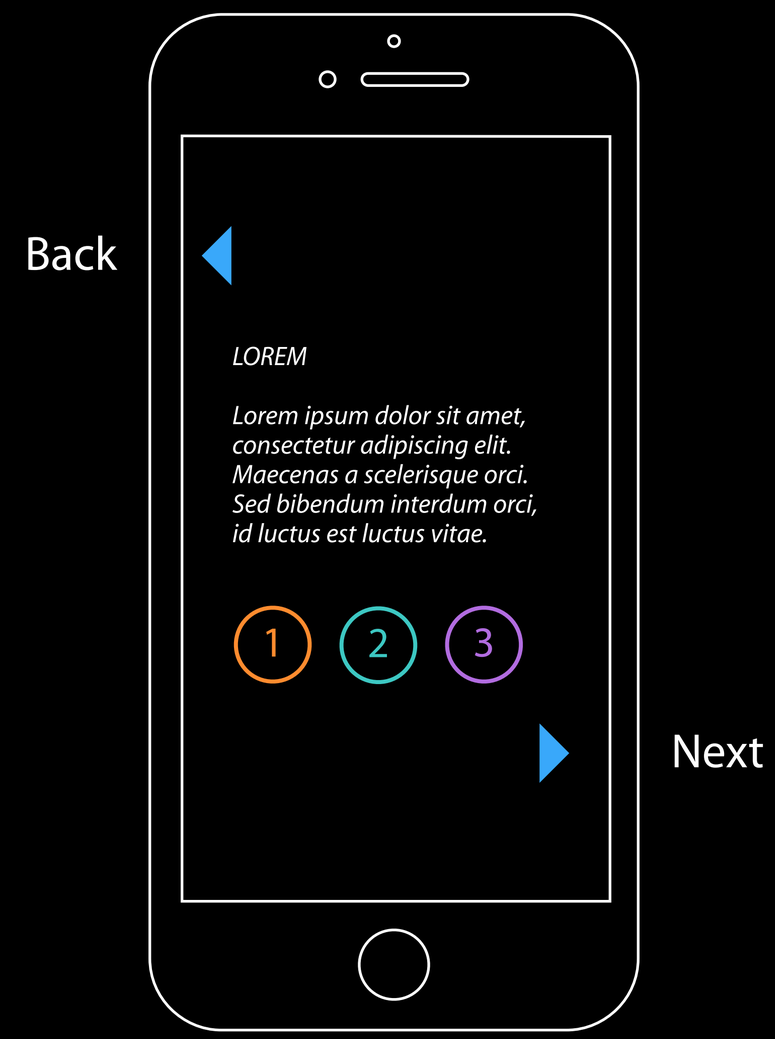
Order



Order



Navigation


Back
LOREM
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Maecenas a scelerisque orci. Sed bibendum interdum orci, id luctus est luctus vitae.
1
2
3
Next

Navigation


Back
LOREM
Lorem ipsum dolor sit amet,
consectetur adipiscing elit.
Maecenas a scelerisque orci.
Sed bibendum interdum orci,
id luctus est luctus vitae.
3
2
1
Next

New in UIKit

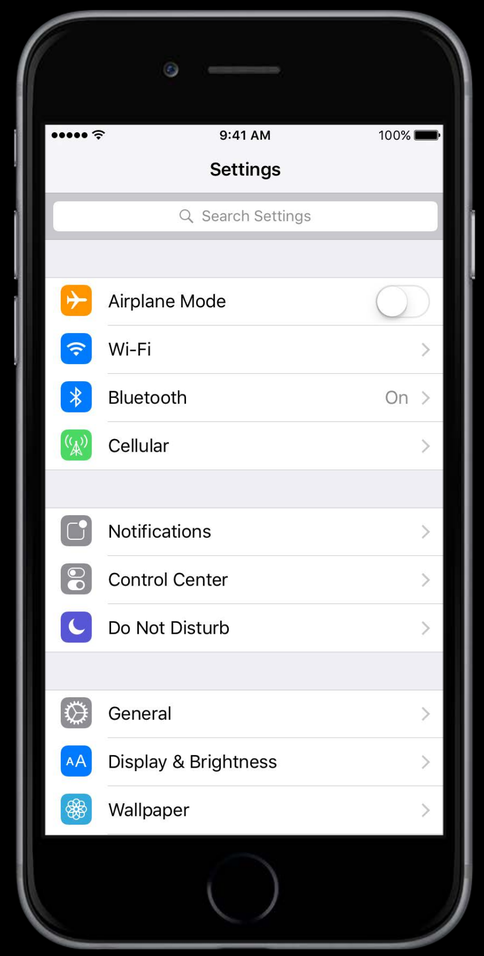
Right-to-left support


٪١٠٠
٩:٤١
الكاميرا
الصور
الخميس
٤
التقويم
٤
الرسائل
الفيديوهات
280
الخرائط
الساعة
الطقس
Wallet
الأسهم
التذكيرات
الملاحظات
صحتي
App Store
iTunes Store
iBooks
الإعدادات
الموسيقى
Safari
١٠
البريد
الهاتف

Standard Controls

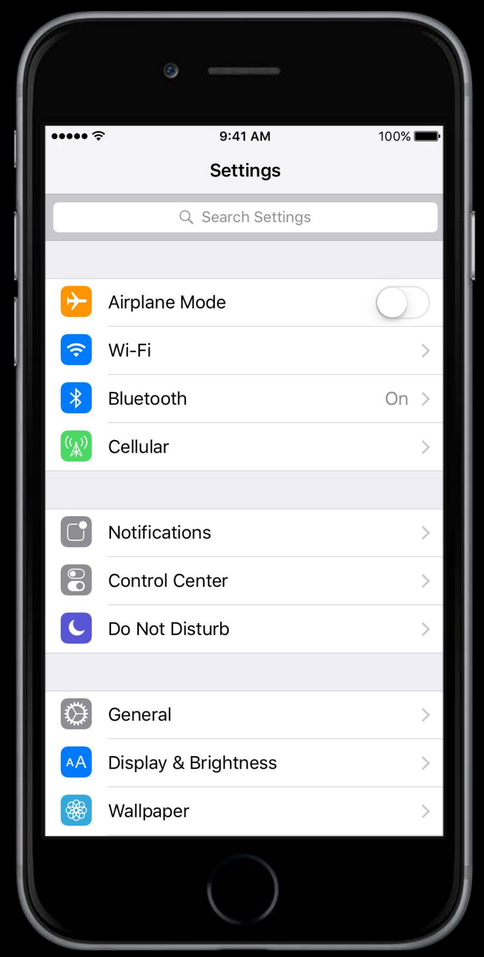
Table Views

Table Views



Left-to-Right

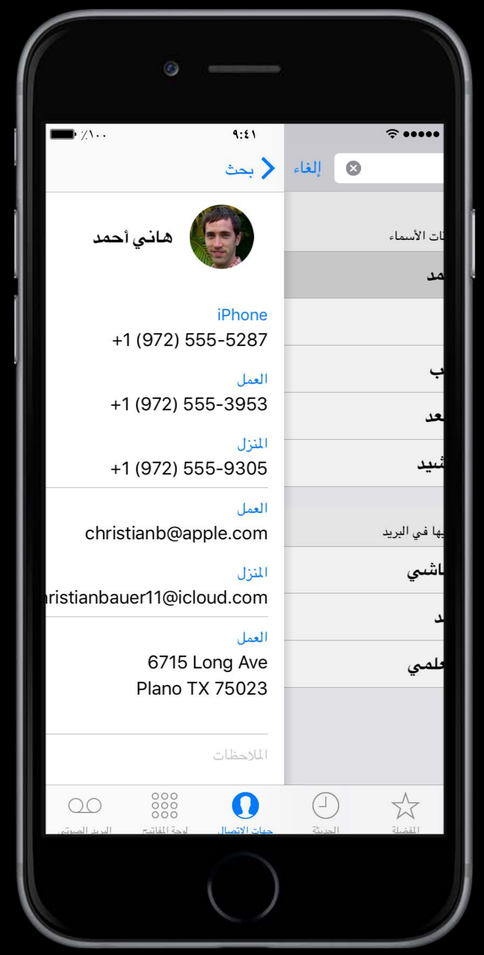
Table Views



Left-to-Right

Right-to-Left

Navigation Controller



Tracking Gestures

Tracking Gestures

Enabling Right-to-Left Support

Enabling Right-to-Left Support

Link against iOS 9

Enabling Right-to-Left Support

Link against iOS 9

As simple as adding a RTL localization

Enabling Right-to-Left Support

Link against iOS 9

As simple as adding a RTL localization

Base.lproj/Main.storyboard

Enabling Right-to-Left Support

Link against iOS 9

As simple as adding a RTL localization

Base.lproj/Main.storyboard

ar

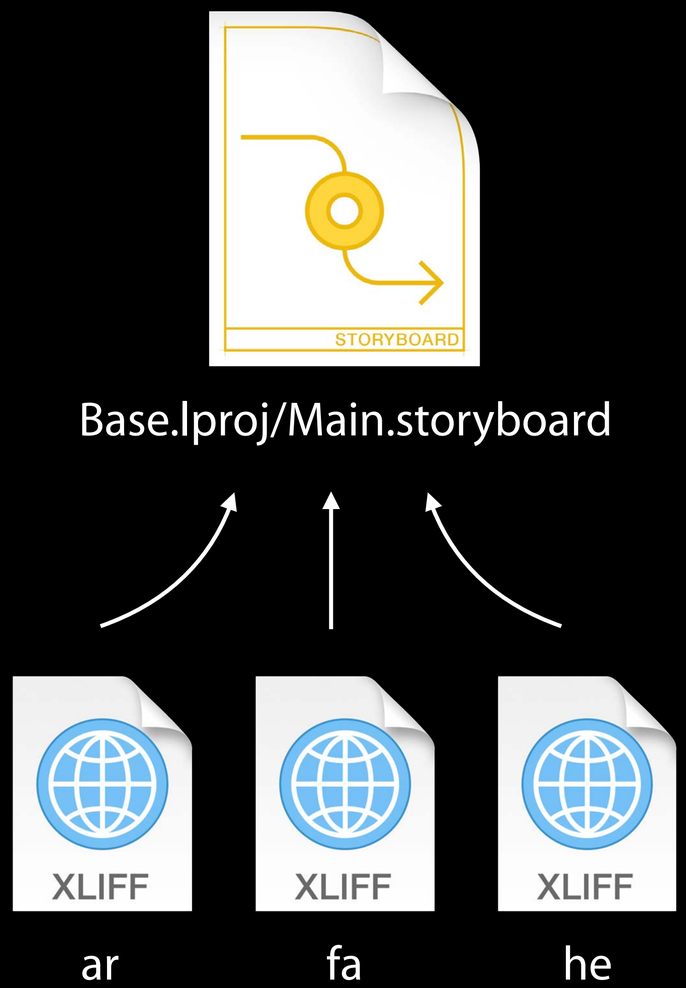
fa

he

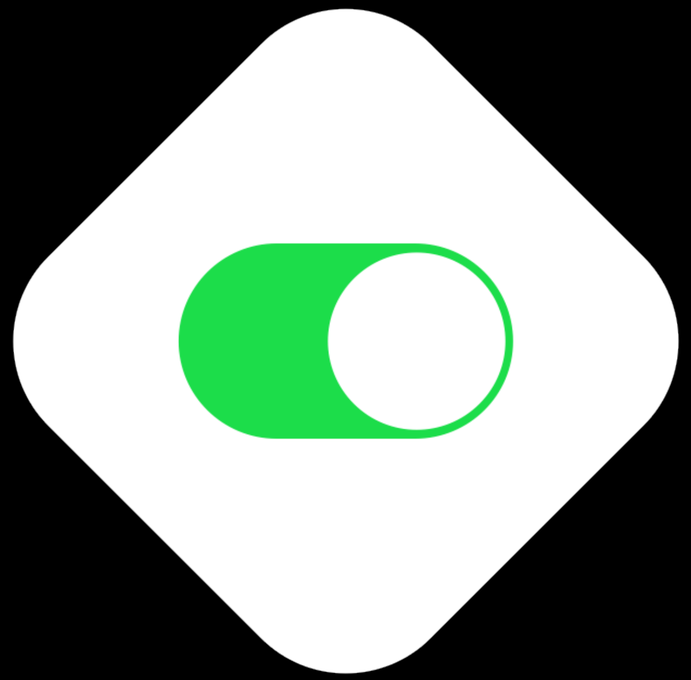
Enabling Right-to-Left Support

Link against iOS 9

As simple as adding a RTL localization



User Interface Testing



TestLocScreen 〉 iOS Device

- Build
  2 targets
- Run
  Debug
- Test
  Debug
- Profile
  Release
- Analyze
  Debug
- Archive
  Release

Info | Arguments | Options | Diagnostics

Core Location	☑ Allow Location Simulation
	Default Location: None
Application Data	None
Routing App Coverage File	None
GPU Frame Capture	Automatically Enabled
	Metal API Validation: Enabled
Background Fetch	☐ Launch due to a background fetch event
Localization Debugging	☐ Show non-localized strings
Application Language	System Language
Application Region	System Region
XPC Services	☑ Debug XPC services used by this application
View Debugging	☑ Enable user interface debugging

Duplicate Scheme | Manage Schemes... | ☐ Shared | Close

TestLocScreen 〉 iOS Device

- **Build** 2 targets
- **Run** Debug
- **Test** Debug
- **Profile** Release
- **Analyze** Debug
- **Archive** Release

Info | Arguments | Options | Diagnostics

Core Location: [x] Allow Location Simulation

Default Location: None

Application Data: None

Routing App Coverage File: None

GPU Frame Capture: Automatically Enabled

Metal API Validation: Enabled

Background Fetch: [ ] Launch due to a background fetch event

Localization Debugging: [ ] Show non-localized strings

Application Languag

Application Regio

XP

- ✓ System Language
- English
- Double Length Pseudolanguage
- Right to Left Pseudolanguage

View Debugging: [x] Enable user interface debugging

Duplicate Scheme | Manage Schemes... | [ ] Shared | Close

Demo

Localization

Custom Layout

API Changes

API Changes

UITextField

- `leftView`/`rightView` and `leftViewMode`/`rightViewMode` flip automatically
- `leftViewRectForBounds(_:)`/`rightViewRectForBounds(_:)` stay unchanged

API Changes

UITextField

- `leftView`/`rightView` and `leftViewMode`/`rightViewMode` flip automatically
- `leftViewRectForBounds(_:)`/`rightViewRectForBounds(_:)` stay unchanged

UITableView

- Insets set using the `separatorInset` property automatically flip left and right measurements

API Changes

API Changes

UISlider

- `minimumValueImage` and `maximumValueImage` flip automatically
- Be aware of adjustments done in `minimumValueImageRectForBounds(_:)` and `maximumValueImageRectForBounds(_:)`

API Changes

UISlider

- `minimumValueImage` and `maximumValueImage` flip automatically
- Be aware of adjustments done in `minimumValueImageRectForBounds(_:)` and `maximumValueImageRectForBounds(_:)`

UINavigationItem

- `leftBarButtonItem(s)` and `rightBarButtonItem(s)` flip automatically
- Beware of views added outside of API

Table View Cells

Table View Cells

Standard cells flip automatically

Table View Cells

Standard cells flip automatically

Custom layouts need to be flipped too

Collection View Flow Layouts

Collection View Flow Layouts

`UICollectionViewFlowLayout` supports right to left

Collection View Flow Layouts

`UICollectionViewFlowLayout` supports right to left

Reverse math for custom flow layouts

Collection View Flow Layouts

`UICollectionViewFlowLayout` supports right to left

Reverse math for custom flow layouts

- Subclass `UICollectionViewFlowLayout`

Collection View Flow Layouts

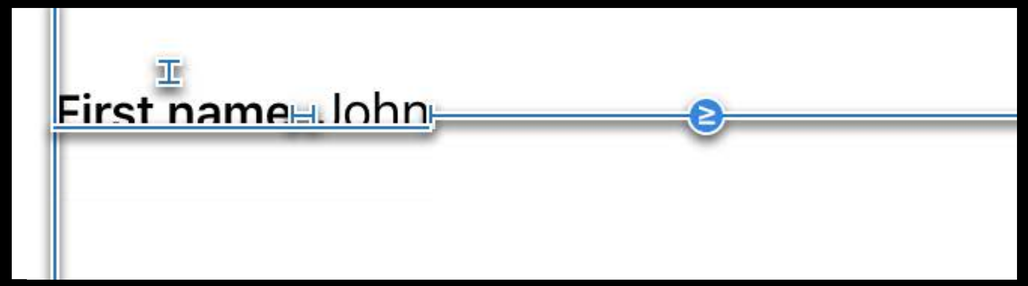
`UICollectionViewFlowLayout` supports right to left

Reverse math for custom flow layouts

- Subclass `UICollectionViewFlowLayout`

Auto Layout

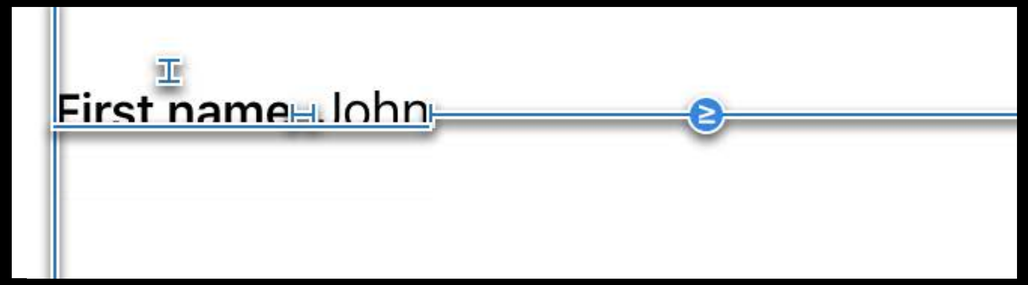
Many reasons to use Auto Layout



Auto Layout

Many reasons to use Auto Layout

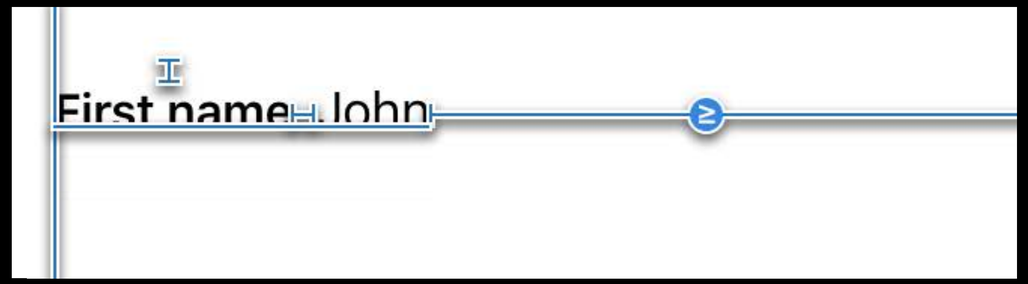
- Available since iOS 6



Auto Layout

Many reasons to use Auto Layout

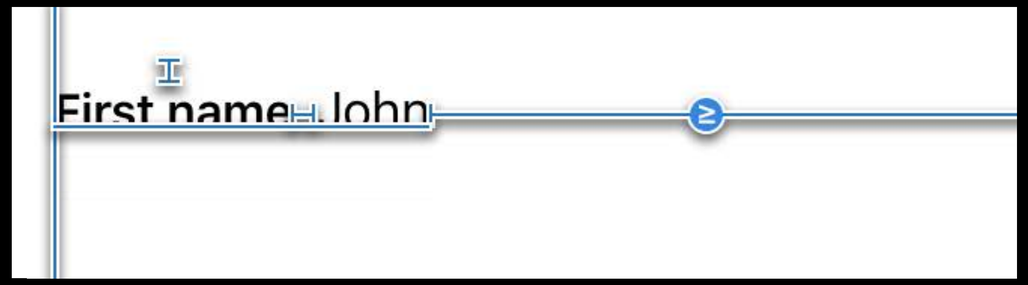
- Available since iOS 6
- Different screen sizes



Auto Layout

Many reasons to use Auto Layout

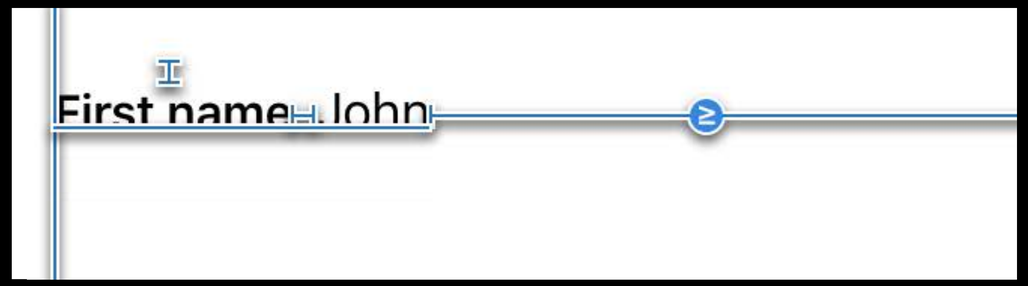
- Available since iOS 6
- Different screen sizes
- Split-screen multitasking



Auto Layout

Many reasons to use Auto Layout

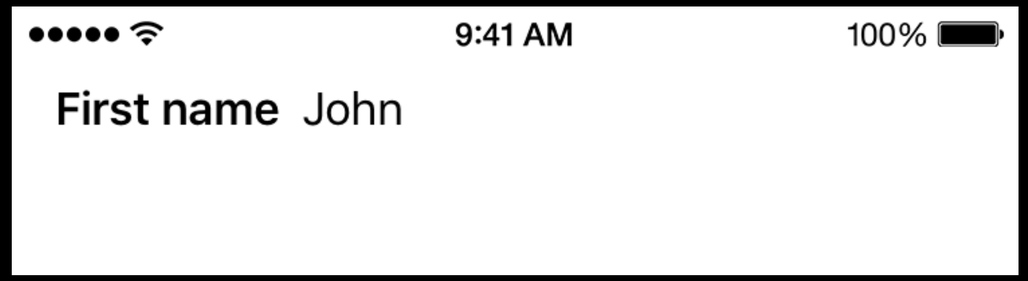
- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization



Auto Layout

Many reasons to use Auto Layout

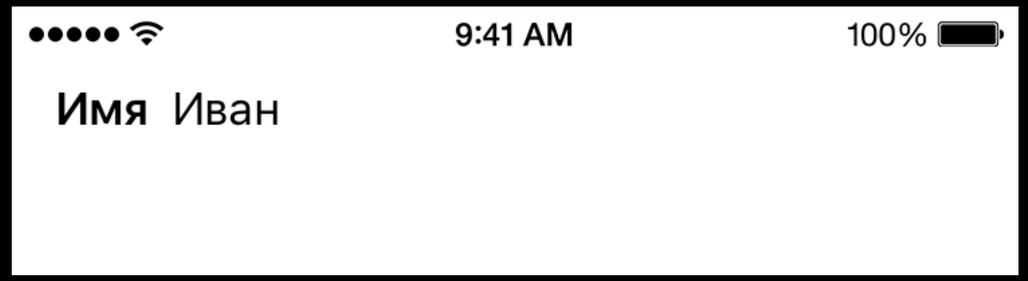
- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization



Auto Layout

Many reasons to use Auto Layout

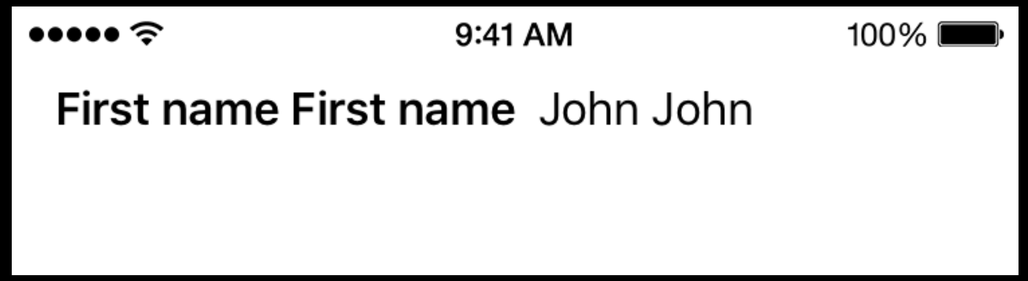
- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization



Auto Layout

Many reasons to use Auto Layout

- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization



Auto Layout

Many reasons to use Auto Layout

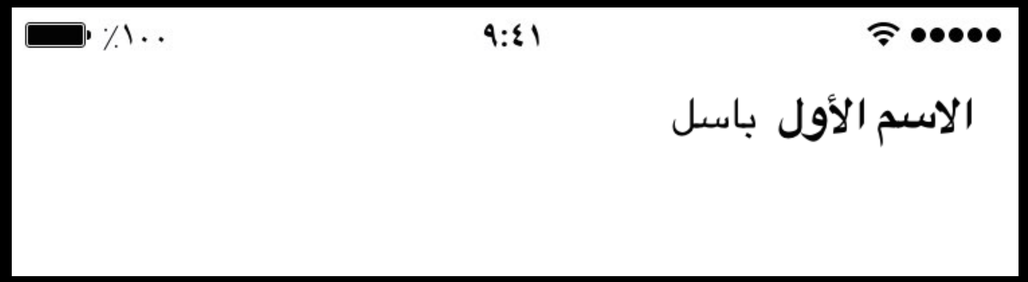
- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization

Auto Layout

Many reasons to use Auto Layout

- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization

One more reason–right-to-left!



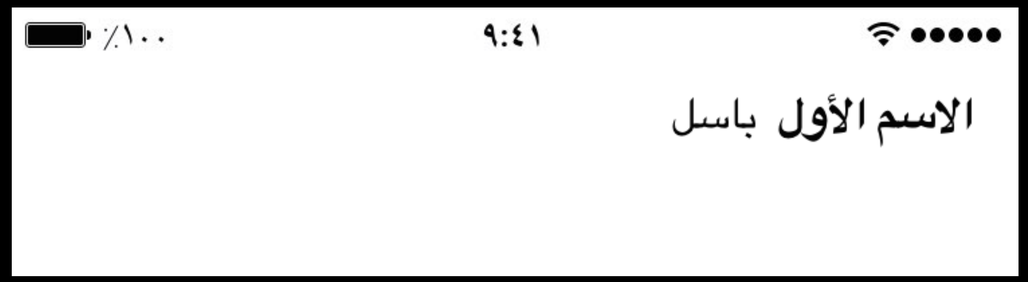
Auto Layout

Many reasons to use Auto Layout

- Available since iOS 6
- Different screen sizes
- Split-screen multitasking
- Localization

One more reason–right-to-left!

Can be used in storyboards, programmatically, or both



Auto Layout

Auto Layout

Use leading and trailing constraints

Auto Layout

Use leading and trailing constraints

Xcode

Auto Layout

Use leading and trailing constraints

Xcode

Auto Layout

Use leading and trailing constraints

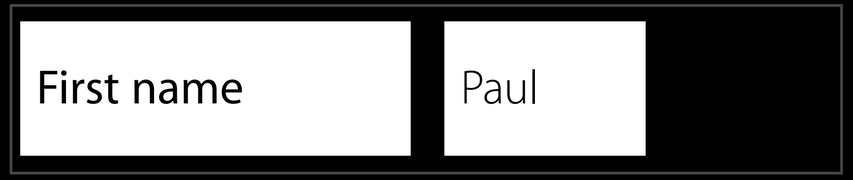
Xcode

Auto Layout

Use leading and trailing constraints

Xcode

Left-to-right

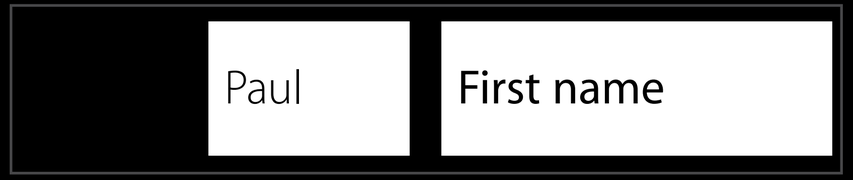


Auto Layout

Use leading and trailing constraints

Xcode

Right-to-left



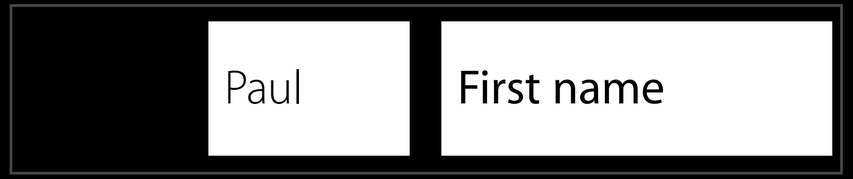
Auto Layout

Use leading and trailing constraints

Storyboards

Xcode

Right-to-left



Auto Layout

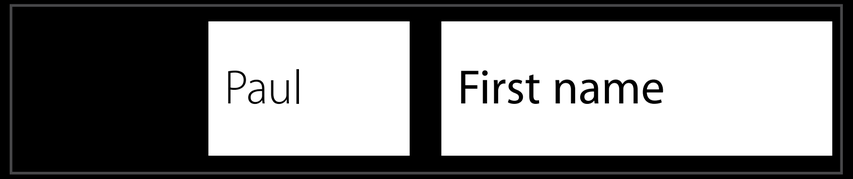
Use leading and trailing constraints

Storyboards

- The default

Xcode

Right-to-left



Auto Layout

Use leading and trailing constraints

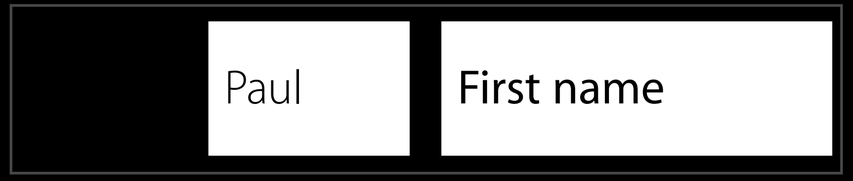
Storyboards

- The default

Code

Xcode

Right-to-left



Auto Layout

Use leading and trailing constraints

Storyboards

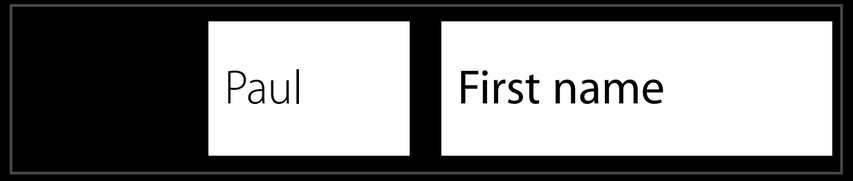
- The default

Code

- The default in visual format language

Xcode

Right-to-left



Auto Layout

Use leading and trailing constraints

Storyboards

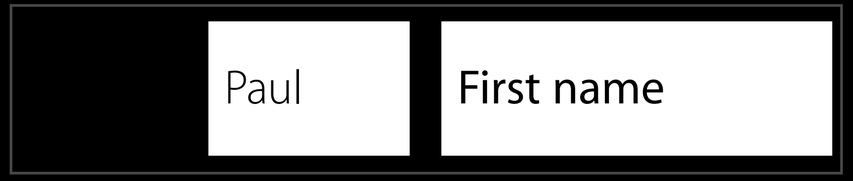
- The default

Code

- The default in visual format language
- Use explicitly for manual constraints and layout anchors

Xcode

Right-to-left



Animations

Flip your x-axis animations if using frames

- Not recommended

Animations

Flip your x-axis animations if using frames

- Not recommended

Animations

Flip your x-axis animations if using frames

- Not recommended

Use Auto Layout with leading and trailing constraints instead

Animations

Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

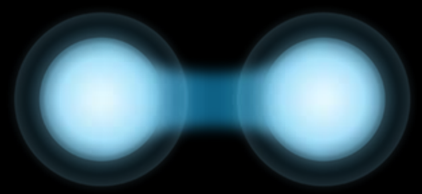
Animations

```
let duration = 0.5 // time in seconds
let newOffset = 10 // new constraint value to animate to
self.layoutIfNeeded() // make sure all frames are at the starting position
UIView.animateWithDuration(duration) {
    self.animatedConstraint?.constant = newOffset
    self.layoutIfNeeded() // layout again to update the frames
}
```

Tracking Gestures

Gesture recognizers remain unchanged

- Inherently physical, map-to-finger movement
- A "flipped" recognizer wouldn't make sense



Tracking Gestures

Tracking Gestures

Be aware of what's being manipulated in UI

Tracking Gestures

Be aware of what's being manipulated in UI

- Paintbrush on a canvas?

Tracking Gestures

Be aware of what's being manipulated in UI

- Paintbrush on a canvas?
- Table view cell?

Tracking Gestures

Be aware of what's being manipulated in UI

- Paintbrush on a canvas?
- Table view cell?

Tracking Gestures

Be aware of what's being manipulated in UI

- Paintbrush on a canvas?
- Table view cell?
- Navigation?

Tracking Gestures

Be aware of what's being manipulated in UI

- Paintbrush on a canvas?
- Table view cell?
- Navigation?

Make sure that position changes correspond to movement

Tracking Gestures

Be aware of what's being manipulated in UI

- Paintbrush on a canvas?
- Table view cell?
- Navigation?

Make sure that position changes correspond to movement

- Use Auto Layout

Demo

Custom layout

Exceptions

Views

Exceptions

Semantic content attributes

NEW

Exceptions

Semantic content attributes

NEW

```
var semanticContentAttribute: UISemanticContentAttribute
```

Exceptions

Semantic content attributes

NEW

```
var semanticContentAttribute: UISemanticContentAttribute
```

Not all UI flips

Exceptions

Semantic content attributes

NEW

```
var semanticContentAttribute: UISemanticContentAttribute
```

Not all UI flips

Default is `.Unspecified`

Exceptions

Semantic content attributes

NEW

```
var semanticContentAttribute: UISemanticContentAttribute
```

Not all UI flips

Default is `.Unspecified`

Some UI needs different semantic content attribute for correct layout

Exceptions

Semantic content attributes

NEW

```
var semanticContentAttribute: UISemanticContentAttribute
```

Not all UI flips

Default is `.Unspecified`

Some UI needs different semantic content attribute for correct layout

Affects resolution of leading and trailing constraints

Exceptions

Semantic content attributes

NEW

`UISemanticContentAttribute.Playback`

Containers of playback controls, playhead scrubbers, etc.

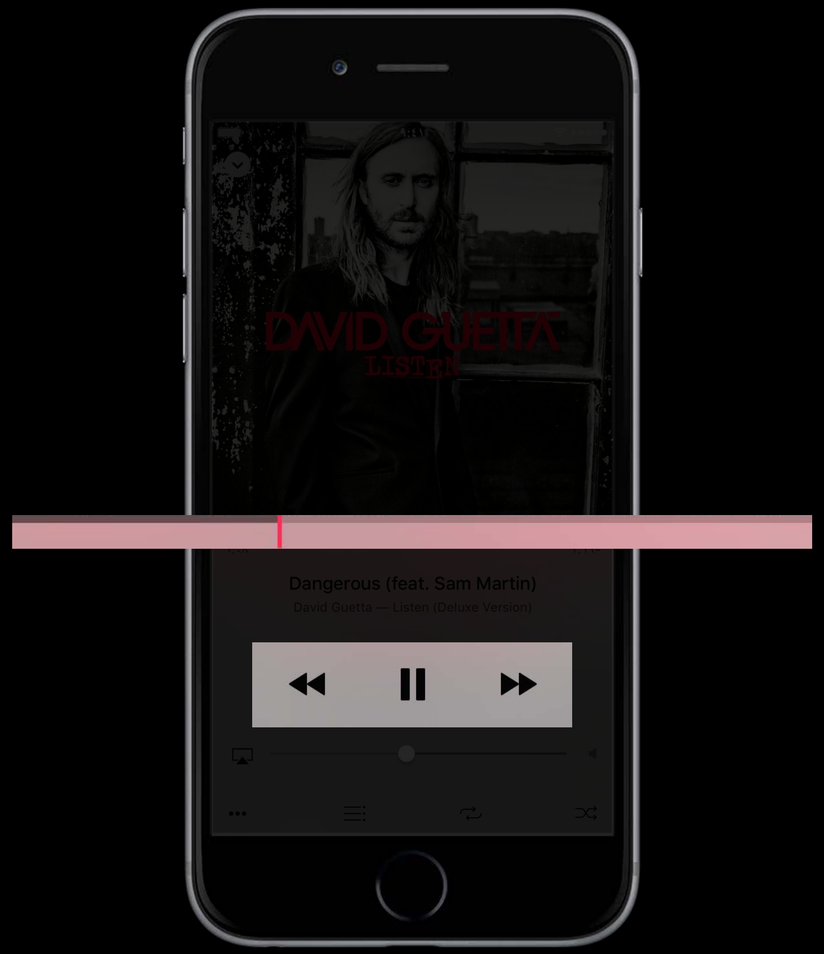
Exceptions

Semantic content attributes

NEW

`UISemanticContentAttribute.Playback`

Containers of playback controls, playhead scrubbers, etc.



Exceptions

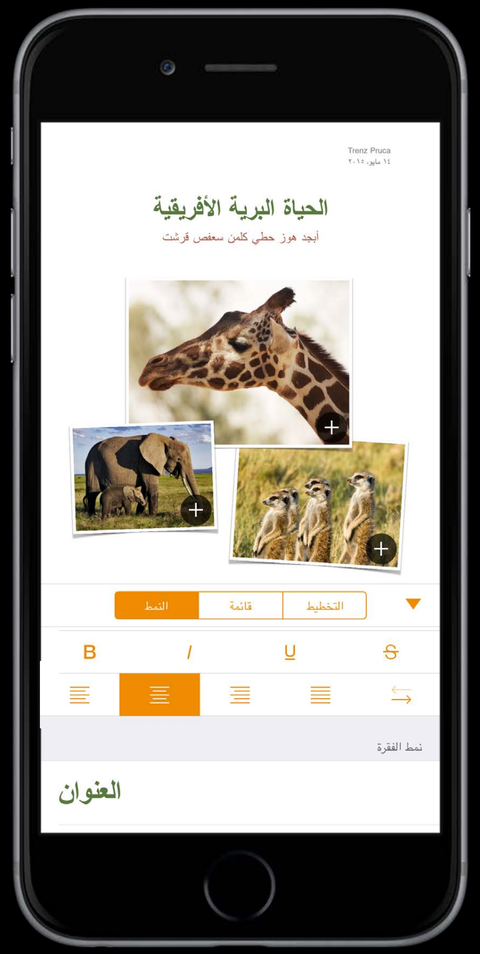
Semantic content attributes

NEW

`UISemanticContentAttribute.Spatial`

Groups of controls for manipulating objects or directional input on the screen

- Game controllers
- Text alignment controls



Exceptions

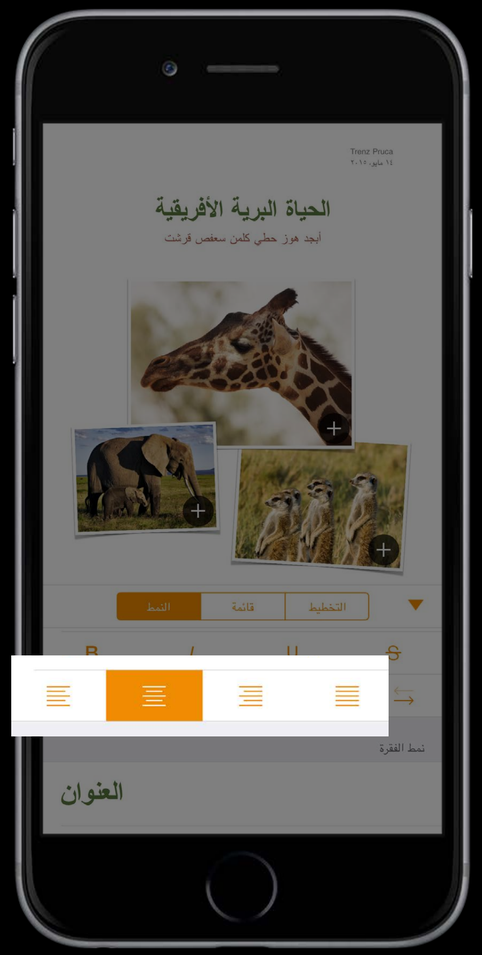
Semantic content attributes

NEW

`UISemanticContentAttribute.Spatial`

Groups of controls for manipulating objects or directional input on the screen

- Game controllers
- Text alignment controls



Exceptions

Semantic content attributes

NEW

Exceptions

Semantic content attributes

NEW

```
UISemanticContentAttribute.ForceLeftToRight
                          .ForceRightToLeft
```

Exceptions

Semantic content attributes

NEW

```
UISemanticContentAttribute.ForceLeftToRight
                          .ForceRightToLeft
```

Explicitly set the layout direction you want

Exceptions

Semantic content attributes

NEW

```
UISemanticContentAttribute.ForceLeftToRight
                           .ForceRightToLeft
```

Explicitly set the layout direction you want

Only `.ForceRightToLeft` affects layout in left-to-right localizations

Exceptions

Semantic content attributes

NEW

```
UISemanticContentAttribute.ForceLeftToRight
                           .ForceRightToLeft
```

Explicitly set the layout direction you want

Only `.ForceRightToLeft` affects layout in left-to-right localizations

Come talk to us in a lab if you want to use these

Best Practices

User interface and text

Right-to-Left User Interface

Right-to-Left User Interface

Use formatters for region-appropriate formatting

Right-to-Left User Interface

Use formatters for region-appropriate formatting

Right-to-Left User Interface

Use formatters for region-appropriate formatting

Never use `NSLocale` or `NSBundle` for UI layout branching

Right-to-Left User Interface

Use formatters for region-appropriate formatting

Never use **`NSLocale`** or **`NSBundle`** for UI layout branching

```
let preferredLang = NSLocale.preferredLanguages().first!
if NSLocale.characterDirectionForLanguage(preferredLang) == .RightToLeft {
    // ...
}
```

Right-to-Left User Interface

Use formatters for region-appropriate formatting

Never use **`NSLocale`** or **`NSBundle`** for UI layout branching

```
let preferredLang = NSLocale.preferredLanguages().first!
if NSLocale.characterDirectionForLanguage(preferredLang) == .RightToLeft {
    // ...
}
```

Right-to-Left User Interface

NEW

Right-to-Left User Interface

NEW

```
class func userInterfaceLayoutDirectionForSemanticContentAttribute(
    attribute: UISemanticContentAttribute) -> UIUserInterfaceLayoutDirection
```

Right-to-Left User Interface

NEW

```
class func userInterfaceLayoutDirectionForSemanticContentAttribute(
    attribute: UISemanticContentAttribute) -> UIUserInterfaceLayoutDirection
```

For custom UI layout

- Do not use to determine regional or formatting settings

Right-to-Left User Interface

NEW

```
class func userInterfaceLayoutDirectionForSemanticContentAttribute(
    attribute: UISemanticContentAttribute) -> UIUserInterfaceLayoutDirection
```

For custom UI layout

- Do not use to determine regional or formatting settings

```
let semanticAttr = myView.semanticContentAttribute
let layoutDirection = UIView.userInterfaceLayoutDirectionForSemanticContentAttribute(semanticAttr)
if layoutDirection == .RightToLeft {
    // ...
}
```

Right-to-Left User Interface

NEW

```
class func userInterfaceLayoutDirectionForSemanticContentAttribute(
    attribute: UISemanticContentAttribute) -> UIUserInterfaceLayoutDirection
```

For custom UI layout

- Do not use to determine regional or formatting settings

```
let semanticAttr = myView.semanticContentAttribute
let layoutDirection = UIView.userInterfaceLayoutDirectionForSemanticContentAttribute(semanticAttr)
if layoutDirection == .RightToLeft {
    // ...
}
```

Right-to-Left Text

NEW

Right-to-Left Text

NEW

Leave alignment and directionality at their default values

Right-to-Left Text

NEW

Leave alignment and directionality at their default values

- Natural alignment is now default on iOS 9

Right-to-Left Text

NEW

Leave alignment and directionality at their default values

- Natural alignment is now default on iOS 9
- Natural base writing direction is default since iOS 7

Right-to-Left Text

NEW

Leave alignment and directionality at their default values

- Natural alignment is now default on iOS 9
- Natural base writing direction is default since iOS 7

Do not make layout decisions based on the alignment or writing direction

Exceptions

Images

Exceptions

Images

NEW

Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

Exceptions

Images

NEW

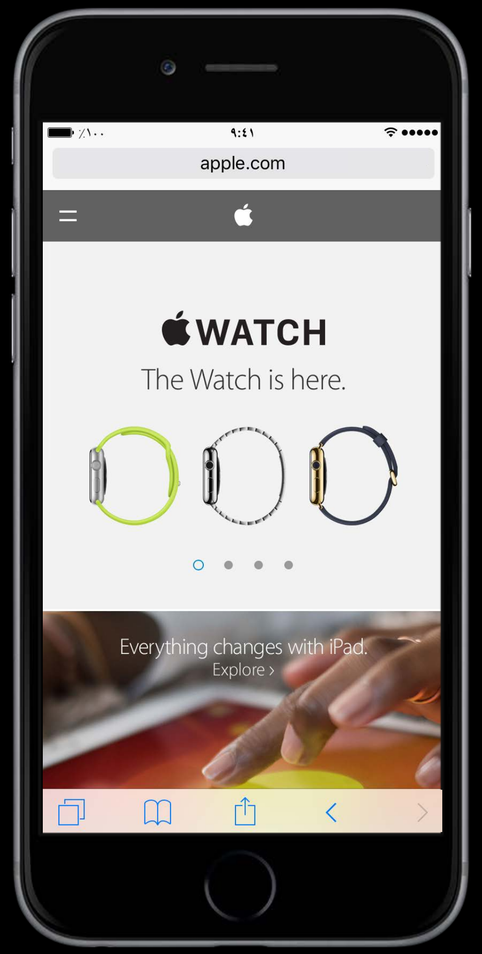
```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons



Exceptions

Images

NEW

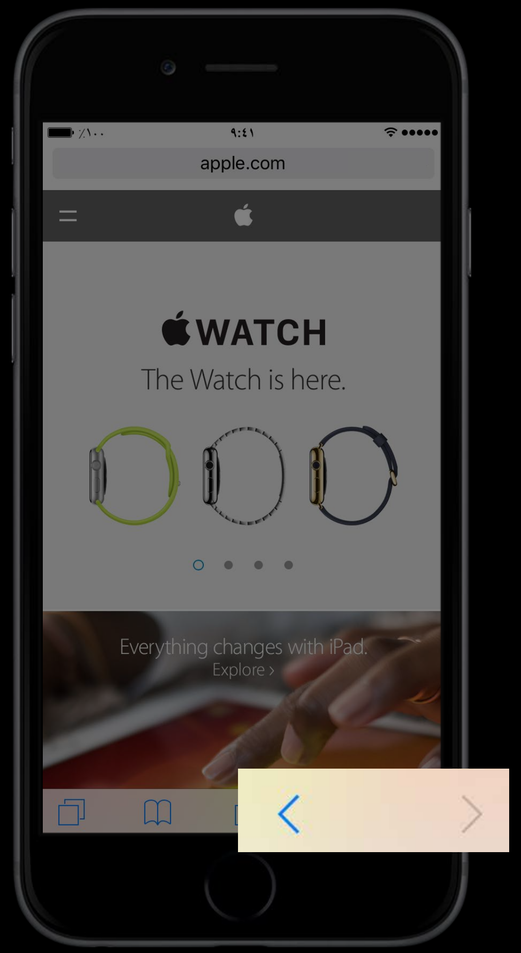
```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons



Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons

Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons
- Some UI icons

Exceptions

Images

NEW

```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons
- Some UI icons

Exceptions

Images

NEW

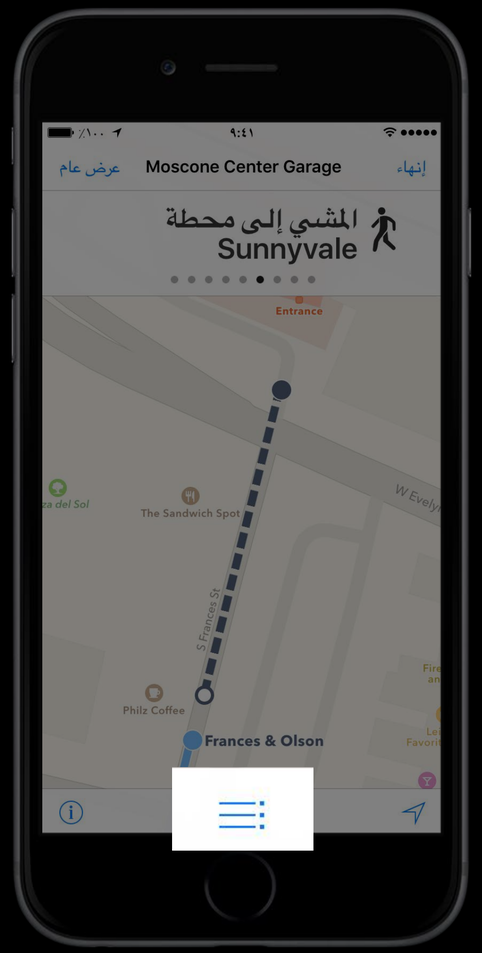
```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons
- Some UI icons



Exceptions

Images

NEW

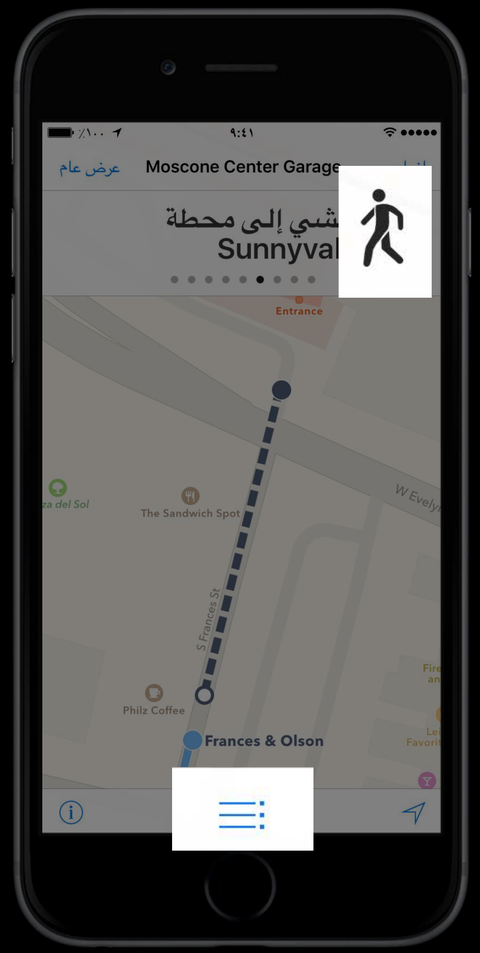
```
func imageFlippedForRightToLeftLayoutDirection() -> UIImage
```

Horizontally flips image in a right-to-left context

- Obeys the `UIImageView`'s semantic content attribute

Only for directional images

- Arrows
- Chevrons
- Some UI icons



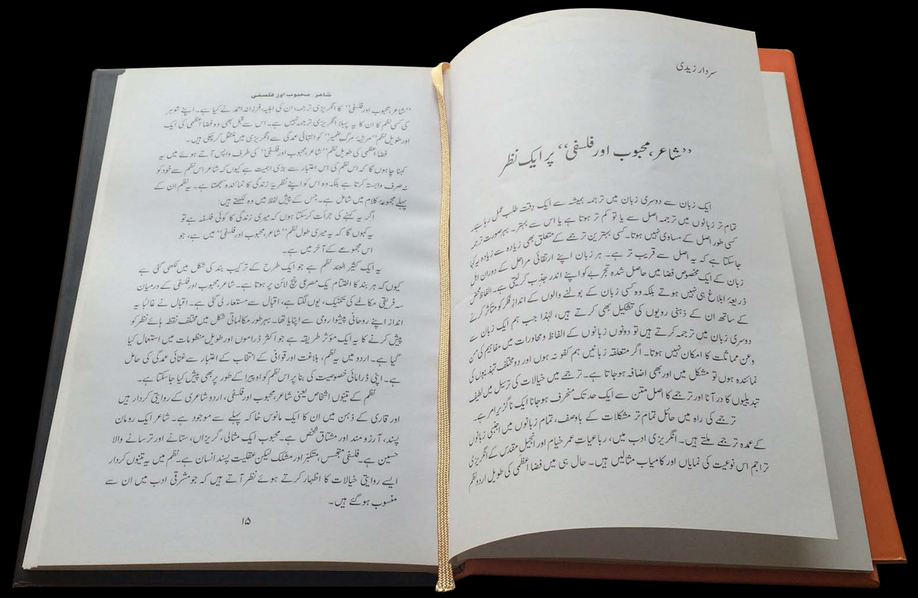
Demo

Exceptions and best practices

Conclusion

Summary

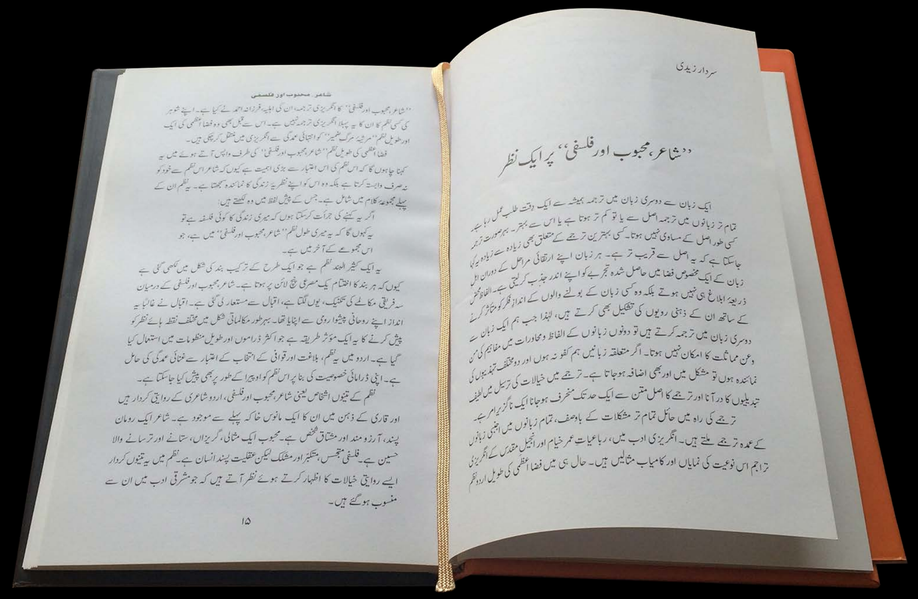
Natives of right-to-left languages expect right-to-left UI



Summary

Natives of right-to-left languages expect right-to-left UI

Perfect opportunity to add right-to-left localizations

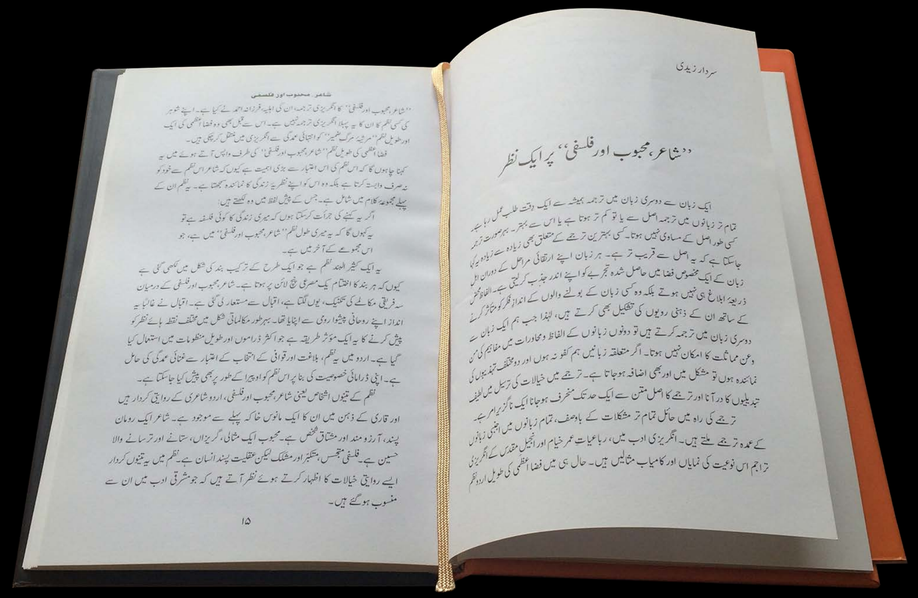


Summary

Natives of right-to-left languages expect right-to-left UI

Perfect opportunity to add right-to-left localizations

API accessible to non-natives



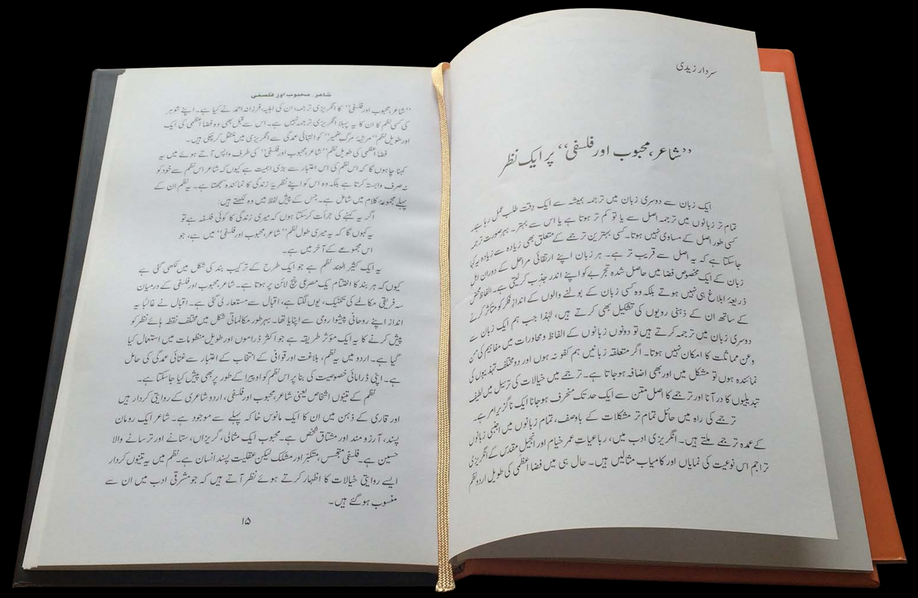
Summary

Natives of right-to-left languages expect right-to-left UI

Perfect opportunity to add right-to-left localizations

API accessible to non-natives

Reach millions of users in new markets



More Information

Documentation and Videos

Internationalization Guide

http://developer.apple.com/internationalization/

Technical Support

Apple Developer Forums

http://developer.apple.com/forums

Related Sessions

Mysteries of Auto Layout, Part 1	Presidio	Thursday 11:00 AM
Mysteries of Auto Layout, Part 2	Presidio	Thursday 1:30 PM
What's New in Internationalization	Pacific Heights	Friday 9:00 AM
Cocoa Touch Best Practices	Presidio	Friday 1:30 PM

Labs

Interface Builder and Auto Layout Lab	Developer Tools Lab C	Now
Internationalization Lab	Frameworks Lab A	Friday 11:00 AM

WWDC15