Beyond size classes

Making better use of large screens

Douglas Hill @qdoug

iOSDevUK

September 2019

09:29 Wed 21 Aug

apple.com

iPad Pro

Overview Design Why iPad Tech Specs Buy

11″

12.9″

09:55 Fri 30 Aug

twitter.com

By using Twitter's services you agree to our Cookies Use. We and our partners operate globally and use cookies, including for analytics, personalisation, and ads.

Home

ccount? Log In

Federico Vi
@viticci

Founder and Edit
@MacStoriesNet
Co-host of @App
@DialogPodcast.
@_RelayFM. vitic

In your web br

macstories.ne

Joined Februa

Federico Viticci
@viticci

Follow

iPad Pro connected to a 4K UltraFine display with support for Bluetooth keyboard, mouse, and DualShock 4 controller.

Just need a single USB-C cable to go from tablet to dream iOS workstation, now with cursor support 😍

(I re-use the same accessories for my Mac mini.)

7:58 am - 19 Jun 2019

180 Retweets **1,149** Likes

88 180 1.1K

Federico Viticci @viticci · Jun 19

This setup is by no means perfect (needs proper display management, cursor is an Accessibility feature, etc.), but it totally makes it possible to work with iOS on a large display, which I find ideal for my eyesight, posture, and focus.

14 6 149

Max "Tim Van Damme" Voltar [emoji] @maxvoltar · Jun 19

Replying to @viticci

Federico Viticci @viticci · Jun 19

iPad Pro connected to a 4K UltraFine display with support for Bluetooth keyboard, mouse, and DualShock 4 controller.

Just need a single USB-C cable to go from tablet to dream iOS workstation, now with cursor support 😍

(I re-use the same accessories for my Mac mini.)

88 180 1.1K

macOS Preview

DC Universe | Morpholio | Post-it® | Jira | American Airlines | Fender | TripIt

With this project management tool, take advantage of native Mac features like push notifications, keyboard shortcuts, Spotlight search, drag and drop, and much more. Use the custom menu bar and toolbars to easily manage projects across all devices.

Overview

- Goals
- iOS design patterns
- 5 design techniques with basic implementation and examples

10:14 Fri 30 Aug

Today

PDF Viewer - Annotation Expert

Fill Forms, Sign, Edit, Redact

OPEN

4.7 ★★★★★
368 Ratings

4+
Age

What's New

Version History

Improves performance when opening large documents, and fixes annotation reordering not always being possible.
Here's the full list of changes:

more

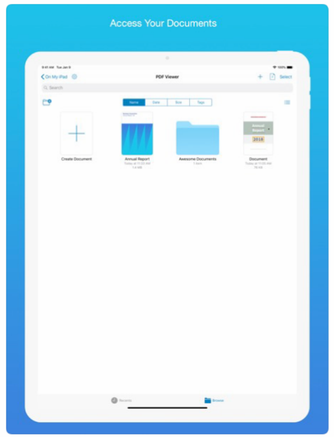
3w ago
Version 3.6.1

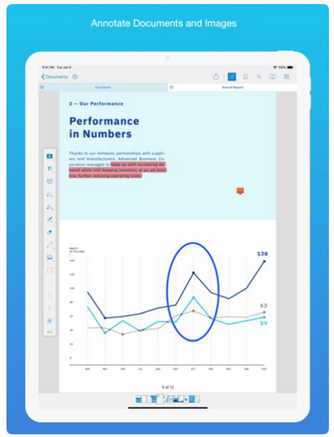
Subscriptions

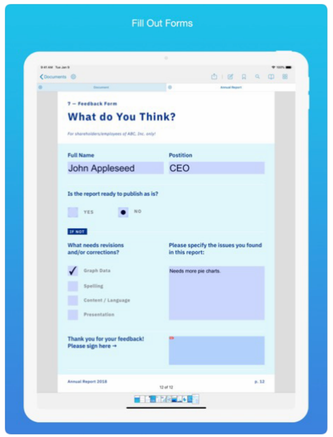
PDF Viewer Pro 3 Months
Pro features: Redaction, PDF Merging & more!
£6.49

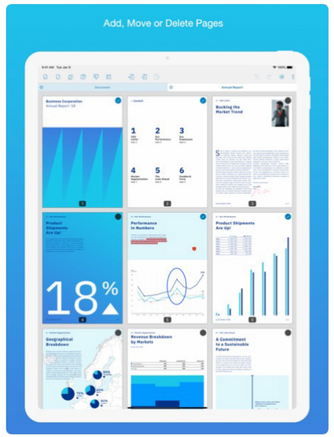
PDF Viewer Pro Yearly
Pro features: Redaction, PDF Merging & more!
SUBSCRIBED

Preview









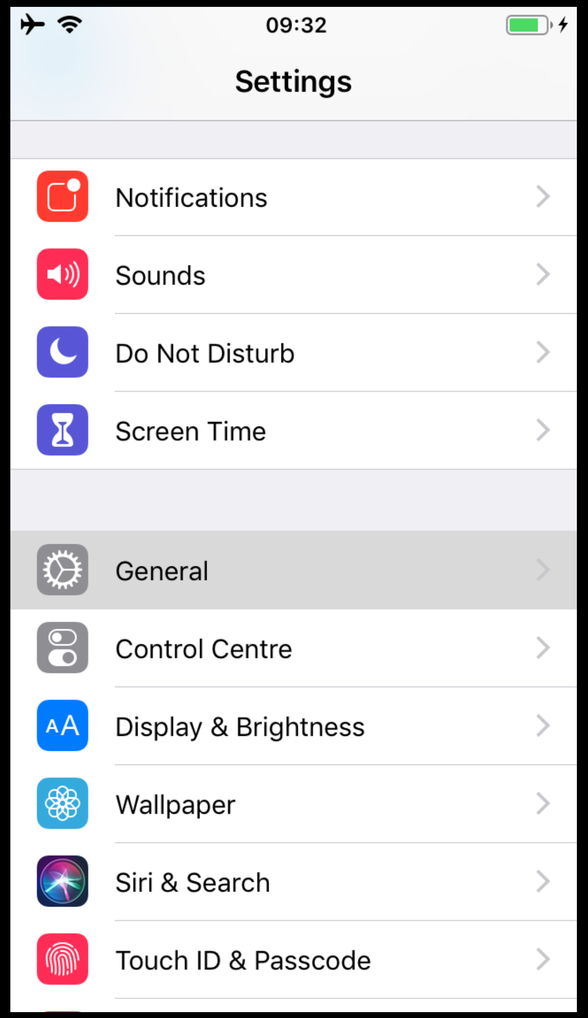
Offers iPhone and iMessage Apps

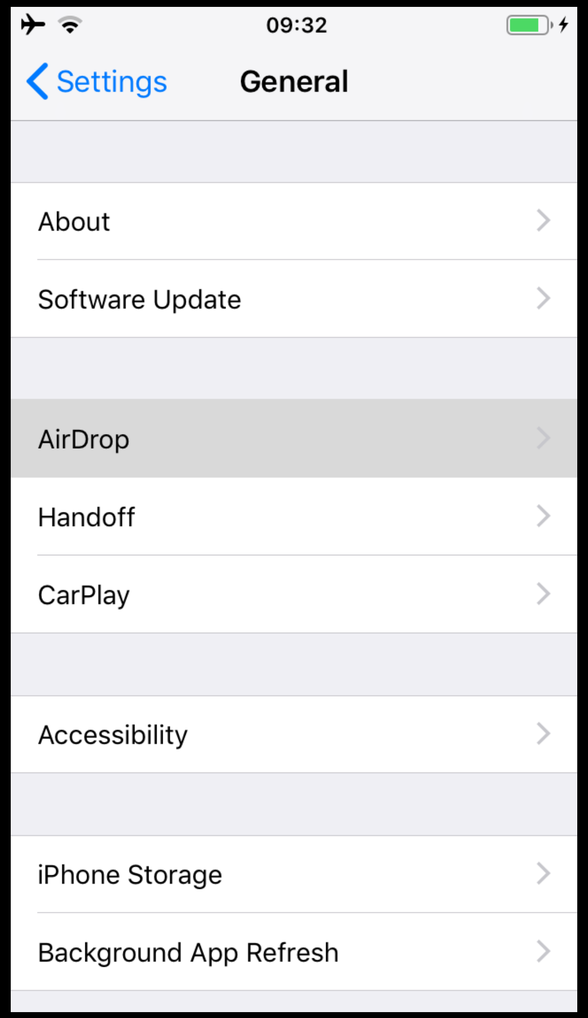
Today | Games | Apps | Updates | Search

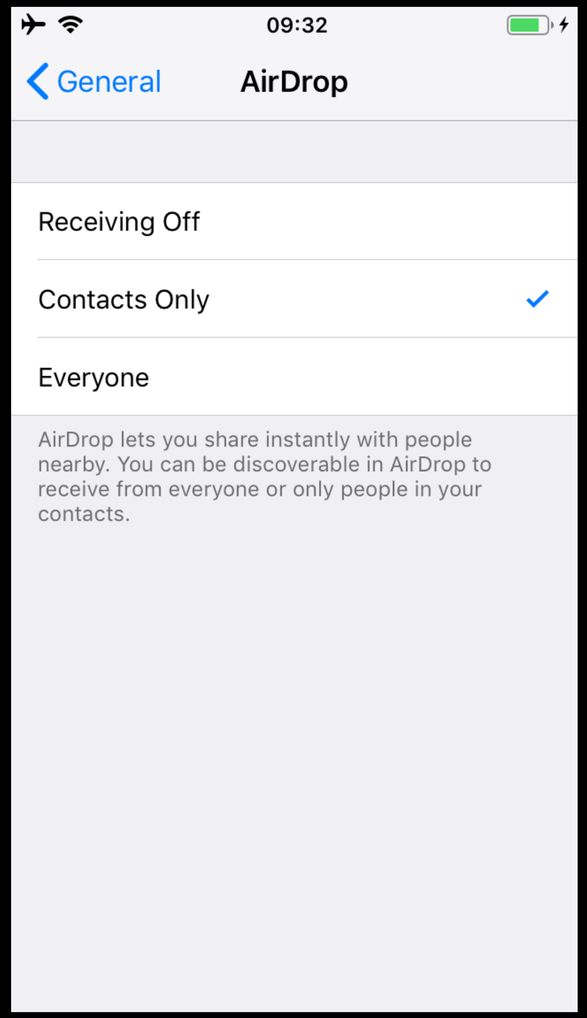
Goals

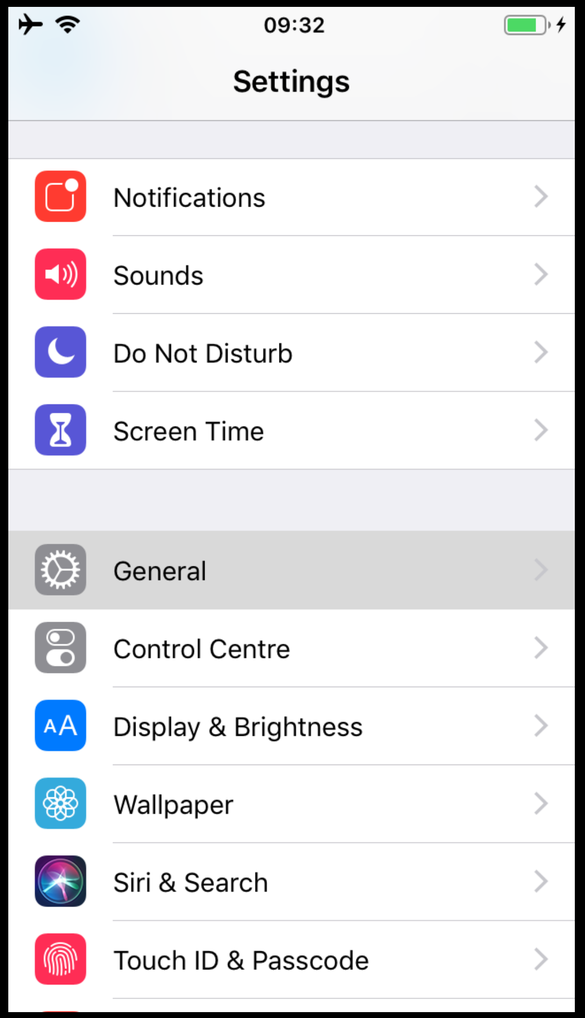
- Show more at once
- Maintain readability
- Avoid overwhelming

iPhone design patterns


09:32
Settings
Notifications
Sounds
Do Not Disturb
Screen Time
General
Control Centre
Display & Brightness
Wallpaper
Siri & Search
Touch ID & Passcode


09:32
Settings
General
About
Software Update
AirDrop
Handoff
CarPlay
Accessibility
iPhone Storage
Background App Refresh

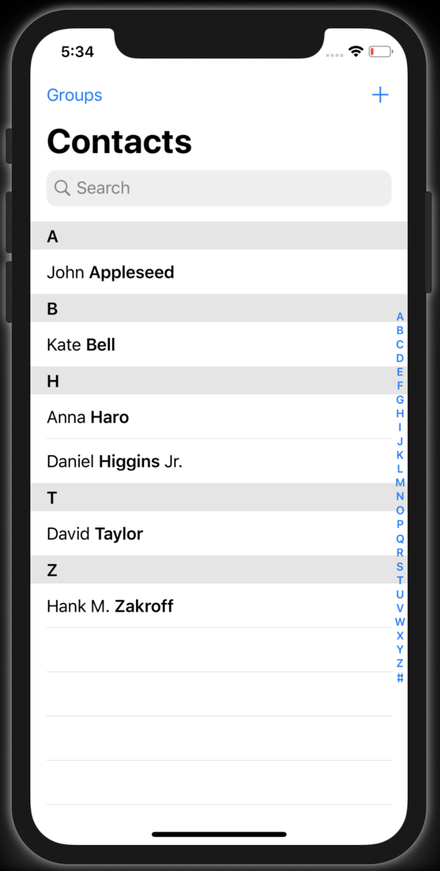

09:32
General
AirDrop
Receiving Off
Contacts Only
Everyone
AirDrop lets you share instantly with people nearby. You can be discoverable in AirDrop to receive from everyone or only people in your contacts.

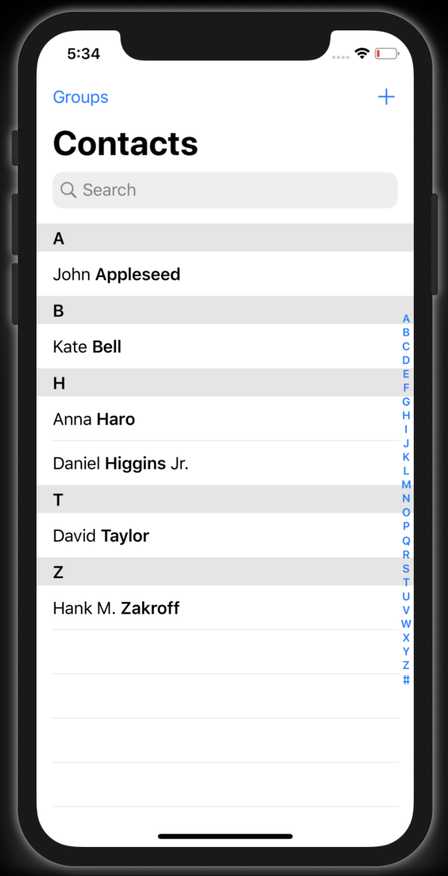

09:32
Settings
Notifications
Sounds
Do Not Disturb
Screen Time
General
Control Centre
Display & Brightness
Wallpaper
Siri & Search
Touch ID & Passcode

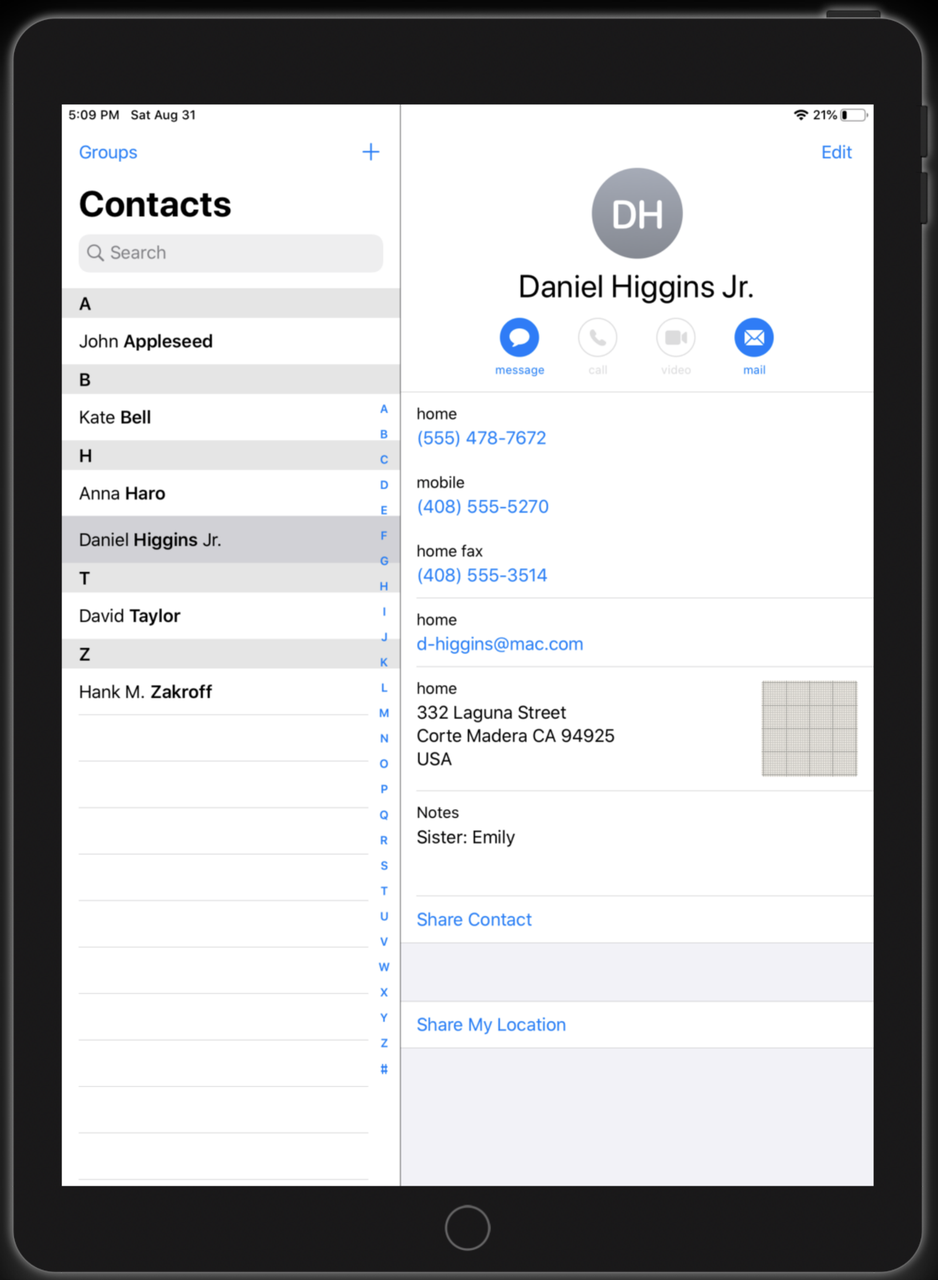
40 – 80 characters

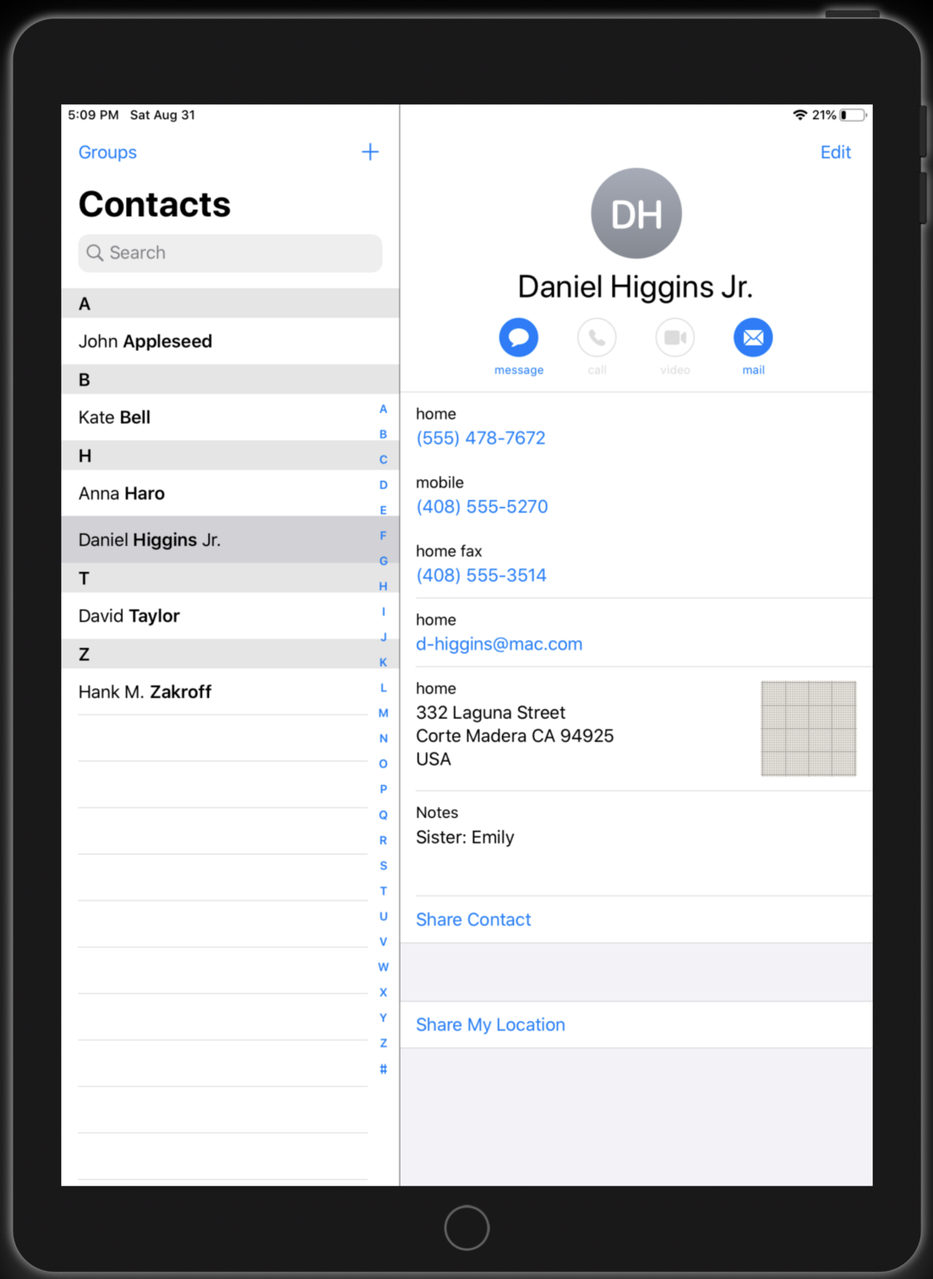
iPad design patterns

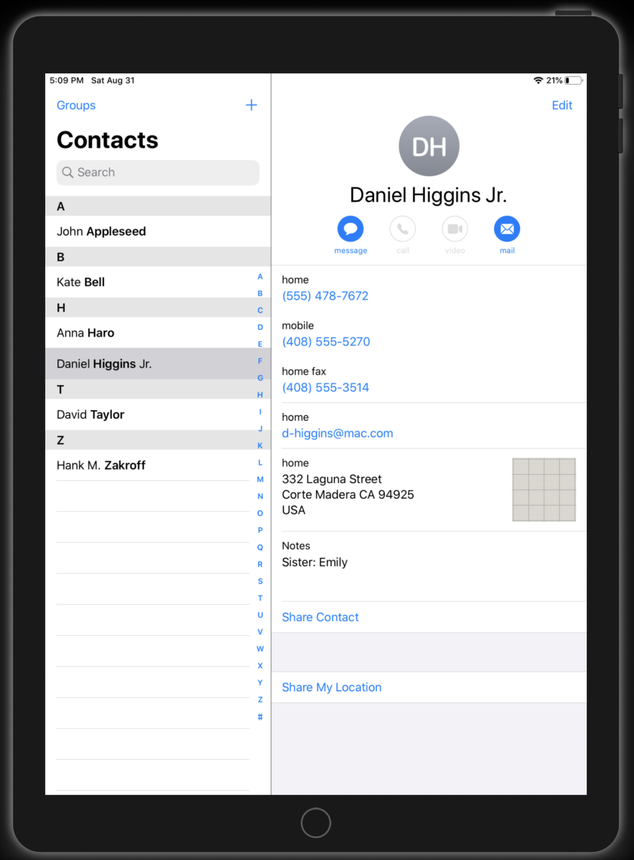
Size classes & `UISplitViewController`

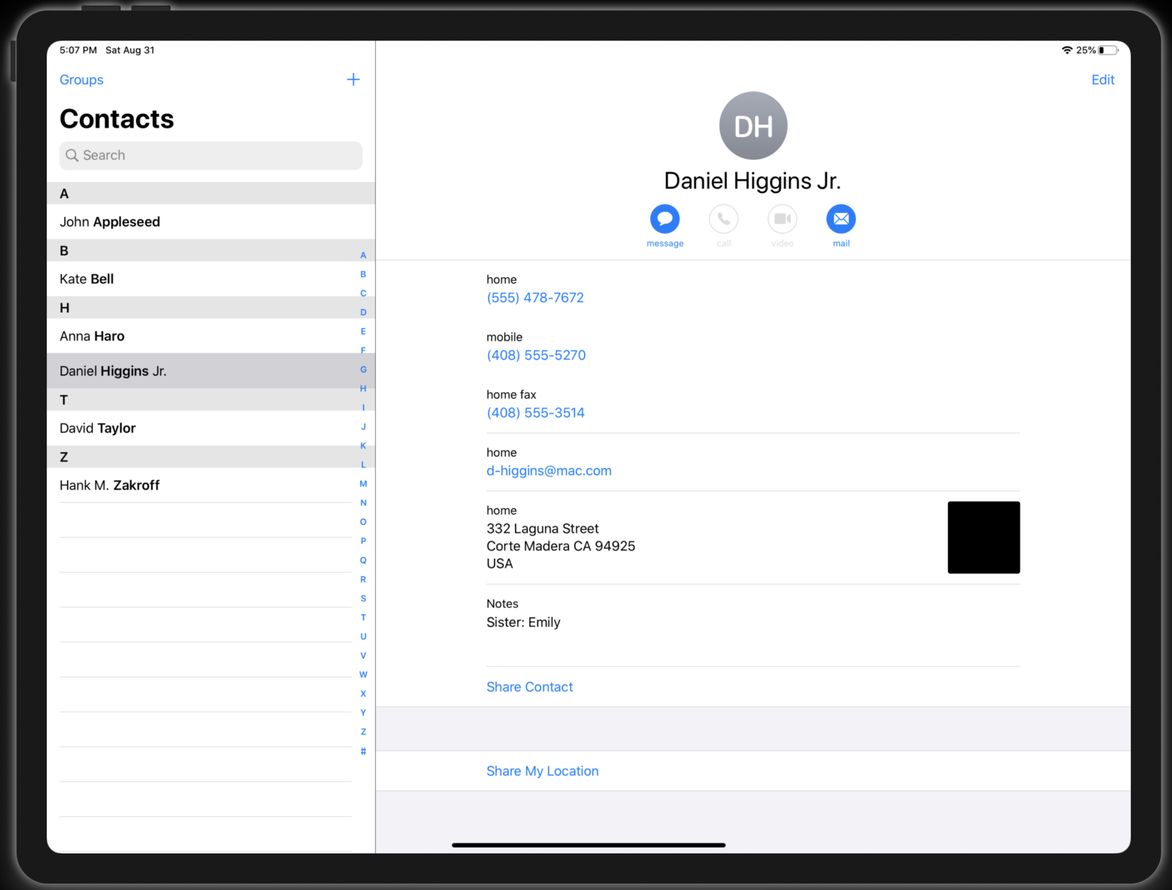

5:34
Groups
Contacts
Search
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #

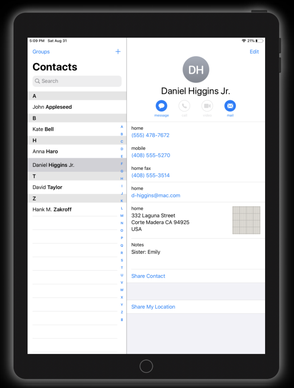

5:34
Groups
Contacts
Search
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #

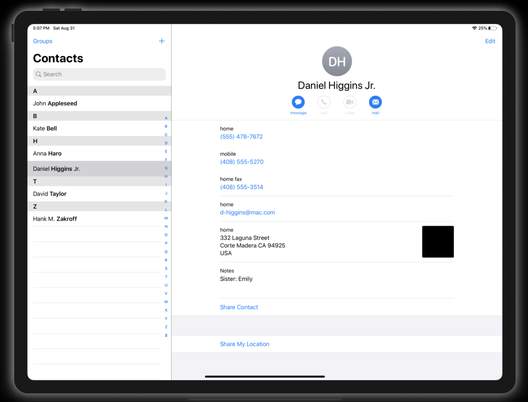

5:09 PM Sat Aug 31
21%
Groups
Contacts
Search
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #
Edit
DH
Daniel Higgins Jr.
message
call
video
mail
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514
home
d-higgins@mac.com
home
332 Laguna Street
Corte Madera CA 94925
USA
Notes
Sister: Emily
Share Contact
Share My Location


5:09 PM Sat Aug 31
21%
Groups
Contacts
Search
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #
Edit
DH
Daniel Higgins Jr.
message
call
video
mail
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514
home
d-higgins@mac.com
home
332 Laguna Street
Corte Madera CA 94925
USA
Notes
Sister: Emily
Share Contact
Share My Location


5:09 PM Sat Aug 31
21%
Groups
Edit
Contacts
Search
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
DH
Daniel Higgins Jr.
message
call
video
mail
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514
home
d-higgins@mac.com
home
332 Laguna Street
Corte Madera CA 94925
USA
Notes
Sister: Emily
Share Contact
Share My Location


5:07 PM Sat Aug 31
25%
Groups
Edit
Contacts
Search
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
DH
Daniel Higgins Jr.
message
call
video
mail
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514
home
d-higgins@mac.com
home
332 Laguna Street
Corte Madera CA 94925
USA
Notes
Sister: Emily
Share Contact
Share My Location


Groups
Contacts
Edit
DH
Daniel Higgins Jr.
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514
home
d-higgins@mac.com
home
332 Laguna Street
Corte Madera CA 94925
USA
Notes
Sister: Emily
Share Contact
Share My Location


Groups
Contacts
Edit
DH
Daniel Higgins Jr.
A
John Appleseed
B
Kate Bell
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
Hank M. Zakroff
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514
home
d-higgins@mac.com
home
332 Laguna Street
Corte Madera CA 94925
USA
Notes
Sister: Emily
Share Contact
Share My Location

Main

Edit

Inbox

Search

// TODO London 14:23

// TODO 2.10 🍎🚲 on Wednesday, September 11

Fabio scheduled a new event in // TODO London Just now // TODO 2.10 🍎🚲 // TODO London Wednesday, September 11, 2019 6:30 PM to 8:30 PM Intercom 80 Great Eastern St London EC2A 3JL Attend Details Please join us at on Wednesday the 11th of September for our //...

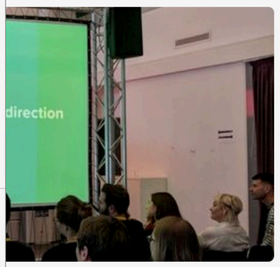
Updated Just Now

Hide

L

September 11

TODO London Just now



Main Edit

Inbox

Search

// TODO London 14:23
// TODO 2.10 🍎🚲 on Wednesday, September 11
Fabio scheduled a new event in // TODO London Just now // TODO 2.10 🍎🚲 // TODO London Wednesday, September 11, 2019 6:30 PM to 8:30 PM Intercom 80 Great Eastern St London EC2A 3JL Attend Details Please join us at on Wednesday the 11th of September for our //...

Updated Just Now

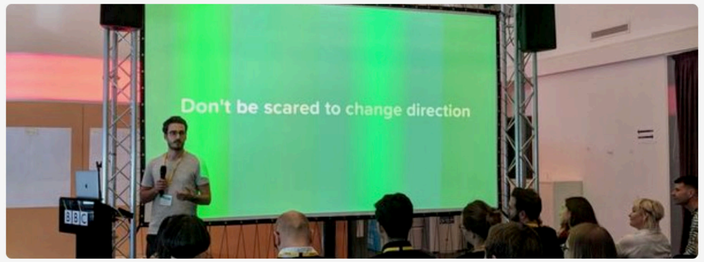
From: // TODO London
To: Douglas Hill Hide

// TODO 2.10 🍎🚲 on Wednesday, September 11

Today at 14:23

Meetup

Fabio scheduled a new event in // TODO London Just now



// TODO 2.10 🍎🚲

// TODO London

Wednesday, September 11, 2019
6:30 PM to 8:30 PM

Intercom
80 Great Eastern St
London EC2A 3JL

Attend

Details

18:58 Sat 31 Aug

Edit

Mailboxes

All Inboxes
Main
PSPDFKit
NSLondon
All Mail
PSPDFKit

MAIN

Inbox
Drafts
Sent
Junk
Trash
Archive
Current
Mark as Spam
Notes
Payments
Reading

PSPDFKIT

NSLONDON

Inbox
Drafts
Sent
Junk

Edit

Inbox

Search

// TODO London 14:23
// TODO 2.10 🍎🚲 on Wednesday, September 11
Fabio scheduled a new event in // TODO London Just now // TODO 2.10 🍎🚲 // TODO London Wednesday, September 11, 2019 6:30 PM to 8:30 PM Intercom 80 Great Eastern St London EC2A 3JL Attend Details Please join us at on Wednesday the 11th of September for our //...

Updated Just Now

From: // TODO London
To: Douglas Hill
Hide

// TODO 2.10 🍎🚲 on Wednesday, September 11

Today at 14:23

Meetup

Fabio scheduled a new event in // TODO London Just now

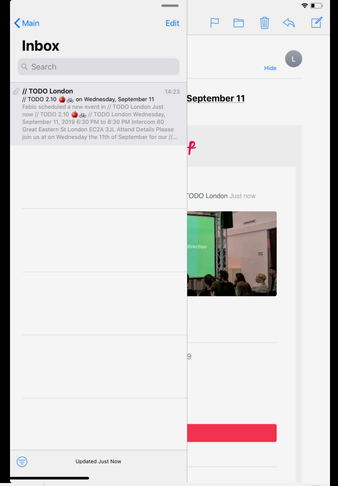
// TODO 2.10 🍎🚲

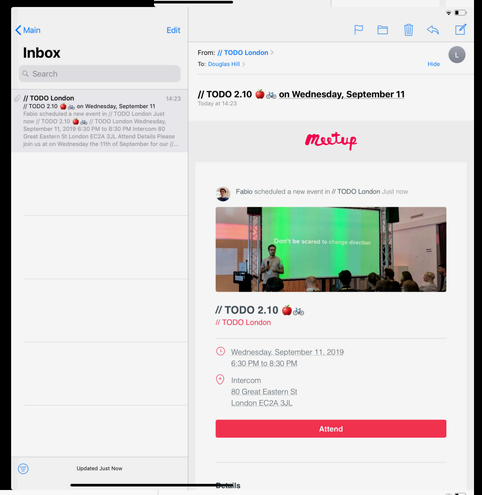
// TODO London

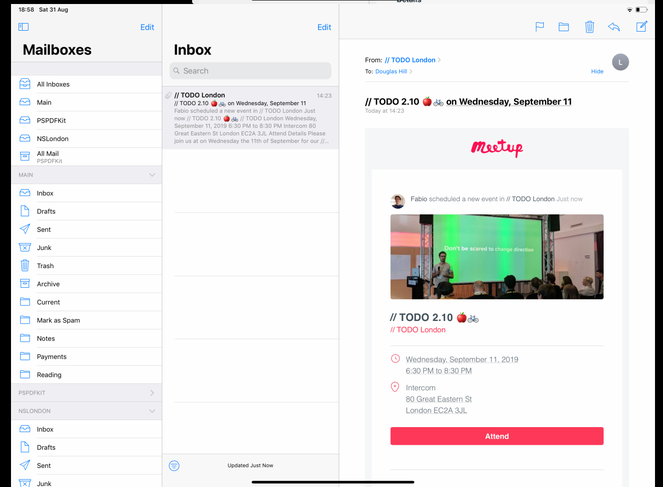
Wednesday, September 11, 2019
6:30 PM to 8:30 PM

Intercom
80 Great Eastern St
London EC2A 3JL

Attend


Main
Edit
Inbox
Search
// TODO London
14:23
// TODO 2.10 on Wednesday, September 11
Fabio scheduled a new event in // TODO London Just now // TODO 2.10 // TODO London Wednesday, September 11, 2019 6:30 PM to 8:30 PM Intercom 80 Great Eastern St London EC2A 3JL Attend Details Please join us at on Wednesday the 11th of September for our //...
Updated Just Now
Hide
September 11
TODO London Just now


Main
Edit
Inbox
Search
// TODO London
14:23
// TODO 2.10 on Wednesday, September 11
Fabio scheduled a new event in // TODO London Just now // TODO 2.10 // TODO London Wednesday, September 11, 2019 6:30 PM to 8:30 PM Intercom 80 Great Eastern St London EC2A 3JL Attend Details Please join us at on Wednesday the 11th of September for our //...
Updated Just Now
From: // TODO London
To: Douglas Hill
Hide
// TODO 2.10 on Wednesday, September 11
Today at 14:23
Meetup
Fabio scheduled a new event in // TODO London Just now
// TODO 2.10
// TODO London
Wednesday, September 11, 2019
6:30 PM to 8:30 PM
Intercom
80 Great Eastern St
London EC2A 3JL
Attend


18:58 Sat 31 Aug
Edit
Mailboxes
All Inboxes
Main
PSPDFKit
NSLondon
All Mail
PSPDFKit
MAIN
Inbox
Drafts
Sent
Junk
Trash
Archive
Current
Mark as Spam
Notes
Payments
Reading
PSPDFKIT
NSLONDON
Inbox
Drafts
Sent
Junk
Edit
Inbox
Search
// TODO London
14:23
// TODO 2.10 on Wednesday, September 11
Fabio scheduled a new event in // TODO London Just now // TODO 2.10 // TODO London Wednesday, September 11, 2019 6:30 PM to 8:30 PM Intercom 80 Great Eastern St London EC2A 3JL Attend Details Please join us at on Wednesday the 11th of September for our //...
Updated Just Now
From: // TODO London
To: Douglas Hill
Hide
// TODO 2.10 on Wednesday, September 11
Today at 14:23
Meetup
Fabio scheduled a new event in // TODO London Just now
// TODO 2.10
// TODO London
Wednesday, September 11, 2019
6:30 PM to 8:30 PM
Intercom
80 Great Eastern St
London EC2A 3JL
Attend

09:37 Wed 21 Aug

Settings

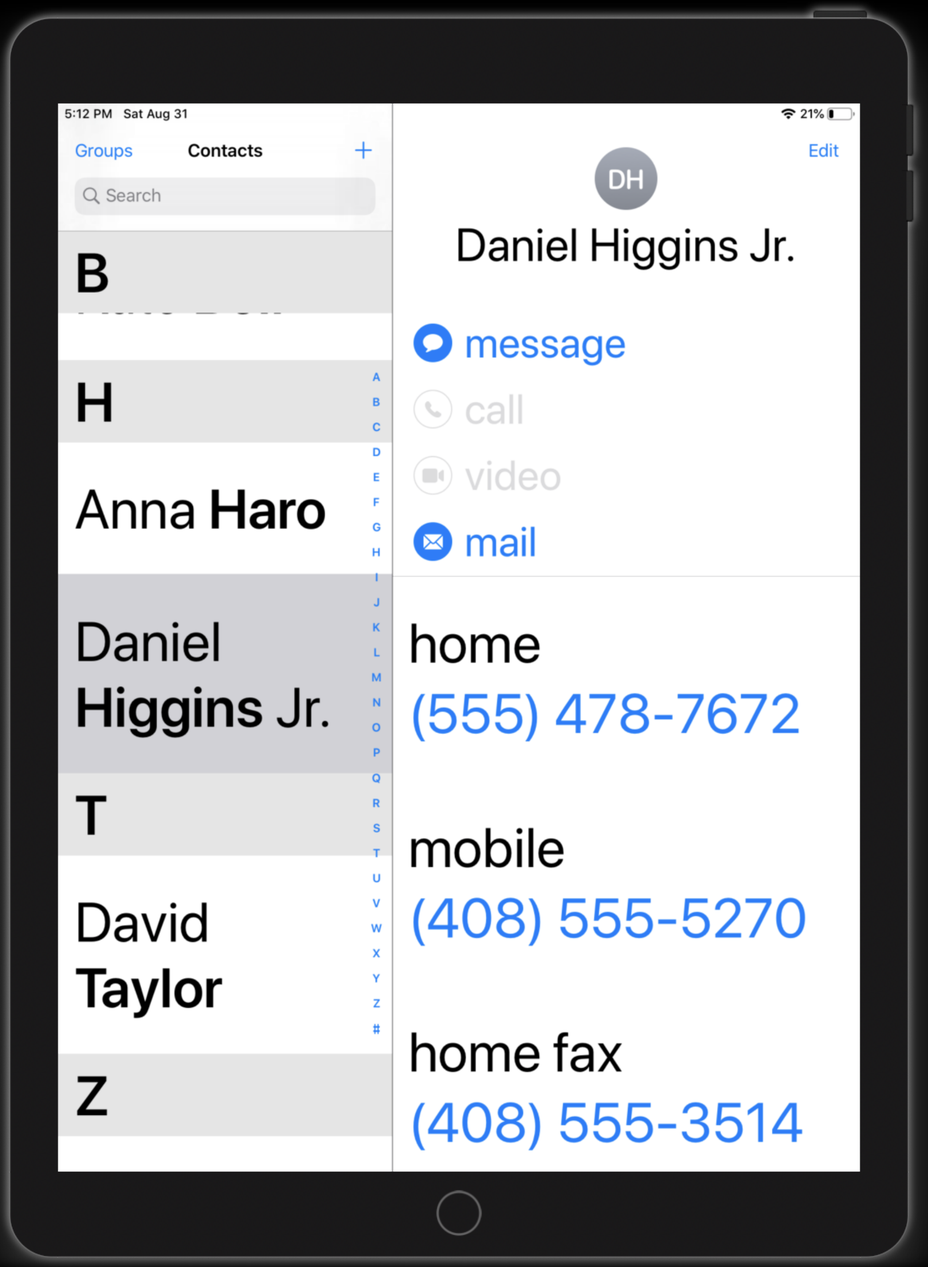
- Notifications
- Sounds
- Do Not Disturb
- Screen Time
- General
- Control Centre
- Display & Brightness
- Wallpaper
- Siri & Search
- Apple Pencil
- Face ID & Passcode

Display & Brightness

Text Size

A A

Apps that support Dynamic Type will adjust to your preferred reading size above.


5:12 PM Sat Aug 31
21%
Groups
Contacts
Search
B
H
Anna Haro
Daniel Higgins Jr.
T
David Taylor
Z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z #
Edit
DH
Daniel Higgins Jr.
message
call
video
mail
home
(555) 478-7672
mobile
(408) 555-5270
home fax
(408) 555-3514

Scalable

5 scalable design techniques

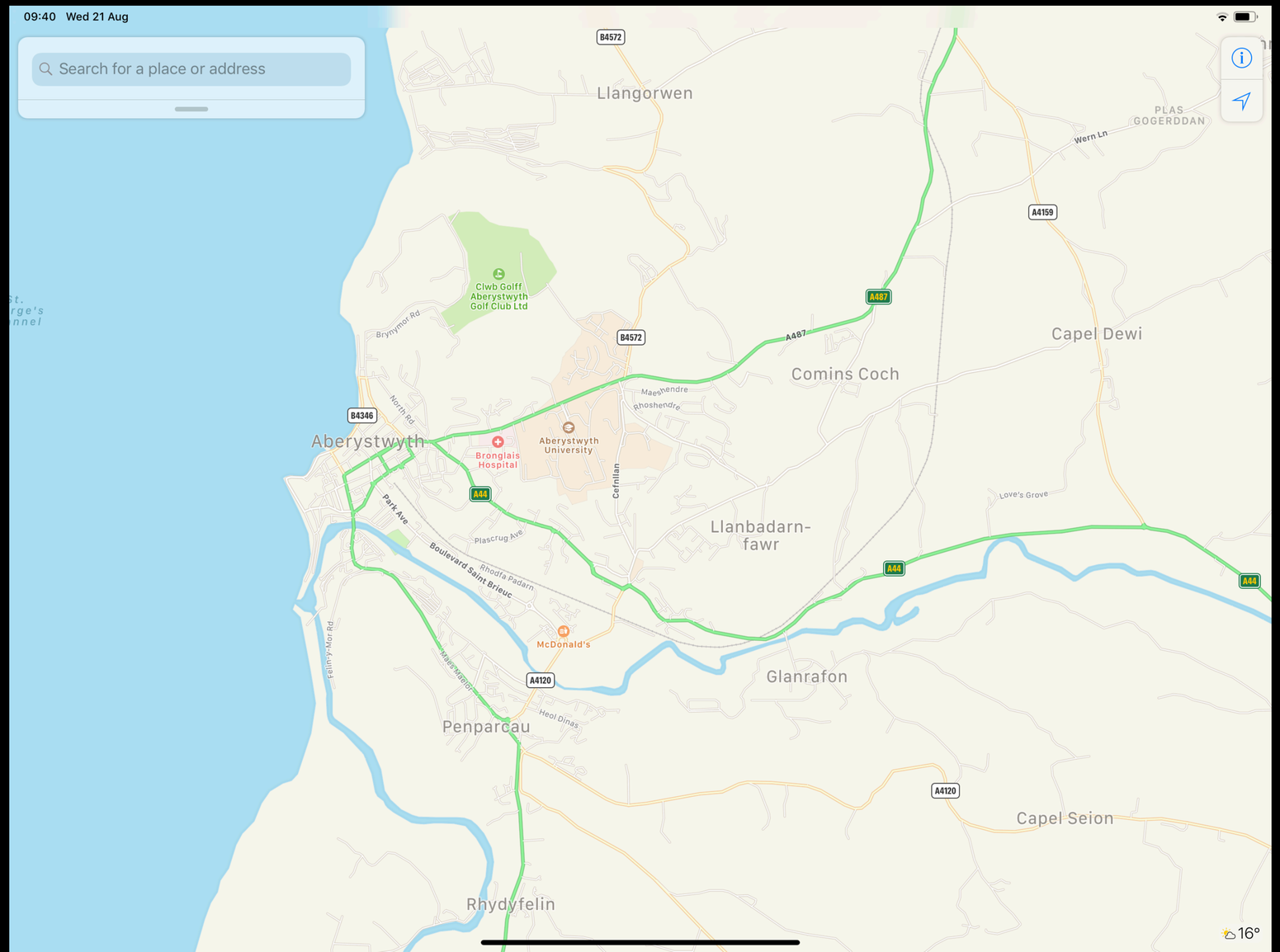
5 scalable design techniques

- Fill the space
- More whitespace
- Grid
- Columns across hierarchy
- Columns within hierarchy

Fill the space

Fill the space

Maps



More whitespace

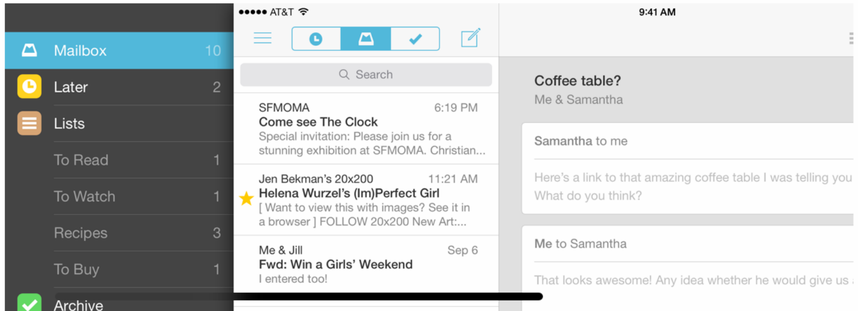
Designing Unread for iPad Part 5 – Comical Amounts of Negative Space

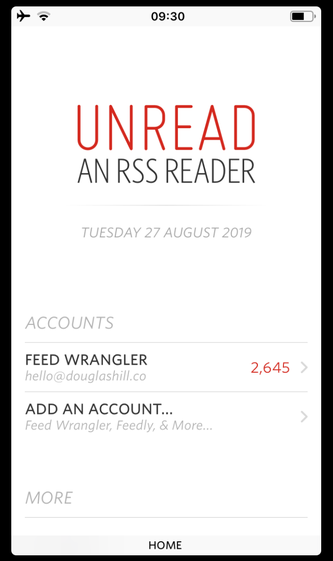
This is Part 5 in an ongoing series about the design of the iPad version of Unread.

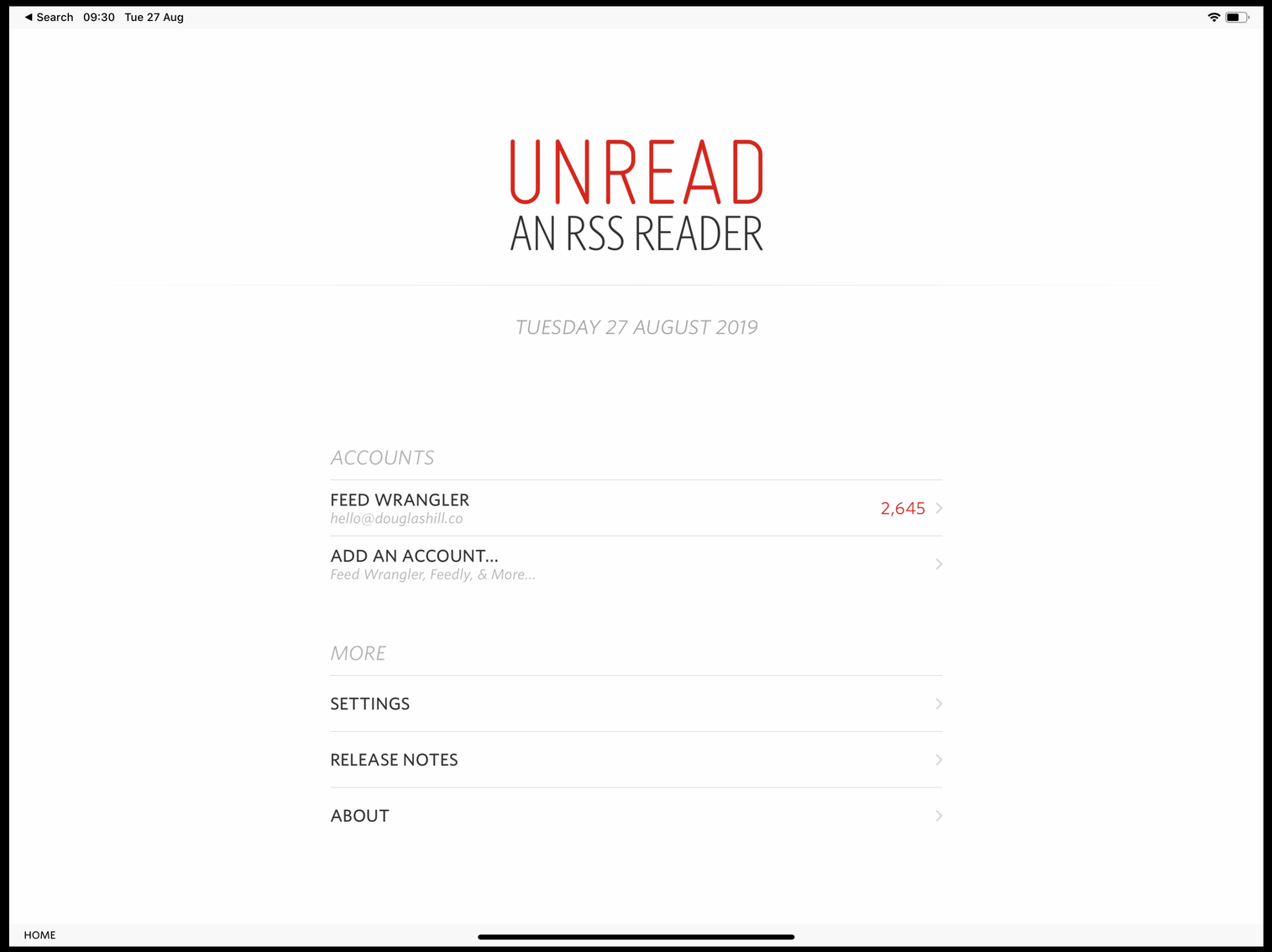
Now that I know how I want the article view to look, I can work backwards through the other screens in the app. Those screens are:

- **Article Summaries List** – This is the second most-viewed screen in Unread. On the iPhone version, it's a long list of article summaries, titles, and image thumbnails.
- **Account Dashboard** – The source list for navigating to an article summaries list. This screen has three sections: *Articles*, *Folders*[1], and *Subscriptions*.
- **Home screen** – This screen has the list of all your accounts, plus a section for extras like settings or contact options. On the iPhone, this screen also has the *UNREAD* masthead, to make the app feel more like a magazine.

At this stage in the design process, the quickest way out is to use a split view controller and call it a day. That was how the earliest iPad apps solved the problem. Many others have mimicked it. But I am almost certain that this would be a mistake for Unread.




09:30
UNREAD
AN RSS READER
TUESDAY 27 AUGUST 2019
ACCOUNTS
FEED WRANGLER
hello@douglashill.co
2,645
ADD AN ACCOUNT...
Feed Wrangler, Feedly, & More...
MORE
HOME


Search 09:30 Tue 27 Aug
UNREAD
AN RSS READER
TUESDAY 27 AUGUST 2019
ACCOUNTS
FEED WRANGLER
hello@douglashill.co
2,645
ADD AN ACCOUNT...
Feed Wrangler, Feedly, & More...
MORE
SETTINGS
RELEASE NOTES
ABOUT
HOME

Persistent History Tracking in Core Data

MICHAEL TSAI
21 Aug 2019 at 21:46

Steffen Ryll:

> At *WWDC* '17, *Apple* introduced a number of new *Core Data* features, one of which is *Persistent History Tracking* or `NSPersistentHistory`. But as of the time of writing, its API is still undocumented. Thus, the only real reference is the What's New in Core Data *WWDC* session.
>
> Since *Persistent History Tracking* makes sharing an `NSPersistentStore` across multiple processes and is one of my favorite new *Core Data* features, it is unfortunate that it mostly seems to fall of the radar.
>
> The purpose of this post is to give a real-world example on how to use it and what makes it so great.

That was written a year and a half ago, and `NSPersistentHistory` remains a really cool feature that's under-discussed and under-documented. Some resources I've found are:

Persistent History Tracking in Core Data

MICHAEL TSAI
21 Aug 2019 at 21:46

Steffen Ryll:

At *WWDC* '17, *Apple* introduced a number of new *Core Data* features, one of which is *Persistent History Tracking* or `NSPersistentHistory`. But as of the time of writing, its API is still undocumented. Thus, the only real reference is the What's New in Core Data *WWDC* session.

Instance Property

readableContentGuide

A layout guide representing an area with a readable width within the view.

Declaration

```
var readableContentGuide: UILayoutGuide { get }
```

Discussion

This layout guide defines an area that can easily be read without forcing users to move their head to track the lines. The readable content area follows the following rules:

1. The readable content guide never extends beyond the view's layout margin guide.
2. The readable content guide is vertically centered inside the layout margin guide.
3. The readable content guide's width is equal to or less than the readable width defined for the current dynamic text size.

Use the readable content guide to lay out a single column of text. If you are laying out multiple columns, you can use the guide's width to determine the optimal width for your columns.

SDKs

iOS 9.0+

tvOS 9.0+

Framework

UIKit

On This Page

See Also

Instance Property

cellLayoutMarginsFollowReadableWidth

A Boolean value that indicates whether the cell margins are derived from the width of the readable content guide.

Declaration

```
var cellLayoutMarginsFollowReadableWidth: Bool { get set }
```

SDKs

iOS 9.0+

tvOS 9.0+

Framework

UIKit

On This Page

See Also

Configuring Cell Height and Layout

`var rowHeight: CGFloat`

The default height (in points) of each row in the table view.

`var estimatedRowHeight: CGFloat`

The estimated height of rows in the table view.

`var insetsContentViewsToSafeArea: Bool`

A Boolean value that indicates whether the table view repositions its content view to be within the current safe area.

```
p, nav, ul, ol {
    max-width: 30em;
}
```

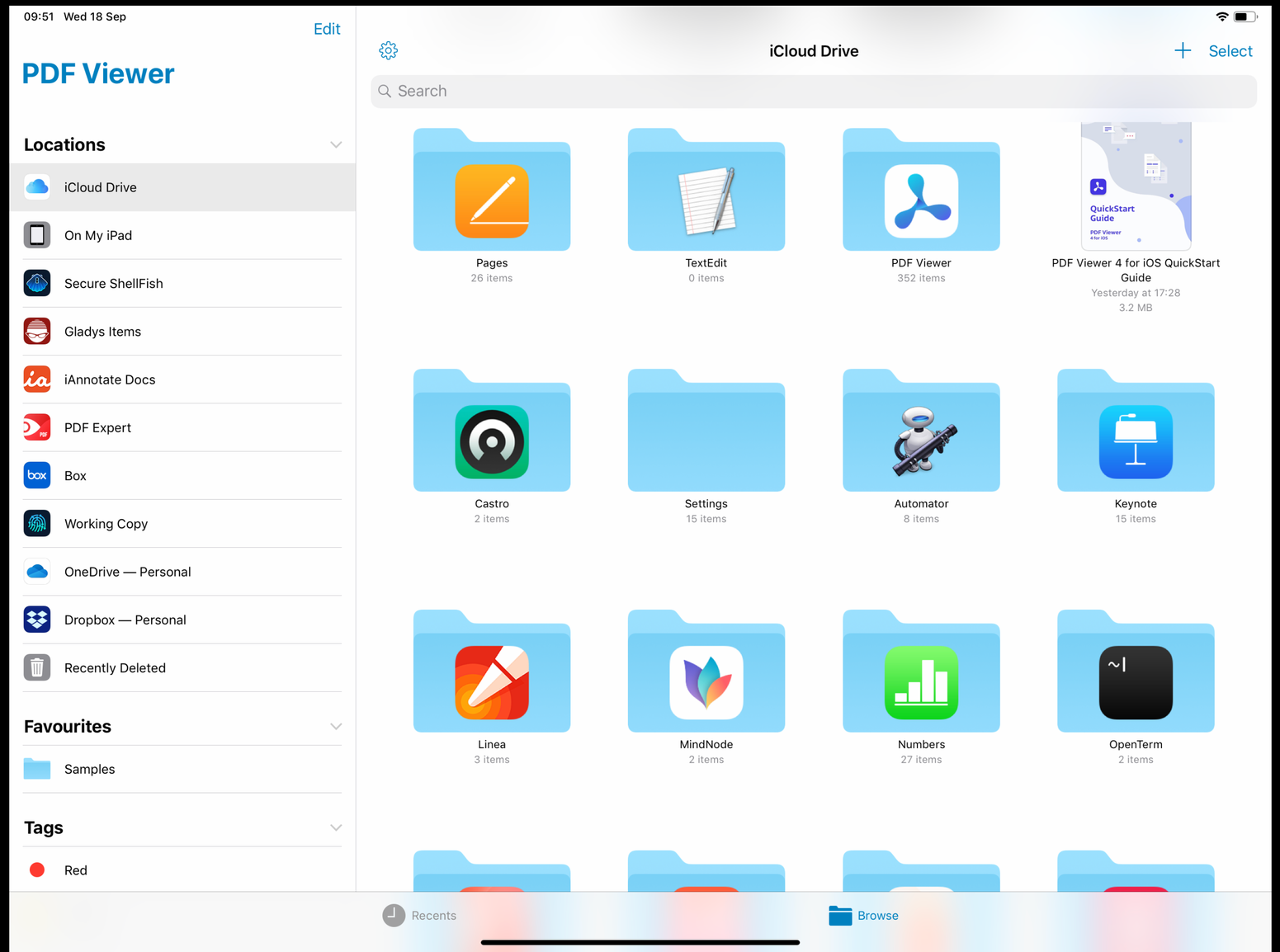
```swift
let font = UIFontDescriptor.preferredFontDescriptor(withTextStyle: .body)

let maxWidth = font.pointSize * 30
```

Grid

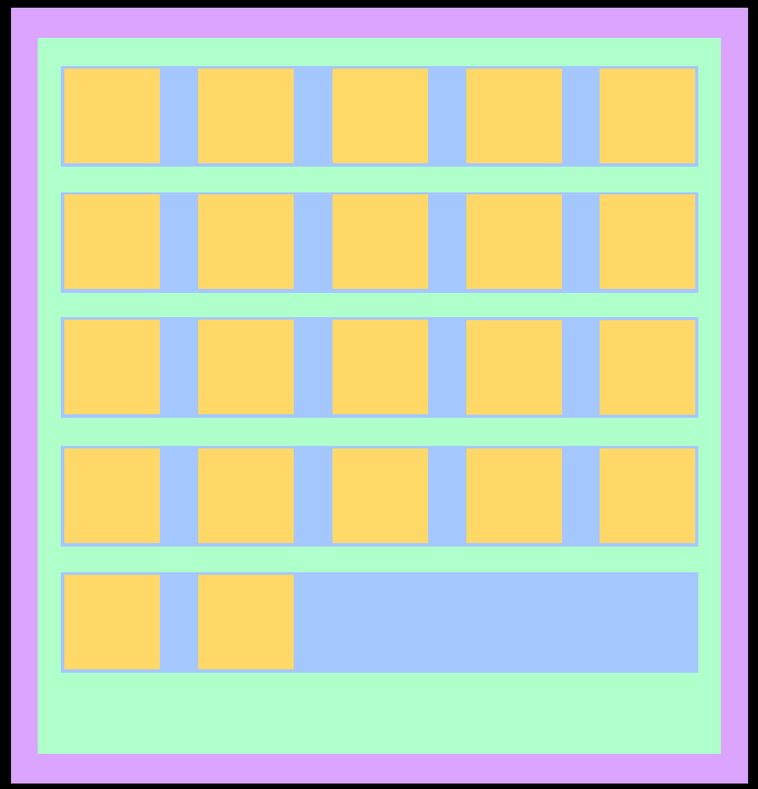
Grid

Files / PDF Viewer



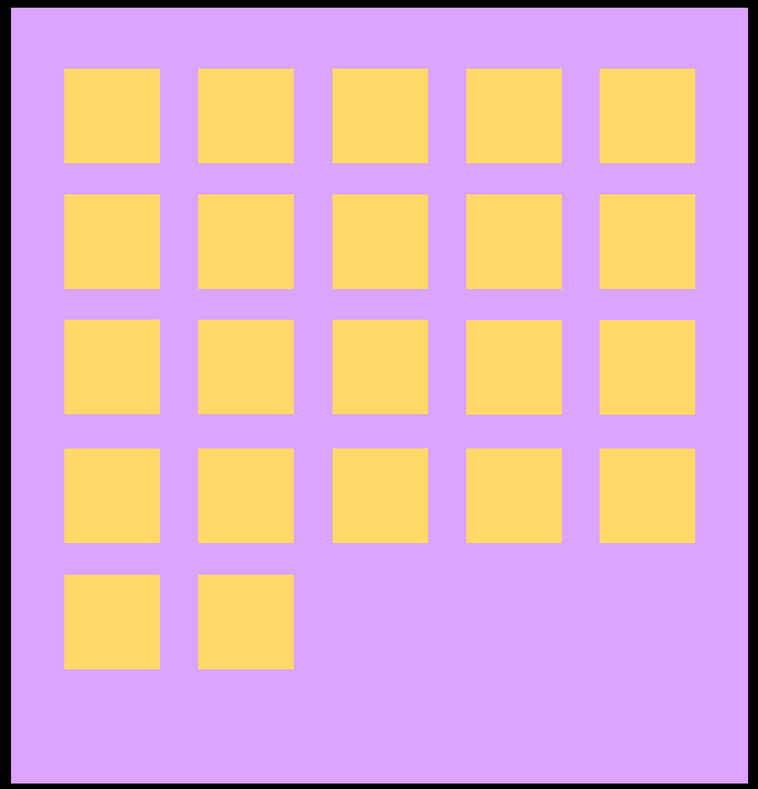
UICollectionViewCompositionalLayout

- Item
- Group
- Section
- Layout



UICollectionViewCompositionalLayout

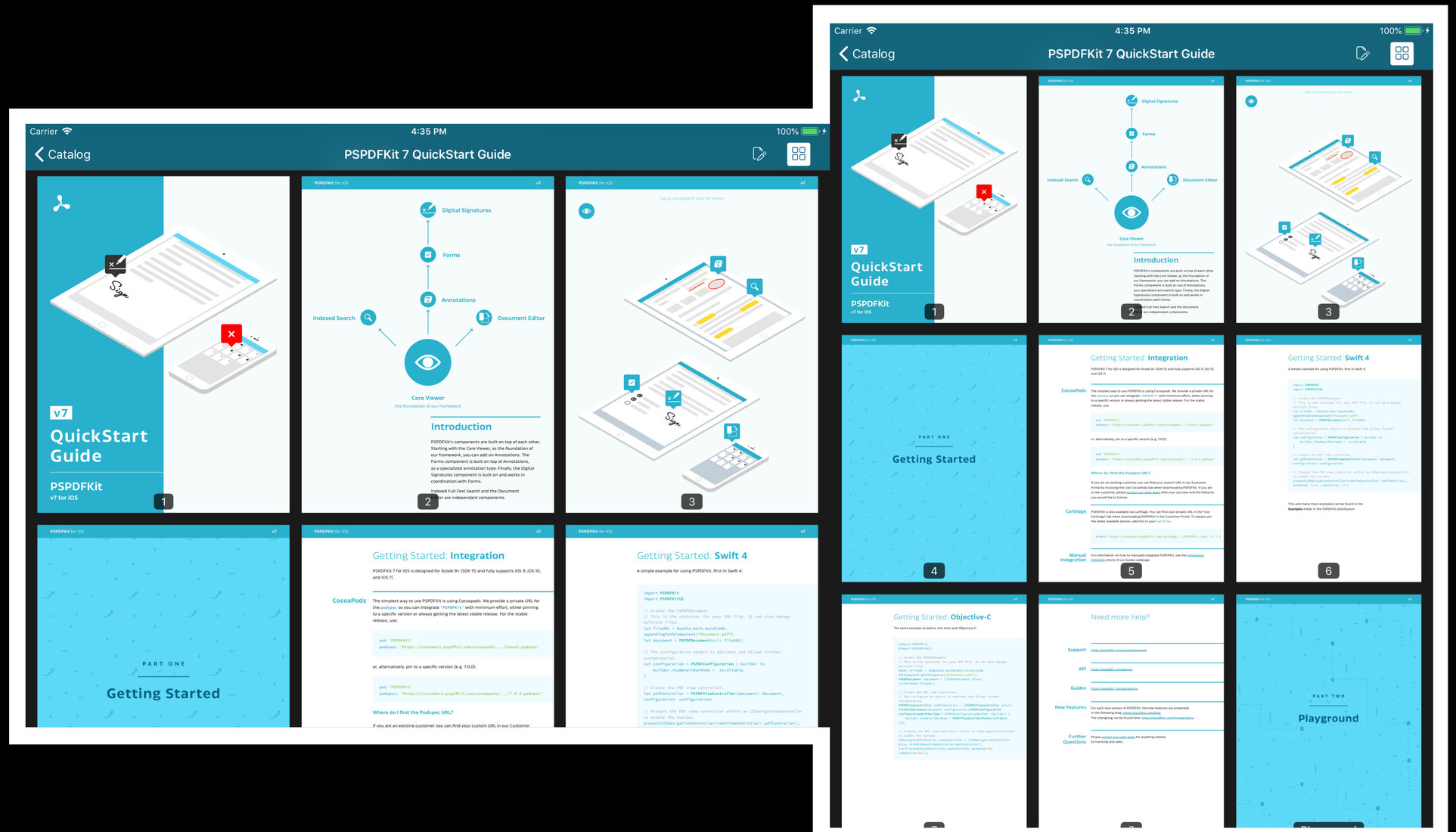
- Item
- Group
- Section
- Layout



Four column grid of ~squares

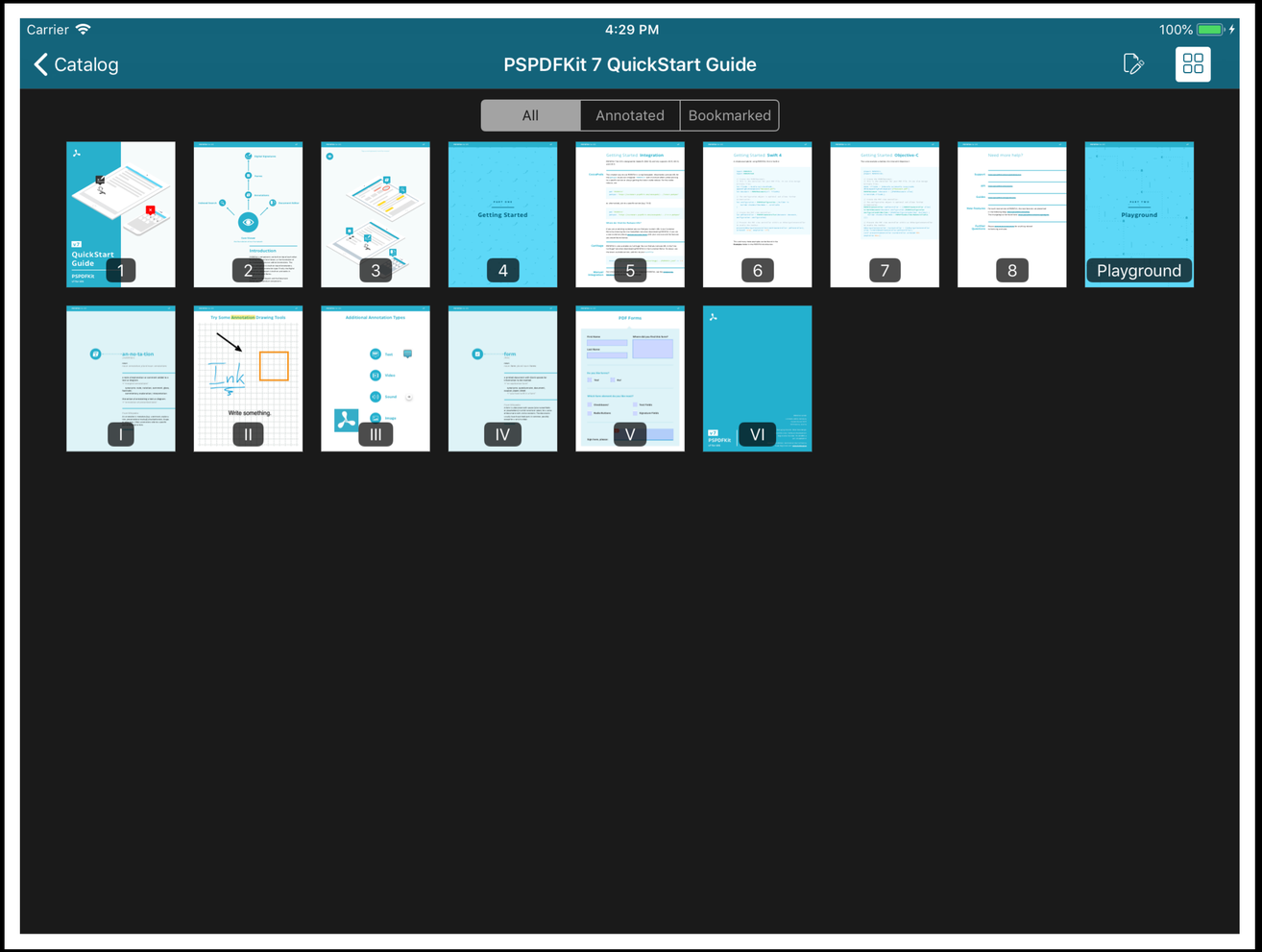
```
let groupSize = NSCollectionLayoutSize(
     widthDimension: .fractionalWidth(1),
    heightDimension: .fractionalWidth(0.25)
)
let group = NSCollectionLayoutGroup.horizontal(
    layoutSize: groupSize,
       subitem: item,
         count: 4
)
```

Fixed number of columns



Fixed size


4:31
Catalog
1
2
3
4
5
6
7
8
Playground
I
II
III
IV
V
VI


Carrier
4:29 PM
100%
Catalog
PSPDFKit 7 QuickStart Guide
All
Annotated
Bookmarked
1
2
3
4
5
6
7
8
Playground
I
II
III
IV
V
VI

Grid with approximate number of items

Grid with approximate number of items

```
// What is the total available area?
let containerArea = containerSize.width * containerSize.height

// 10 thumbnails is useful without being overwhelming.
let approximateNumThumbnails: CGFloat = 10

// Divide the available area by the target number of thumbnails to
get the approximate area per thumbnail.
let approxThumbnailArea = containerArea / approximateNumThumbnails

// We want to handle pages of any aspect ratio.
let pageAspectRatio = pageSize.width / pageSize.height
```

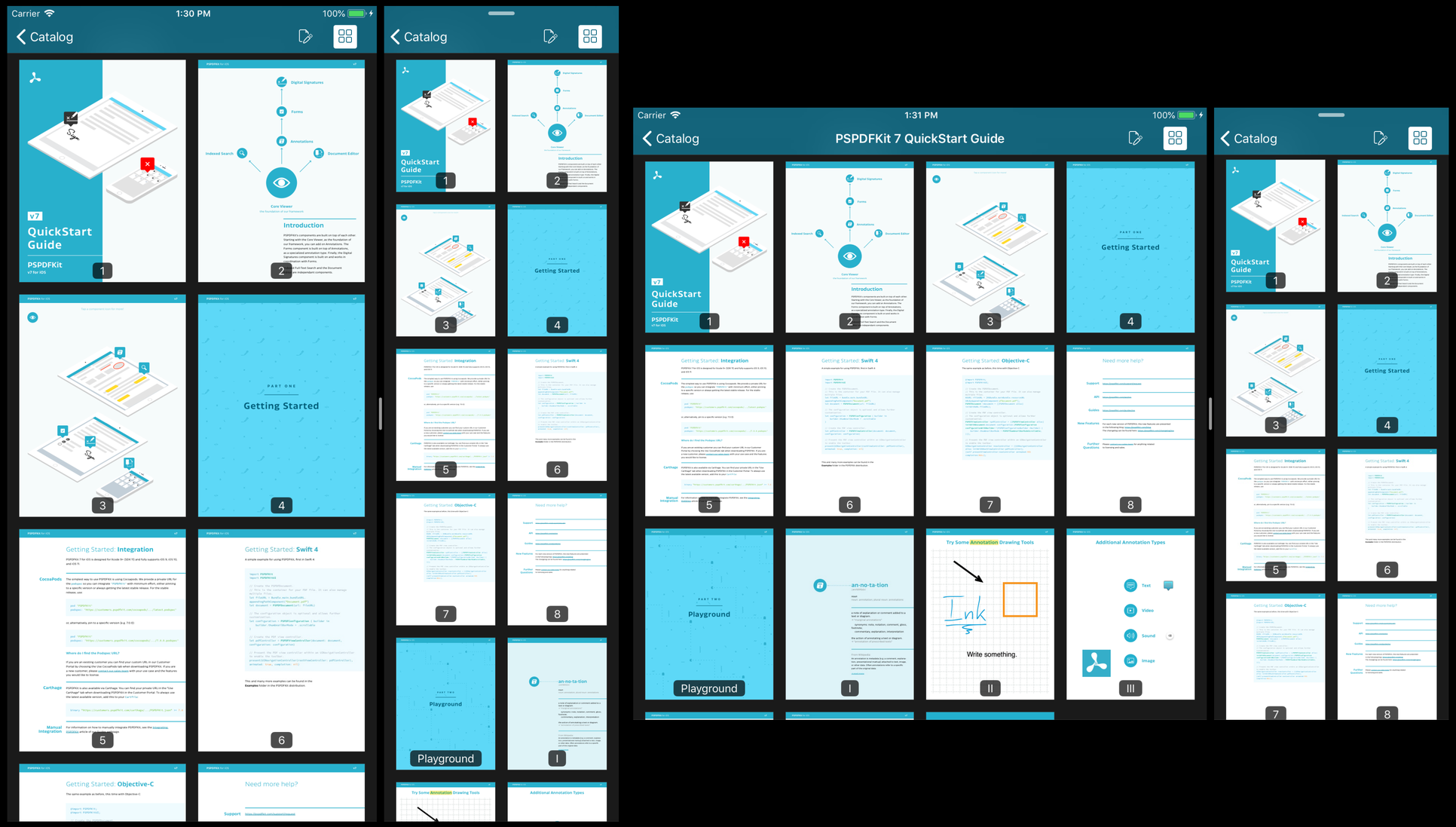
Grid with approximate number of items

```
//    width × height = area
//    width / height = aspect ratio
// => width × width  = area × aspect ratio
let approxThumbnailWidthSquared = approxThumbnailArea * pageAspectRatio

// Take the square root of the value calculated above to find the
approximate thumbnail width.
let approximateThumbnailWidth = sqrt(approximateThumbnailWidthSquared)

// We need a whole number of columns.
let numberOfColumns = round(containerSize.width /
approximateThumbnailWidth)
```

Grid with approximate number of items



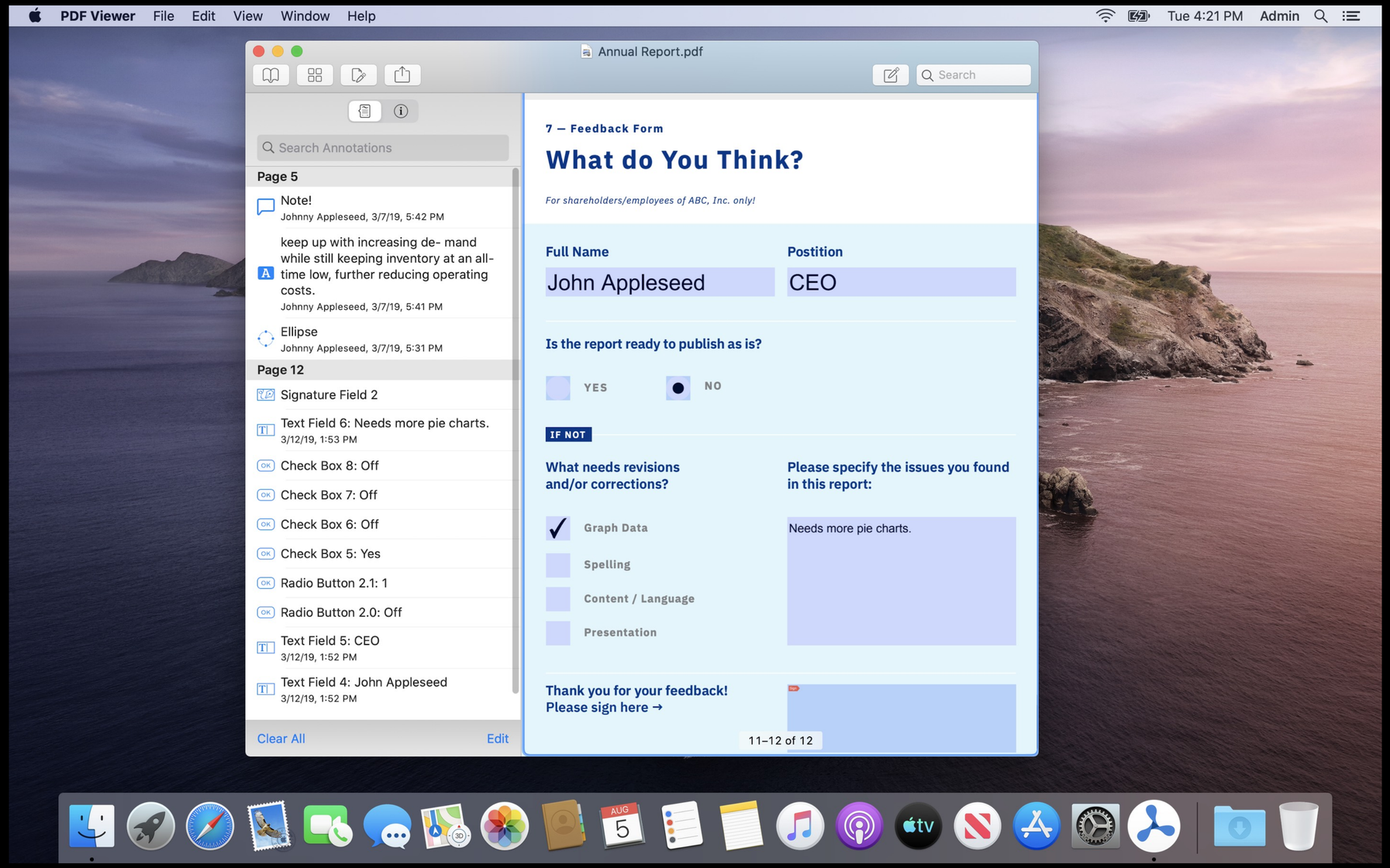
Columns across the information hierarchy

Two columns

`UISplitViewController`

Columns across hierarchy

PDF Viewer for Mac



Three or more columns

macOS Preview

Jira | American Airlines | Fender | TripIt | Crew | Rosetta Stone | Proloquo2Go

With TripIt on Mac, you can review all your plans in one place, edit and share your trip details, and research your destination, all while multitasking with other Mac apps. Your itinerary is even available offline, whether you're at your desk or at 35,000 feet.

Columns across hierarchy

Fiery Feeds

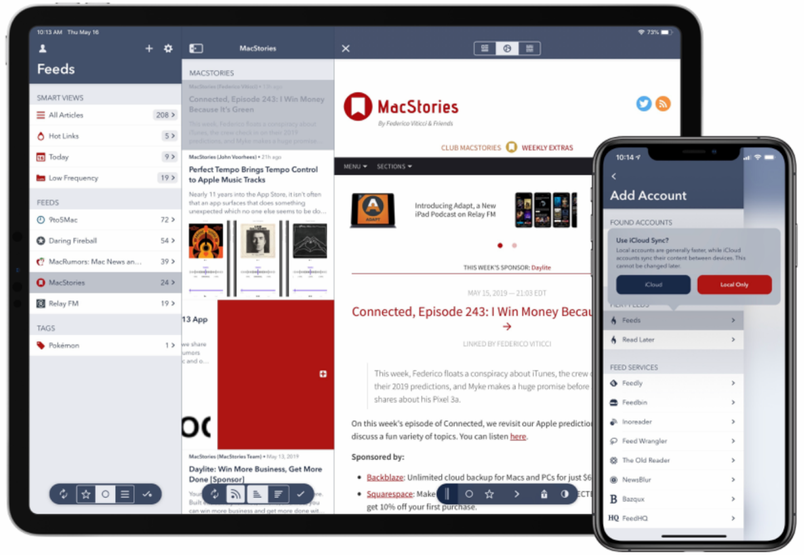
12:10 Sun 18 Aug

macstories.net

MAY 17, 2019 — 16:40 BST

Fiery Feeds Adds iCloud Accounts, Three-Pane iPad View, New Customization Tools, and More

BY RYAN CHRISTOFFEL



(View full size)

Fiery Feeds, the modern, flexible RSS client for iOS, was updated today with a variety of new features that take the app to new heights: enabling iCloud-based accounts for RSS and Read Later so you don't need third-party services, adding a three-pane layout on iPad, offering new, configurable methods for navigation, and a lot more. There's something for everyone, from users who may be new to RSS to Fiery Feeds veterans who will appreciate the additional power offered here.

iCloud Accounts & Sync

“Three-pane layouts were added to the system Notes and Mail apps in iOS 10, but since then the model set by Apple has been adopted by surprisingly few apps. Reducing the need to constantly navigate between menus is always a good thing”

– Ryan Christoffel

Columns across hierarchy

Bear

BEAR

97;&
101;r-gr&
105;nf&
109;a' + 'i&
117;&
blogdraft
c7a'\));g.p(1,'
define
define-ing
details
endif
gettingthemost...
if
import
import\s<([a-z
include
json
name
object id
parameter
path
pragma

GETTINGTH...ULTITASKING

1M **WWDC 2019 session 258 Architecting Your App for Multiple Windows…**

1M **WWDC 2019 session 246 Window Management in Your Multitasking App…**

2M **WWDC 2019 Session 259 Targeting Content With Multiple Windows**
Alex Schaefer, iOS System UI This…

WWDC 2019 Session 259 Targeting Content With Multiple Windows

Alex Schaefer, iOS System UI

This was part of what was live as ‘Getting the Most Out of Multitasking’ (session 242) #GettingTheMostOutOfMultitasking

Which scene should activate in response to a launch event? For example a notification or a URL. If the app isn’t running, the system does not want to wait until the app is loaded to make this decision.

The system can describe what the launch event is about and scenes can describe what capabilities they have.

Therefore the system remembers what activation conditions each scene has in a way that can be serialised and evaluated before the app is running.

These conditions are provided via the `activationConditions` property on `UIScene`. https://developer.apple.com/documentation/uikit/uiscene/3238055-activationconditions

It has the type `UISceneActivationConditions`. https://developer.apple.com/documentation/uikit/uisceneactivationconditions

It’s a couple of predicates:

- *Can* This is the main one. What content the scene can display.
- *Prefers*: Is the scene is especially interested in particular type of content? The system uses this to choose a scene from several possibilities.

These predicates act on a *target content identifier*. This is ultimately a string. It represent data in the app’s model.

You could use the actual string of a Universal Link as a target content identifier.

- An APNS dictionary can have a target content identifier. There’s a key for this.
- A shortcut item can use a target content identifier.
- An NSUserActivity can have a target content identifier.

Columns across hierarchy

Bear

23:25 Sat 31 Aug

39%

GETTINGTH...ULTITASKING

1M **WWDC 2019 session 258 Architecting Your App for Multiple Windows…**

1M **WWDC 2019 session 246 Window Management in Your Multitasking App…**

2M **WWDC 2019 Session 259 Targeting Content With Multiple Windows**
Alex Schaefer, iOS System UI This…

H1 **WWDC 2019 Session 259 Targeting Content With Multiple Windows**

Alex Schaefer, iOS System UI

This was part of what was live as ‘Getting the Most Out of Multitasking’ (session 242) #GettingTheMostOutOfMultitasking

Which scene should activate in response to a launch event? For example a notification or a URL. If the app isn’t running, the system does not want to wait until the app is loaded to make this decision.

The system can describe what the launch event is about and scenes can describe what capabilities they have.

Therefore the system remembers what activation conditions each scene has in a way that can be serialised and evaluated before the app is running.

These conditions are provided via the `activationConditions` property on `UIScene`. https://developer.apple.com/documentation/uikit/uiscene/3238055-activationconditions

It has the type `UISceneActivationConditions`. https://developer.apple.com/documentation/uikit/uisceneactivationconditions

It’s a couple of predicates:

- *Can* This is the main one. What content the scene can display.
- *Prefers*: Is the scene is especially interested in particular type of content? The system uses this to choose a scene from several possibilities.

These predicates act on a *target content identifier*. This is ultimately a string. It represent data in the app’s model.

You could use the actual string of a Universal Link as a target content identifier.

- An APNS dictionary can have a target content identifier. There’s a key for this.
- A shortcut item can use a target content identifier.
- An NSUserActivity can have a target content identifier.

Columns across hierarchy

Bear

23:25 Sat 31 Aug

39%

WWDC_2019 Session_259 Targeting Content With Multiple Windows

Alex Schaefer, iOS System UI

This was part of what was live as ‘Getting the Most Out of Multitasking’ (session 242) #GettingTheMostOutOfMultitasking

Which scene should activate in response to a launch event? For example a notification or a URL. If the app isn’t running, the system does not want to wait until the app is loaded to make this decision.

The system can describe what the launch event is about and scenes can describe what capabilities they have.

Therefore the system remembers what activation conditions each scene has in a way that can be serialised and evaluated before the app is running.

These conditions are provided via the `activationConditions` property on `UIScene`. https://developer.apple.com/documentation/uikit/uiscene/3238055-activationconditions

It has the type `UISceneActivationConditions`. https://developer.apple.com/documentation/uikit/uisceneactivationconditions

It’s a couple of predicates:

- *Can* This is the main one. What content the scene can display.
- *Prefers*: Is the scene is especially interested in particular type of content? The system uses this to choose a scene from several possibilities.

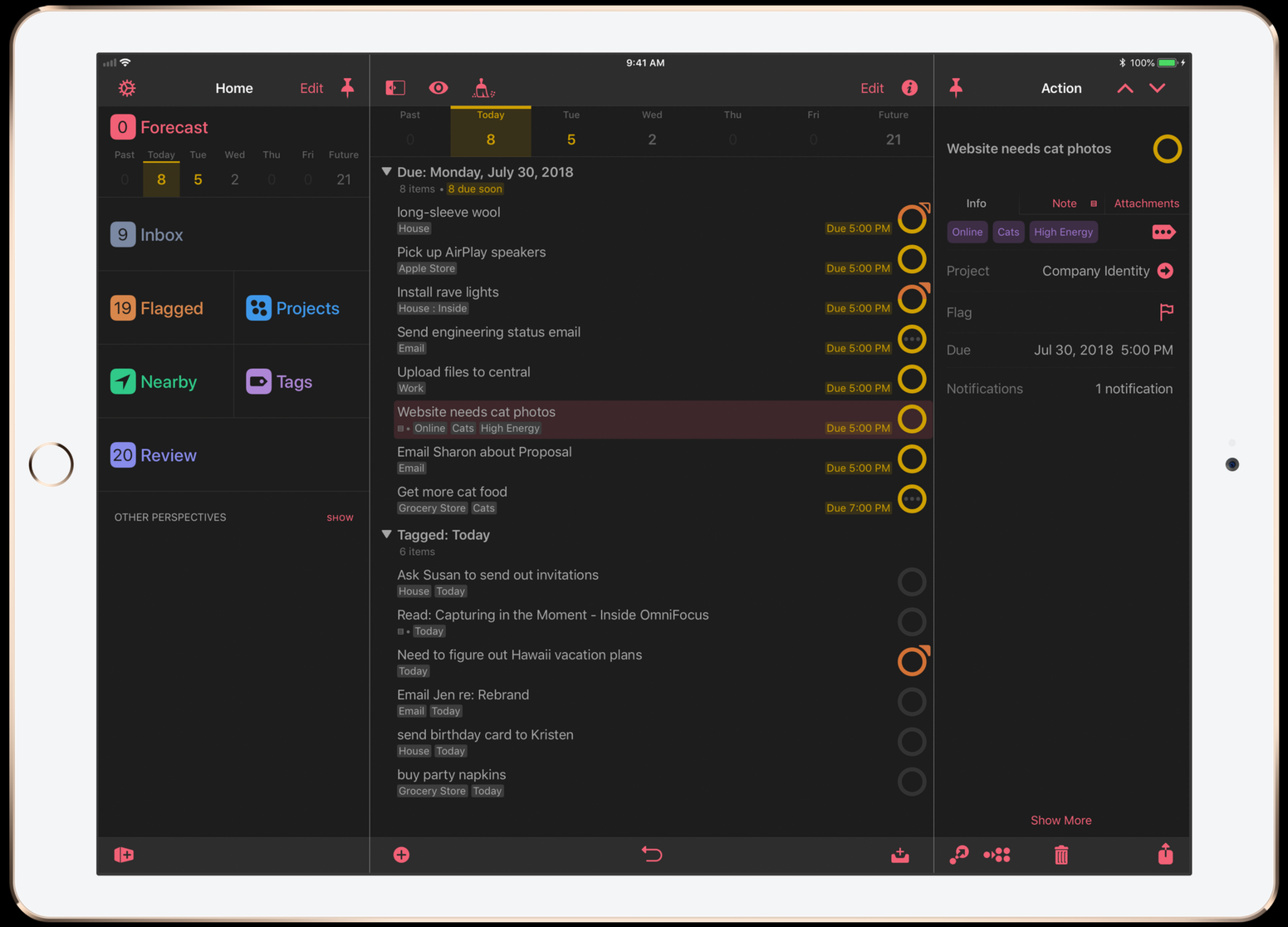
These predicates act on a *target content identifier*. This is ultimately a string. It represent data in the app’s model.

You could use the actual string of a Universal Link as a target content identifier.

i the this

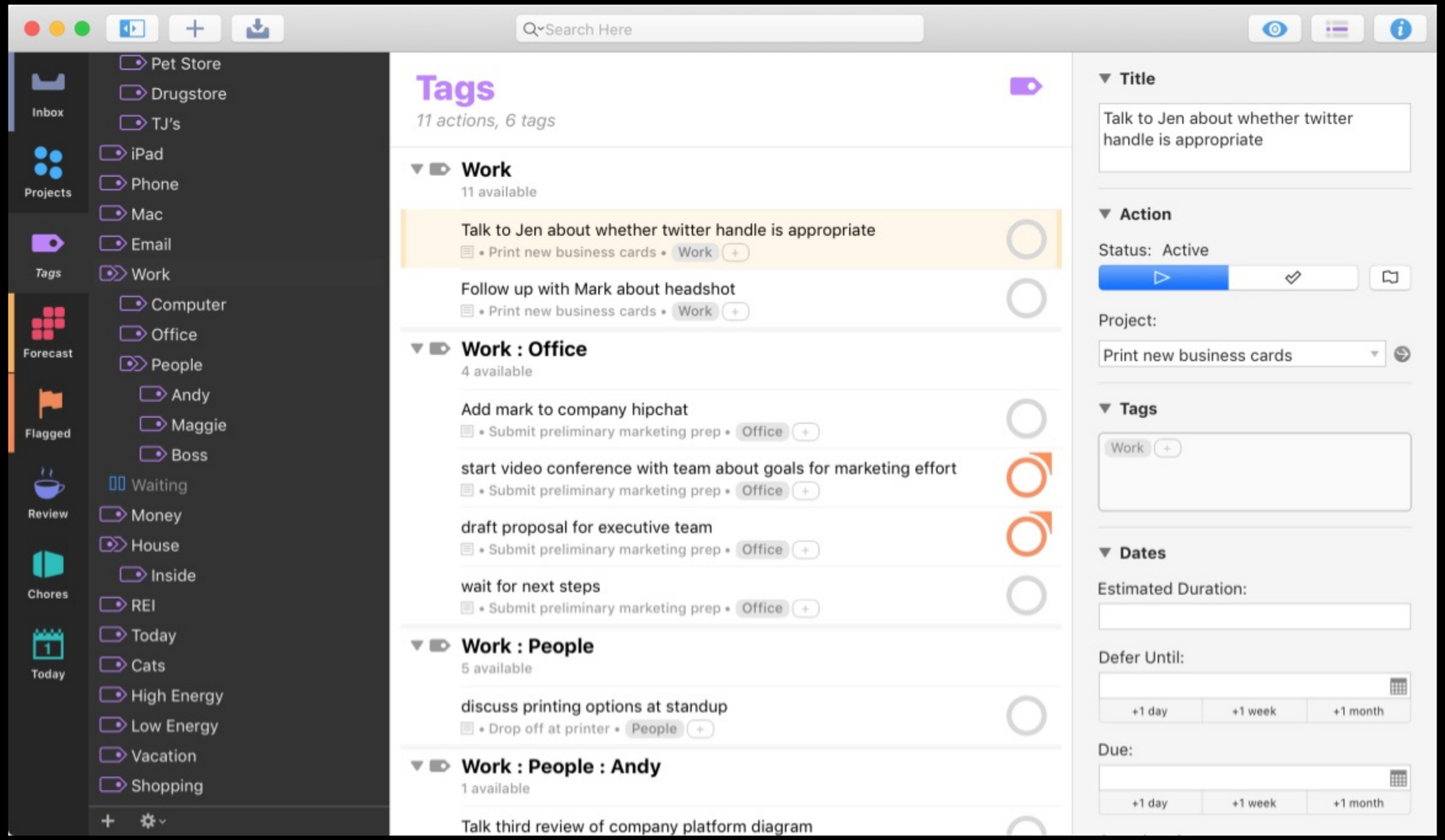
Columns across hierarchy

OmniFocus



Columns across hierarchy

OmniFocus for Mac



Implementation

- Use a container view controller
- View appearance callbacks are hard
- Use `preferredContentSize` to see what fits
- Implement `preferredContentSize` (obviously)
- Collapse into a UINavigationController when not fitting

Columns within
one level of the
information hierarchy

Columns within hierarchy

Unread

09:33 Tue 27 Aug

WEDNESDAY 21

Persistent History Tracking in Core Data

MICHAEL TSAI

WED 21

Steffen Ryll: At WWDC '17, Apple introduced a number of new Core Data features, one of which is Persistent History Tracking or NSPersistentHistory .

Catalina's Path Changes

MICHAEL TSAI

WED 21

Howard Oakley: If you write scripts of any kind for macOS – shell scripts, AppleScripts, or anything similar – now is the time to join the Catalina beta programme (if you're not already signed up) if you want those scripts to work with macOS 10.15 when it's released in a month or so.

Nullable References in C# 8.0

MICHAEL TSAI

WED 21

Erik Sink: Bottom line, C# 8.0 uses the same syntax for nullability of reference types that we have been using for value types: string x; // not nullable string?

Under the Hood with OmniFocus for the Web

THE OMNI GROUP

WED 21

We just published a new article on Inside OmniFocus — OmniFocus for the Web: Under the Hood — which lists the various pieces that make up the web app.

PSPDFKit 2.3 for Windows

ALL ARTICLES

Columns within hierarchy

PDF Viewer

09:52 Sun 25 Aug

Documents

PDF32000.book

If the NoZoom flag is set, the annotation shall always maintain the same fixed size on the screen and shall be unaffected by the magnification level at which the page itself is displayed. Similarly, if the NoRotate flag is set, the annotation shall retain its original orientation on the screen when the page is rotated (by changing the Rotate entry in the page object; see 7.7.3, “Page Tree”).

In either case, the annotation’s position shall be determined by the coordinates of the upper-left corner of its annotation rectangle, as defined by the **Rect** entry in the annotation dictionary and interpreted in the default user space of the page. When the default user space is scaled or rotated, the positions of the other three corners of the annotation rectangle are different in the altered user space than they were in the original user space. The conforming reader shall perform this alteration automatically. However, it shall not actually change the annotation’s **Rect** entry, which continues to describe the annotation’s relationship with the unscaled, unrotated user space.

NOTE Figure 58 shows how an annotation whose NoRotate flag is set remains upright when the page it is on is rotated 90 degrees clockwise. The upper-left corner of the annotation remains at the same point in default user space; the annotation pivots around that point.

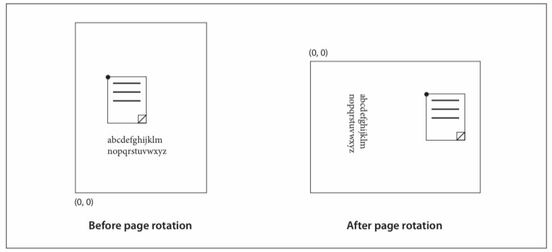


Figure 58 – Coordinate adjustment with the NoRotate flag

12.5.4 Border Styles

An annotation may optionally be surrounded by a border when displayed or printed. If present, the border shall be drawn completely inside the annotation rectangle. In PDF 1.1, the characteristics of the border shall be specified by the **Border** entry in the annotation dictionary (see Table 164). Beginning with PDF 1.2, the border characteristics for some types of annotations may instead be specified in a *border style dictionary* designated by the annotation’s **BS** entry. Such dictionaries may also be used to specify the width and dash pattern for the lines drawn by line, square, circle, and ink annotations. Table 166 summarizes the contents of the border style dictionary. If neither the **Border** nor the **BS** entry is present, the border shall be drawn as a solid line with a width of 1 point.

Table 166 – Entries in a border style dictionary

Key	Type	Value
Type	name	*(Optional)* The type of PDF object that this dictionary describes; if present, shall be **Border** for a border style dictionary.
W	number	*(Optional)* The border width in points. If this value is 0, no border shall drawn. Default value: 1.

Table 166 – Entries in a border style dictionary (continued)

Key	Type	Value
S	name	*(Optional)* The border style: S (Solid) A solid rectangle surrounding the annotation. D (Dashed) A dashed rectangle surrounding the annotation. The dash pattern may be specified by the **D** entry. B (Beveled) A simulated embossed rectangle that appears to be raised above the surface of the page. I (Inset) A simulated engraved rectangle that appears to be recessed below the surface of the page. U (Underline) A single line along the bottom of the annotation rectangle. A conforming reader shall tolerate other border styles that it does not recognize and shall use the default value.
D	array	*(Optional)* A *dash array* defining a pattern of dashes and gaps that shall be used in drawing a dashed border (border style D in the **S** entry). The dash array shall be specified in the same format as in the line dash pattern parameter of the graphics state (see 8.4.3.6, “Line Dash Pattern”). The dash phase is not specified and shall be assumed to be 0. EXAMPLE A **D** entry of [3 2] specifies a border drawn with 3-point dashes alternating with 2-point gaps. Default value: [3].

Beginning with PDF 1.5, some annotations (square, circle, and polygon) may have a **BE** entry, which is a *border effect dictionary* that specifies an effect that shall be applied to the border of the annotations. Beginning with PDF 1.6, the free text annotation may also have a **BE** entry. Table 167 describes the entries in a border effect dictionary.

Table 167 – Entries in a border effect dictionary

Key	Type	Value
S	name	*(Optional)* A name representing the border effect to apply. Possible values are: S No effect: the border shall be as described by the annotation dictionary’s **BS** entry. C The border should appear “cloudy”. The width and dash array specified by **BS** shall be honored. Default value: S.
I	number	*(Optional; valid only if the value of **S** is C)* A number describing the intensity of the effect, in the range 0 to 2. Default value: 0.

12.5.5 Appearance Streams

Beginning with PDF 1.2, an annotation may specify one or more *appearance streams* as an alternative to the simple border and colour characteristics available in earlier versions. Appearance streams enable the annotation to be presented visually in different ways to reflect its interactions with the user. Each appearance stream is a form XObject (see 8.10, “Form XObjects”): a self-contained content stream that shall be rendered inside the annotation rectangle.

The algorithm outlined in this sub-clause shall be used to map from the coordinate system of the appearance XObject (as defined by its **Matrix** entry; see Table 97) to the annotation’s rectangle in default user space:

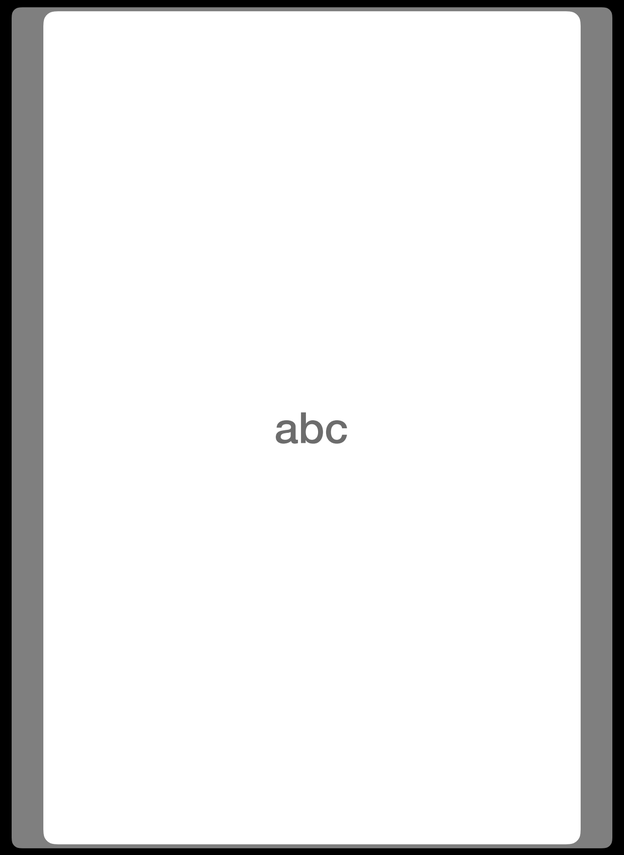
Page 438

394–395 of 756

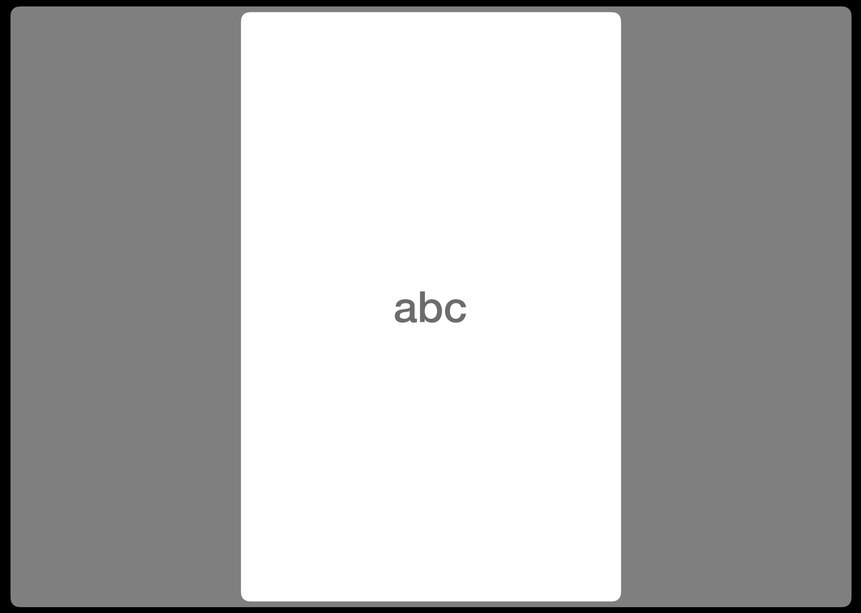
386

387

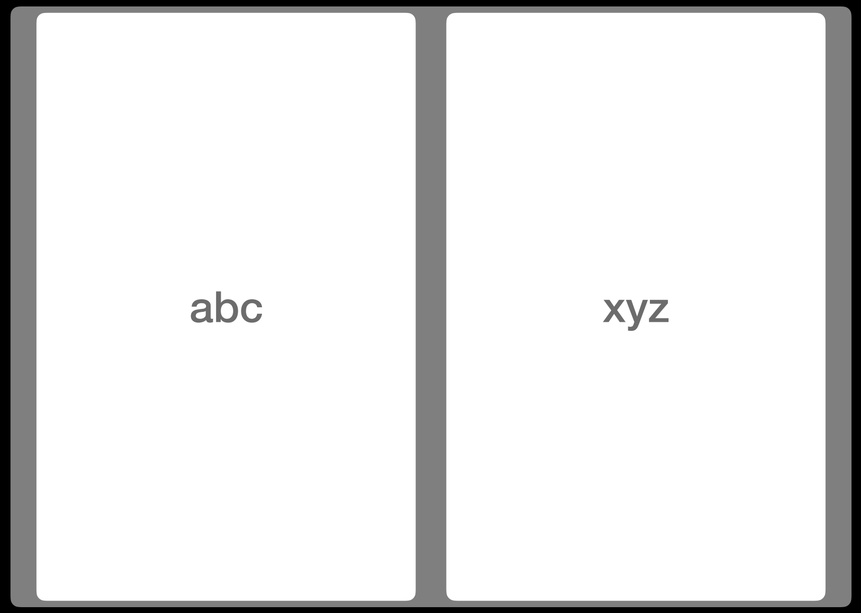
Automatic double page mode



Automatic double page mode



Automatic double page mode



Automatic double page mode

```
let pageAspectRatio = pageSize.width / pageSize.height
let viewAspectRatio = bounds.width / bounds.height

let isDoublePageMode = viewAspectRatio > pageAspectRatio * 1.8
```

Michael Tsai - Blog - Persistent History Tracking in Core Data

Steffen Ryll:

> *At* WWDC *'17,* Apple *introduced a number of new* Core Data *features, one of which is* Persistent History Tracking *or* `NSPersistentHistory`*. But as of the time of writing, its API is still undocumented. Thus, the only real reference is the* *What's New in Core Data* WWDC *session.*
>
> *Since* Persistent History Tracking *makes sharing an* `NSPersistentStore` *across multiple processes and is one of my favorite new* Core Data *features, it is unfortunate that it mostly seems to fall of the radar.*
>
> *The purpose of this post is to give a real-world example on how to use it and what makes it so great.*

That was written a year and a half ago, and `NSPersistentHistory` remains a really cool feature that's under-discussed and under-documented. Some resources I've found are:

- WWDC 2017 Session 210: What's New in Core Data
- WWDC 2019 Session 230: Making Apps with Core Data
- Consuming Relevant Store Changes
- Persistent History

Here are some things I figured out by exploring:

- The history is stored directly in the same SQLite database(s) as the persistent store.
- It uses tables that look kind of like Core Data tables, only with a different prefix.
- But you couldn't create them yourself using Core Data, since the same column can store the primary key for different types of entities (you would think `NSObjectIDAttributeType` could do that, but it actually can't be used in stores), and likewise for the columns that store the tombstone values.
- The tables are updated using SQLite triggers, which are again not directly exposed in Core Data (though this year's new

Michael Tsai - Blog - Persistent History Tracking in Core Data

Steffen Ryll:

> *At* WWDC *'17,* Apple *introduced a number of new* Core Data *features, one of which is* Persistent History Tracking *or* `NSPersistentHistory`*. But as of the time of writing, its API is still undocumented. Thus, the only real reference is the* *What's New in Core Data* WWDC *session.*
>
> *Since* Persistent History Tracking *makes sharing an* `NSPersistentStore` *across multiple processes and is one of my favorite new* Core Data *features, it is unfortunate that it mostly seems to fall of the radar.*
>
> *The purpose of this post is to give a real-world example on how to use it and what makes it so great.*

That was written a year and a half ago, and `NSPersistentHistory` remains a really cool feature that's under-discussed and under-documented. Some resources I've found are:

- WWDC 2017 Session 210: What's New in Core Data
- WWDC 2019 Session 230: Making Apps with Core Data
- Consuming Relevant Store Changes
- Persistent History

Here are some things I figured out by exploring:

- The history is stored directly in the same SQLite database(s) as the persistent store.
- It uses tables that look kind of like Core Data tables, only with a different prefix.
- But you couldn't create them yourself using Core Data, since the same column can store the primary key for different types of entities (you would think `NSObjectIDAttributeType` could do that, but it actually can't be used in stores), and likewise for the columns that store the tombstone values.
- The tables are updated using SQLite triggers, which are again not directly exposed in Core Data (though this year's new derived attributes also use them).
- The triggers are fired for all database changes, so unlike the managed object context's change tracking, they also work for batch updates and deletions (and, presumably, the forthcoming batch insertions).
- The tables look very compact, with repeated string values interned and the list of modified columns stored as a bit vector.
- Core Data automatically updates the schema of the history tracking tables when you do a migration.
- Enabling history tracking does not change the version of your model. But, in practice, you'll get incorrect results if you don't enable it consistently.
- Setting an attribute to be preserved after deletion (i.e. for the tombstone) does change the model's version hash, however.
- There's no public API to set this flag on an attribute in the model, only a checkbox in Xcode. However, you can use key-value coding to set or query `NSPropertyDescription.preserveValueOnDeletionInPersistentHistory`.
- So, overall, it seems tricky to use persistent history on a store that will be shared with OS versions that don't support history tracking. You might have to roll your own in that case.
- Querying and pruning the history works as you would expect.
- The `NSPersistentHistoryTransaction.objectIDNotification()` does not generate a `NSManagedObjectContextDidSaveNotification`, but rather a private `NSManagedObjectContextDidSaveObjectIDsNotification` notification.
- Rather than containing full objects under keys like `NSUpdatedObjectsKey`, it contains object IDs under keys like `updated_objectIDs`. This is a bit unexpected, because `NSManagedObjectContext` is already documented to

Michael Tsai - Blog - Persistent History Tracking in Core Data

Steffen Ryll:

> *At* WWDC *'17,* Apple *introduced a number of new* Core Data *features, one of which is* Persistent History Tracking *or* `NSPersistentHistory`*. But as of the time of writing, its API is still undocumented. Thus, the only real reference is the* *What's New in Core Data* WWDC *session.*
>
> *Since* Persistent History Tracking *makes sharing an* `NSPersistentStore` *across multiple processes and is one of my favorite new* Core Data *features, it is unfortunate that it mostly seems to fall of the radar.*
>
> *The purpose of this post is to give a real-world example on how to use it and what makes it so great.*

That was written a year and a half ago, and `NSPersistentHistory` remains a really cool feature that's under-discussed and under-documented. Some resources I've found are:

- WWDC 2017 Session 210: What's New in Core Data
- WWDC 2019 Session 230: Making Apps with Core Data
- Consuming Relevant Store Changes
- Persistent History

Here are some things I figured out by exploring:

- The history is stored directly in the same SQLite database(s) as the persistent store.
- It uses tables that look kind of like Core Data tables, only with a different prefix.
- But you couldn't create them yourself using Core Data, since the same column can store the primary key for different types of entities (you would think `NSObjectIDAttributeType` could do that, but it actually can't be used in stores), and likewise for the columns that store the tombstone values.
- The tables are updated using SQLite triggers, which are again not directly exposed in Core Data (though this year's new derived attributes also use them).
- The triggers are fired for all database changes, so unlike the managed object context's change tracking, they also work for batch updates and deletions (and, presumably, the forthcoming batch insertions).
- The tables look very compact, with repeated string values interned and the list of modified columns stored as a bit vector.
- Core Data automatically updates the schema of the history tracking tables when you do a migration.
- Enabling history tracking does not change the version of your model. But, in practice, you'll get incorrect results if you don't enable it consistently.
- Setting an attribute to be preserved after deletion (i.e. for the tombstone) does change the model's version hash, however.
- There's no public API to set this flag on an attribute in the model, only a checkbox in Xcode. However, you can use key-value coding to set or query `NSPropertyDescription.preserveValueOnDeletionInPersistentHistory`.
- So, overall, it seems tricky to use persistent history on a store that will be shared with OS versions that don't support history tracking. You might have to roll your own in that case.
- Querying and pruning the history works as you would expect.
- The `NSPersistentHistoryTransaction.objectIDNotification()` does not generate a `NSManagedObjectContextDidSaveNotification`, but rather a private `NSManagedObjectContextDidSaveObjectIDsNotification` notification.
- Rather than containing full objects under keys like `NSUpdatedObjectsKey`, it contains object IDs under keys like `updated_objectIDs`. This is a bit unexpected, because `NSManagedObjectContext` is already documented to support `NSManagedObjectID` or `NSURL` objects under the `NSUpdatedObjectsKey` key.
- In any case, you get IDs because it isn't storing the changed values. Instead, when merging, it fetches the latest values from the store.
- This makes sense given the data model, but it means that, perhaps counterintuitively, merging will update *all* the attributes, not just those those changed in the transaction that generated the notification. And they'll be updated to the *current* values, which may be much newer than the ones at the time of the transaction. This is not version control, just a way to see what has changed.

Update (2019-08-22): Deeje Cooley:

> *I incorporated Persistent History Tracking into CloudCore, an open-source CoreData-CloudKit sync engine, specifically to support offline sync. Check it out!*

Core Data Documentation iOS iOS 11 iOS 12 Key-Value Coding (KVC) Mac macOS 10.13 High Sierra macOS 10.14 Mojave Programming SQLite

Stay up-to-date by subscribing to the Comments RSS Feed for this post.

Number of columns

```
let availableContentWidth =
      view.bounds.inset(by: view.layoutMargins).width

let idealColWidth = fontSize * 30

let columnWidth = min(availableContentWidth, idealColWidth)

let columnsPerPage = floor(availableContentWidth / columnWidth)
```

Beyond size classes

Making better use of large screens

Douglas Hill @qdoug

iOSDevUK

September 2019