What’s New in Cocoa Touch

Session 201

Eliza Block, UIKit
Josh Shaffer, UIKit
Productivity

UI refinements

API enhancements
Productivity

UI refinements

API enhancements
Drag and Drop
Geckos
Drag and Drop

Enabling drags
Drag and Drop
Enabling drags
Drag and Drop

Enabling drags

```swift
let drag = UIDragInteraction(delegate: self)
iconView.addInteraction(drag)
```
Drag and Drop
Enabling drags

```swift
let drag = UIDragInteraction(delegate: self)
iconView.addInteraction(drag)
```
Drag and Drop

Enabling drags

```
let drag = UIDragInteraction(delegate: self)
iconView.addInteraction(drag)
```

UIDragInteractionDelegate

- Provide data for dragged item
- Customize lift animation
- Customize preview
Drag and Drop
Enabling drops
Drag and Drop
Enabling drops
Drag and Drop

Enabling drops

let drop = UIDropInteraction(delegate: self)
iconContainerView.addInteraction(drop)
Drag and Drop
Enabling drops

```swift
let drop = UIDropInteraction(delegate: self)
iconContainerView.addInteraction(drop)
```

**UIDropInteractionDelegate**

- Update UI as drag moves
- Receive data on drop
- Customize drop animation
Drag and Drop
Easy to adopt
Drag and Drop
Easy to adopt

Built-in support
• TableView, CollectionView, TextView, TextField, WebView
Drag and Drop
Easy to adopt

Built-in support
• TableView, CollectionView, TextView, TextField, WebView

Integration with UIPasteConfiguration
Drag and Drop
Easy to adopt

Built-in support
- TableView, CollectionView, TextView, TextField, WebView

Integration with UIPasteConfiguration

<table>
<thead>
<tr>
<th>Event</th>
<th>Hall</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing Drag and Drop</td>
<td>Hall 3</td>
<td>Tuesday 11:20AM</td>
</tr>
<tr>
<td>Mastering Drag and Drop</td>
<td>Executive Ballroom</td>
<td>Wednesday 11:00AM</td>
</tr>
<tr>
<td>Drag and Drop with Collection and Table View</td>
<td>Hall 2</td>
<td>Thursday 9:00AM</td>
</tr>
<tr>
<td>Data Delivery with Drag and Drop</td>
<td>Hall 2</td>
<td>Thursday 10:00AM</td>
</tr>
</tbody>
</table>
File Management
File Management
UIDocumentBrowserViewController
File Management
UIDocumentBrowserViewController

class UIDocumentBrowserViewController {
    init(forOpeningFilesWithContentTypes: [String]?)
    var delegate: UIDocumentBrowserViewControllerDelegate?
}

class UIDocumentBrowserViewController {
    init(forOpeningFilesWithContentTypes: [String]?)
    var delegate: UIDocumentBrowserViewControllerDelegate?
}

Highly customizable
File Management
UIDocumentBrowserViewController

class UIDocumentBrowserViewController {
    init(forOpeningFilesWithContentTypes: [String]?)
    var delegate: UIDocumentBrowserViewControllerDelegate?
}

Highly customizable

Access to local documents and cloud storage
File Management
UIDocumentBrowserViewController

```swift
class UIDocumentBrowserViewController {
init(forOpeningFilesWithContentTypes: [String]?)
var delegate: UIDocumentBrowserViewControllerDelegate?
}
```

Highly customizable

Access to local documents and cloud storage

Be sure to coordinate file access
Highly customizable

Access to local documents and cloud storage

Be sure to coordinate file access

• NSFFileCoordinator or UIDocument
File Management

UIDocumentBrowserViewController

class UIDocumentBrowserViewController {
    init(forOpeningFilesWithContentTypes: [String]?)
    var delegate: UIDocumentBrowserViewControllerDelegate?
}

Highly customizable

Access to local documents and cloud storage

Be sure to coordinate file access
• NSFileCoordinator or UIDocument
Productivity
UI refinements
API enhancements
Enabling Large Titles

class UINavigationBar {
    var prefersLargeTitle: Bool
}

Enabling Large Titles

class UINavigationBar {
    var prefersLargeTitle: Bool
}

class UINavigationItem {
    var largeTitleDisplayMode: LargeTitleDisplayMode
}
Enabling Unified Search

class UINavigationItem {
    var searchController: UISearchController?
}

<table>
<thead>
<tr>
<th>App</th>
<th>Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>App Store</td>
<td>Badges, Banners</td>
</tr>
<tr>
<td>Calendar</td>
<td>Badges, Sounds, Alerts</td>
</tr>
<tr>
<td>FaceTime</td>
<td>Badges, Sounds, Alerts</td>
</tr>
<tr>
<td>Games</td>
<td>Badges, Sounds, Banners</td>
</tr>
<tr>
<td>Health</td>
<td>Badges</td>
</tr>
<tr>
<td>Home</td>
<td>Badges, Sounds, Banners</td>
</tr>
<tr>
<td>Mail</td>
<td>Badges, Sounds</td>
</tr>
<tr>
<td>Maps</td>
<td></td>
</tr>
</tbody>
</table>
Mailboxes

- Inbox
- VIP
- Unread
- Attachments
- Today
- Trash

Updated Just Now
Mailboxes

- Inbox
- VIP
- Unread
- Attachments
- Today
- Trash
Safe Area Insets

class UIView {

    // auto layout
    var safeAreaLayoutGuide: UILayoutGuide { get }

    // manual layout
    var safeAreaInsets: UIEdgeInsets { get }

    func safeAreaInsetsDidChange()

}
Safe Area Insets

class UIScrollView {

    var contentInsetAdjustmentBehavior: UIScrollViewContentInsetAdjustmentBehavior

    var adjustedContentInset: UIEdgeInsets { get }

}
<table>
<thead>
<tr>
<th>Name</th>
<th>Time</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Megan</td>
<td>8:30 AM</td>
<td>Hope you have a great flight! See you tonight!</td>
</tr>
<tr>
<td>Gram</td>
<td>Yesterday</td>
<td>Thinking about you a lot this week</td>
</tr>
<tr>
<td>Samantha</td>
<td>Saturday</td>
<td>How's the new place in San Francisco?</td>
</tr>
<tr>
<td>Mom</td>
<td>Friday</td>
<td>Looking forward to seeing you soon!</td>
</tr>
</tbody>
</table>
Megan
Hope you have a great flight! See you tonight!

Gram
Thinking about you a lot this week

Samantha
How's the new place in San Francisco?

Mom
Looking forward to seeing you soon!
class UISwipeActionsConfiguration {

    init(actions: [UIContextualAction])

    var performsFirstActionWithFullSwipe: Bool

}

class UIContextualAction {

    init(style: Style, title: String?, handler: UIContextualActionHandler)

}
<table>
<thead>
<tr>
<th>Name</th>
<th>Time</th>
<th>Message</th>
</tr>
</thead>
<tbody>
<tr>
<td>Megan</td>
<td>8:30 AM</td>
<td>Hope you have a great flight! See you tonight!</td>
</tr>
<tr>
<td>Gram</td>
<td>Yesterday</td>
<td>Thinking about you a lot this week</td>
</tr>
<tr>
<td>Samantha</td>
<td>Saturday</td>
<td>How’s the new place in San Francisco?</td>
</tr>
<tr>
<td>Mom</td>
<td>Friday</td>
<td>Looking forward to seeing you soon!</td>
</tr>
</tbody>
</table>
class UITableViewCell {

  var separatorInsetReference: UITableViewSeparatorInsetReference

}
<table>
<thead>
<tr>
<th>Event</th>
<th>Hall</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Updating Your App for iOS 11</td>
<td>3</td>
<td>Tuesday 4:10PM</td>
</tr>
<tr>
<td>Design Studio Shorts 2</td>
<td>3</td>
<td>Thursday 1:50PM</td>
</tr>
</tbody>
</table>
Productivity

UI refinements

API enhancements
Swift 4 and Foundation
Swift 4 and Foundation
Archiving Swift native types

```
enum Animal {
    case chicken
    case cow
    case sheep
}

struct Farm {
    let name: String
    let animals: [Animal]
}
```
Swift 4 and Foundation
Archiving Swift native types

```swift
enum Animal : Codable {
    case chicken
    case cow
    case sheep
}

struct Farm : Codable {
    let name: String
    let animals: [Animal]
}
```
Swift 4 and Foundation
Archiving Swift native types

```swift
ever Animal : Codable {
    case chicken
    case cow
    case sheep
}
```

```swift
struct Farm : Codable {
    let name: String
    let animals: [Animal]
}
```
Swift 4 and Foundation
Archiving Swift native types

```swift
enum Animal: Codable {
    case chicken
    case cow
    case sheep
}

struct Farm: Codable {
    let name: String
    let animals: [Animal]
}
```

NSCoding

JSON
Swift 4 and Foundation
Archiving Swift native types

```swift
enum Animal: Codable {
    case chicken
    case cow
    case sheep
}

struct Farm: Codable {
    let name: String
    let animals: [Animal]
}
```

NSCoding

JSON

Property lists
Swift 4 and Foundation

Key paths
Swift 4 and Foundation

Key paths

New KeyPath type

• Literal syntax

• Getters and setters
Swift 4 and Foundation

Key paths

New KeyPath type

• Literal syntax
• Getters and setters

let copresenterNameKeyPath = \Presenter.copresenter.name
Swift 4 and Foundation
Key paths

New KeyPath type
• Literal syntax
• Getters and setters

```swift
let copresenterNameKeyPath = \Presenter.copresenter.name

eliza[keyPath: copresenterNameKeyPath] // "Josh"
```
Swift 4 and Foundation

Key paths

New KeyPath type
• Literal syntax
• Getters and setters

```swift
let copresenterNameKeyPath = Presenter.copresenter.name
eliza[keyPath: copresenterNameKeyPath]  // “Josh”
eliza[keyPath: .copresenter.name]      // “Josh”
```
private var myContext = 42

func startObserving() {
    let eliza = ...
    eliza.addObserver(self, forKeyPath: "copresenter", ..., context: &myContext)
}

override func observeValue(forKeyPath keyPath: String?, of object: Any?, ..., context: UnsafeMutableRawPointer?) {
    if context == &myContext {
        print("Eliza’s co-presenter is now \(object?.copresenter.name)")
    } else {
        super.observeValue(forKeyPath: keyPath, of: object, ..., context: context)
    }
}
var token: NSKeyValueObservation? = nil

func startObserving() {
    let eliza = ...
    token = eliza.observe(.copresenter) { (object, change) in
        print("Eliza's co-presenter is now \(object.copresenter.name)")
    }
}
Swift 4 and Foundation
New block-based KVO!

```swift
var token: NSKeyValueObservation? = nil

func startObserving() {
    let eliza = ...
    token = eliza.observe(.copresenter) { (object, change) in
        print("Eliza's co-presenter is now \(object.copresenter.name)")
    }
}
```

What's New in Foundation
Hall 2
Wednesday 11:00AM
Deferring System Gestures
class UIViewController {
    func preferredScreenEdgesDeferringSystemGestures() -> UIRectEdge
}

Auto Layout
Auto Layout and Scroll View
Frame vs. content
Auto Layout and Scroll View
Frame vs. content

class UIScrollView {

    var contentLayoutGuide: UILayoutGuide { get }

    var frameLayoutGuide: UILayoutGuide { get }

}
imageView.centerXAnchor.constraint(equalTo: scrollView.contentLayoutGuide.centerXAnchor)
imageView.centerXAnchor.constraint(equalTo: scrollView.contentLayoutGuide.centerXAnchor)
imageView.centerYAnchor.constraint(equalTo: scrollView.contentLayoutGuide.centerYAnchor)
Dynamic Type
What's New in Cocoa Touch
San Jose McEnery Convention Center
365 S Market St, San Jose, CA 95113, United States

Tuesday, Jun 6, 2017
from 10:20 AM to 11 AM

What's New in Cocoa Touch San Jose McEnery Convention Center
11 AM
Lunch with Josh Favorite Diner
Noon

Calendar

Alert

Time to Leave

Delete Event
What's New in Cocoa Touch
San Jose McEnery Convention Center
365 S Market St, San Jose, CA 95113, United States

Tuesday, Jun 6, 2017
from 10:20 AM to 11 AM

11 AM
What’s New in Cocoa Touch San Jose McEnery Con...

Noon
Lunch with Josh Favorite Diner

Calendar

Alert
Time to Leave

Delete Event
What's New in Cocoa Touch
San Jose McEnery Convention Center
365 S Market St, San Jose, CA 95113, United States

Tuesday, Jun 6, 2017
from 10:20 AM to 11 AM
What’s New in Cocoa Touch
San Jose McEnery Convention Center
365 S Market St, San Jose, CA 95113, United States

Tuesday, Jun 6, 2017
from 10:20 AM to 11 AM

10 AM
What’s New in Cocoa Touch Sa...
11 AM
Lunch with Josh
Dynamic Type
Sizing your text
Dynamic Type
Sizing your text

How do you choose a font appropriate for your user’s dynamic type size?
Dynamic Type
Sizing your text

How do you choose a font appropriate for your user’s dynamic type size?

Old way

```swift
let bodyFont = UIFont.preferredFont(forTextStyle: .body)
let titleFont = UIFont.preferredFont(forTextStyle: .title1)
```
Dynamic Type
Sizing your text

How do you choose a font appropriate for your user’s dynamic type size?

Old way

```swift
let bodyFont = UIFont.preferredFont(forTextStyle: .body)
let titleFont = UIFont.preferredFont(forTextStyle: .title1)
```

But what if your app needs a custom font?
Dynamic Type
UIFontMetrics
Dynamic Type
UIFontMetrics

Scale any font

```swift
let bodyMetrics = UIFontMetrics(forTextStyle: .body)
```
Dynamic Type
UIFontMetrics

Scale any font

```swift
let bodyMetrics = UIFontMetrics(forTextStyle: .body)

let standardFont = ... // any font you want, for standard type size
```
Dynamic Type
UIFontMetrics

Scale any font

```swift
let bodyMetrics = UIFontMetrics(forTextStyle: .body)

let standardFont = ... // any font you want, for standard type size
let font = bodyMetrics.scaledFont(for: standardFont)
```
Dynamic Type
UIFontMetrics

Scale any font

```
let bodyMetrics = UIFontMetrics(forTextStyle: .body)

let standardFont = ... // any font you want, for standard type size
let font = bodyMetricsscaledFont(for: standardFont)
```

Can also scale arbitrary layout values
Dynamic Type
UIFontMetrics

Scale any font

```swift
let bodyMetrics = UIFontMetrics(forTextStyle: .body)

let standardFont = ... // any font you want, for standard type size
let font = bodyMetrics.scaledFont(for: standardFont)
```

Can also scale arbitrary layout values

```swift
let titleMetrics = UIFontMetrics(forTextStyle: .title3)

let standardHeight = ... // button height for standard type size
let height = titleMetrics.scaledValue(forValue: standardHeight)
```
Dynamic Type

AutoLayout
Dynamic Type
AutoLayout

Tuesday 12:30pm
Lunch with Josh
Dynamic Type

AutoLayout

Tuesday 12:30pm

Lunch with Josh
Tuesday 12:30pm
Lunch with Josh
Dynamic Type
AutoLayout

Tuesday 12:30pm
Lunch with Josh

Tuesday 12:30pm
Lunch with Josh
<table>
<thead>
<tr>
<th>Time</th>
<th>Activity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
<tr>
<td>Time</td>
<td>Event</td>
</tr>
<tr>
<td>------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
</tbody>
</table>

```swift
let topAnchor = topLabel.lastBaselineAnchor
let bottomAnchor = bottomLabel.firstBaselineAnchor
```
let topAnchor = topLabel.lastBaselineAnchor
let bottomAnchor = bottomLabel.firstBaselineAnchor
bottomAnchor.constraintEqualToSystemSpacing(below: topAnchor)
<table>
<thead>
<tr>
<th>Time</th>
<th>Event</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tuesday 12:30pm</td>
<td>Lunch with Josh</td>
</tr>
</tbody>
</table>

```swift
let topAnchor = topLabel.lastBaselineAnchor
let bottomAnchor = bottomLabel.firstBaselineAnchor
bottomAnchor.constraintEqualToSystemSpacing(below: topAnchor)
```
Dynamic Type

AutoLayout

Works with VFL too

```swift
NSLayoutConstraint.constraintsWithVisualFormat(
    "V:|-[topLabel]-[bottomLabel]-|", // '-' gives you system spacing
    options: [NSLayoutOption spacingBaselineToBaseline],
    metrics: nil,
    views: ...)
```
Dynamic Type
AutoLayout

Works with VFL too

```swift
NSLayoutConstraints.constraintsWithVisualFormat(
    "V:-[topLabel]-[bottomLabel]-|", // '-' gives you system spacing
    options: [spacingBaselineToBaseline],
    metrics: nil,
    views: ...)
```

And with UIStackView

```swift
stackView.baselineRelativeArrangement = true
stackView.spacing = .spacingUseSystem
```
Dynamic Type

AutoLayout

Works with VFL too

```swift
NSLayoutConstraint.constraintsWithVisualFormat(
    "V:|-[topLabel]-[bottomLabel]-|",       // '-' gives you system spacing
    options: [NSLayoutOptions(spacingBaselineToBaseline)],
    metrics: nil,
    views: ...)
```

And with UIStackView

```swift
stackView.baselineRelativeArrangement = true
stackView.spacing = .spacingUseSystem
```
Password AutoFill
apple.com — rmondello
apple.com

bbc.co.uk — rmondello
bbc.co.uk

bing.com — rmondello
bing.com

example.com — rmondello
shiny.example.com

icloud.com — rmondello
icloud.com

twitter.com — rmondello
mobile.twitter.com

weather.com — rmondello
weather.com

webkit.org — rmondello
webkit.org

wikipedia.org — rmondello
wikipedia.org
Introducing Password AutoFill for Apps

Grand Ballroom A

Wednesday 9:00AM
Asset Catalogs
Asset Catalogs
Colors and icons

class UIColor {
    init?(named name: String)
}

Asset Catalogs
Colors and icons

```swift
class UIColor {

    init?(named name: String)

}
```

Wide gamut color support
Asset Catalogs
Colors and icons

class UIColor {

    init?(named name: String)

}

Wide gamut color support

Wide gamut colors for icons
Asset Catalogs

Colors and icons

class UIColor {
    init?(named name: String)
}

Wide gamut color support

Wide gamut colors for icons

App thinning for icons
Asset Catalogs
PDF-backed images
Asset Catalogs

PDF-backed images
Asset Catalogs
PDF-backed images
Asset Catalogs
PDF-backed images

Preserve vector data if
Asset Catalogs
PDF-backed images

Preserve vector data if
• Image used at multiple sizes
Asset Catalogs
PDF-backed images

Preserve vector data if
• Image used at multiple sizes
• Symbolic glyph that resizes with dynamic type
Asset Catalogs
PDF-backed images

Preserve vector data if
• Image used at multiple sizes
• Symbolic glyph that resizes with dynamic type
• Tab bar image
ProMotion
Variable Refresh Rate
Query screen’s refresh rate

class UIScreen {

    var maximumFramesPerSecond: Int { get }

}

Variable Refresh Rate
Best practices for UIKit apps
Variable Refresh Rate
Best practices forUIKit apps

Optimize your app’s performance
Variable Refresh Rate
Best practices for UIKit apps

Optimize your app’s performance

• Use Instruments to check your frame rate during scrolling and animations
Variable Refresh Rate
Best practices for Metal apps
Variable Refresh Rate
Best practices for Metal apps

Use flexible presentation times to ensure smooth playback
Variable Refresh Rate
Best practices for Metal apps

Use flexible presentation times to ensure smooth playback

Present frames less frequently to conserve power
Variable Refresh Rate
Best practices for Metal apps

Use flexible presentation times to ensure smooth playback

Present frames less frequently to conserve power

Use Xcode’s GPU Report feature to inspect performance characteristics
Variable Refresh Rate

Best practices for Metal apps

Use flexible presentation times to ensure smooth playback

Present frames less frequently to conserve power

Use Xcode’s GPU Report feature to inspect performance characteristics
What’s New
Localization
الإعدادات

Joshua Shaffer
App Store, iTunes, iCloud, Apple ID

تحديث البرامج
عدد المهام
انترنت
البيانات الخلوية
إيفر:
مساحة تخزين iPad
تجديد التطبيقات في الخلفية
الإشعارات
مركز التحكم
عدم الإزعاج
لوحة المفاتيح
<table>
<thead>
<tr>
<th>Arabic</th>
<th>English</th>
</tr>
</thead>
<tbody>
<tr>
<td>حول</td>
<td>About</td>
</tr>
<tr>
<td>تحديث البرامج</td>
<td>Update Software</td>
</tr>
<tr>
<td>تعدد المهام</td>
<td>Multiple Tasks</td>
</tr>
<tr>
<td>إمكانية وصول دوي الاحتياجات الخاصة</td>
<td>AssistiveTouch Accessibility Features</td>
</tr>
<tr>
<td>مساحة تخزين iPad</td>
<td>iPad Storage</td>
</tr>
<tr>
<td>تحديث التطبيقات في الخلفية</td>
<td>Update Apps in Background</td>
</tr>
<tr>
<td>الإشعارات</td>
<td>Notifications</td>
</tr>
<tr>
<td>مركز التحكم</td>
<td>Control Center</td>
</tr>
<tr>
<td>عدم الإزعاج</td>
<td>Do Not Disturb</td>
</tr>
</tbody>
</table>

More Information

https://developer.apple.com/wwdc17/201
# Related Sessions

<table>
<thead>
<tr>
<th>Session</th>
<th>Location</th>
<th>Date</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introducing Drag and Drop</td>
<td>Hall 3</td>
<td>Tuesday</td>
<td>11:20AM</td>
</tr>
<tr>
<td>Mastering Drag and Drop</td>
<td>Executive Ballroom</td>
<td>Wednesday</td>
<td>11:00AM</td>
</tr>
<tr>
<td>Drag and Drop with Collection and Table View</td>
<td>Hall 2</td>
<td>Thursday</td>
<td>9:00AM</td>
</tr>
<tr>
<td>Data Delivery with Drag and Drop</td>
<td>Hall 2</td>
<td>Thursday</td>
<td>10:00AM</td>
</tr>
<tr>
<td>Building Great Document-Based Apps in iOS 11</td>
<td>Hall 2</td>
<td>Thursday</td>
<td>1:50PM</td>
</tr>
<tr>
<td>Updating Your App for iOS 11</td>
<td>Hall 3</td>
<td>Tuesday</td>
<td>4:10PM</td>
</tr>
<tr>
<td>Design Studio Shorts 2</td>
<td>Hall 3</td>
<td>Thursday</td>
<td>1:50PM</td>
</tr>
</tbody>
</table>
## Related Sessions

<p>| Session                                                        | Location            | Date and Time        |
|                                                               |                     |                      |
| What’s New in Foundation                                      | Hall 2              | Wednesday 11:00AM    |
| Building Apps with Dynamic Type                               | Executive Ballroom  | Friday 1:50PM        |
| Introducing Password AutoFill for Apps                        | Grand Ballroom A    | Wednesday 9:00AM     |
| Introducing Metal 2                                            | Executive Ballroom  | Tuesday 1:50PM       |
| What’s New in International User Interfaces                   |                     | WWDC 2016            |
| Increase Usage of Your App With Proactive Suggestions          |                     | WWDC 2016            |</p>
<table>
<thead>
<tr>
<th>Lab</th>
<th>Lab Location</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>UIKit and Drag and Drop Lab</td>
<td>Technology Lab C</td>
<td>Tue 1:50PM–4:10PM</td>
</tr>
<tr>
<td>Cocoa Touch Lab</td>
<td>Technology Lab I</td>
<td>Wed 3:10PM-6:00PM</td>
</tr>
<tr>
<td>UIKit and Collection View Lab</td>
<td>Technology Lab B</td>
<td>Thur 10:00AM-12:30PM</td>
</tr>
<tr>
<td>Cocoa Touch and Haptics Lab</td>
<td>Technology Lab C</td>
<td>Fri 12:00PM-1:50PM</td>
</tr>
<tr>
<td>Text and Fonts Lab</td>
<td>Technology Lab H</td>
<td>Thur 1:50PM-3:50PM</td>
</tr>
<tr>
<td>Accessibility and Dynamic Type Lab</td>
<td>Technology Lab C</td>
<td>Fri 12:30PM-4:00PM</td>
</tr>
<tr>
<td>Safari, WebKit, and AutoFill Lab</td>
<td>Technology Lab D</td>
<td>Wed 10:00AM-12:00PM</td>
</tr>
<tr>
<td>Foundation Lab</td>
<td>Technology Lab C</td>
<td>Wed 1:00PM-3:10PM</td>
</tr>
</tbody>
</table>