#WWDC19

Architecting Your App for Multiple Windows

Janum Trivedi, UIKit

Changes to app lifecycle

Using the scene delegate

Architecture

Changes to app lifecycle

Using the scene delegate

Architecture

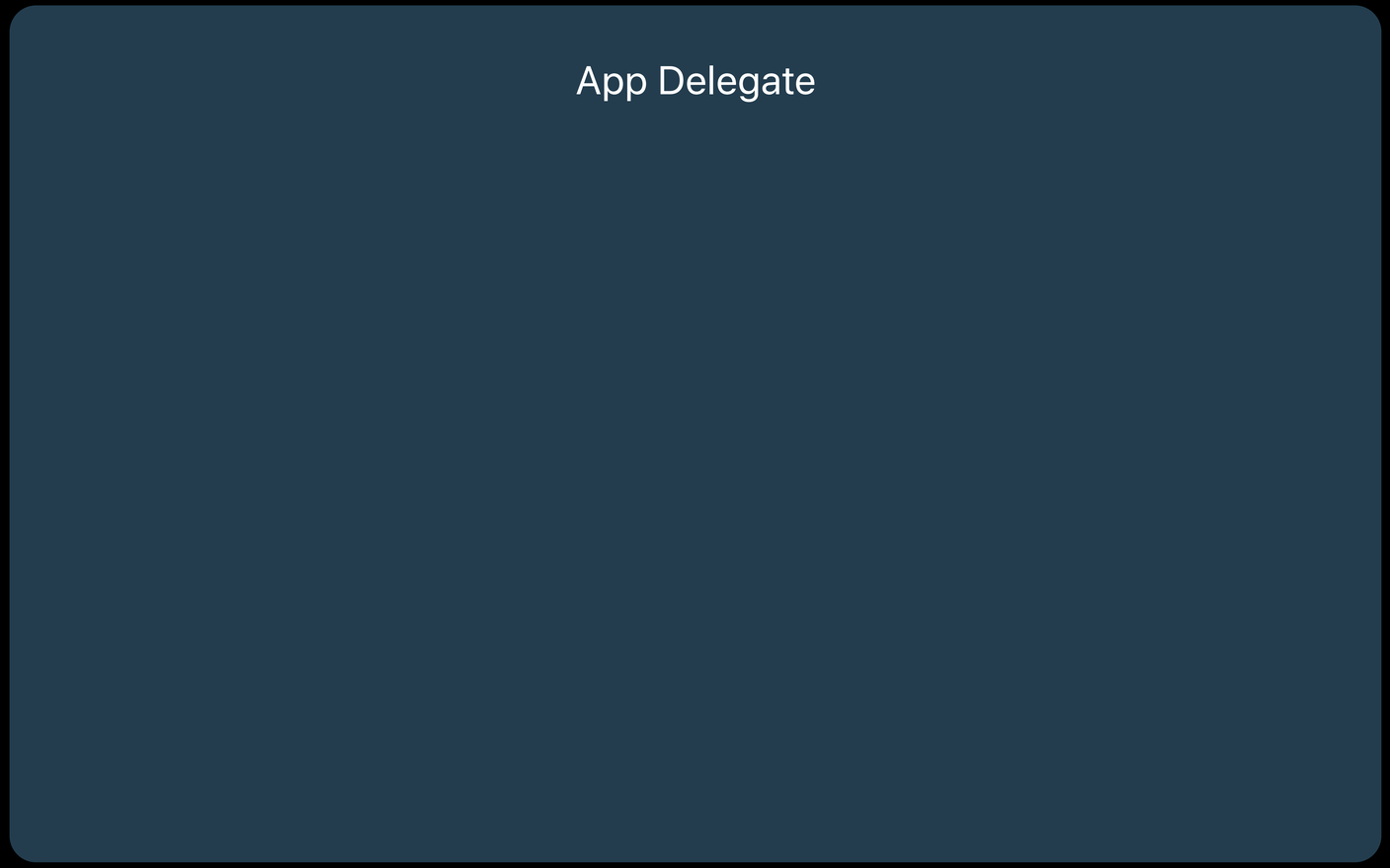
Changes to app lifecycle

Using the scene delegate

Architecture

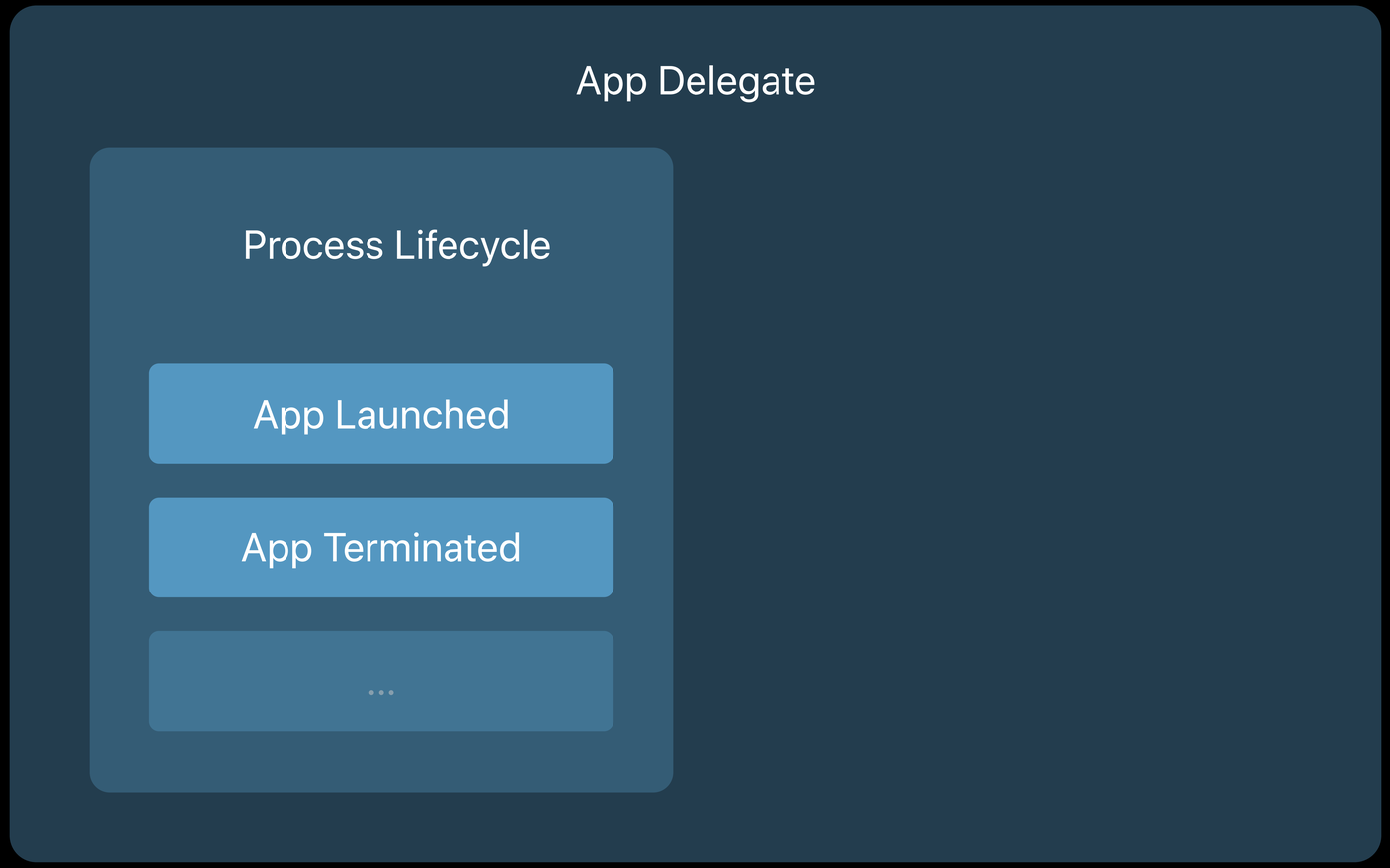
App Delegate Responsibilities

iOS 12 and earlier



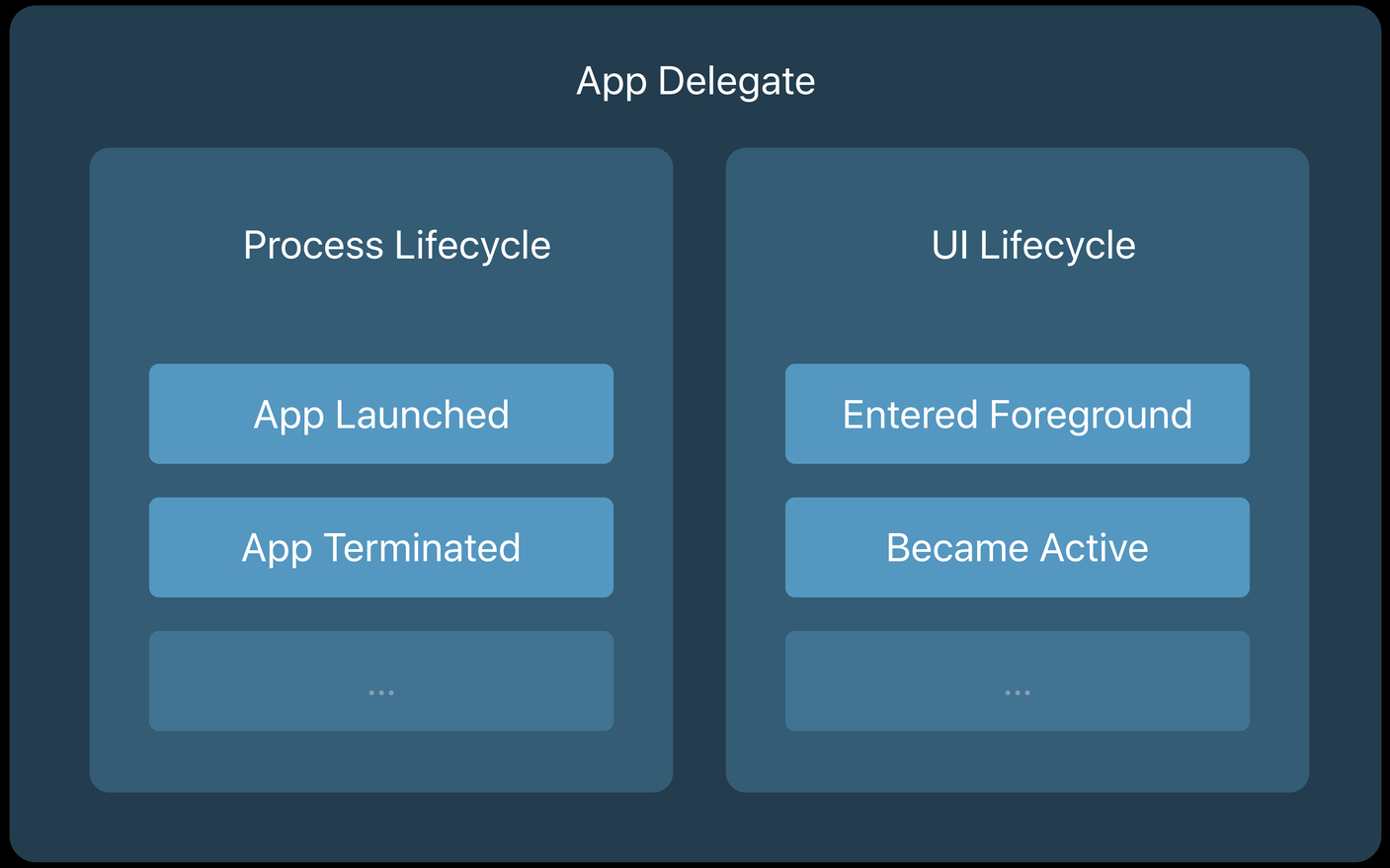
App Delegate Responsibilities

iOS 12 and earlier



App Delegate Responsibilities

iOS 12 and earlier




App A
App B
App C
App D

```swift
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

    func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]?) -> Bool {

        // Do some global setup
        Database.connect()

        // Set up the UI...
        window = UIWindow()
    }
}
```

```
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

    func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]?) -> Bool {

        // Do some global setup
        Database.connect()

        // Set up the UI...
        window = UIWindow()
    }
}
```

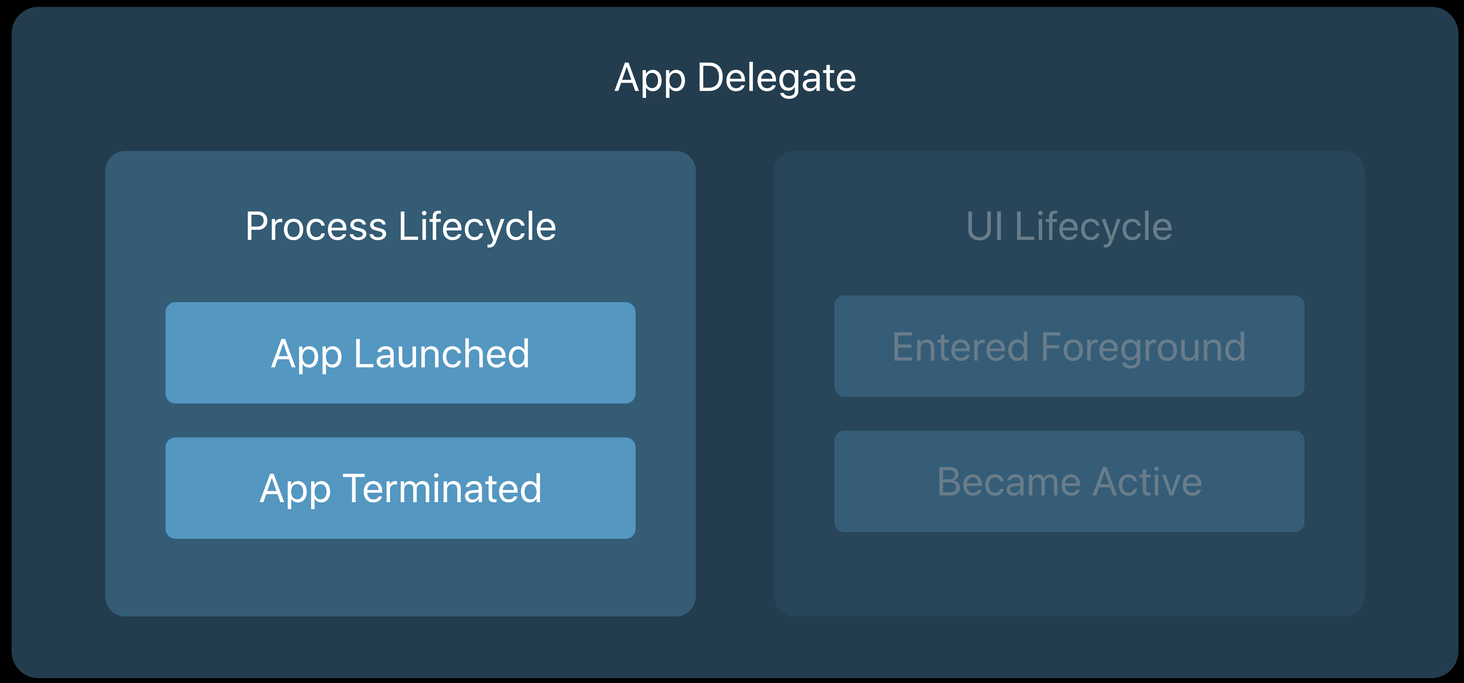
```swift
@UIApplicationMain
class AppDelegate: UIResponder, UIApplicationDelegate {
    var window: UIWindow?

    func application(_ application: UIApplication, didFinishLaunchingWithOptions launchOptions:
[UIApplication.LaunchOptionsKey: Any]?) -> Bool {

        // Do some global setup
        Database.connect()

        // Set up the UI...
        window = UIWindow()
    }
}
```


App A
App B
App C
App D


Session A
Session B
Session C
Session D

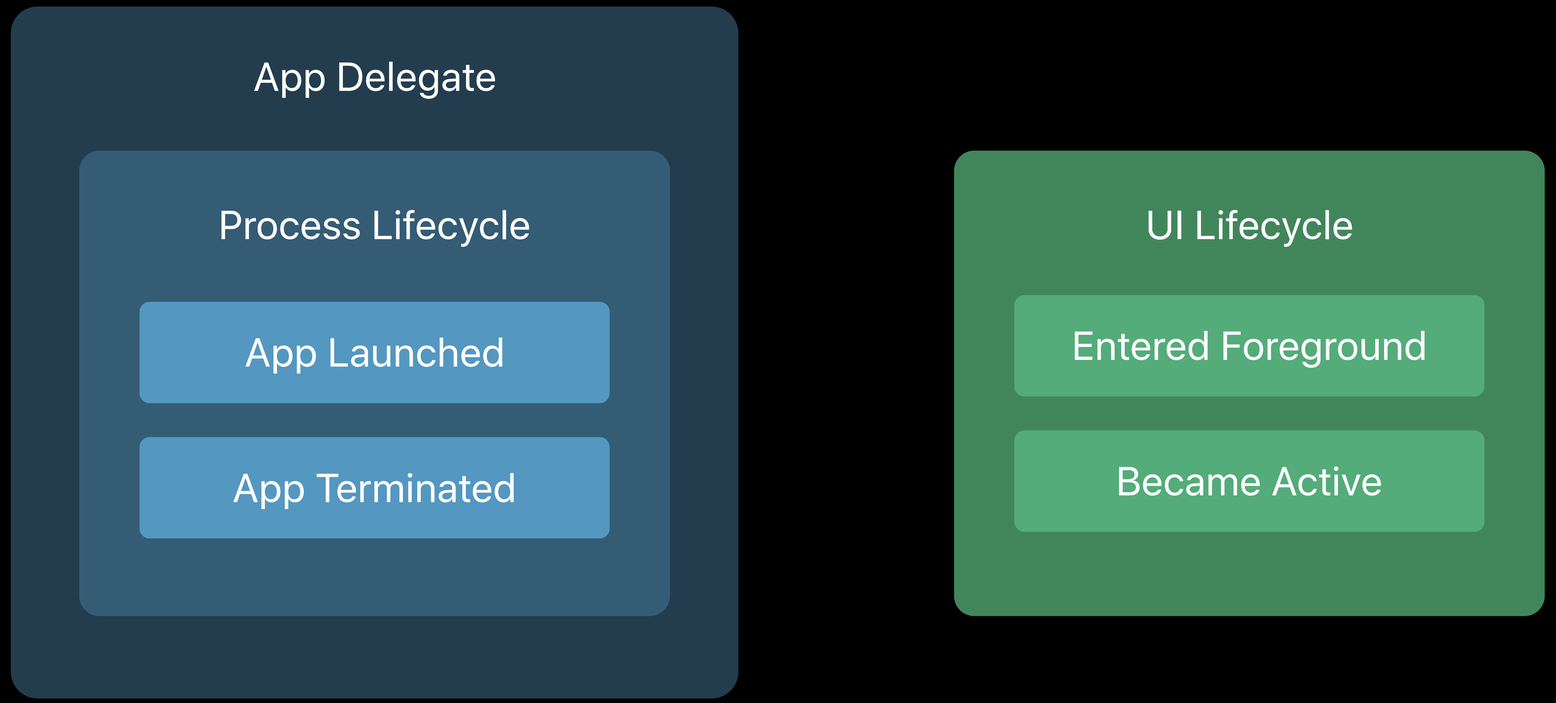
App Delegate Changes

iOS 13



App Delegate Changes

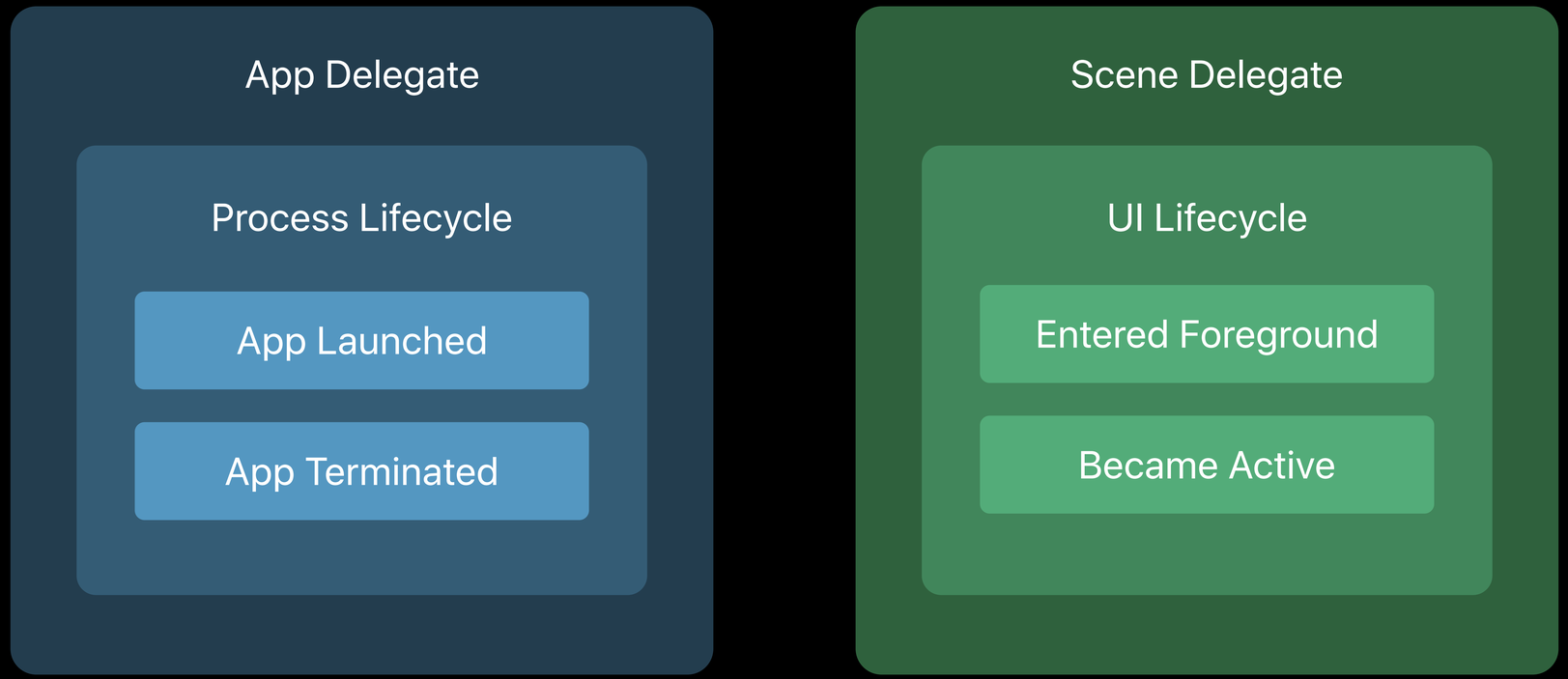
iOS 13



App Delegate Changes

iOS 13

NEW



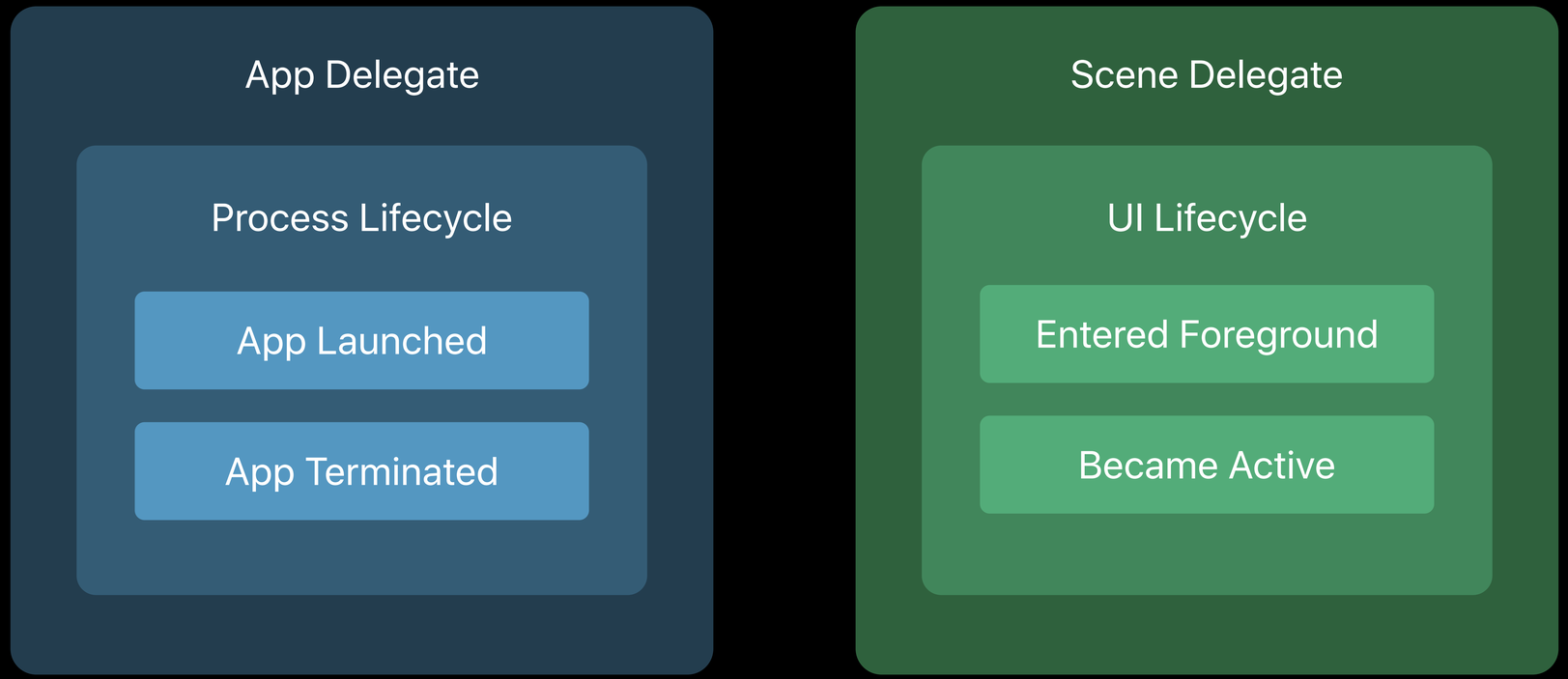
iOS 13

UIApplicationDelegate	UISceneDelegate
`application:willEnterForeground`	`scene:willEnterForeground`
`application:didEnterBackground`	`scene:didEnterBackground`
`application:willResignActive`	`scene:willResignActive`
`application:didBecomeActive`	`scene:didBecomeActive`

Session Lifecycle

iOS 13

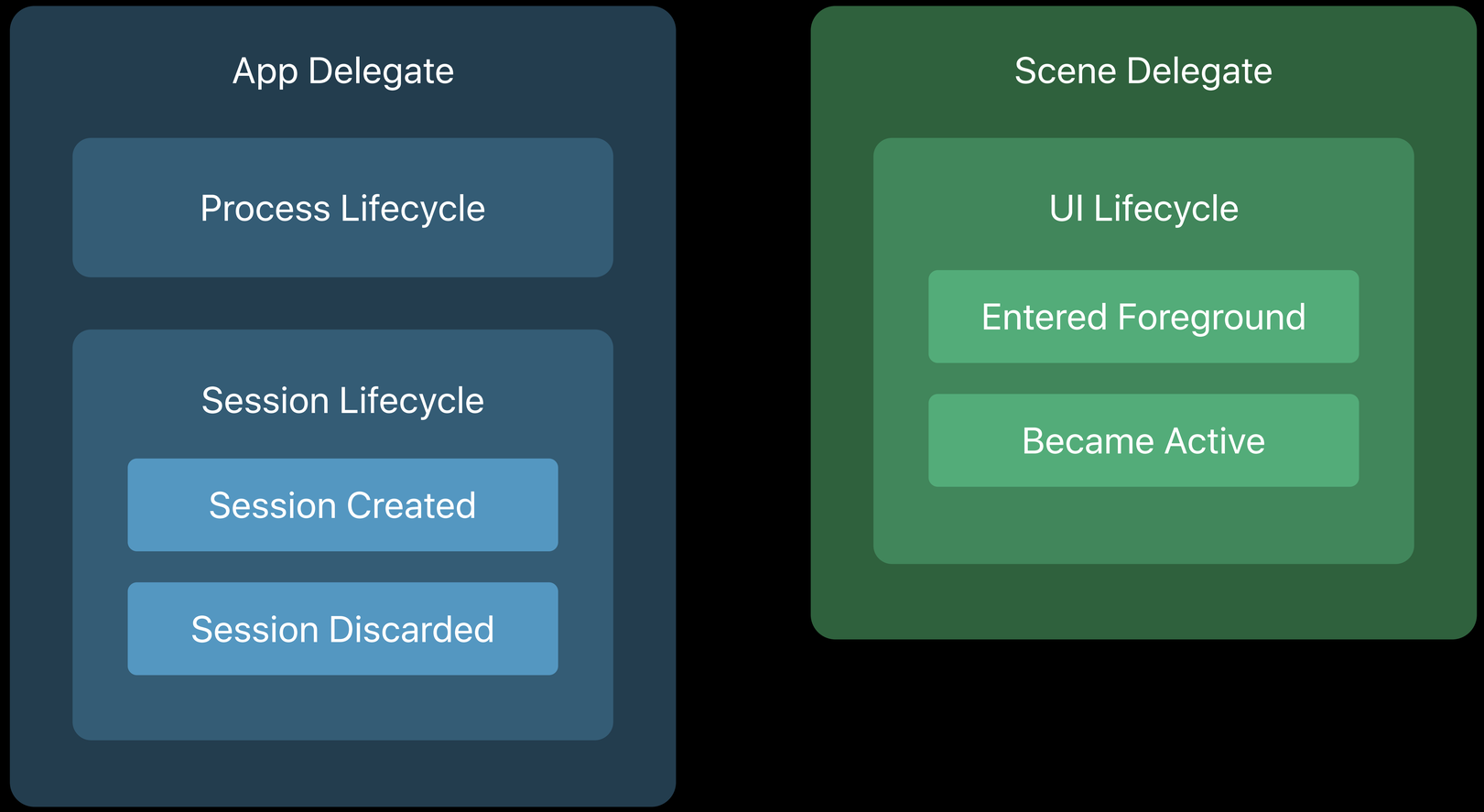
NEW

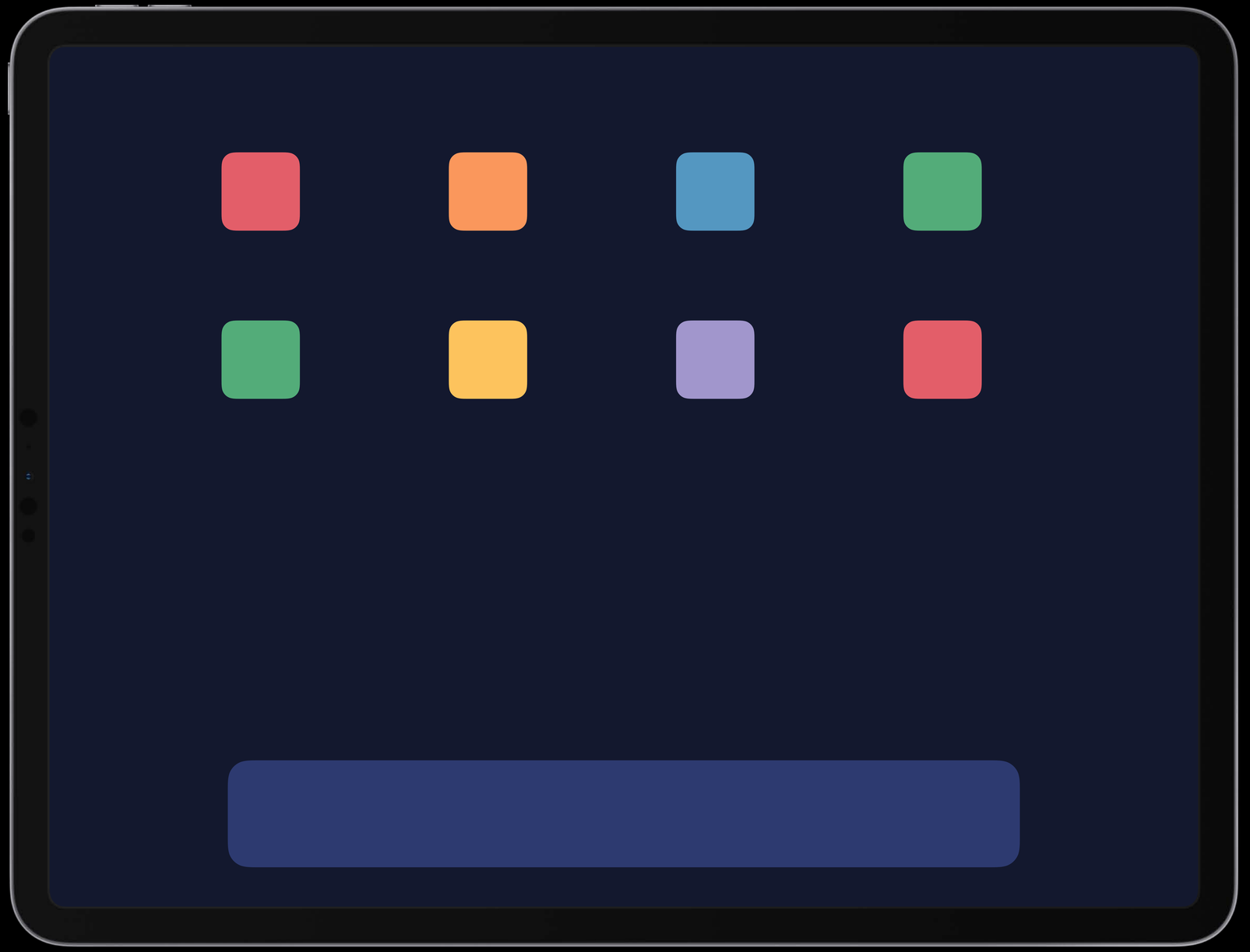


Session Lifecycle

iOS 13

NEW





App Delegate

didFinishLaunching

App Delegate

didFinishLaunching

configurationForSession

Configuring New Sessions

```
func application(_ application: UIApplication,
                 configurationForConnecting connectingSceneSession: UISceneSession,
                 options: UIScene.ConnectionOptions) -> UISceneConfiguration
```

Select a scene *configuration*

Provides user activities, URLs

Static and dynamic definition

Configuring New Sessions

```
func application(_ application: UIApplication,
                 configurationForConnecting connectingSceneSession: UISceneSession,
                 options: UIScene.ConnectionOptions) -> UISceneConfiguration
```

Select a scene *configuration*

Provides user activities, URLs

Static and dynamic definition

```swift
// Configuring incoming scenes

class AppDelegate: UIResponder, UIApplicationDelegate {

    func application(_ application: UIApplication,
                     configurationForConnecting connectingSceneSession: UISceneSession,
                     options: UIScene.ConnectionOptions) -> UISceneConfiguration {

        // First check the options...
        return UISceneConfiguration(name: "Default", sessionRole: connectingSceneSession.role)
    }

}
```

```
// Configuring incoming scenes

class AppDelegate: UIResponder, UIApplicationDelegate {

    func application(_ application: UIApplication,
                     configurationForConnecting connectingSceneSession: UISceneSession,
                     options: UIScene.ConnectionOptions) -> UISceneConfiguration {

        // First check the options...
        return UISceneConfiguration(name: "Default", sessionRole: connectingSceneSession.role)
    }

}
```

App Delegate

didFinishLaunching

configurationForSession

Scene Delegate

willConnectToSession

```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

    var window: UIWindow?

    func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)

        window = UIWindow(windowScene: scene as! UIWindowScene)

        if let activity = options.userActivities.first ?? session.stateRestorationActivity {
            configure(window: window, with: activity)
        }

    }

}
```

```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

    var window: UIWindow?

    func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)

        window = UIWindow(windowScene: scene as! UIWindowScene)

        if let activity = options.userActivities.first ?? session.stateRestorationActivity {
            configure(window: window, with: activity)
        }

    }

}
```

App Delegate

didFinishLaunching

configurationForSession

Scene Delegate

willConnectToSession

Scene Delegate

willConnectToSession

Scene Delegate

willConnectToSession

willResignActive

didEnterBackground

Scene Delegate

willConnectToSession

willResignActive

didEnterBackground

didDisconnect

Scene Disconnection

```
func sceneDidDisconnect(_ scene: UIScene)
```

System is releasing the scene

May be called any time

Release associated resources

The scene may return!

Scene Disconnection

```
func sceneDidDisconnect(_ scene: UIScene)
```

System is releasing the scene

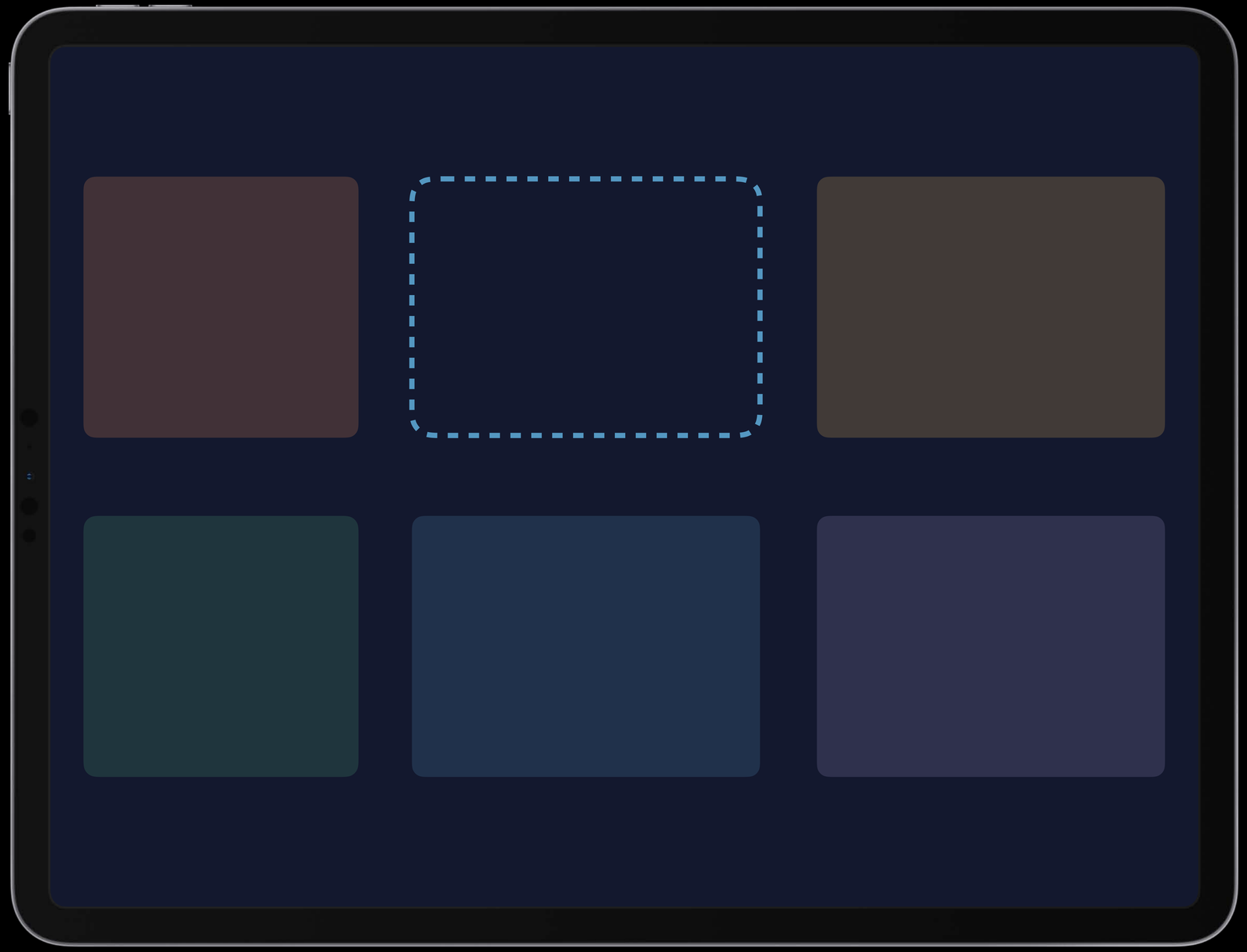
May be called any time

Release associated resources

The scene may return!

didEnterBackground

didDisconnect



didEnterBackground

didDisconnect

App Delegate

didDiscardSceneSession

Cleaning up Discarded Sessions

```
func application(_ application: UIApplication,
                 didDiscardSceneSessions sceneSessions: Set<UISceneSession>)
```

For permanently discarded sessions

Delete associated data

May be called after next launch

Cleaning up Discarded Sessions

```
func application(_ application: UIApplication,
                 didDiscardSceneSessions sceneSessions: Set<UISceneSession>)
```

For permanently discarded sessions

Delete associated data

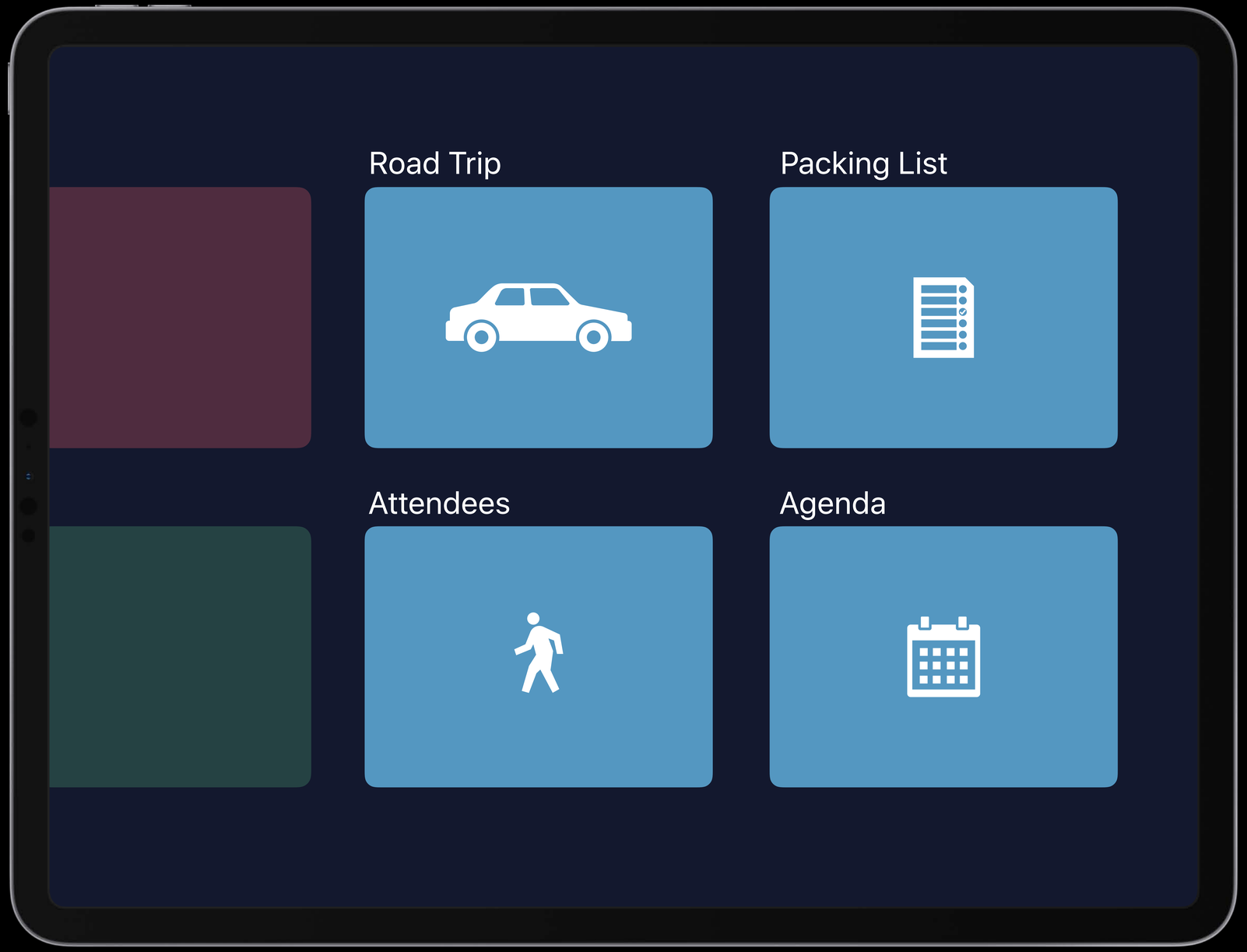
May be called after next launch

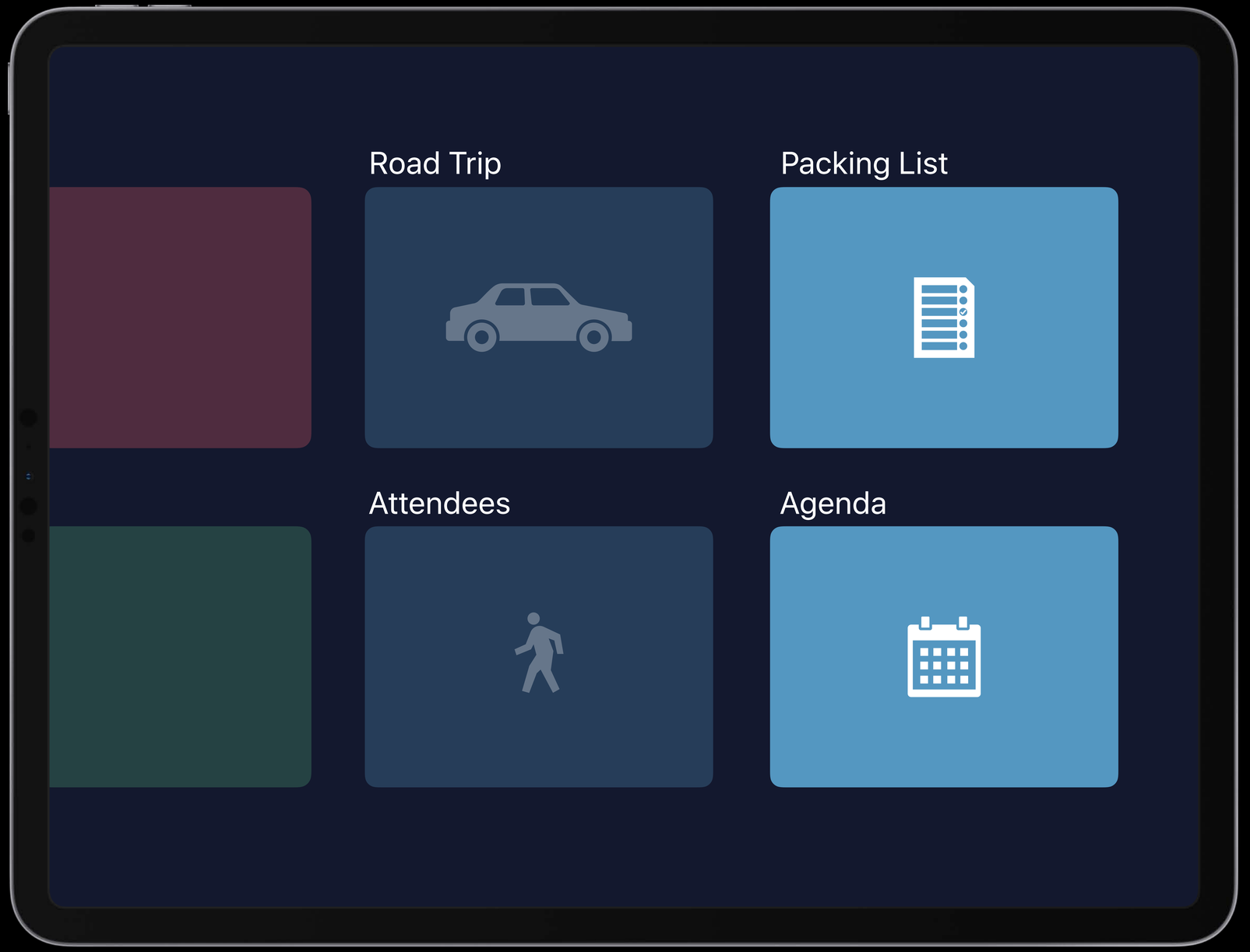
Changes to app lifecycle

Using the scene delegate

Architecture

State Restoration


Road Trip
Packing List
Attendees
Agenda


Road Trip
Packing List
Attendees
Agenda

Per-Scene State Restoration

NEW

```
func stateRestorationActivity(for scene: UIScene) -> NSUserActivity?
```

Called on scene background

Encode state via NSUserActivity

Data protection

Per-Scene State Restoration

NEW

```
func stateRestorationActivity(for scene: UIScene) -> NSUserActivity?
```

Called on scene background

Encode state via NSUserActivity

Data protection

```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

   func stateRestorationActivity(for scene: UIScene) -> NSUserActivity? {
      let currentActivity = fetchCurrentUserActivity(for: self.window)
      return currentActivity
   }

   func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)
      if let restorationActivity = session.stateRestorationActivity {
         self.configure(window: window, with: restorationActivity)
      }
  }
```

```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

    func stateRestorationActivity(for scene: UIScene) -> NSUserActivity? {
        let currentActivity = fetchCurrentUserActivity(for: self.window)
        return currentActivity
    }

    func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)
        if let restorationActivity = session.stateRestorationActivity {
            self.configure(window: window, with: restorationActivity)
        }
    }
```

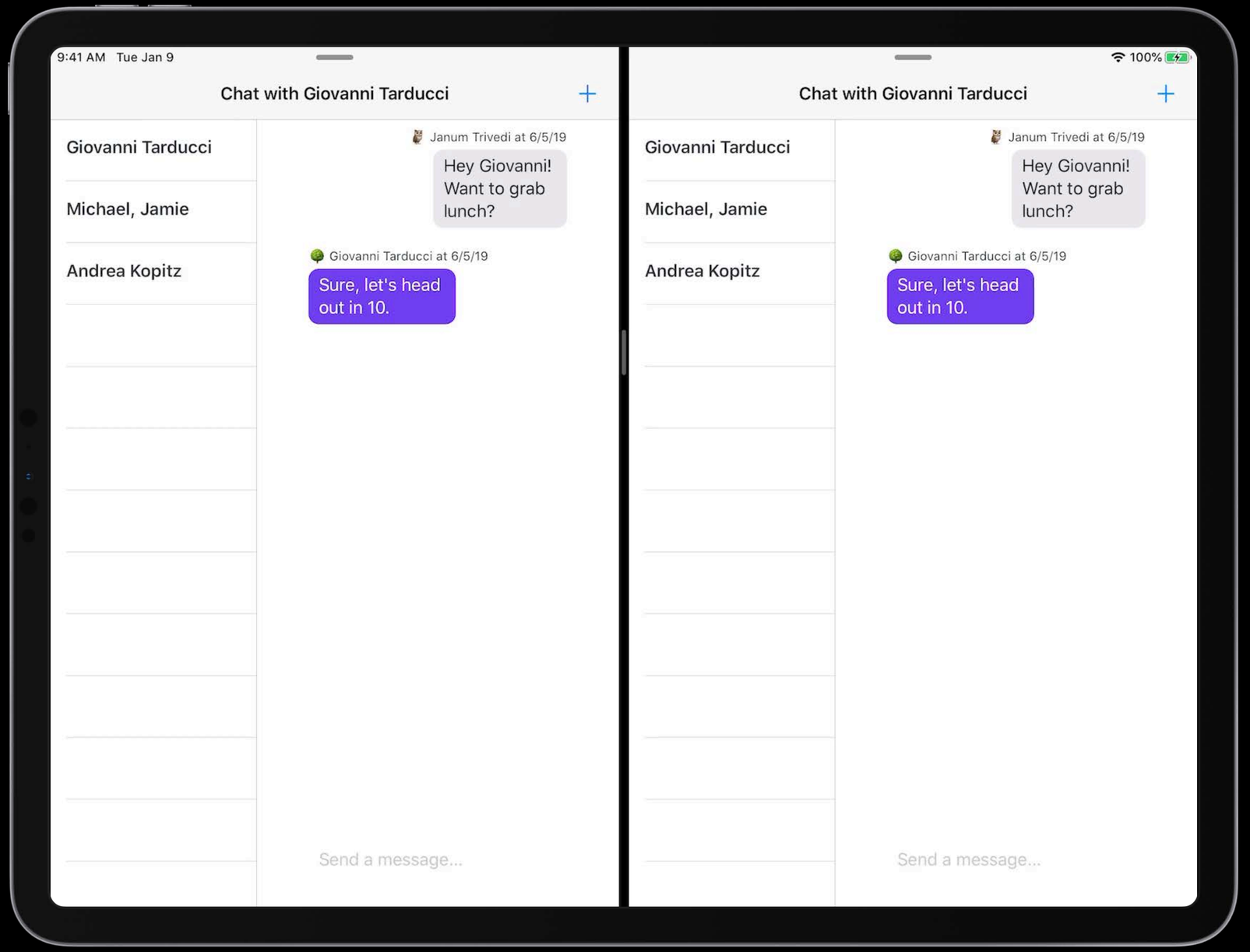
```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

   func stateRestorationActivity(for scene: UIScene) -> NSUserActivity? {
      let currentActivity = fetchCurrentUserActivity(for: self.window)
      return currentActivity
   }

   func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)
      if let restorationActivity = session.stateRestorationActivity {
         self.configure(window: window, with: restorationActivity)
      }
  }
```

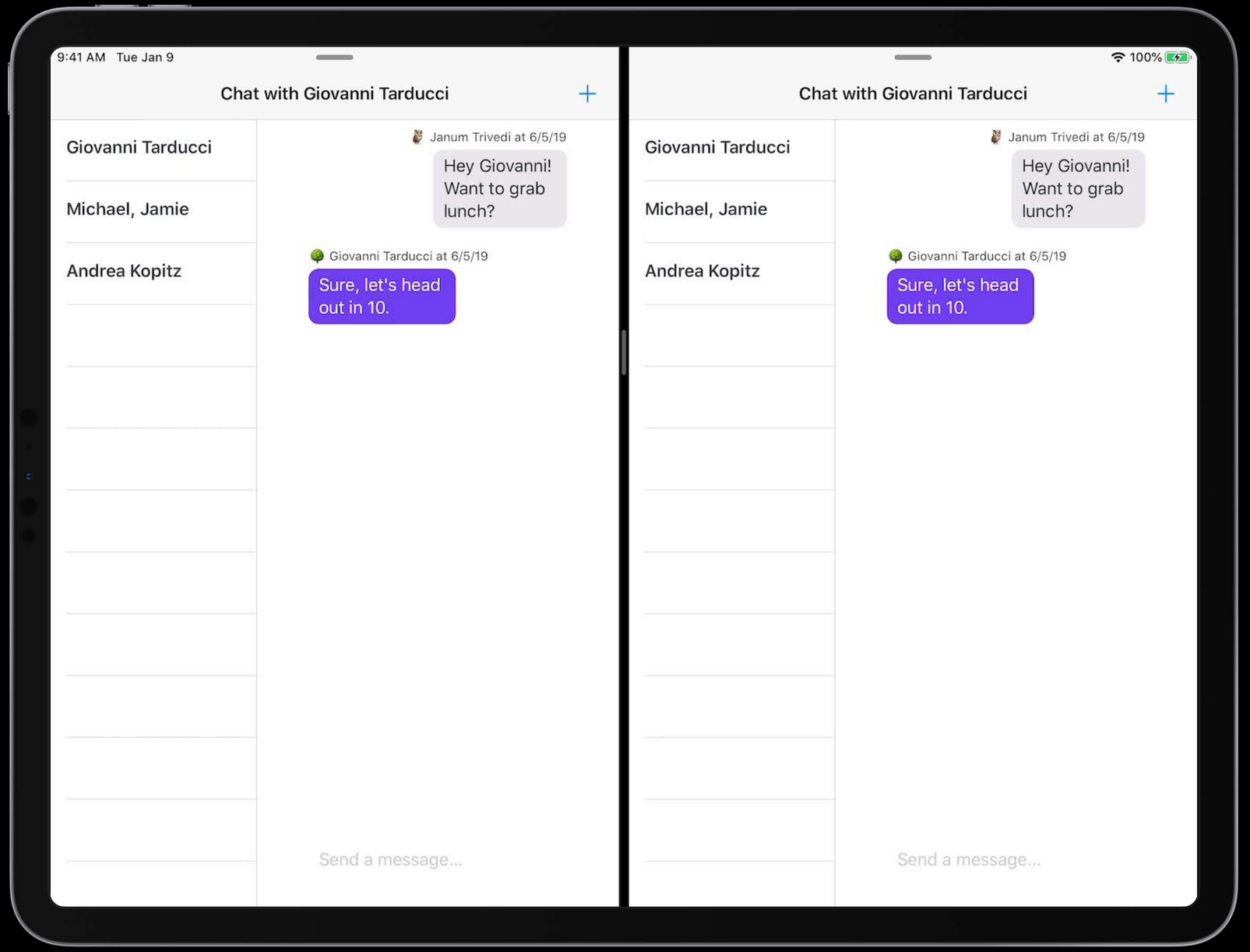
```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

   func stateRestorationActivity(for scene: UIScene) -> NSUserActivity? {
      let currentActivity = fetchCurrentUserActivity(for: self.window)
      return currentActivity
   }

   func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)
      if let restorationActivity = session.stateRestorationActivity {
         self.configure(window: window, with: restorationActivity)
      }
  }
```

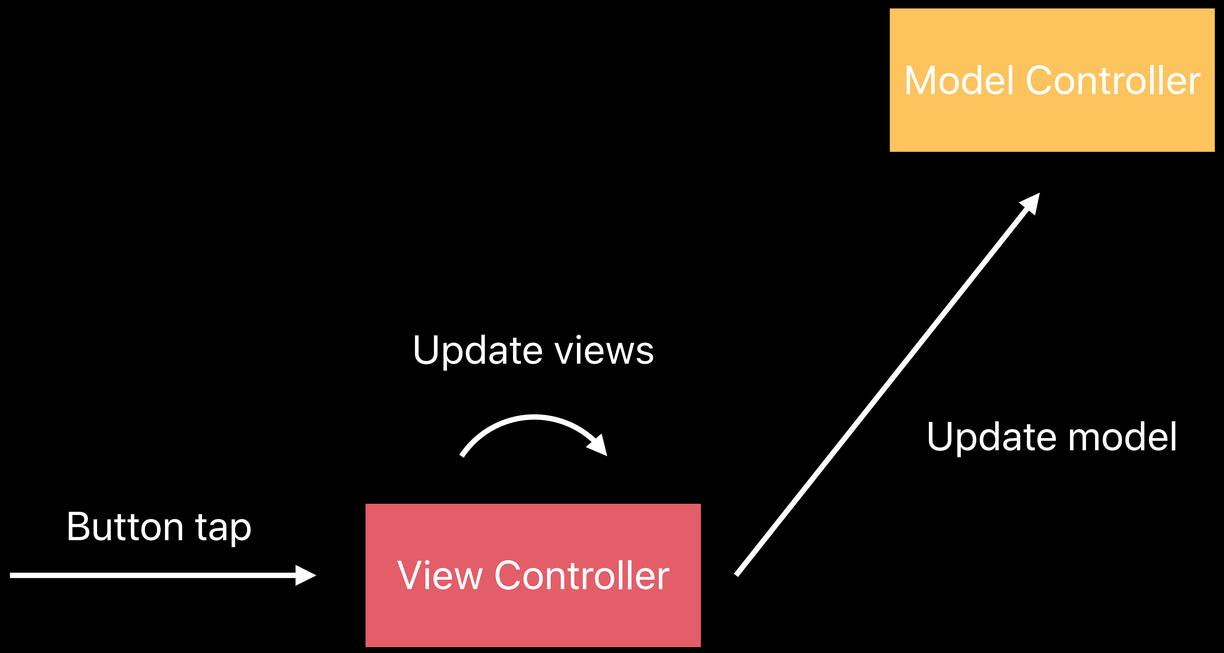
```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

   func stateRestorationActivity(for scene: UIScene) -> NSUserActivity? {
      let currentActivity = fetchCurrentUserActivity(for: self.window)
      return currentActivity
   }

   func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)
      if let restorationActivity = session.stateRestorationActivity {
         self.configure(window: window, with: restorationActivity)
      }
  }
```

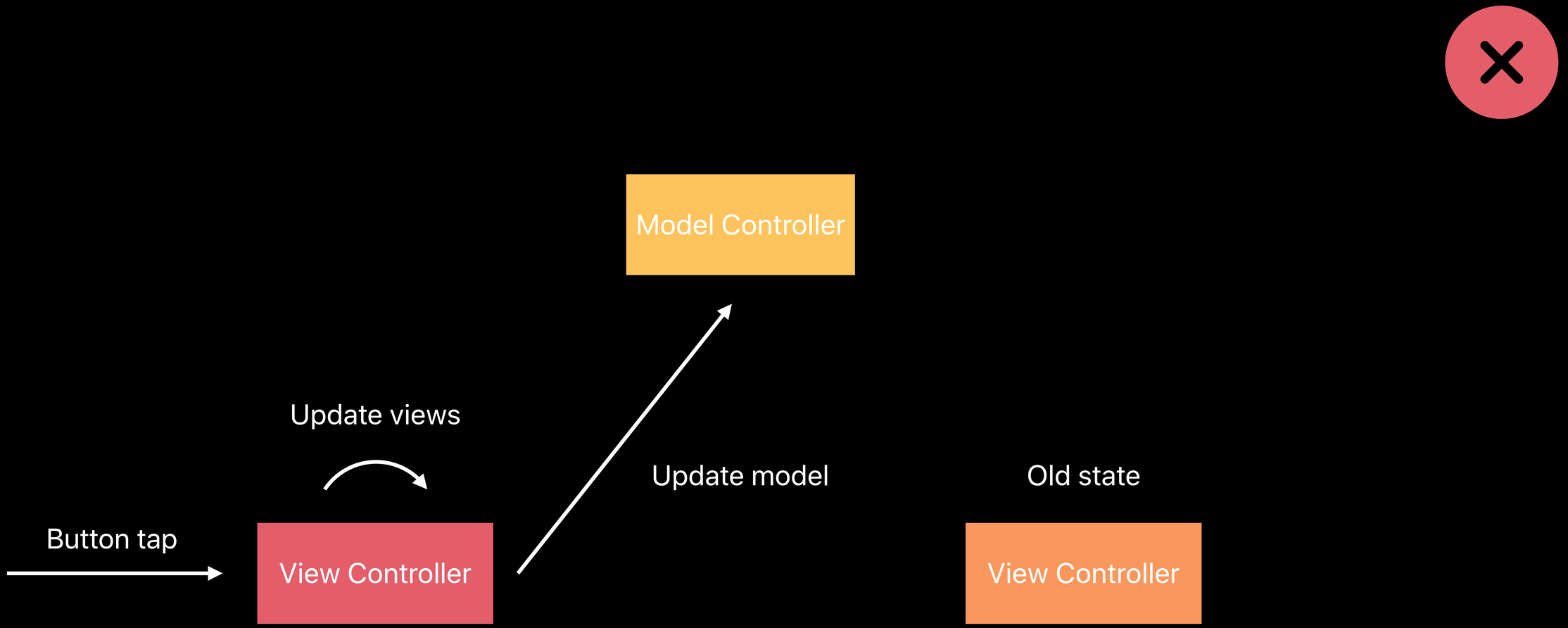
```
class SceneDelegate: UIResponder, UIWindowSceneDelegate {

   func stateRestorationActivity(for scene: UIScene) -> NSUserActivity? {
      let currentActivity = fetchCurrentUserActivity(for: self.window)
      return currentActivity
   }

   func scene(_ scene: UIScene, willConnectTo session: UISceneSession,
options: .ConnectionOptions)
      if let restorationActivity = session.stateRestorationActivity {
         self.configure(window: window, with: restorationActivity)
      }
  }
```

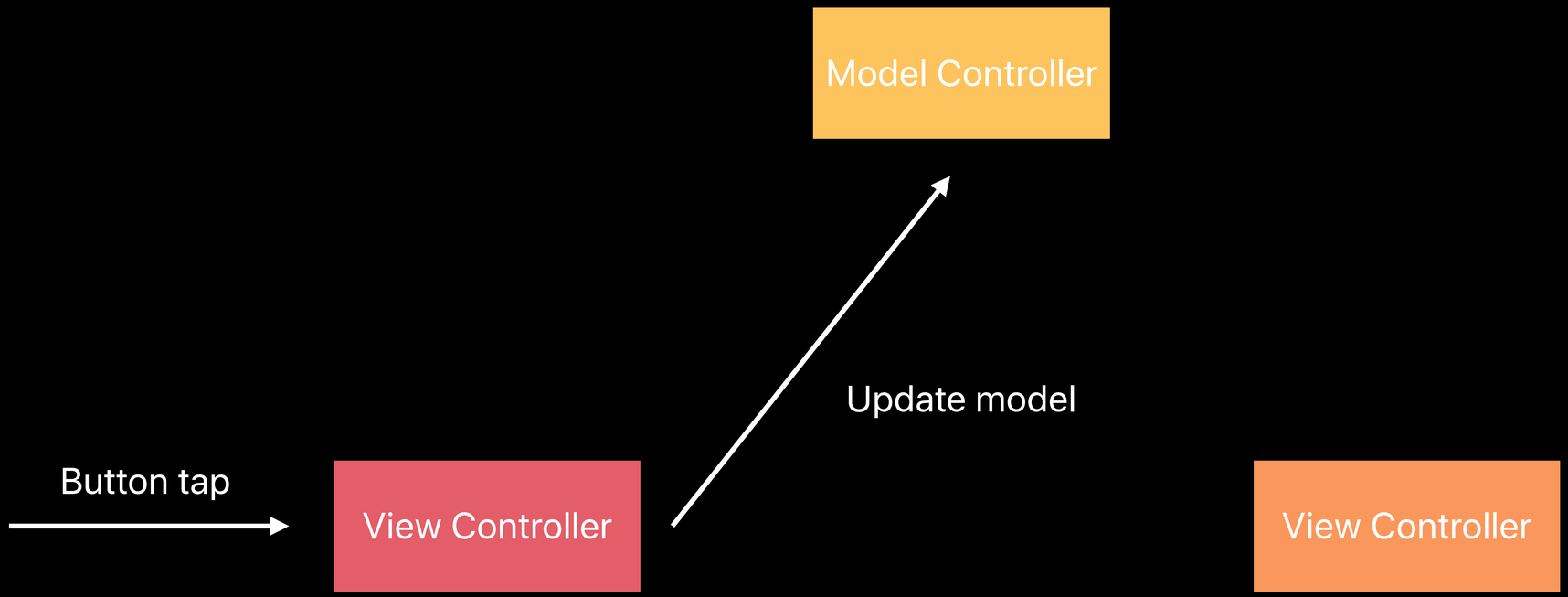
Keeping Scenes in Sync

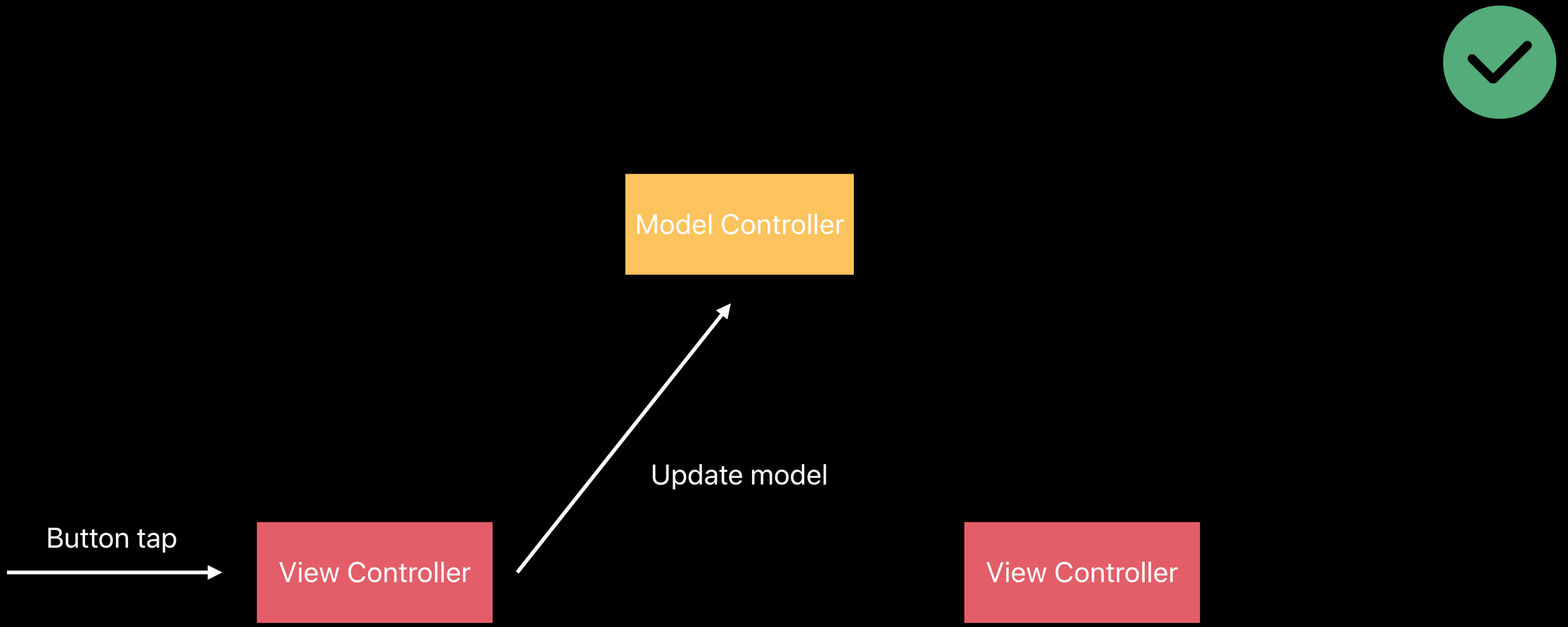

9:41 AM Tue Jan 9
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...
100%
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...

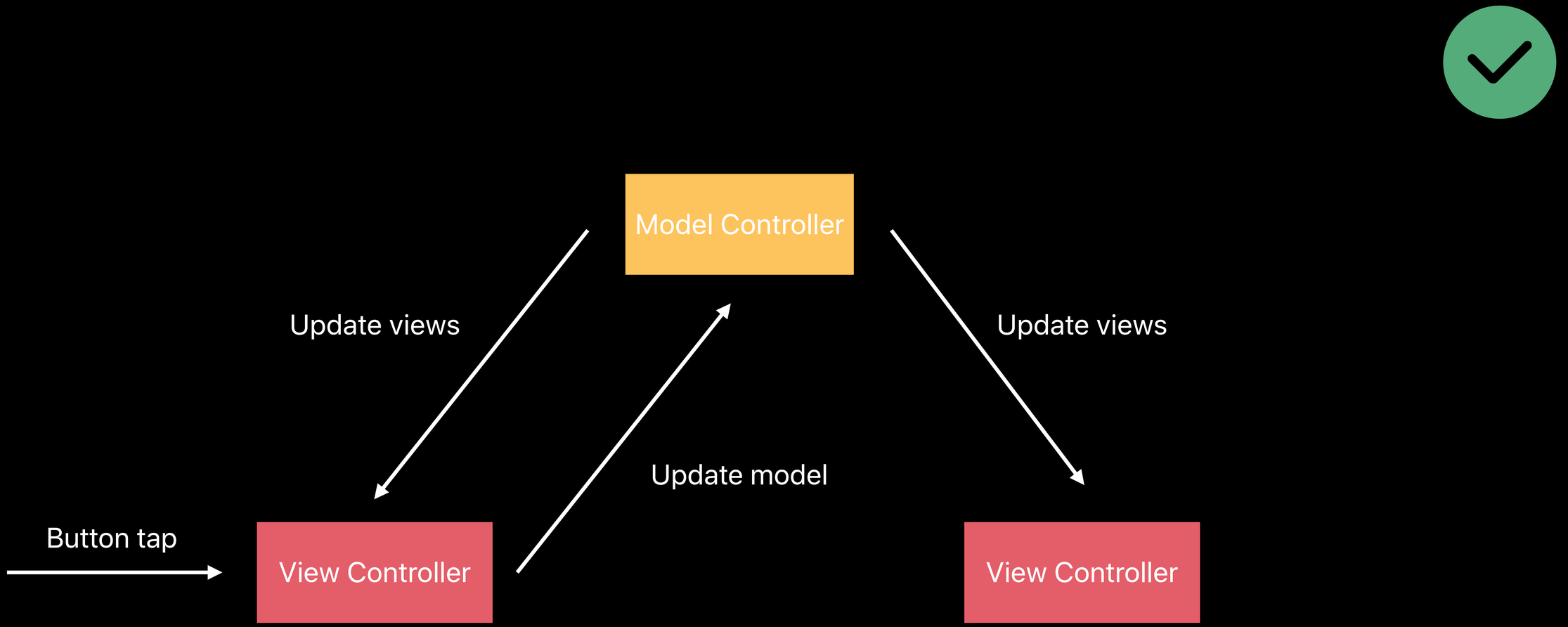

9:41 AM Tue Jan 9
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...
100%
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...


Model Controller
Update views
Update model
Button tap
View Controller


Model Controller
Update views
Update model
Old state
Button tap
View Controller
View Controller


Model Controller
Update model
Button tap
View Controller
View Controller


Model Controller
Update model
Button tap
View Controller
View Controller


Model Controller
Update views
Update views
Update model
Button tap
View Controller
View Controller

```
class ChatViewController: UIViewController {

    @objc func didEnterMessage(sender: UITextField) {

        let message = Message(text: sender.text)

        // Update views
        self.animateNewRow(for: message)
        self.updateBadgeCount()

        // Update the model
        ChatModelController.shared.add(message: message)
    }

}
```

```
class ChatViewController: UIViewController {

    @objc func didEnterMessage(sender: UITextField) {

        let message = Message(text: sender.text)

        // Update the model
        ChatModelController.shared.add(message: message)
    }

}
```

```
class ChatModelController {

    static let shared = ChatModelController()

    func add(message: Message) {
        saveToDisk(message)
    }

}
```

How will we send the update down?

```
enum UpdateEvent {

}
```

```
enum UpdateEvent {
   case NewMessage(message: Message)
}
```

```
enum UpdateEvent {
   case NewMessage(message: Message)

   static let NewMessageNotificationName = Notification.Name(rawValue: "NewMessage")

   func post() {
      // Notify subscribers
   }

}
```

```swift
enum UpdateEvent {
    case NewMessage(message: Message)

    static let NewMessageNotificationName = Notification.Name(rawValue: "NewMessage")

    func post() {
        // Notify subscribers
        switch self {
        case .NewMessage(message: _): NotificationCenter.default.post(
                name: UpdateEvent.NewMessageNotificationName,
                object: self
            )
        }
    }

}
```

```
class ChatModelController {

    static let shared = ChatModelController()

    func add(message: Message) {
        saveToDisk(message)

        let event = UpdateEvent.NewMessage(message: message)
        event.post()
    }

}
```

```
class ChatModelController {

    static let shared = ChatModelController()

    func add(message: Message) {
        saveToDisk(message)

        let event = UpdateEvent.NewMessage(message: message)
        event.post()
    }

}
```

```
class ChatViewController: UIViewController {

   override func viewDidLoad() {
      NotificationCenter.default.addObserver(selector: ..., name: .NewMessageNotificationName)
   }

}
```

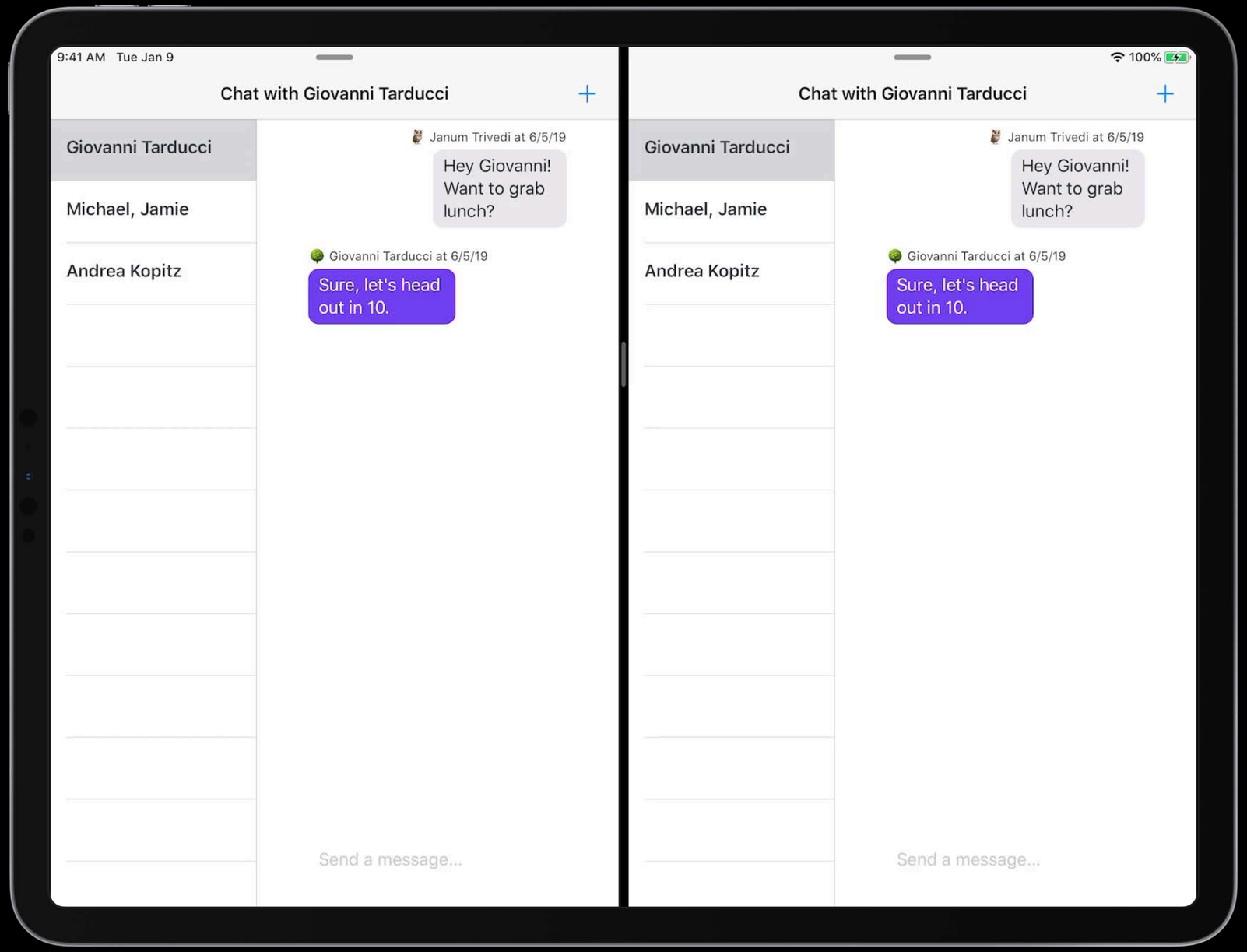
```
class ChatViewController: UIViewController {

    override func viewDidLoad() {
        NotificationCenter.default.addObserver(selector: ..., name: .NewMessageNotificationName)
    }

}
```

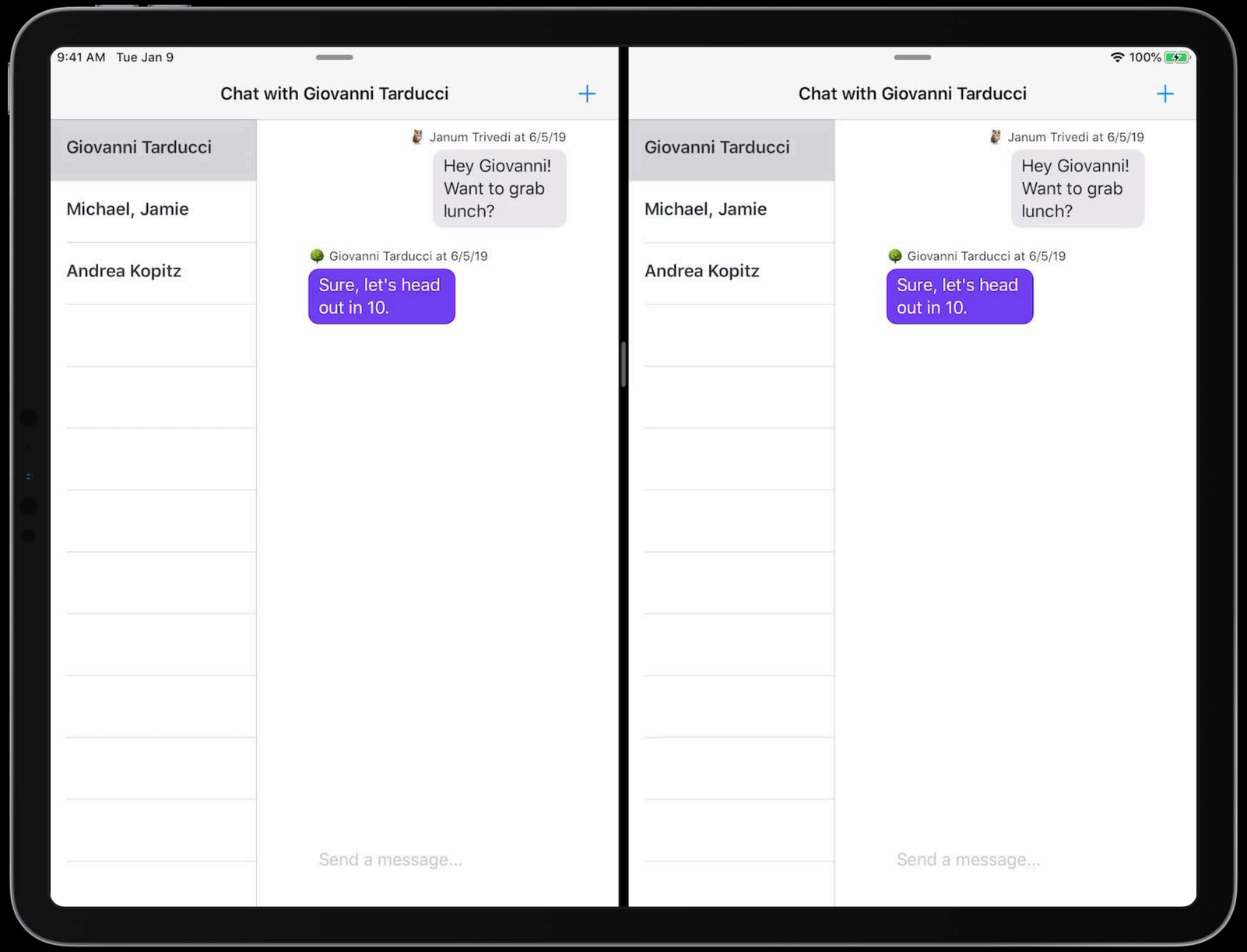
```
class ChatViewController: UIViewController {

    override func viewDidLoad() {
        NotificationCenter.default.addObserver(selector: ..., name: .NewMessageNotificationName)
    }

    @objc func handle(notification: Notification) {
        let event = notification.object as! UpdateEvent

        switch event {
        case .NewMessage(message: newMessage):
            // Update the UI
            self.animateNewRow(for: newMessage)
            self.updateBadgeCount()
        }
    }

`
```

```
class ChatViewController: UIViewController {

    override func viewDidLoad() {
        NotificationCenter.default.addObserver(selector: ..., name: .NewMessageNotificationName)
    }

    @objc func handle(notification: Notification) {
        let event = notification.object as! UpdateEvent

        switch event {
        case .NewMessage(message: newMessage):
            // Update the UI
            self.animateNewRow(for: newMessage)
            self.updateBadgeCount()
        }
    }

`
```

```
class ChatViewController: UIViewController {

    override func viewDidLoad() {
        NotificationCenter.default.addObserver(selector: ..., name: .NewMessageNotificationName)
    }

    @objc func handle(notification: Notification) {
        let event = notification.object as! UpdateEvent

        switch event {
        case .NewMessage(message: newMessage):
            // Update the UI
            self.animateNewRow(for: newMessage)
            self.updateBadgeCount()
        }
    }

`
```

```
class ChatViewController: UIViewController {

    override func viewDidLoad() {
        NotificationCenter.default.addObserver(selector: ..., name: .NewMessageNotificationName)
    }

    @objc func handle(notification: Notification) {
        let event = notification.object as! UpdateEvent

        switch event {
        case .NewMessage(message: newMessage):
            // Update the UI
            self.animateNewRow(for: newMessage)
            self.updateBadgeCount()
        }
    }

`
```


9:41 AM Tue Jan 9
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...
100%
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...


9:41 AM Tue Jan 9
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...
100%
Chat with Giovanni Tarducci
Giovanni Tarducci
Michael, Jamie
Andrea Kopitz
Janum Trivedi at 6/5/19
Hey Giovanni! Want to grab lunch?
Giovanni Tarducci at 6/5/19
Sure, let's head out in 10.
Send a message...

Review

App vs. scene delegate

State restoration

Keeping scenes in sync

More Information

developer.apple.com/wwdc19/258


WWDC19