iOS-101

iOS-Incubator @nfactorial @abylay

El Plan

- SwiftUI Layout Basics // Let’s dive deeper into the SwiftUI Layout
- State // What is State?
- Binding // What is Binding?
- Observable Object // What is ObservableObject?
- Navigation // How navigation in SwiftUI works
- Gesture Recognizers // Learn how to use gesture recognizers
- User Defaults // What is UserDefaults

-> State

- `State` is used to represent the internal state of a SwiftUI view
- It automatically make a view update when that state was changed
- SwiftUI manages the property’s storage. When the value changes, SwiftUI updates the parts of the view hierarchy that depend on the value
- Therefore most often a good idea to keep - properties wrapped properties private
- Use state as the single source of truth for a given value type that you store in a view hierarchy

- Example: Ask student to show red/blue reading what was written on paper, we can’t change the value on the paper, but if we will give him a smartphone where color is written, we can remotely update the value and update the state of the flag shown / Radio Example

-> Let’s code

-> Binding

- Use a binding to create a two-way connection between a property that stores data, and a view that displays and changes the data.
- A *binding* connects a property to a source of truth stored elsewhere, instead of storing data directly.
- For example, a button that toggles between play and pause can create a binding to a property of its parent view using the *Binding* property wrapper.
- The parent view declares a property to hold the playing state, using the *State* property wrapper to indicate that this property is the value's source of truth.
- Example: Radio when subscriber asks to change the song, song is changed for everyone.

-> Let’s code

-> ObservableObject

- A type of object with a publisher that emits before the object has changed.
- By default an ObservableObject synthesizes an objectWillChange publisher that emits the changed value before any of its @Published properties changes.

- Works similar as *@State*, main difference is that we can use ObservableObject between some independent views
- Views subscribe for the changes of ObservableObject and when changes arrive it updates itself corresponding for the data.
- Example:

-> Let’s code

-> GestureRecognizers

- OnTapGesture - Adds an action to perform when this view recognizes a tap gesture.
- Use this method to perform the specified action when the user clicks or taps on the view or container count times.

- SwiftUI manages the property’s storage. When the value changes, SwiftUI updates the parts of the view hierarchy that depend on the value
- It can be added to any view

-> Let’s code

-> Navigation

- NavigationStackView is used for navigating through your application, between the screens
- Use a navigation stack to present a stack of views over a root view
- People can add views to the top of the stack by clicking or tapping a NavigationLink
- Remove views using built-in, platform-appropriate controls, like a Back button or a swipe gesture
- The stack always displays the most recently added view that hasn’t been removed, and doesn’t allow the root view to be removed.
- We will not review NavigationSplitView in this lecture, cause it’s used for desgning navigation logic for Ipad and MacOS systems

- Example: Plates on each other

-> Let’s code

-> UserDefaults

- The UserDefaults class provides a programmatic interface for interacting with the defaults system.
- The defaults system allows an app to customize its behavior to match a user’s preferences. For example, you can allow users to specify their preferred units of measurement or media playback speed
- Apps store these preferences by assigning values to a set of parameters in a user’s defaults database

-> Let’s code

-> Swift Basics

- Closures
- **Completion**
- **Initializers**
- **Enum**
- **Collections**
- **Example: Ask student to show red/blue reading what was written on paper, we can’t change the value on the paper, but if we will give him a smartphone where color is written, we can remotely update the value and update the state of the flag shown**