iOS Bootcamp - Meeting 4

Hosted by App Team Carolina

Agenda

What can you expect this meeting?

1. GeoGuesser
2. @State Review
3. @Binding Review
4. NavigationStack
5. Project time!

Attendance!

Please fill this out!

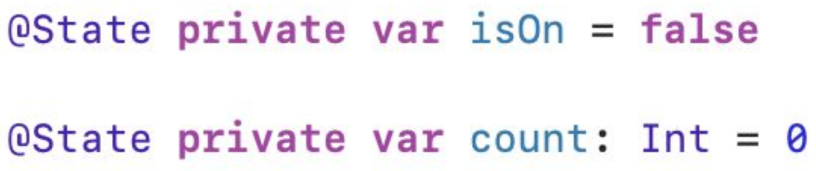
@State Review

@State

Review

@State performs two very important jobs:

- Stores the data somewhere safe
- **Updates the UI when its written to**



```
@State private var isOn = false

@State private var count: Int = 0
```

When an @State var changes, SwiftUI

redraws the View with the new value.

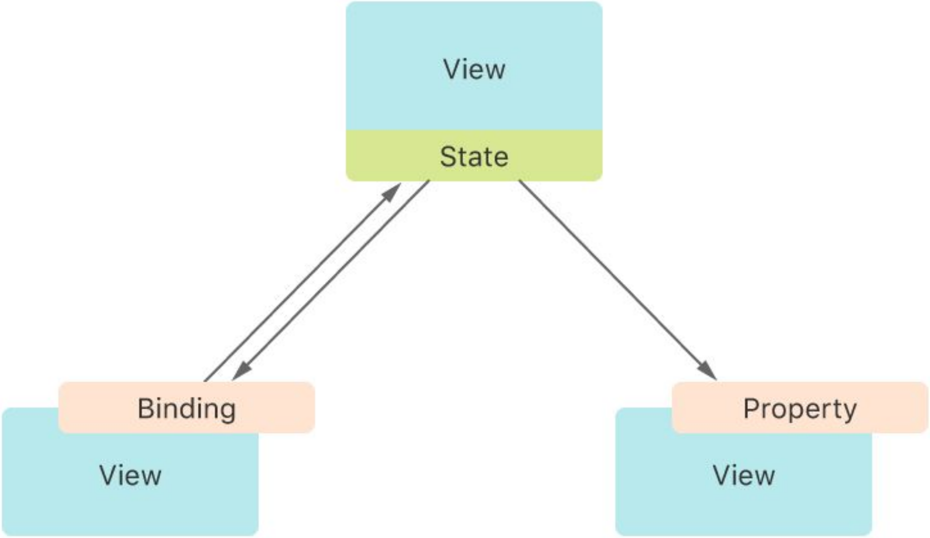
Always make @State variables private. This will prevent obscure issues!

@Binding Review

@Binding

Recall

@Binding is a property wrapper used for taking in @State variables that establishes a read **and write** relationship with that variable.



If a view wants to modify an @State variable it does not own, @Binding **must** be used

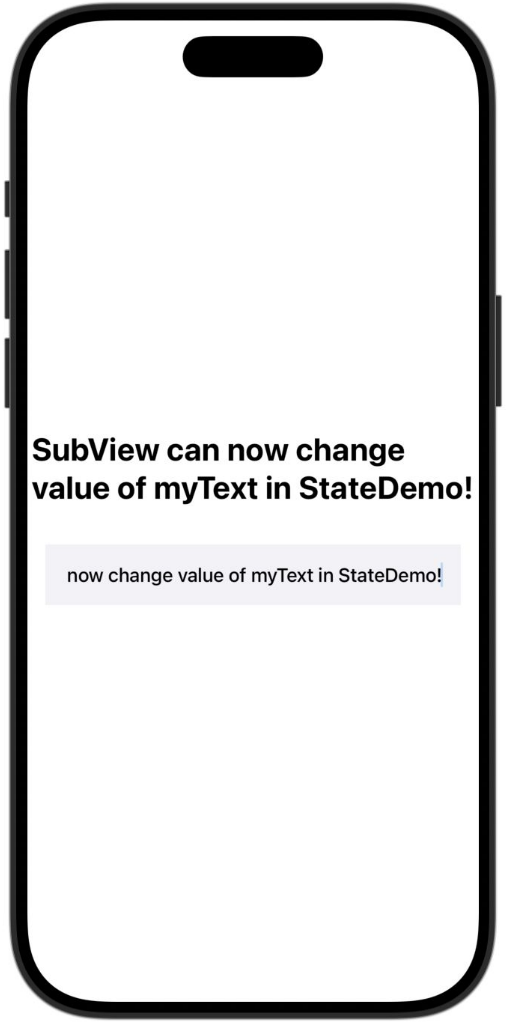
@Binding

Trivial Example

```
struct StateDemo: View {
    @State private var myText: String = ""
    var body: some View {
            Text(myText)
            BindingSubView(someText: $myText)
    }
}
```

```
struct BindingSubView: View {
    @Binding var someText: String

    var body: some View {
        TextField("My text Field", text: $someText)
    }
}
```



SubView can now change value of myText in StateDemo!

now change value of myText in StateDemo!

@Binding

Areas of Confusion

Despite looking similar to @State, the two **cannot** be used interchangeably! While @State owns the data, @Binding lets a view borrow the data– allowing read **and write** privilege.

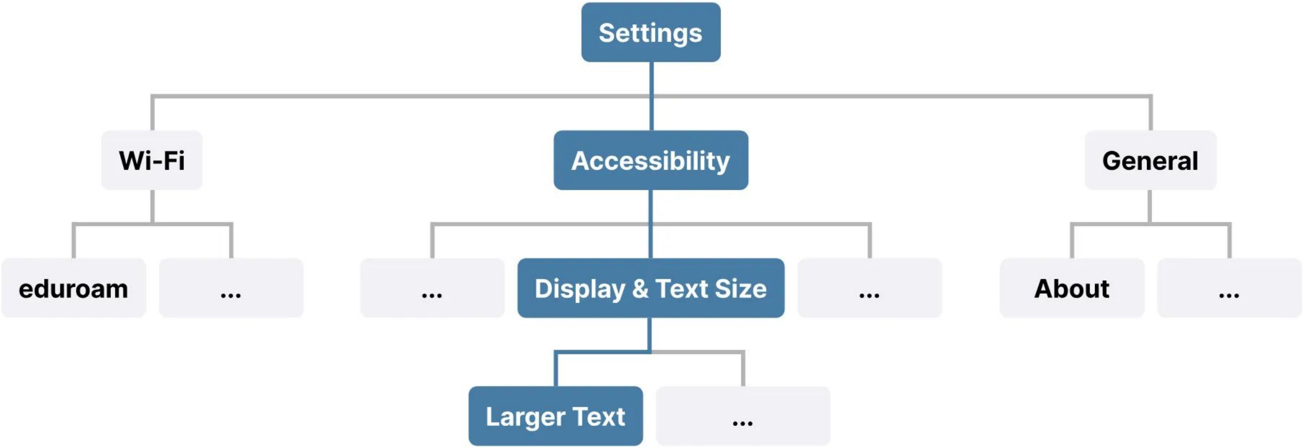
Without @Binding, a view can only read a copy of the @State variable.

NavigationStack

Navigation

Intro

The apps you create will require managing and displaying more information as they increase in complexity. How do you manage traversing between dozens of views?



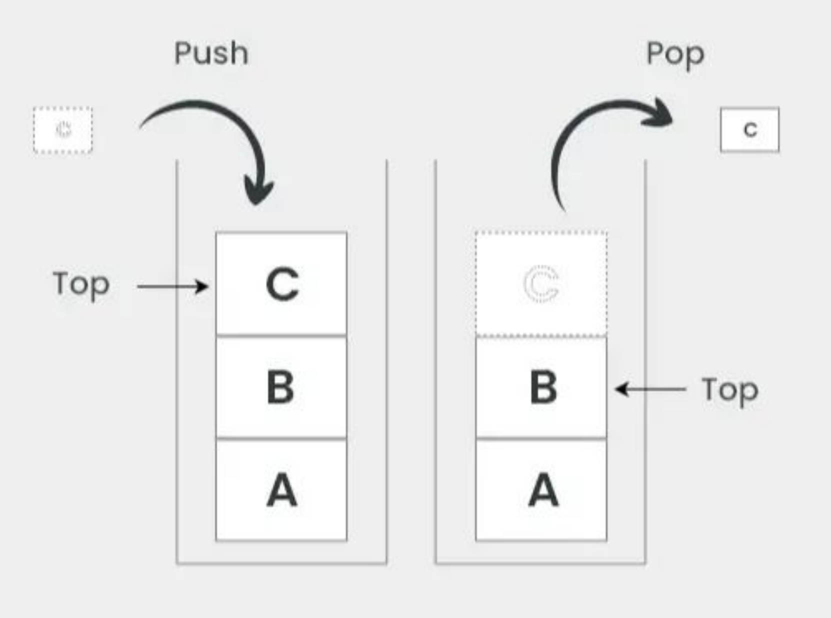
NavigationStack

Intro

In SwiftUI, **NavigationStack** is a container view that allows navigation between views via a (LIFO) stack data structure.

In a NavigationStack, views are:

- **Pushed:** Added to top of stack (the current screen)
- **Popped:** Removed from top of stack



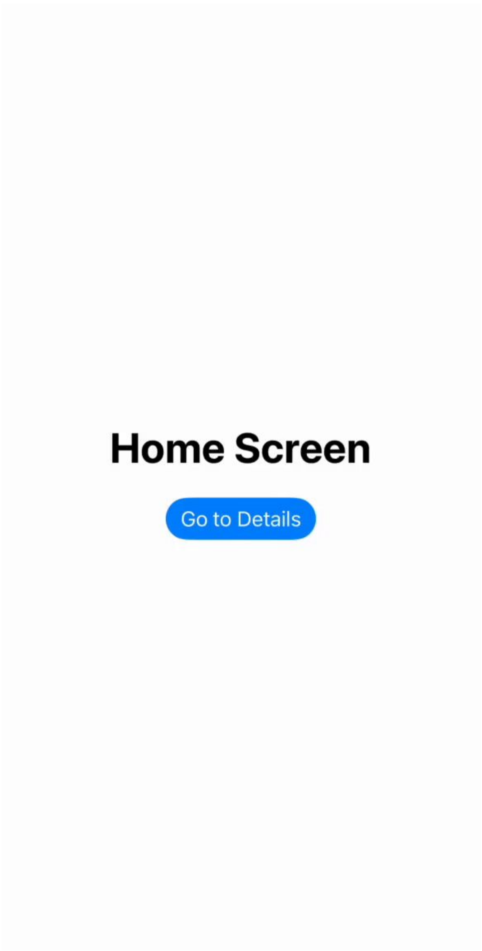
Stack diagram from **GeeksforGeeks**

NavigationStack

Example

```swift
struct NavView: View {
    var body: some View {
        NavigationStack {
            VStack(spacing: 20) {
                Text("Home Screen")
                    .font(.largeTitle)
                    .bold()

                NavigationLink("Go to Details") {
                    DetailView()
                }
                .buttonStyle(.borderedProminent)
            }
        }
    }
}
```

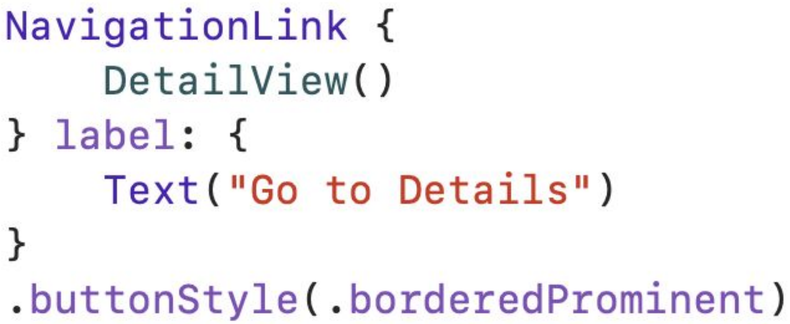


NavigationStack

NavigationLink

NavigationLink has two parameters:

1. **Destination** – the view that will be shown when the link is tapped
2. **Label** – a View that determines how the NavigationLink appears



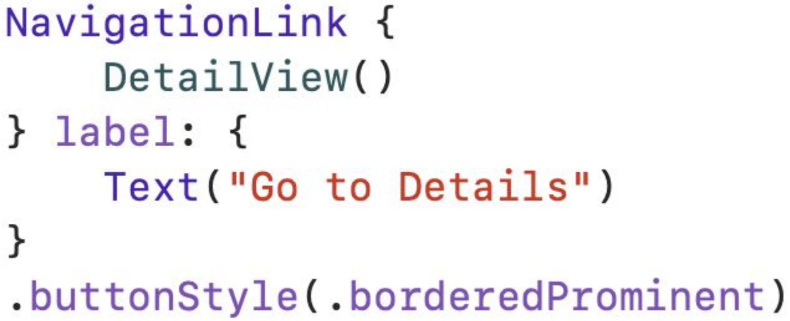
```
NavigationLink {
    DetailView()
} label: {
    Text("Go to Details")
}
.buttonStyle(.borderedProminent)
```

NavigationStack

NavigationLink

NavigationLink has two parameters:

1. **Destination** – the view that will be shown when the link is tapped
2. **Label** – a View that determines how the NavigationLink appears



```
NavigationLink {
    DetailView()
} label: {
    Text("Go to Details")
}
.buttonStyle(.borderedProminent)
```

NavigationLink only works in a **NavigationStack**!

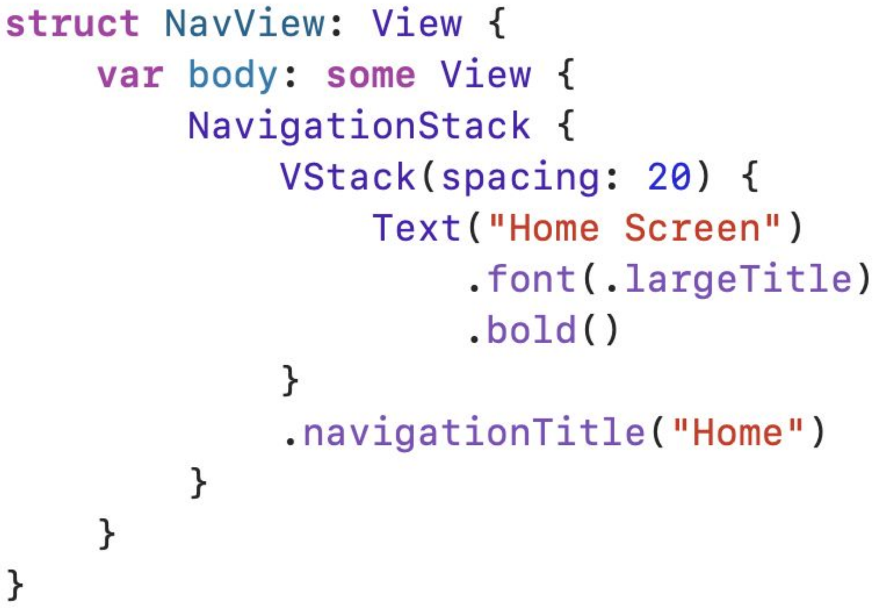
NavigationStack

NavigationTitle

NavigationTitle has one parameters:

1. **Title** – The title text that will appear in the navigation bar

Modifiers on NavigationStack are applied ***within*** the NavigationStack itself!



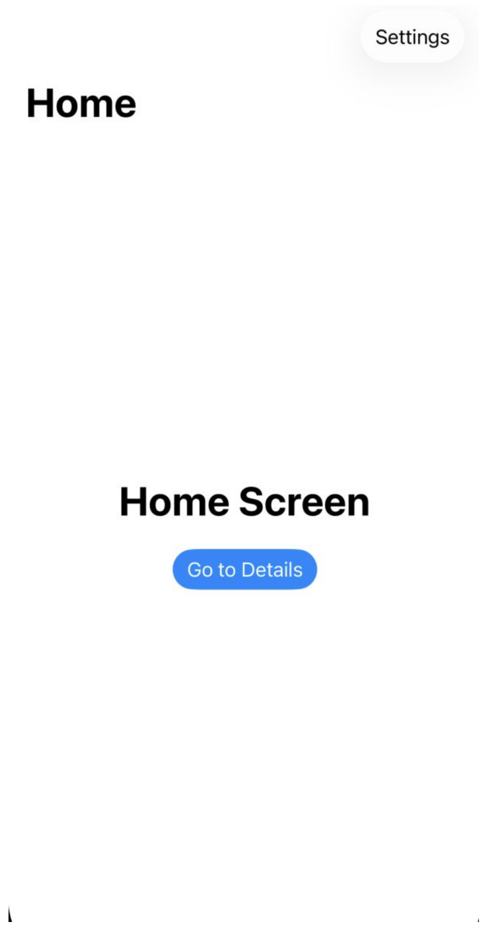
```
struct NavView: View {
    var body: some View {
        NavigationStack {
            VStack(spacing: 20) {
                Text("Home Screen")
                    .font(.largeTitle)
                    .bold()
            }
            .navigationTitle("Home")
        }
    }
}
```

NavigationStack

Toolbar

```
struct NavView: View {
    var body: some View {
        NavigationStack {
            VStack(spacing: 20) {
                Text("Home Screen")
                    .font(.largeTitle)
                    .bold()

                NavigationLink {
                    DetailView()
                } label: {
                    Text("Go to Details")
                }
                .buttonStyle(.borderedProminent)
            }
            .navigationTitle("Home")
           .toolbar {
               ToolbarItem(placement: .topBarTrailing) {
                   Button("Settings") {}
               }
           }
        }
    }
}
```



Practice NavigationStack

Return to Notion!

Project Time!

Work on projects you haven’t completed