SwiftUI

SwiftUI

SwiftUI is an innovative and simple way to build user interfaces using a *declarative* syntax.

- With imperative programming, we tell the program *what to do* and *how to do it*.
- With declarative programming, we tell the program what to do, *but not how*.

Declarative Programming

- does not involve control flow.
- describes what needs to happen: it presents a desired outcome called *state*, and leaves the implementation up to another program or framework.

A good example of a declarative language is HTML.

- HTML (and CSS) are used to build Web pages by providing *markup* and styling rules.
- You don't use HTML to draw pixels on a screen, or tell a browser how to render your web page. Instead, you tell it what components to put on the screen, and leave the rest to your browser.

Adding UIKit objects to SwiftUI

To use `UIView` subclasses from within `SwiftUI`, you wrap the other view in a `SwiftUI` view that conforms to the `UIViewRepresentable` protocol.

`UIViewRepresentable` is a protocol that allows us to use `UIKit` objects that have not yet been natively integrated into `SwiftUI`. It requires us to implement two methods:

- `makeUIView()`: responsible for the creation of our `UIView` as well as providing it to `SwiftUI` in a compatible some View format allowed by the `SwiftUI` protocol.
- `updateUIView()`: invoked whenever our `UIView` needs to be updated to reflect a change in the user interface (a bit like the `body` property we saw in `SwiftUI` native objects).