📒

Chapter 33/35

ForEach with Lists and Tab Views

ForEach Structure

- conforms to the `Identifiable` protocol
    - requires an `id` property that should be a unique value
    - `UUID()` generates a unique random hexadecimal string
- a view that **generates** a multitude of other views for each **identifiable element** in an **array of objects**

```
EditButton() // ability to move and delete list elements
List {
    ForEach(<array>) {
        <element> in <View>
    }
    .onDelete { offset in
        <array>.remove(atOffsets: offset)
    }
    .onMove { offset, index in
        <array>.move(fromOffsets: offset,
                     toOffset: index)
    }
}
```

Tab Views

- allows the user to navigate between different child views by selecting tab items located in a tab bar
- creates separate views that are shown when you clicked on the associated tab item

```
TabView {
    <View> // changes when tab is clicked on
      .tabItem {
          Image(systemName: "car")
          Text("Crosswalks")
      }
}
```

API - Working with the Web

- sharing and reusing data from web companies services
- `JSON` - a file type consisting of key/value pairs
- `Codable`
  - protocol that allows structures to be encoded/decoded in order to represent JSONs
  - primitive data types are automatically codable

Data Persistance

Sandboxes

- help keep files secure
- individual/isolated folders accessible to each application using them