10: Pokédex II

Due: Noon, 31 October

Description

In this assignment you will add features to your Pokedex app. You will again be graded on the visual aspects of your app, along with the functionality, style, and organization of your code.

- **Home view:**
  - The home view for your app will now display rows of Pokemon 'cards.' There should be a row for each `PokemonType`. Each row should be appropriately titled. Because Pokemon have multiple types, they will appear in multiple rows.
  - Each card should *somehow* visually indicate whether a Pokemon has been captured.
  - Tapping a card should navigate to that Pokemon's detail page.
  - If any Pokemon have been captured, they should be displayed in a row at the top. The row should be appropriately titled. If no Pokemon have been captured, do not display the row at all (it would be empty anyway).
  - Rows should be scrollable to ensure all Pokemon in a row can be displayed.
  - From the home view there should be an option to view a full list of all Pokemon.
- **Pokemon list:**
  - Each row in the list should *somehow* indicate whether a Pokemon has been captured
  - The user should be able to filter Pokemon in the list by `PokemonType`, e.g., filter the list so it only displays Pokemon of type Dragon. The same mechanism for filtering should also allow no filter to be applied so that all Pokemon are again in the list.
- **Pokemon detail screen:**
  - There should be a button which allows the user to mark the Pokemon as captured/released.
    - The user should know what action they are taking (capturing/releasing) so ensure that the button displays the appropriate action based on the current status.
  - There should be sections displaying the Pokemon cards for the evolutionary predecessors (`prevEvolution`) and successors (`nextEvolution`) of the Pokemon.
    - Clicking on a Pokemon in either section should navigate to its detail page.
- **Persistence:**
  - The captured status of Pokemon should persist.

Testing

1. Include appropriate catch clauses with your decoder to identify errors.
2. Test persistence by marking some Pokemon as captured and closing/reopening the app.
3. Check each of the requirements in the Description section above.

Hints

10: Pokédex II

Due: Noon, 31 October

1. You will need `PokemonType` conforming to `CaseIterable` to use `.allCases`.
2. The `Pokemon` type should have a `captured` property and your handling of the JSON file should accommodate this.
3. Create single-responsibility views to keep your code readable and reusable. For example, your Pokemon 'card' View can be used on the 'home' View and in your detail pages.
4. To allow the user to *optionally* filter by PokemonType you will need to use a Picker with a binding to a state variable of type `PokemonType?` and include an option for None (no filtering).
5. Evolutionary predecessors and successors are represented in the Pokemon type as lists of IDs referencing other Pokemon. Displaying them as a scrollable list Pokemon of cards makes for a consistent feel to the app.
6. Be sure that your home view and all navigation destinations have a navigation bar title.

Submission

Your submission should be pushed on the master branch. Be sure to verify that your project builds and remove all cruft and compiler warnings.