WeatherAggregatorApp

Initial point of the application.

ViewModel created and attached to weatherView.

WeatherView

Holds viewModel object, and create the UI and update as per requests from viewModel

Direct interaction between them. Backward communication Is done using publisher to update UI when data updated.

DatabaseService

Holds NSPersistentContainer as a shared object which will be provided to all storage managers

WeatherStorageManager

Holds logic for add weather entity, fetch weather entity.

No direct dependency

WeatherViewModel

Holds complete business logic (delegate fetch api from server, delegate save and fetch from database, holds sort and search logic, update UI to load new content,)

No direct dependency

NetworkService

Holds logic for delegating fetching data from API and parsing it to required swift models.

No direct dependency

APIService

Holds logic for fetching data from API.

ParsingService

Holds logic for parsing data to required swift models.

No direct dependency

Kept different classes for API fetch and data parsing so that single responsibility if given to each class.

No direct object of weatherStorageManager and NetworkService is created in viewModel, instead protocol abstraction is used for the ease of Unit tests and dependency inversion for a loosely coupled code.