Here are some notes from the sources you provided regarding Live Data, View Models, and Data Binding.

**Android Jetpack: Live Data + View Model + Data Binding**

* **Section Goal:** Learn to use ViewModels and LiveData to persist data through configuration changes (like screen rotation).
* **Key Concepts:**
  + **ViewModel:**
    - A class that prepares and manages data for UI components, such as activities or fragments.
    - Handles communication between the UI component and the rest of the application.
    - Tied to the lifecycle of the activity or fragment and will be retained as long as its owner is alive (e.g., for an activity, until it is finished).
    - **Most importantly, it will not be destroyed if its owner is destroyed due to a configuration change (like screen rotation). This means it can be used to cache data so the UI doesn't need to refetch it after a change**.
    - Acts as a business logic or screen-level state holder.
    - Exposes the UI's state and encapsulates related business logic.
    - **Benefit:** Caches state and persists it through configuration changes, so the UI doesn't have to fetch data again.
  + **LiveData:**
    - An observable data holder class.
    - **Lifecycle-aware:** It only updates app component observers in an active lifecycle state (e.g., started or resumed).
    - **Benefits:**
      * Eliminates memory leaks caused by multiple callbacks.
      * Decouples the data mediator and UI to avoid crashed activities.
      * UI components can continuously monitor data and LiveData handles update tasks automatically as the lifecycle changes.
      * Even if the activity or fragment is recreated, it immediately receives the latest information from LiveData.
      * Can be used to wrap system services for sharing within the app.
  + **Data Binding:**
    - Allows you to more easily link data in the ViewModel to the UI components, reducing the amount of code needed.
* **Example: Simple Counter App**
  + The source code demonstrates these concepts using a simple counter application with the following functionality:
    - The UI displays the counter and a button to increment it.
    - The counter is stored in a ViewModel.
    - When the button is clicked, the ViewModel's counter is incremented.
    - The counter's value in the ViewModel is wrapped in LiveData so the UI will update automatically.
    - The ViewModel and its data persist through screen rotation thanks to the ViewModel's lifecycle awareness.