Android Apps with Kotlin: ViewModel and Lifecycle

MANAGING ACTIVITY STATE WITH VIEWMODEL

Jim Wilson
MOBILE SOLUTIONS DEVELOPER & ARCHITECT
@hedgehogjim blog.jwhh.com

What to Expect from This Course

Managing activity state with ViewModel

Maintaining activity state during system-initiated shutdowns

Persisting complex activity state

Subscribing to Lifecycle events

Determining Lifecycle state

What to Expect from This Module

Challenges in maintaining activity state

The role of ViewModel and related types

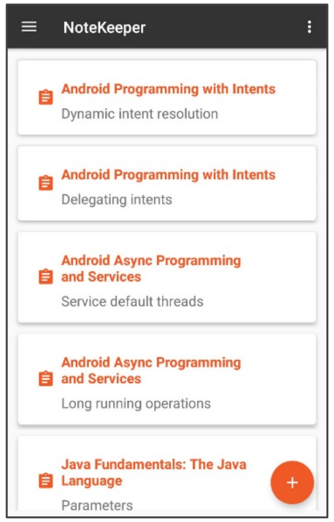
Identifying a state-related bug

Adding ViewModel and build dependencies

Accessing an activity's ViewModel

Dealing with more complex state

Activities – More Than Just a Pretty Face



NoteKeeper

Android Programming with Intents
Dynamic intent resolution

Android Programming with Intents
Delegating intents

Android Async Programming and Services
Service default threads

Android Async Programming and Services
Long running operations

Java Fundamentals: The Java Language
Parameters

App user experience provided by activities

Appear to user as simple app screens

But there's much more going on

Activities have a lifecycle

Our code needs to cooperate with that lifecycle

Life, Death, and Life of an Activity

Created

Has app-defined initial state

User Interaction

State reflects user's action

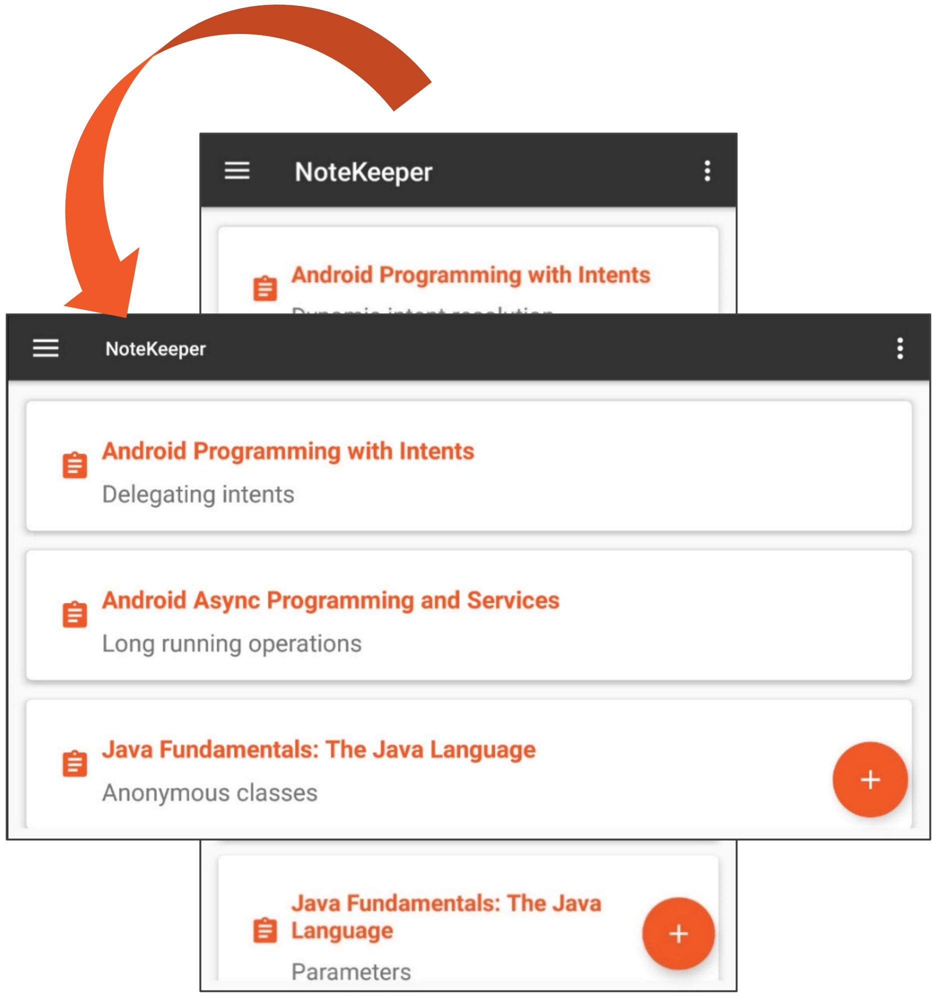
Destroyed

State stored within activity instance is lost

Recreated

Should restore previous state

Configuration Changes


NoteKeeper
Android Programming with Intents
NoteKeeper
Android Programming with Intents
Delegating intents
Android Async Programming and Services
Long running operations
Java Fundamentals: The Java Language
Anonymous classes
Java Fundamentals: The Java Language
Parameters

Managing Activity State

Maintaining activity state

- Writing to a persistent store is expensive
- Need a better solution for maintaining state across configuration changes

ViewModel

- Stores activity state in-process
- State stored separate from the activity
- Extend ViewModel class to customize
- Add properties and methods specific to your activity's state requirements

Managing Activity State

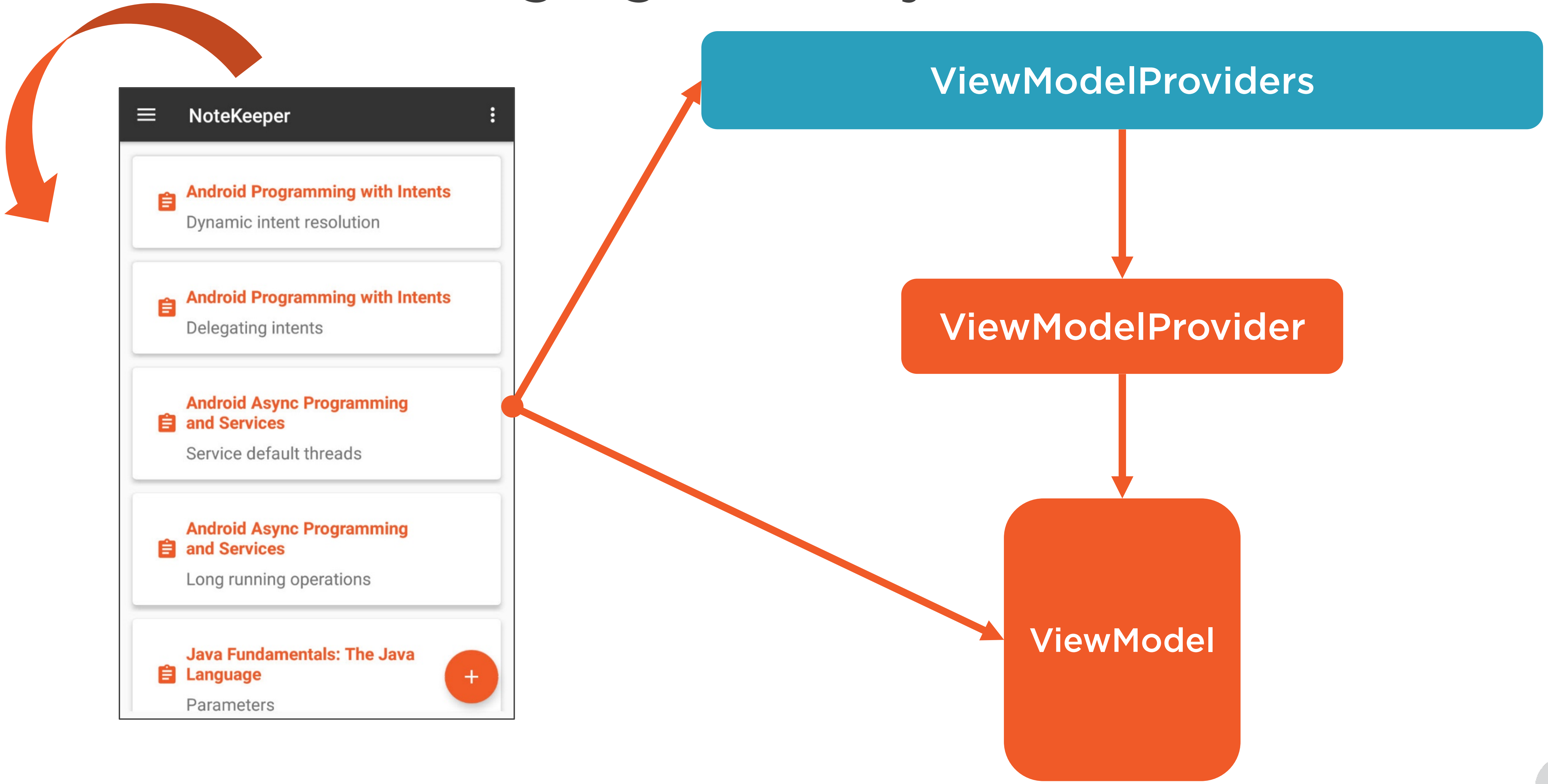
ViewModelProvider

- Manages ViewModel instances
- Creates new instance when needed
- Retrieves existing when available

ViewModelProviders

- Manages association between activities and ViewModelProvider instances

Managing Activity State



NoteKeeper

Android Programming with Intents
Dynamic intent resolution

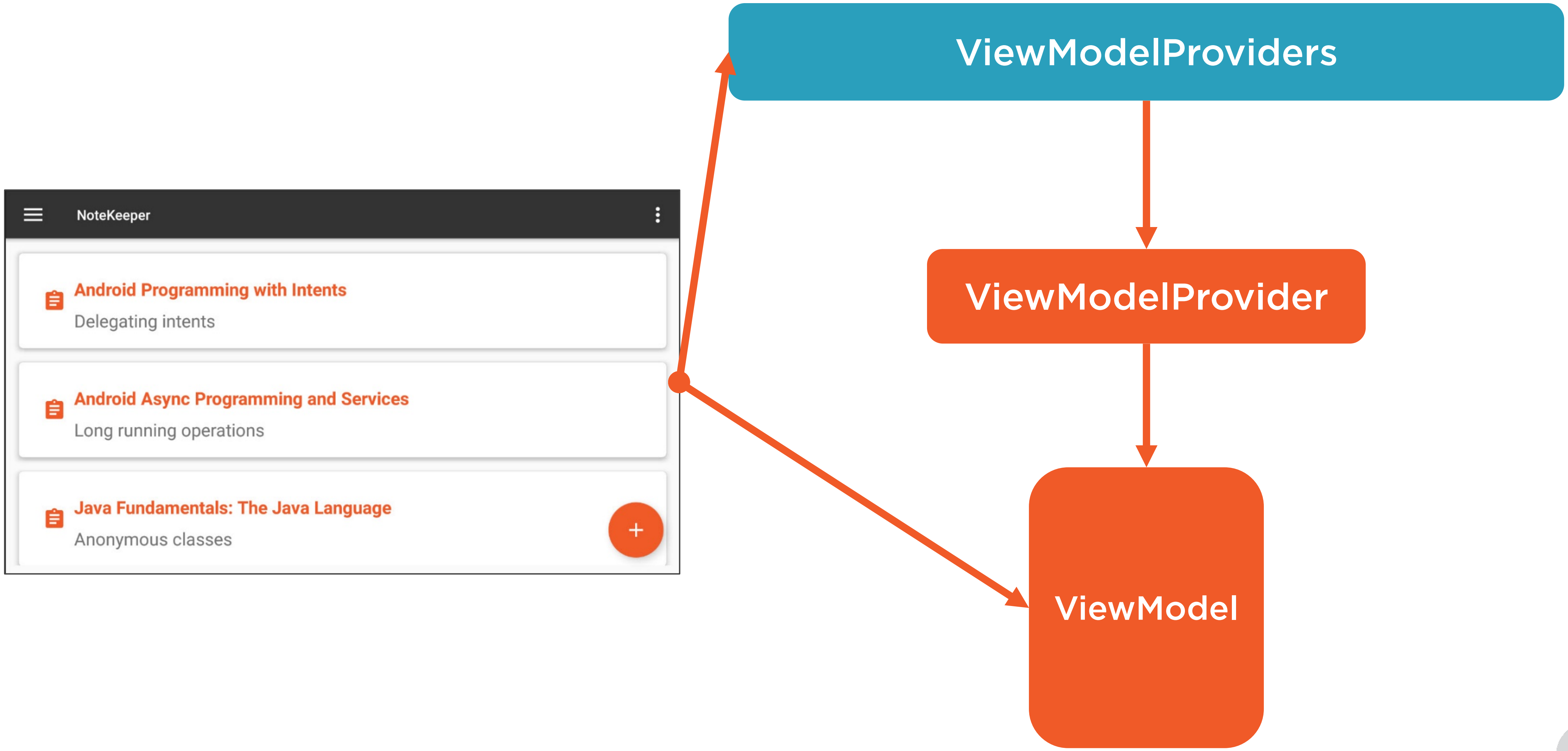
Android Programming with Intents
Delegating intents

Android Async Programming and Services
Service default threads

Android Async Programming and Services
Long running operations

Java Fundamentals: The Java Language
Parameters

Managing Activity State



Summary

Configuration changes impact activities

- System destroys and recreates
- State stored directly in activity is lost
- App responsible to provide consistent user experience

ViewModel

- Stores activity state in-process
- State stored separate from the activity
- Extend ViewModel class to customize

Summary

ViewModelProvider

- Manages ViewModel instances
- Creates new instance when needed
- Retrieves existing when available

ViewModelProviders

- Manages association between activities and ViewModelProvider instances

Summary

Benefits of using ViewModel

- Separates activity state from the activity
- Retains state across config changes
- Reduces the amount of code that’s placed directly in the activity class