droidcon academy

Course wrap-up

Piotr Prus

@piotr_prus

in/piotrprus

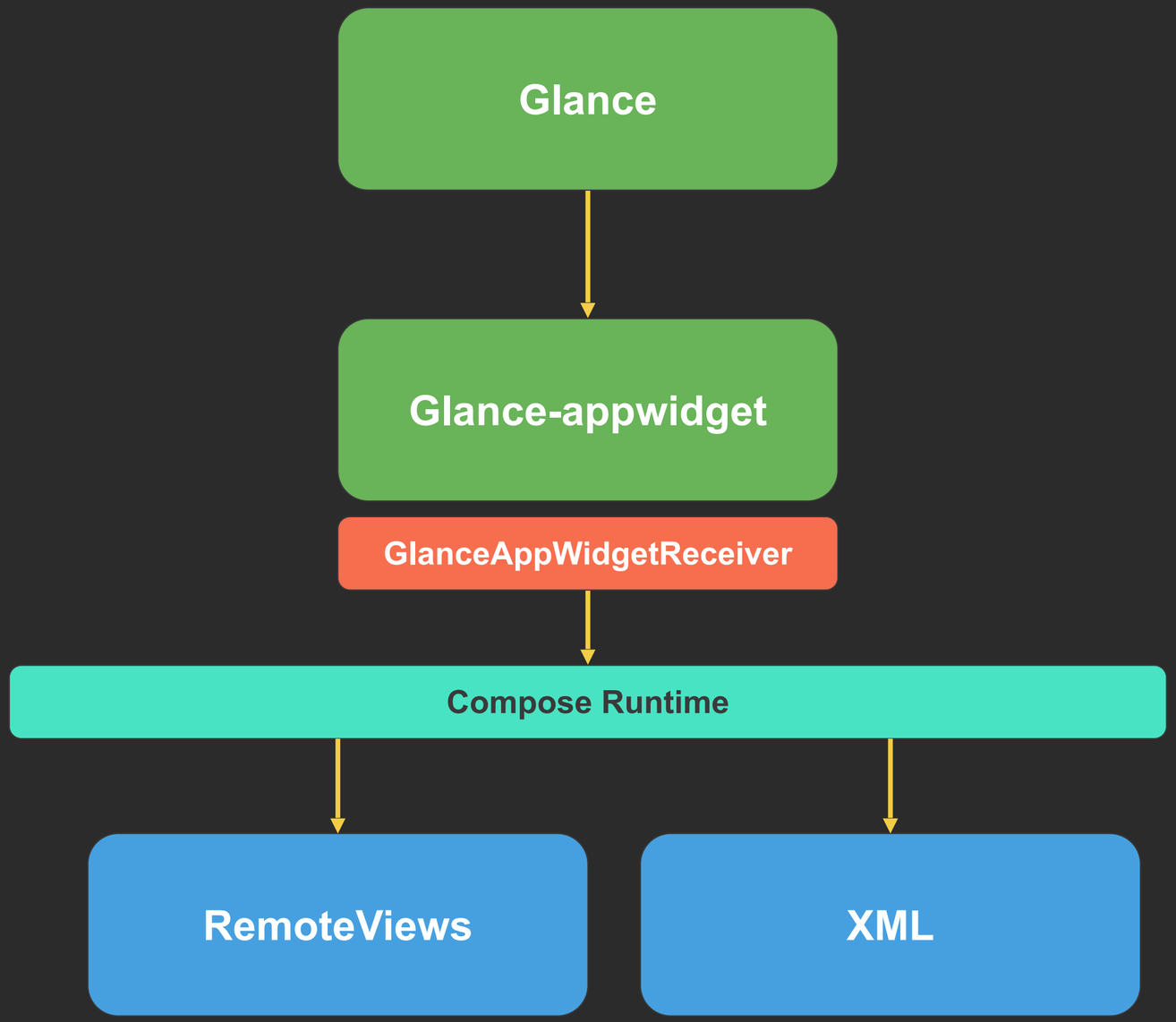
Section Overview

- Recap on the most important concepts of Jetpack Glance
- Go through the code to recap what we have build
- Bonus

droidcon academy

The most important concepts of Jetpack Glance

droidcon academy


Glance
Glance-appwidget
GlanceAppWidgetReceiver
Compose Runtime
RemoteViews
XML

widget_provider.xml

initial_widget_layout.xml

ic_refresh.xml

ic_location.xml

ic_wind.xml

WeatherWidget.kt

```
// Compose imports

import androidx.compose.foundation.layout.Row
import androidx.compose.foundation.layout.Column
import androidx.compose.foundation.layout.Spacer

// Glance imports
import androidx.glance.layout.Alignment
import androidx.glance.layout.Box
import androidx.glance.layout.Column
import androidx.glance.layout.Row
import androidx.glance.layout.Spacer
```

WeatherWidget.kt

```
override suspend fun provideGlance(context: Context, id: GlanceId) {
    provideContent {
        GlanceTheme(GlanceColorScheme.colors) {
            WidgetContent()
        }
    }
}

object GlanceColorScheme {
    val colors = ColorProviders(
        light = LightColorScheme,
        dark = DarkColorScheme
    )
}
```

GlanceId

droidcon academy

GlanceId

```
val glanceId = GlanceAppWidgetManager(context).getGlanceIds(
        WeatherWidget::class.java).last()

override suspend fun onAction(
    context: Context,
    glanceId: GlanceId,
    parameters: ActionParameters
)
```

GlanceStateDefinition

droidcon academy

GlanceStateDefinition.kt

```kotlin
/**
 * Configuration definition for [GlanceState]. This defines where the data is stored and how the
 * underlying data store is created. Use a unique [GlanceStateDefinition] to get a
[GlanceState], once
 * defined, the data should be updated using the state directly, this definition should not
change.
 */
interface GlanceStateDefinition<T>
```

Worker Manager

droidcon academy

WeatherWidgetWorker.kt

```
class WeatherWidgetWorker(
    private val repository: WeatherRepository,
    private val appContext: Context,
    private val workerParameters: WorkerParameters
) : CoroutineWorker(appContext, workerParameters) {
override suspend fun doWork(): Result {
   WidgetStateHelper.setLoading(true)
   repository.getData(latitude, longitude)
      .onSuccess { item →
          WidgetStateHelper.save(item)
           return Result.success()
      }
      .onFailure { throwable →
           WidgetStateHelper.setLoading(false)
          return Result.retry()
      }
}
```

ConfigurationActivity.kt

```
WeatherWidget().apply {
    updateAppWidgetState(context, glanceId) {
        WidgetStateHelper.saveLocation(
            it,
            latitude = item.latitude,
            longitude = item.longitude
        )
        WidgetStateHelper.saveAddress(it, address = item.name)
    }
    update(context, glanceId)
}
startWeatherWorker(latitude = item.latitude, longitude = item.longitude)
```

User Interactions

droidcon academy

actionStartActivity

actionRunCallback

actionStartService

actionSendBroadcast

droidcon academy

Open the Android Studio

droidcon academy

droidcon academy

Building a Widget using Jetpack Glance in Android

Thanks for joining

Piotr Prus

@piotr_prus

in/piotrprus