

# Understanding Activities and Activity Layout Interaction

---



**Jim Wilson**

MOBILE SOLUTIONS DEVELOPER & ARCHITECT

@hedgehogjim [blog.jwhh.com](http://blog.jwhh.com)



# What to Expect from This Module



**What is an Activity?**

**Activity UI**

**Layout Classes**

**ConstraintLayout Class**

**Activity/Layout Relationship**

**Populating a Spinner**



# What is an Activity?

An activity is a single, focused thing that the user can do.



# A Little More Detail About Activities

## **Serve as the place to present the UI**

- Provide a window
- UI is built with View-derived classes

## **Have a lifecycle**

- More than just a “screen”
- Lifecycle calls a series of methods
- Use the onCreate method to initialize the Activity



# Activity UI

## View

- Basic building block of UI
- Drawing and event handling
- Many specialized classes available

## ViewGroup

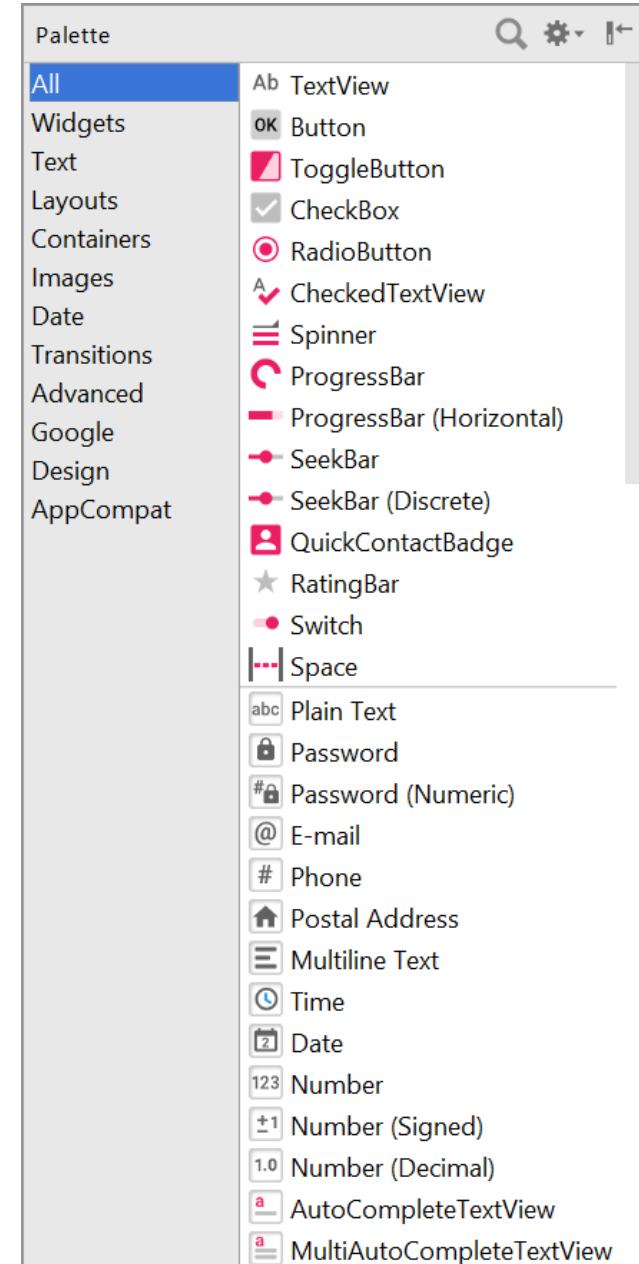
- Special View that holds other views

## Layout

- Special invisible ViewGroup
- Handle View positioning behavior
- Many specialized classes available



# Common View Classes



# Layout Classes

## **Activity UIs need to be responsive**

- Device display characteristics vary
- UI must adapt
- Absolute positioning would be limiting

## **Layout classes provide positioning flexibility**

- Arrange child Views
  - Children can include other layout classes
- Specific positioning behavior depends on the layout class



# Some Common Layout Classes

## **FrameLayout**

- Provides a blocked-out area
- Generally has only one direct child

## **ScrollView**

- Provides a scrollable area

## **LinearLayout**

- Horizontal or vertical arrangement
- Supports weighted distribution

## **RelativeLayout**

- Relative positioning
- Relative to one another or parent





# Simplifying Layout Creation

## Traditional layout classes have challenges

- UIs have become much richer

## Run-time challenges

- Sometimes have to nest layout classes
- Deep/complex nesting impacts speed

## Design-time challenges

- Achieving desired result with designer sometimes challenging
- In some cases end up working in the XML



# ConstraintLayout t Class

## **Extremely flexible layout class**

- Often the only layout class needed

## **First-class design-time experience**

- Closely integrated with designer
- Rarely need to resort to XML



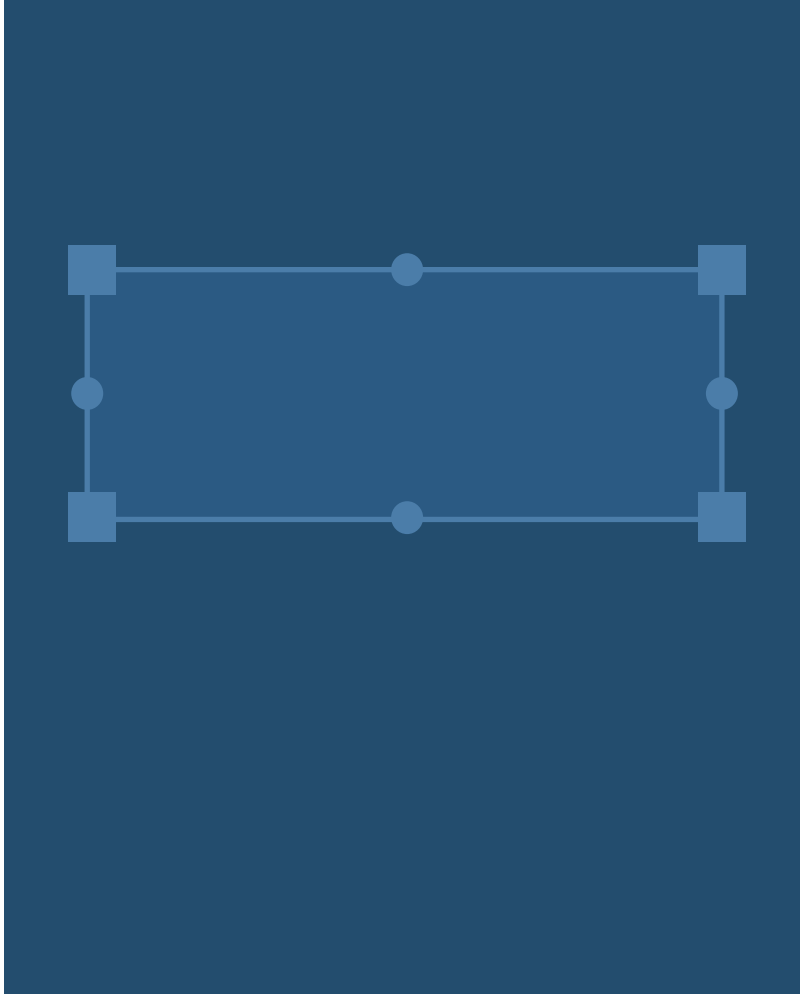
# ConstraintLayout t Class

## Children leverage constraints

- Relative size/position
- Ratio-base size/position
- Group size/position distribution
  - Known as “chains”
- Weighted relationships
- Guideline-based size/position



# ConstraintLayout



## Should set horizontal & vertical constraints

- Positions at 0,0 without constraints
- Can set more than one of each

## Setting constraints with the designer

- Drag circle at mid-line to relationship

## Setting fixed size with the designer

- Drag corner squares

# Creating the Activity UI

## Programmatically

- Use Java code to create class instances
- Relationships and properties set in code

## Layout files

- XML files describe View hierarchy
- Usually created using the Android Studio UI Designer



# Activity/Layout Relationship

## There is no implicit relationship

- Activity must load layout
  - Use setContentView
- Activity must request View reference
  - Use findViewById

## Relies on generated class R

- Contains nested classes
- Layouts names in R.layout
- Id names in R.id



```

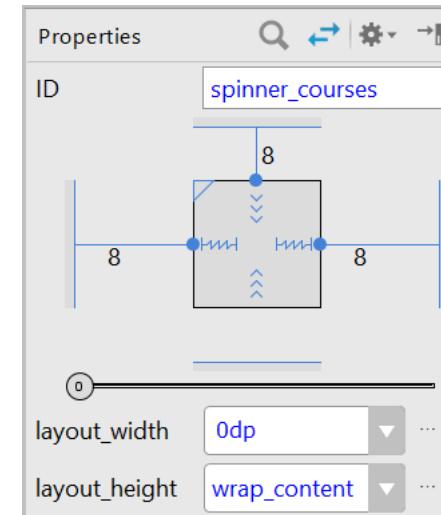
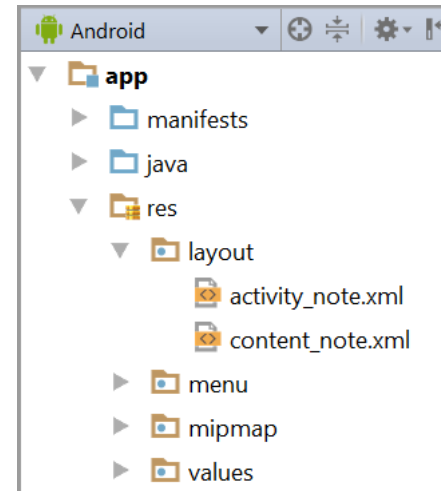
public final class R {
    public static final class layout {
        public static final
            int activity_note = ... ;

        // ...
    }

    public static final class id {
        public static final
            int spinner_courses = ... ;

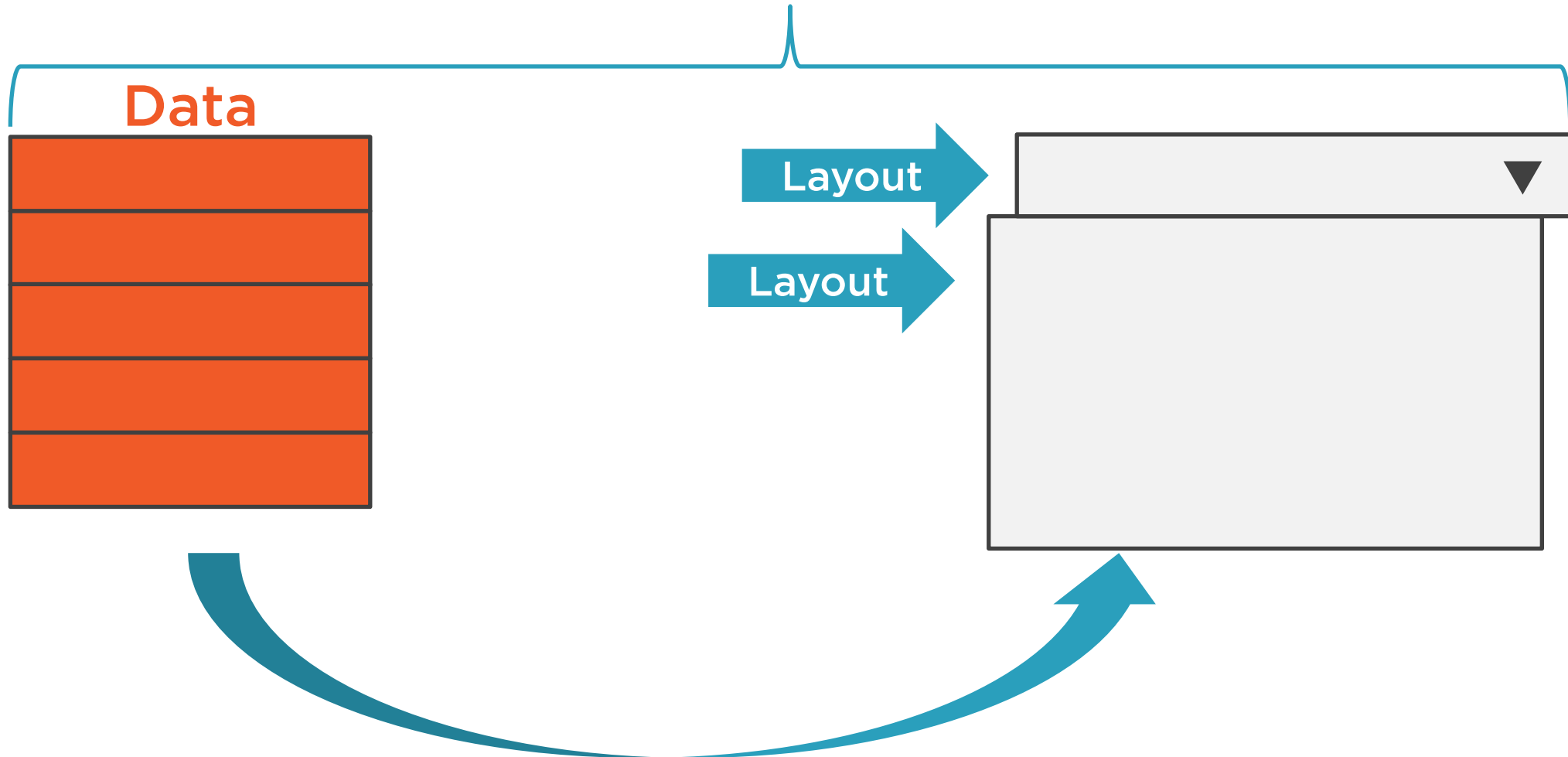
        // ...
    }
}

```



# Populating a Spinner

## Adapter





# Summary



**Activities present the UI**

**Views are the basic UI building blocks**

**Layouts**

- Handle positioning behavior
- Important for creating responsive UI

**ConstraintLayout class**

- Powerful and flexible
- Closely integrated with Android Studio Designer

