Mobile Applications (420-P84-AB)

John Abbott College

Constraint Layout

Aakash Malhotra

Constraint Layout

`ConstraintLayout` allows you to create large and complex layouts with a flat view hierarchy (no nested view groups). It's similar to `RelativeLayout` in that all views are laid out according to relationships between sibling views and the parent layout, but it's more flexible than `RelativeLayout` and easier to use with Android Studio's Layout Editor.

All the power of `ConstraintLayout` is available directly from the Layout Editor's visual tools, because the layout API and the Layout Editor were specially built for each other. So you can build your layout with `ConstraintLayout` entirely by drag-and-dropping instead of editing the XML

Constraint Layout

Constraints overview

To define a view's position in `ConstraintLayout`, you must add at least one horizontal and one vertical constraint for the view. Each constraint represents a connection or alignment to another view, the parent layout, or an invisible guideline. Each constraint defines the view's position along either the vertical or horizontal axis; so each view must have a minimum of one constraint for each axis, but often more are necessary.

When you drop a view into the Layout Editor, it stays where you leave it even if it has no constraints. However, this is only to make editing easier; if a view has no constraints when you run your layout on a device, it is drawn at position [0,0] (the top-left corner).

In figure 1, the layout looks good in the editor, but there's no vertical constraint on view C. When this layout draws on a device, view C horizontally aligns with the left and right edges of view A, but appears at the top of the screen because it has no vertical constraint.

Constraints overview

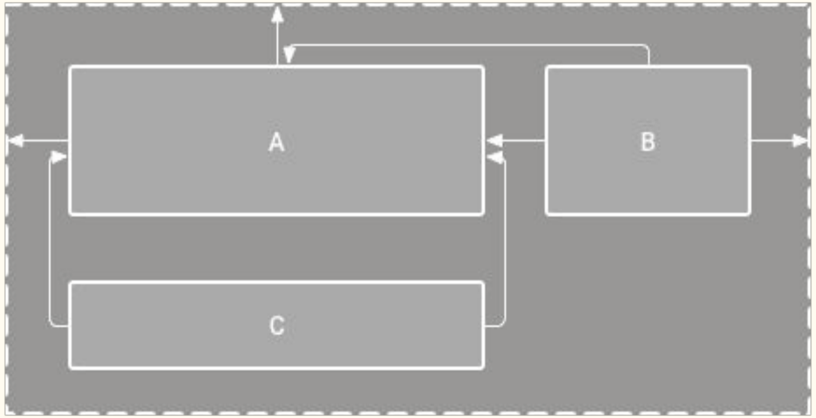


Figure 1. The editor shows view C below A, but it has no vertical constraint

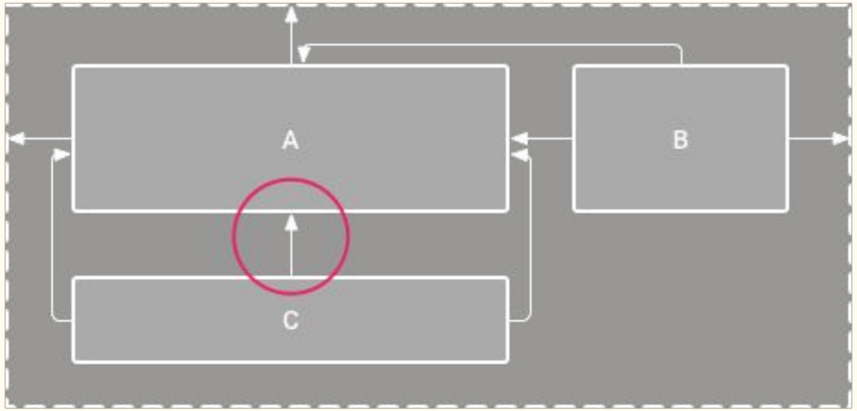


Figure 2. View C is now vertically constrained below view A

Constraints overview

Although a missing constraint won't cause a compilation error, the Layout Editor indicates missing constraints as an error in the toolbar. To view the errors and other warnings, click **Show Warnings and Errors** . To help you avoid missing constraints, the Layout Editor can automatically add constraints for you with the Autoconnect and infer constraints features.

Convert a layout

To convert an existing layout to a constraint layout, follow these steps:

1. Open your layout in Android Studio and click the **Design** tab at the bottom of the editor window.
2. In the **Component Tree** window, right-click the layout and click **Convert *layout* to ConstraintLayout**.

Create a new layout

To start a new constraint layout file, follow these steps:

1. In the **Project** window, click the module folder and then select **File > New > XML > Layout XML**.
2. Enter a name for the layout file and enter "android.support.constraint.ConstraintLayout" for the **Root Tag**.
3. Click **Finish**.

Add or remove a constraint

To add a constraint, do the following:

1. Drag a view from the **Palette** window into the editor.
2. When you add a view in a `ConstraintLayout`, it displays a bounding box with square resizing handles on each corner and circular constraint handles on each side.
3. Click the view to select it.
4. Do one of the following:
    - Click a constraint handle and drag it to an available anchor point (the edge of another view, the edge of the layout, or a guideline).
    - Click **Create a connection** in the view inspector at the top of the **Attributes** window.

When the constraint is created, the editor gives it a default margin to separate the two views.

Add or remove a constraint

When the constraint is created, the editor gives it a default margin to separate the two views.

When creating constraints, remember the following rules:

- Every view must have at least two constraints: one horizontal and one vertical.
- You can create constraints only between a constraint handle and an anchor point that share the same plane. So a vertical plane (the left and right sides) of a view can be constrained only to another vertical plane; and baselines can constrain only to other baselines.
- Each constraint handle can be used for just one constraint, but you can create multiple constraints (from different views) to the same anchor point.

Add or remove a constraint

To remove a constraint, select the view and then click the constraint handle. Or remove all the constraints by selecting the view and then clicking **Delete Constraints** .

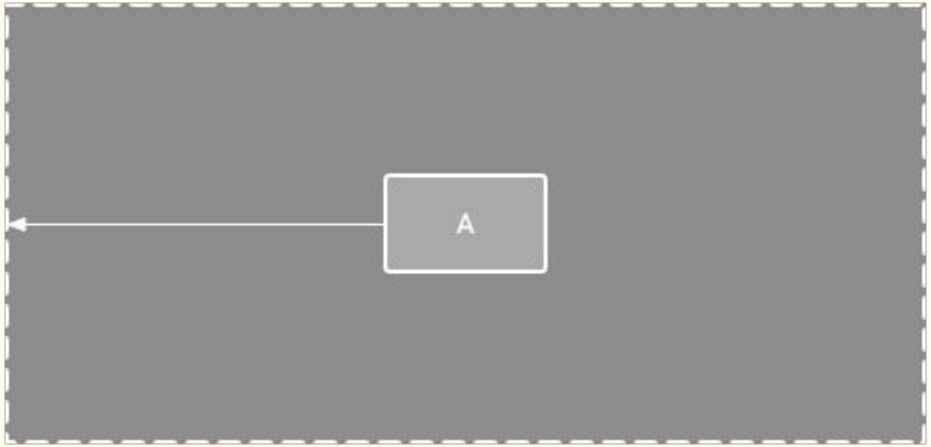
If you add opposing constraints on a view, the constraint lines become squiggly like a spring to indicate the opposing forces. The effect is most visible when the view size is set to "fixed" or "wrap content," in which case the view is centered between the constraints. If you instead want the view to stretch its size to meet the constraints, switch the size to "match constraints"; or if you want to keep the current size but move the view so that it is not centered, adjust the constraint bias.

You can use constraints to achieve different types of layout behavior.

Parent position

Constrain the side of a view to the corresponding edge of the layout.

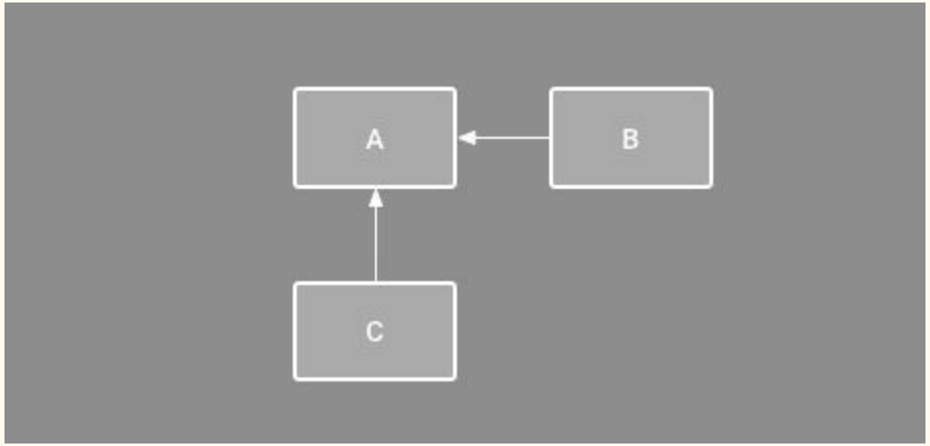
In figure, the left side of the view is connected to the left edge of the parent layout. You can define the distance from the edge with margin.



Order Position

Define the order of appearance for two views, either vertically or horizontally.

In figure 5, B is constrained to always be to the right of A, and C is constrained below A. However, these constraints do not imply alignment, so B can still move up and down.



Alignment

Align the edge of a view to the same edge of another view.

In figure, the left side of B is aligned to the left side of A. If you want to align the view centers, create a constraint on both sides.

You can offset the alignment by dragging the view inward from the constraint. For example, figure 7 shows B with a 24dp offset alignment. The offset is defined by the constrained view's margin.

You can also select all the views you want to align, and then click **Align** in the toolbar to select the alignment type.

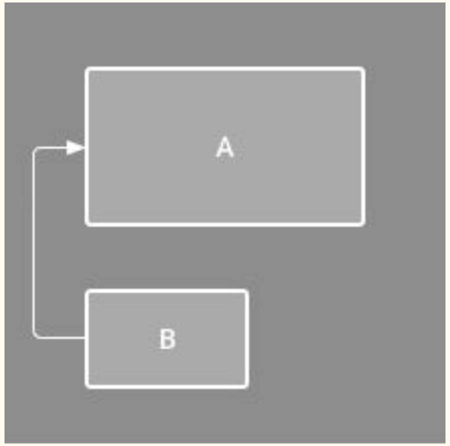


Figure 6. A horizontal alignment constraint

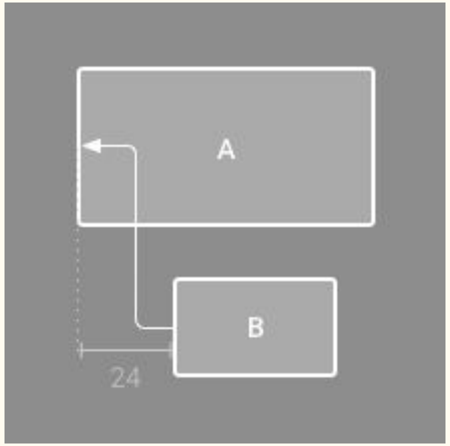


Figure 7. An offset horizontal alignment constraint

Baseline alignment

Align the text baseline of a view to the text baseline of another view.

In figure 8, the first line of B is aligned with the text in A.

To create a baseline constraint, select the text view you want to constrain and then click **Edit Baseline**, which appears below the view. Then click the text baseline and drag the line to another baseline.

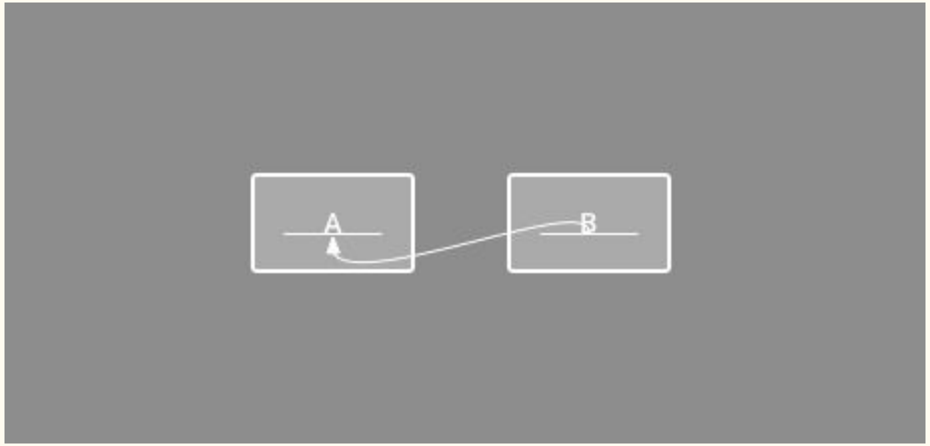


Figure 8. A baseline alignment constraint

References:

- https://developer.android.com/training/constraint-layout/