Mastering RecyclerView Layouts

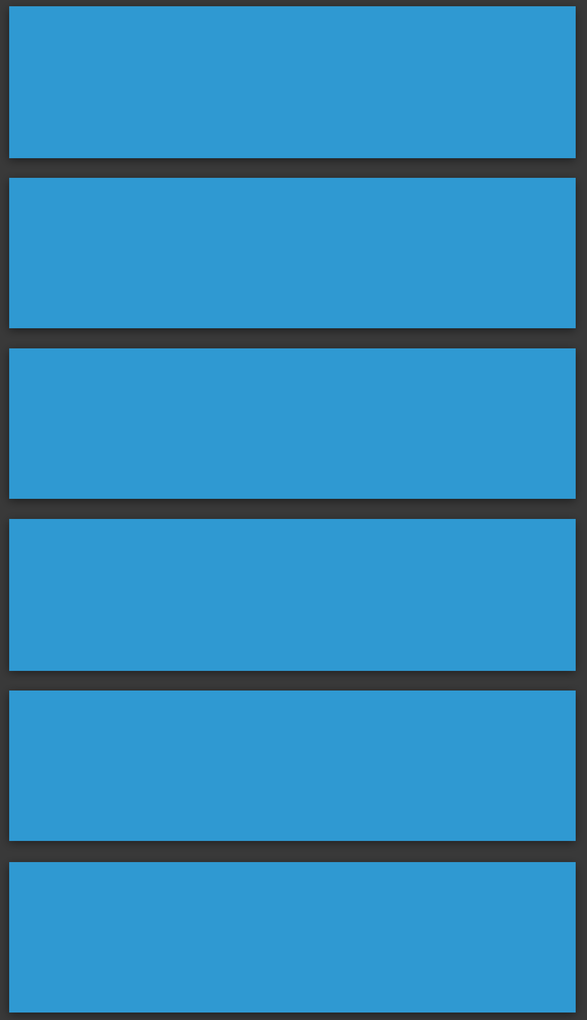
Dave Smith • NewCircle, Inc.
@devunwired
+DaveSmithDev

Why Am I Here?

Help Reduce Sanitarium Admissions caused by RecyclerView Layouts.

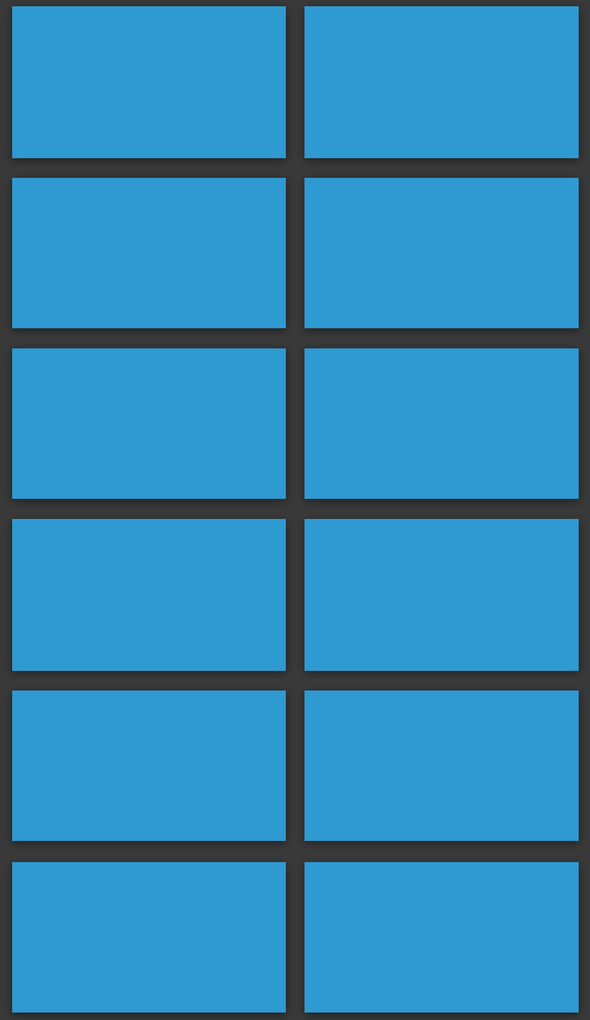
```
RecyclerView recyclerView = new RecyclerView(this);
recyclerView.setLayoutManager(…);

setContentView(recyclerView);
```

LinearLayoutManager



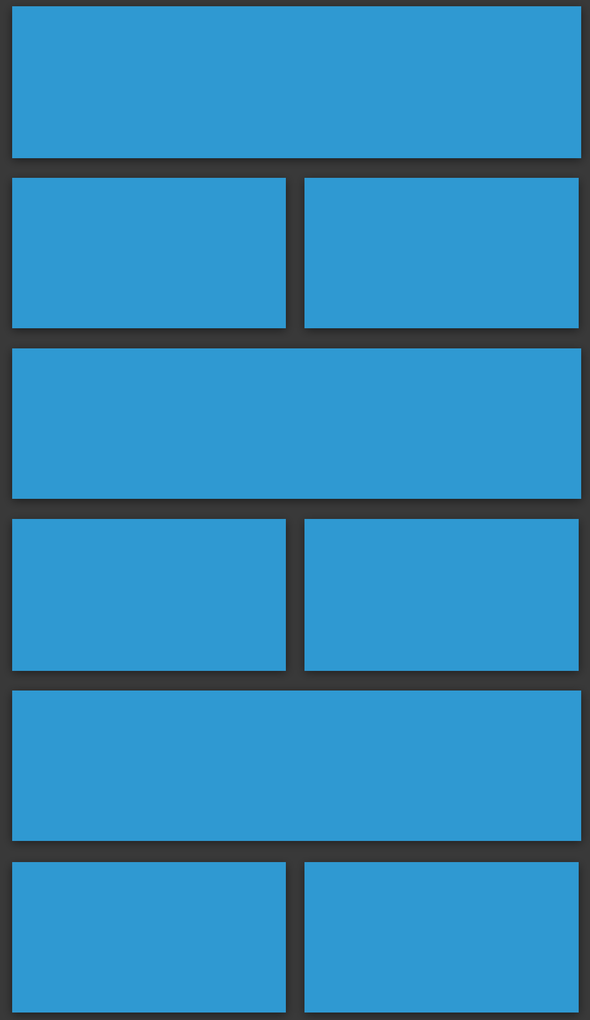
```
LinearLayoutManager manager = new LinearLayoutManager(
        this,
        LinearLayoutManager.VERTICAL,
        false);
```

GridLayoutManager



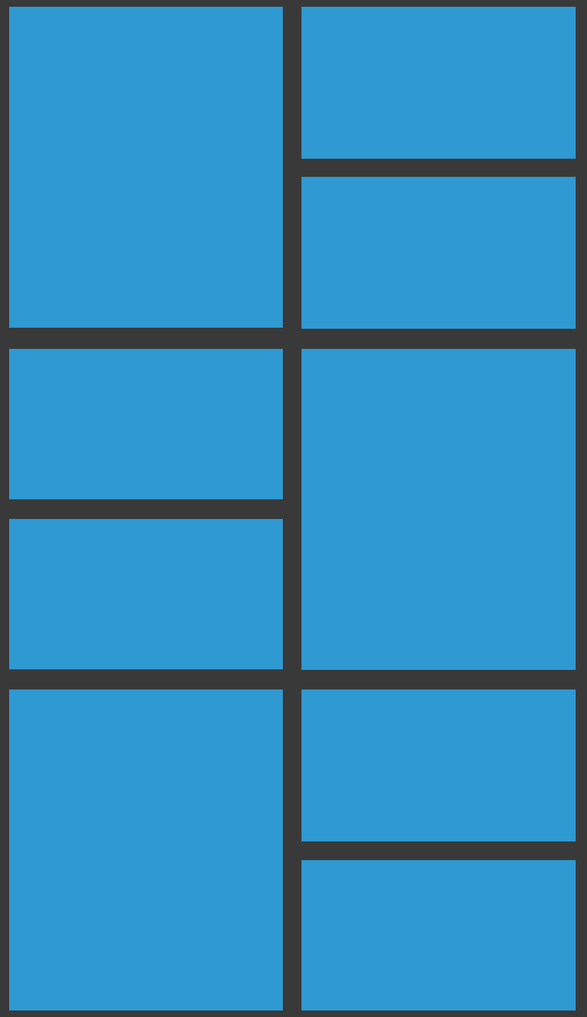
```
GridLayoutManager manager = new GridLayoutManager(
        this,
        2, /* Number of grid columns */
        GridLayoutManager.VERTICAL,
        false);
```

GridLayoutManager



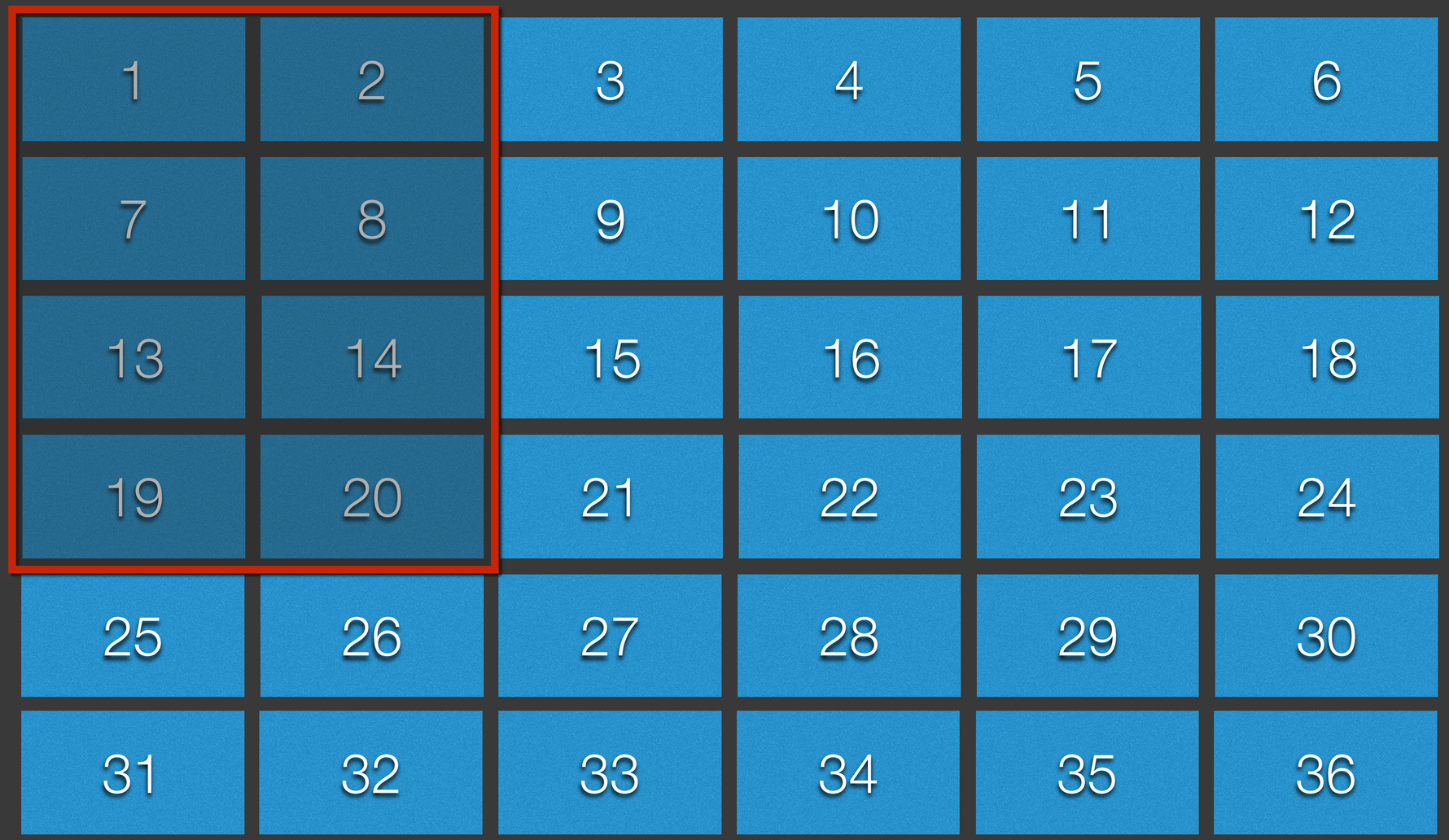
```
GridLayoutManager manager = new GridLayoutManager(
        this,
        2, /* Number of grid columns */
        GridLayoutManager.VERTICAL,
        false);

manager.setSpanSizeLookup(
        new GridLayoutManager.SpanSizeLookup() {
            @Override
            public int getSpanSize(int position) {
                //Stagger every other row
                return (position % 3 == 0 ? 2 : 1);
            }
        });
```

StaggeredGridLayoutManager

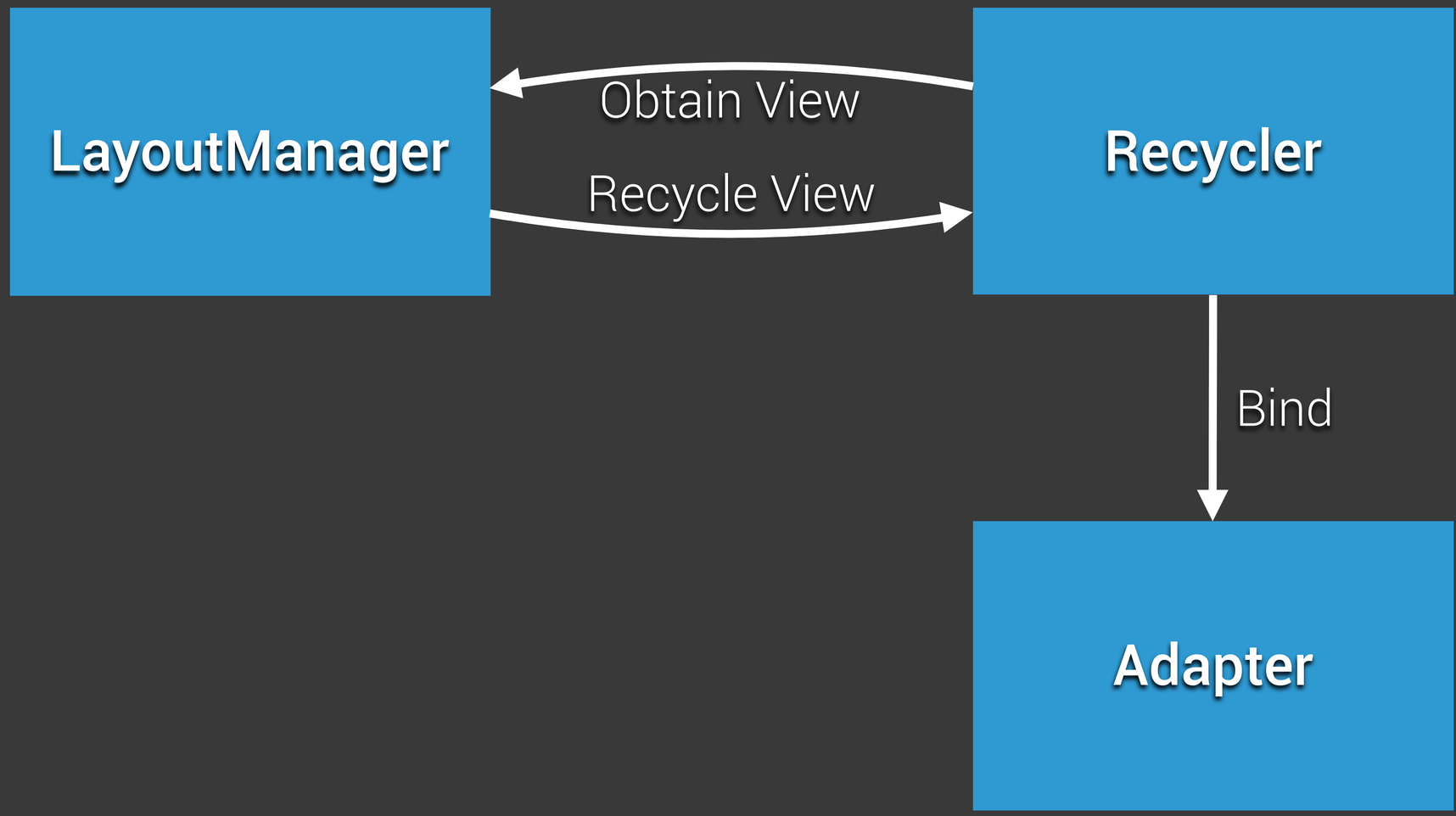


```
StaggeredGridLayoutManager manager =
        new StaggeredGridLayoutManager(
                2, /* Number of grid columns */
                StaggeredGridLayoutManager.VERTICAL);
```

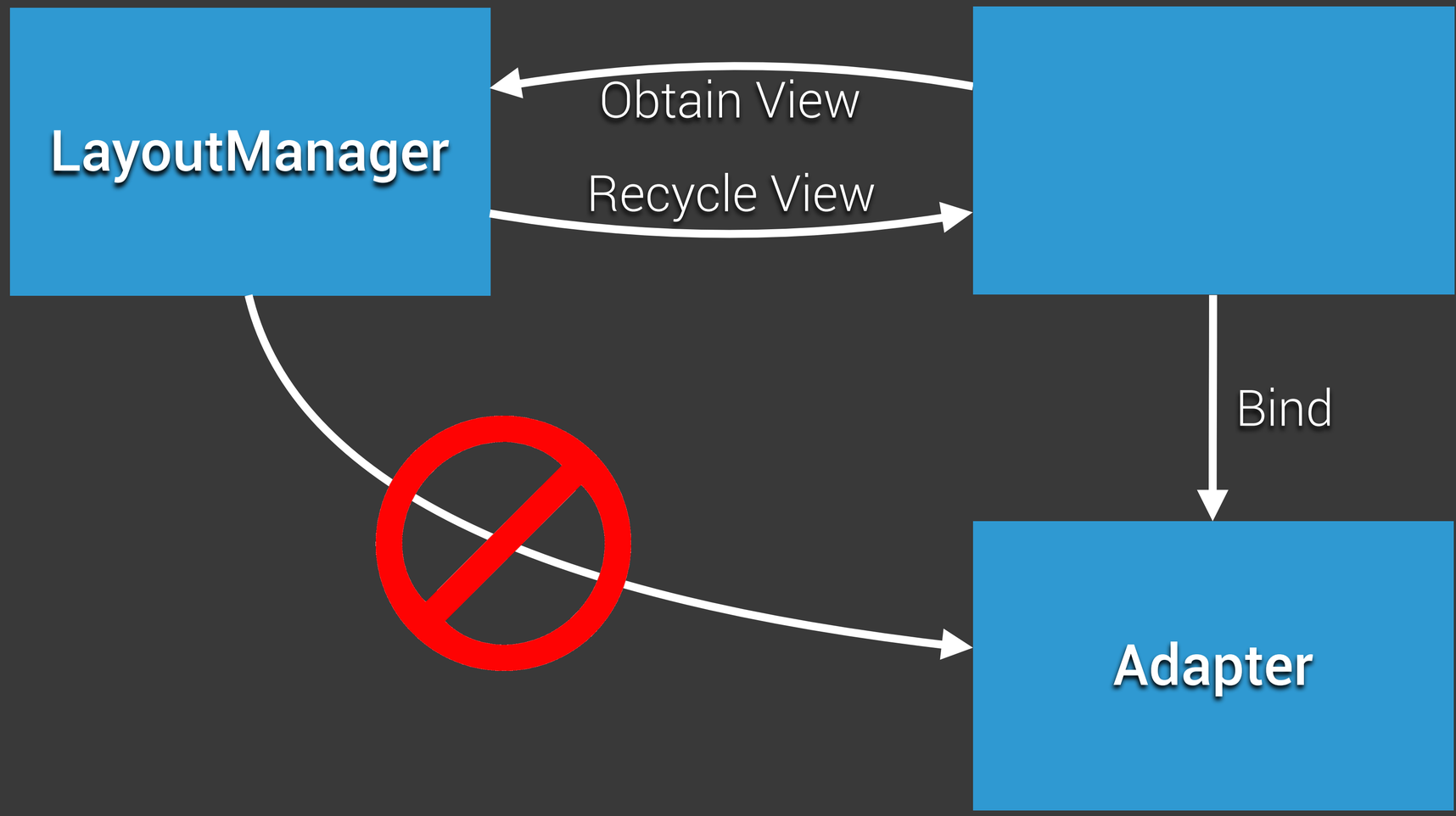
Case Study: FixedGridLayoutManager



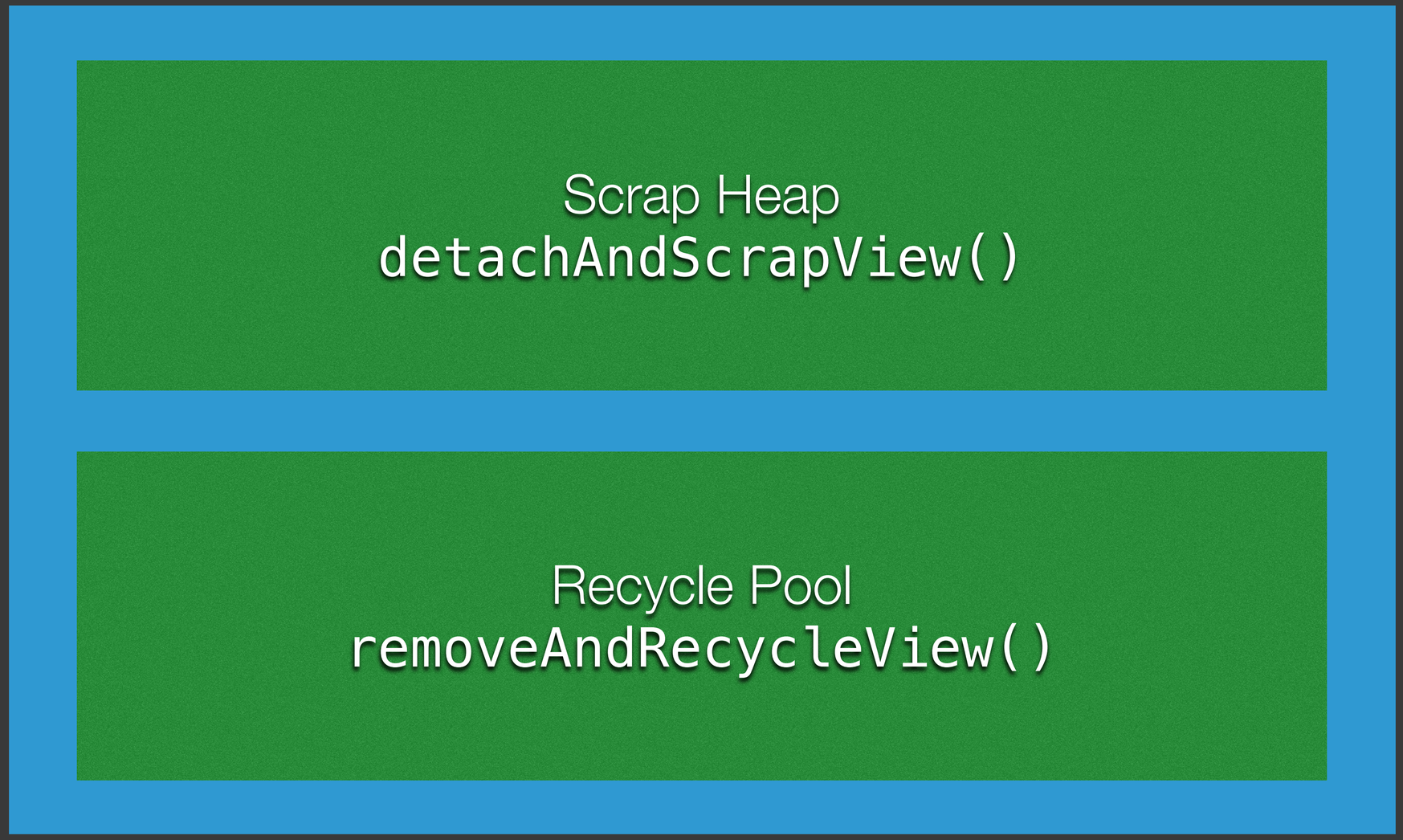
The Recycler



The Recycler

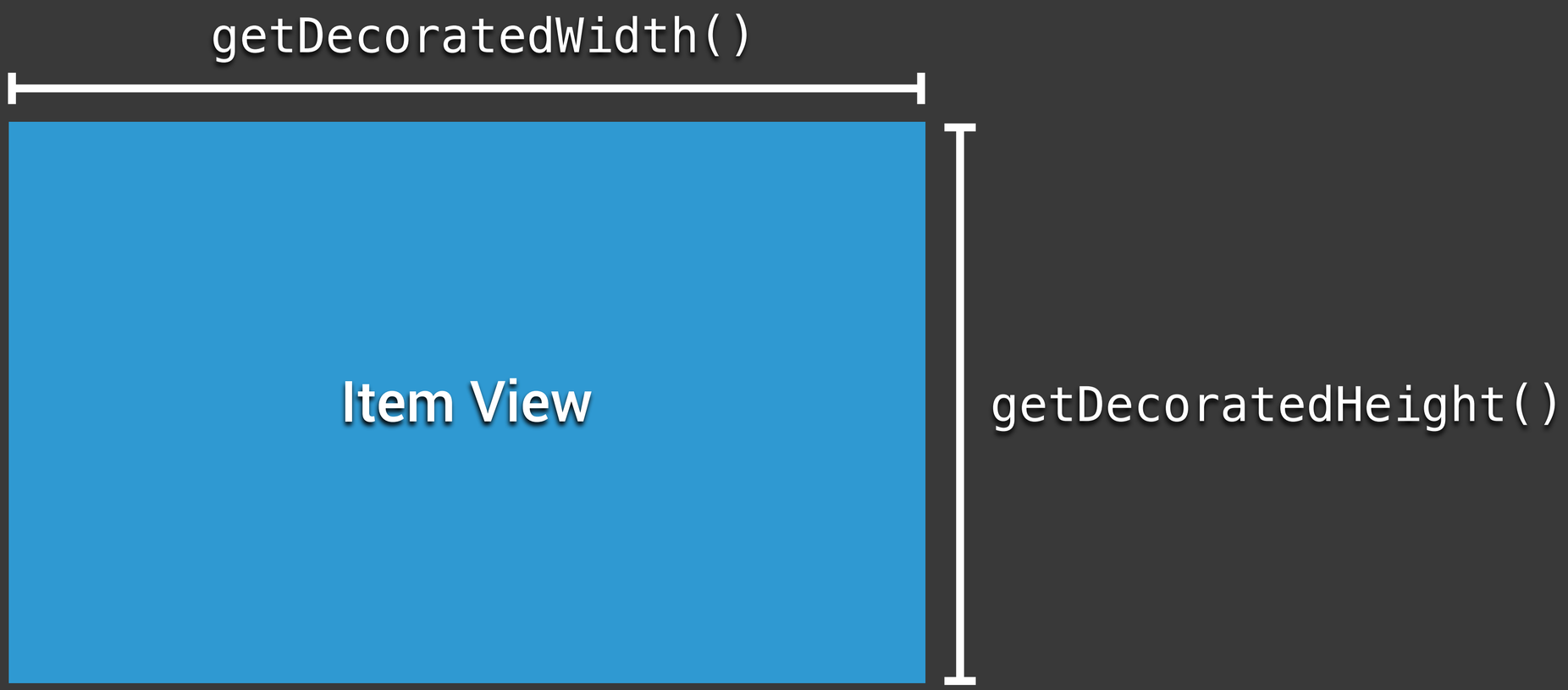


The Recycler

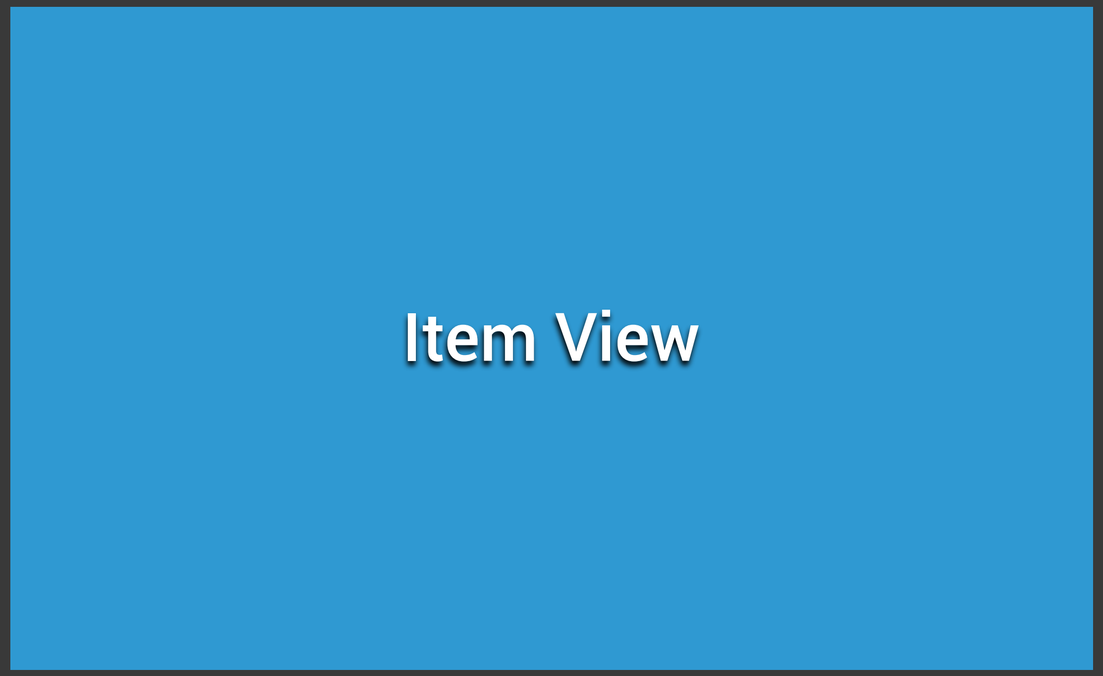


Handling Decorations

Handling Decorations



Handling Decorations




```
layoutDecorated()
measureChild()
```

The Fill Technique

fillGaps()

Discover first visible position/location

Find layout gaps (at the edges)

Scrap everything

Lay out all visible positions

Save as little state as possible.

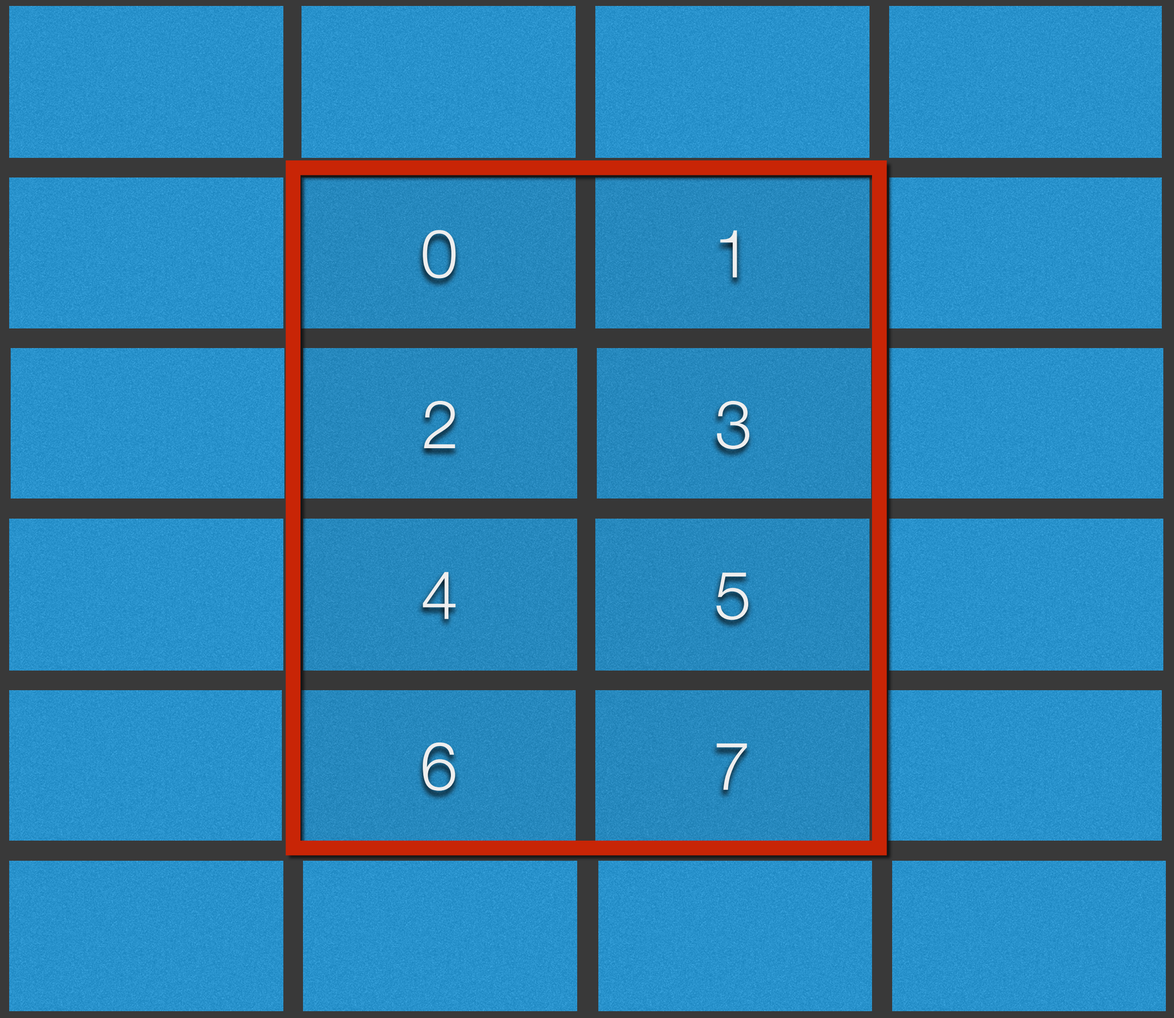
Is there space before the first visible position?

Is there space after the last visible position?

The Fill Technique

```
private void fillGrid(int direction, …,
                      RecyclerView.Recycler recycler,
                      RecyclerView.State state) {

    //…Obtain the first visible item position…

    detachAndScrapAttachedViews(recycler);

    for (…) {
        int nextPosition = …;

        View View = recycler.getViewForPosition(nextPosition);
        addView(view);

        measureChildWithMargins(view, …);
        layoutDecorated(view, …);
    }
```

```
private void fillGrid(int direction, …,
                      RecyclerView.Recycler recycler,
                      RecyclerView.State state) {

    //…Obtain the first visible item position…

    detachAndScrapAttachedViews(recycler);

    for (…) {
        int nextPosition = …;

        View View = recycler.getViewForPosition(nextPosition);
        addView(view);

        measureChildWithMargins(view, …);
        layoutDecorated(view, …);
    }

    //Remove anything that is left behind
    final List<RecyclerView.ViewHolder> scrapList = recycler.getScrapList();
    for (int i=0; i < scrapList.size(); i++) {
        final View removingView = scrapList.get(i);
        recycler.recycleView(removingView);
    }
}
```



Use Attach/ Detach to quickly reorder view indices.

Level 1: Make It Work

onLayoutChildren()

Run a FILL operation

canScrollVertically()
canScrollHorizontally()

Which axes enable scrolling?

scrollHorizontallyBy()
scrollVerticallyBy()

Clamp supplied delta against boundary conditions

Shift all views in the layout

Trigger a FILL operation

Report back actual distance scrolled

Level 1: Make It Work

```
public int scrollHorizontallyBy(int dx,
        RecyclerView.Recycler recycler, RecyclerView.State state) {
    …

    int delta;
    if (dx > 0) { // Contents are scrolling left
        delta = …;
    } else { // Contents are scrolling right
        delta = …;
    }

    offsetChildrenHorizontal(delta);

    if (dx > 0) {
        fillGrid(DIRECTION_START, …, recycler, state);
    } else {
        fillGrid(DIRECTION_END, …, recycler, state);
    }

    return -delta;
}
```

Level 2: Data Set Changes

onAdapterChanged()

removeAllViews()

notifyDataSetChanged() –> onLayoutChanged()

Treat as a new layout, and just run a FILL…

Level 3: Add Some Flair

scrollToPosition()

Track Requested Position

Trigger requestLayout()

smoothScrollToPosition()

Create a SmoothScroller instance

Set the target position

Invoke startSmoothScroll()

Level 3: Add Some Flair

```
LinearSmoothScroller scroller =
        new LinearSmoothScroller(recyclerView.getContext()) {

    @Override
    public PointF computeScrollVectorForPosition(int targetPosition) {
        final int rowOffset = …;
        final int columnOffset = …;

        return new PointF(columnOffset * stepWidth,
                rowOffset * stepHeight);
    }
};

scroller.setTargetPosition(position);
startSmoothScroll(scroller);
```

Level 4: Zen Master

supportsPredictiveItemAnimations()

We have more to say about animations…

onLayoutChildren() [isPreLayout() == true]

Note any removed views **–>** LayoutParams.isViewRemoved()

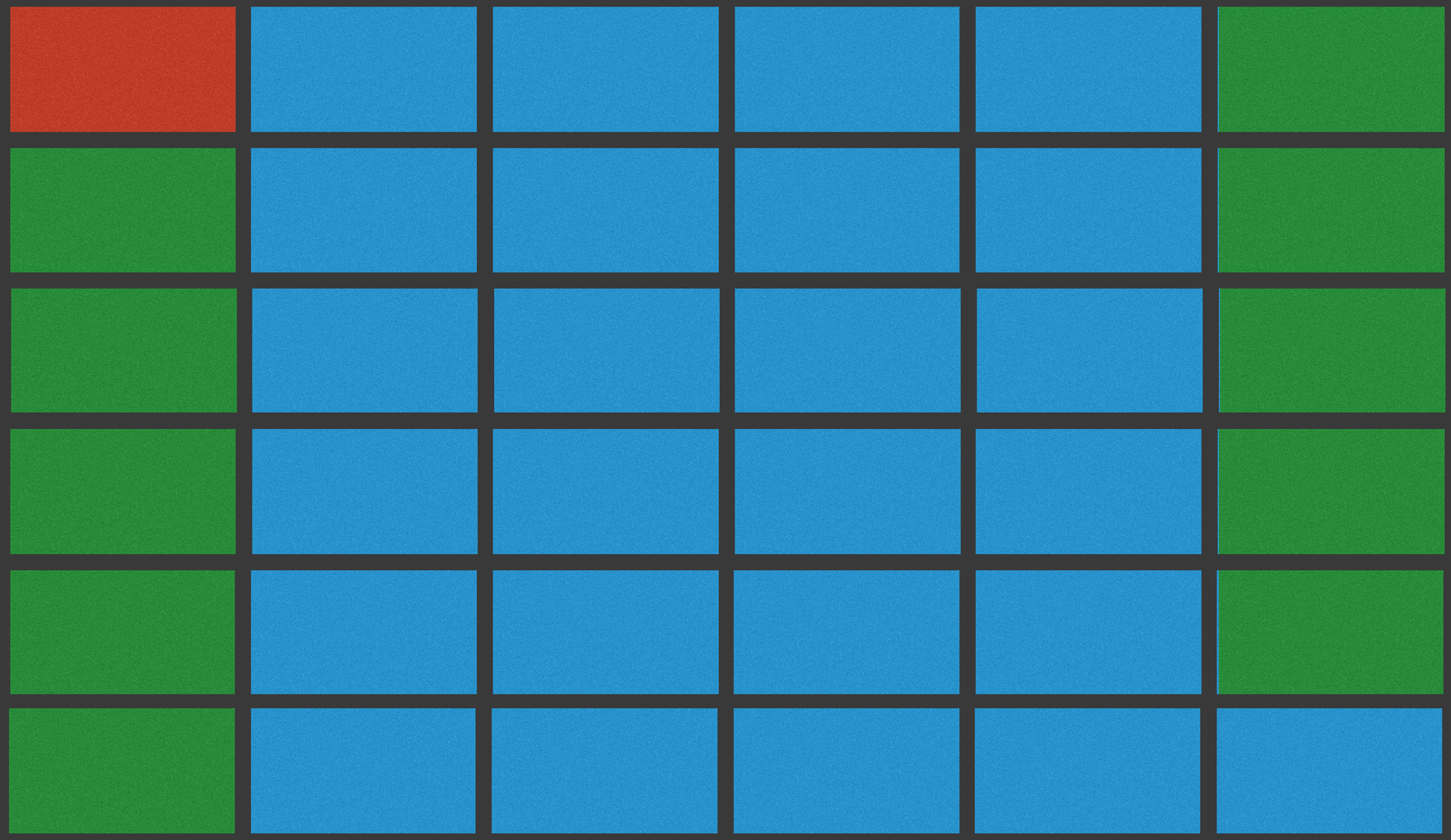
Add extra views during FILL to cover space left behind

onLayoutChildren()

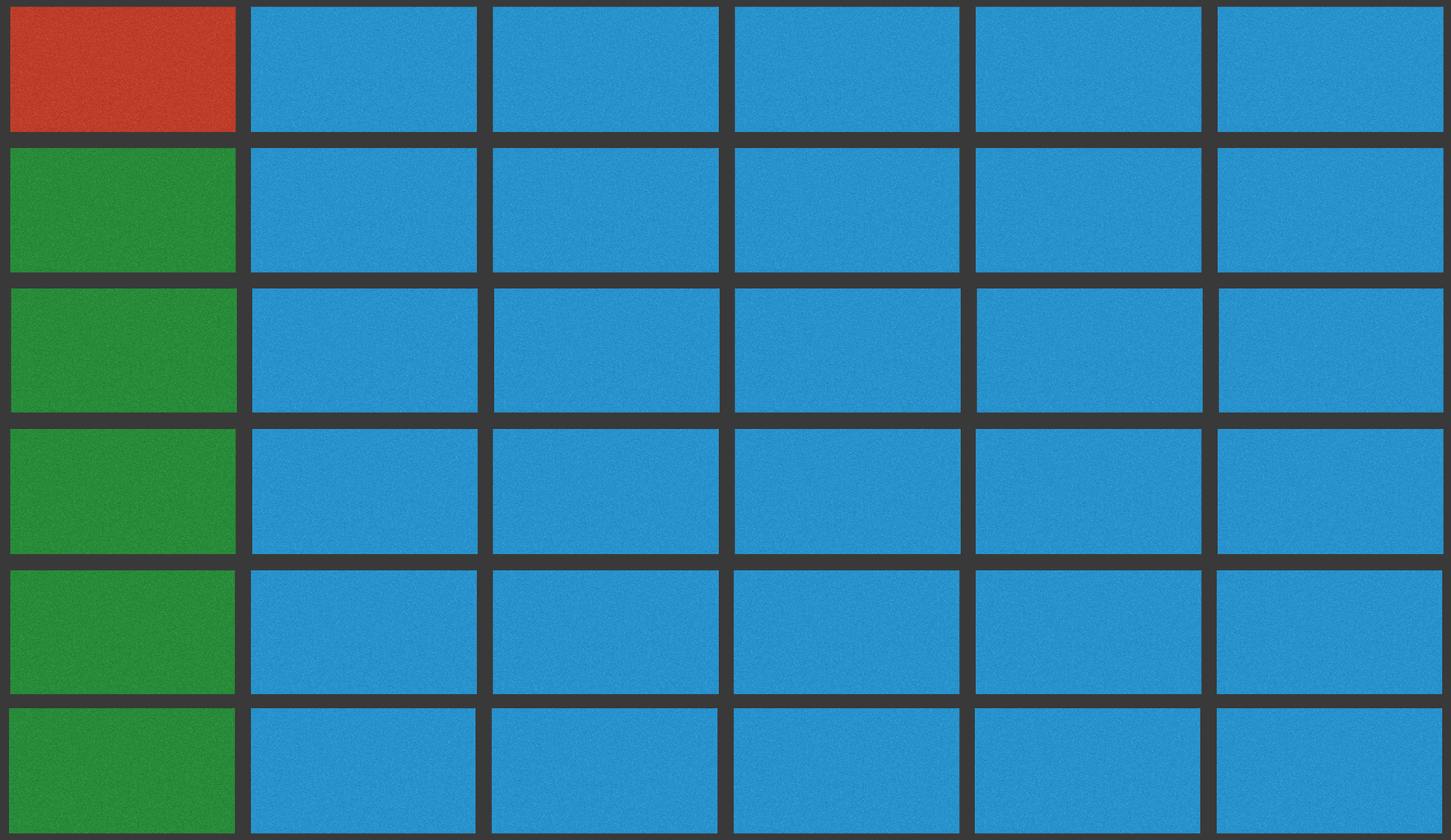
Place disappearing views off-screen

```
public void onLayoutChildren(RecyclerView.Recycler recycler, RecyclerView.State state) {
    …

    if (state.isPreLayout()) {
        final View child = getChildAt(…);
        final LayoutParams lp = (LayoutParams) child.getLayoutParams();

        if (lp.isItemRemoved()) { //Track and count view removals… }
    }

    //Clear all attached views into the recycle bin
    detachAndScrapAttachedViews(recycler);

    //Fill the grid for the initial layout of views
    fillGrid(…);

    //Anything left? Lay out for disappearing animation
    if (!state.isPreLayout() && !recycler.getScrapList().isEmpty()) {
        final List<RecyclerView.ViewHolder> scrapList = recycler.getScrapList();

        //Laying out a scrap item removes it from the list…
        //Beware concurrent modification!

        …
    }
}
```

Default Item Animations



Predictive Item Animations



Resources

- RecyclerView Playground
  - https://github.com/devunwired/recyclerview-playground
- Building a RecyclerView LayoutManager
  - http://wiresareobsolete.com/
  - Redux - Handling Disconnected Ranges
- Android SDK Reference
  - https://developer.android.com/training/material/lists-cards.html