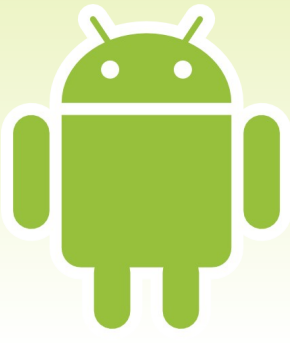
RecyclerView Basics

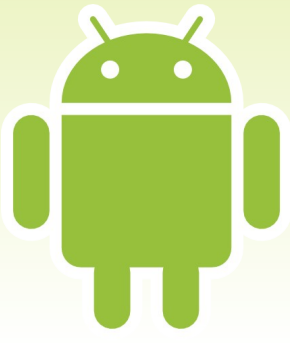
The Role of RecyclerView

- Definite Replacement for ListView, GridView, third-party “staggered grids”
- Probable Replacement
  - StackView
  - ExpandableListView
  - Gallery
- Possible Replacement
  - Spinner
  - AutoCompleteTextView



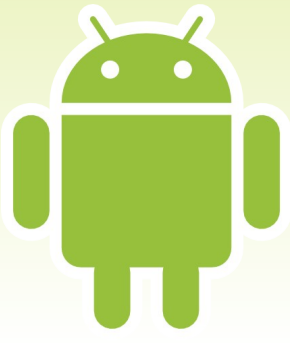
"With great power comes great responsibility."

– "Uncle" Ben Parker



"With great power comes... **great googly moogly**, this is complicated!"

– a balding guy



What We Get From RecyclerView

- More Flexible API
  - Pluggable implementations for most key bits of functionality
- More Hooks
  - Example: animated effects for data changes
- More Headaches
  - No out-of-the-box replacements, must roll yourself or use third-party libraries

Pieces and Parts

- Dependency on `recyclerview-v7`
- The RecyclerView itself
- A LayoutManager
  - No, not a layout manager, like LinearLayout...
- An Adapter
  - No, not a ListAdapter, like ArrayAdapter
- A ViewHolder
  - No, not just any cobbled-together holder

```
dependencies {
    compile 'com.android.support:recyclerview-v7:21.0.3'
}
```

```
import android.app.Activity;
import android.support.v7.widget.RecyclerView;

public class RecyclerViewActivity extends Activity {
  private RecyclerView rv=null;

  public void setAdapter(RecyclerView.Adapter adapter) {
    getRecyclerView().setAdapter(adapter);
  }

  public RecyclerView.Adapter getAdapter() {
    return(getRecyclerView().getAdapter());
  }
```

```
public void setLayoutManager(RecyclerView.LayoutManager mgr) {
  getRecyclerView().setLayoutManager(mgr);
}

public RecyclerView getRecyclerView() {
  if (rv==null) {
    rv=new RecyclerView(this);
    rv.setHasFixedSize(true);
    setContentView(rv);
  }

  return(rv);
}
```

```
public class MainActivity extends RecyclerViewActivity {
  private static final String[] items={"lorem", "ipsum", "dolor",
          "sit", "amet",
          "consectetuer", "adipiscing", "elit", "morbi", "vel",
          "ligula", "vitae", "arcu", "aliquet", "mollis",
          "etiam", "vel", "erat", "placerat", "ante",
          "porttitor", "sodales", "pellentesque", "augue", "purus"};

  @Override
  public void onCreate(Bundle icicle) {
    super.onCreate(icicle);

    setLayoutManager(new LinearLayoutManager(this));
    setAdapter(new IconicAdapter());
  }
```

```
class IconicAdapter extends RecyclerView.Adapter<RowHolder> {
  @Override
  public RowHolder onCreateViewHolder(ViewGroup parent, int viewType) {
    return(new RowHolder(getLayoutInflater()
                         .inflate(R.layout.row, parent, false)));
  }

  @Override
  public void onBindViewHolder(RowHolder holder, int position) {
    holder.bindModel(items[position]);
  }

  @Override
  public int getItemCount() {
    return(items.length);
  }
}
```

```
static class RowHolder extends RecyclerView.ViewHolder {
  TextView label=null;
  TextView size=null;
  ImageView icon=null;
  String template=null;

  RowHolder(View row) {
    super(row);

    label=(TextView)row.findViewById(R.id.label);
    size=(TextView)row.findViewById(R.id.size);
    icon=(ImageView)row.findViewById(R.id.icon);

    template=size.getContext().getString(R.string.size_template);
  }
```

```
void bindModel(String item) {
  label.setText(item);
  size.setText(String.format(template, item.length()));

  if (item.length()>4) {
    icon.setImageResource(R.drawable.delete);
  }
  else {
    icon.setImageResource(R.drawable.ok);
  }
}
```

What We Are Still Missing

- Dividers
  - Come for free with ListView
- Click Events
  - Finding out about them
  - Visually responding to them (list selectors)

Divider Options

- Implement in Row Design
  - Example: CardView
- Implement via ItemDecoration
  - Low-level drawing on the Canvas that is our row
- Implement via a Third-Party Library

```
dependencies {
    compile 'com.android.support:recyclerview-v7:21.0.3'
    compile 'com.android.support:cardview-v7:21.0.3'
}
```

```
<android.support.v7.widget.CardView
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:cardview="http://schemas.android.com/apk/res-auto"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_margin="4dp"
  cardview:cardCornerRadius="4dp">

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal">
```

```
@Override
public void onCreate(Bundle icicle) {
  super.onCreate(icicle);

  setLayoutManager(new LinearLayoutManager(this));

  Drawable divider=getResources().getDrawable(R.drawable.item_divider);

  getRecyclerView().addItemDecoration(new HorizontalDividerItemDecoration(divider));
  setAdapter(new IconicAdapter());
}
```

```
// inspired by https://gist.github.com/polbins/e37206fbc444207c0e92

public class HorizontalDividerItemDecoration extends RecyclerView.ItemDecoration {
  private Drawable divider;

  public HorizontalDividerItemDecoration(Drawable divider) {
    this.divider=divider.mutate();
  }
```

```
@Override
public void onDrawOver(Canvas c, RecyclerView parent, RecyclerView.State state) {
  int left=parent.getPaddingLeft();
  int right=parent.getWidth()-parent.getPaddingRight();

  int childCount=parent.getChildCount();

  for (int i=0; i<childCount; i++) {
    View child=parent.getChildAt(i);
    RecyclerView.LayoutParams params=
        (RecyclerView.LayoutParams)child.getLayoutParams();

    int top=child.getBottom()+params.bottomMargin;
    int bottom=top+divider.getIntrinsicHeight();

    divider.setBounds(left, top, right, bottom);
    divider.draw(c);
  }
}
```

Responding to Clicks

- Handle via Listeners on Row Widgets
  - Nothing at the RecyclerView level for this
  - Need to choose scope
    - Static row contents: apply to row container
    - Interactive row contents: apply to everything as needed, possibly with different listeners for different scenarios
  - Apply in ViewHolder

```
class RowController extends RecyclerView.ViewHolder
    implements View.OnClickListener {
  TextView label=null;
  TextView size=null;
  ImageView icon=null;
  String template=null;

  RowController(View row) {
    super(row);

    label=(TextView)row.findViewById(R.id.label);
    size=(TextView)row.findViewById(R.id.size);
    icon=(ImageView)row.findViewById(R.id.icon);

    template=size.getContext().getString(R.string.size_template);

    row.setOnClickListener(this);
  }
```

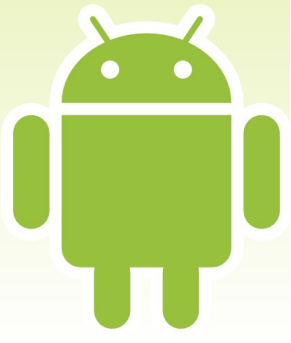
```
@Override
public void onClick(View v) {
  Toast.makeText(v.getContext(),
      String.format("Clicked on position %d", getPosition()),
      Toast.LENGTH_SHORT).show();
}

void bindModel(String item) {
  label.setText(item);
  size.setText(String.format(template, item.length()));

  if (item.length()>4) {
    icon.setImageResource(R.drawable.delete);
  }
  else {
    icon.setImageResource(R.drawable.ok);
  }
}
```

Visualizing the Clicks

- Where?
  - In the row layouts and/or in the ViewHolder
- No, I Mean “Where?”
  - Selector over the row
  - Selector background on the row



```
  </LinearLayout>

  <View
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="?android:attr/selectableItemBackground" />

</android.support.v7.widget.CardView>
```

```
<android.support.v7.widget.CardView
  xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:cardview="http://schemas.android.com/apk/res-auto"
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:layout_margin="4dp"
  cardview:cardCornerRadius="4dp">

  <LinearLayout
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:background="?android:attr/selectableItemBackground">
```

Ripple Selectors

- Official Selector of Material Design
  - Ripple animation emanating from touch point
- Problem: Default = Centered Animation
  - Backgrounds do not know the touch point where the user tapped on the row
- Solution: `setHotspot()`
  - Called on the RippleDrawable background image
  - Changes default emanation point for ripple

```
RowController(View row) {
  super(row);

  label=(TextView)row.findViewById(R.id.label);
  size=(TextView)row.findViewById(R.id.size);
  icon=(ImageView)row.findViewById(R.id.icon);

  template=size.getContext().getString(R.string.size_template);

  row.setOnClickListener(this);
```

```
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP) {
  row.setOnTouchListener(new View.OnTouchListener() {
    @TargetApi(Build.VERSION_CODES.LOLLIPOP)
    @Override
    public boolean onTouch(View v, MotionEvent event) {
      v
        .findViewById(R.id.row_content)
        .getBackground()
        .setHotspot(event.getX(), event.getY());

      return(false);
    }
  });
}
```

Getting Beyond the List

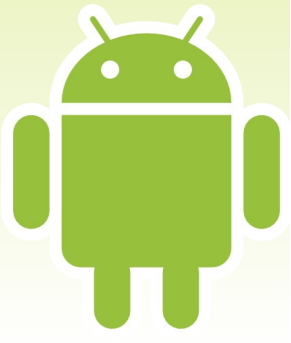
- GridLayoutManager
  - Configure, attach to RecyclerView instead of LinearLayoutManager
  - Simple: all cells the same size
  - Complex: cells vary in size
    - E.g., different widths for different columns

```
@Override
public void onCreate(Bundle icicle) {
  super.onCreate(icicle);

  setLayoutManager(new GridLayoutManager(this, 2));
  setAdapter(new IconicAdapter());
}
```

Getting Beyond the List

- StaggeredGridLayoutManager
- Third-Party Layout Managers
  - https://android-arsenal.com/search?q=recyclerview



Beyond the Basics

- Models Beyond Arrays
- Going Beyond Single Row Layouts
  - E.g., header-and-detail
- Going Beyond Static Rows
  - E.g., checklists
- Going Beyond Static Content
  - Adding and removing model data, with animated effects