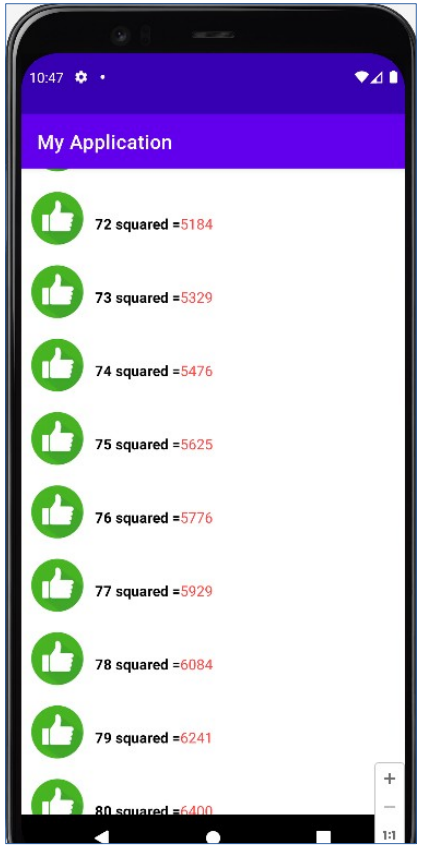
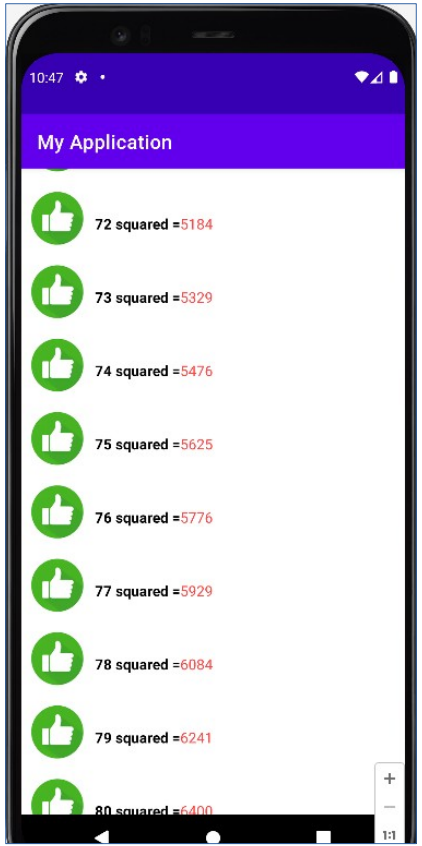
# Each row is a separate RowViewHolder




```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

# Each row is a separate RowViewHolder



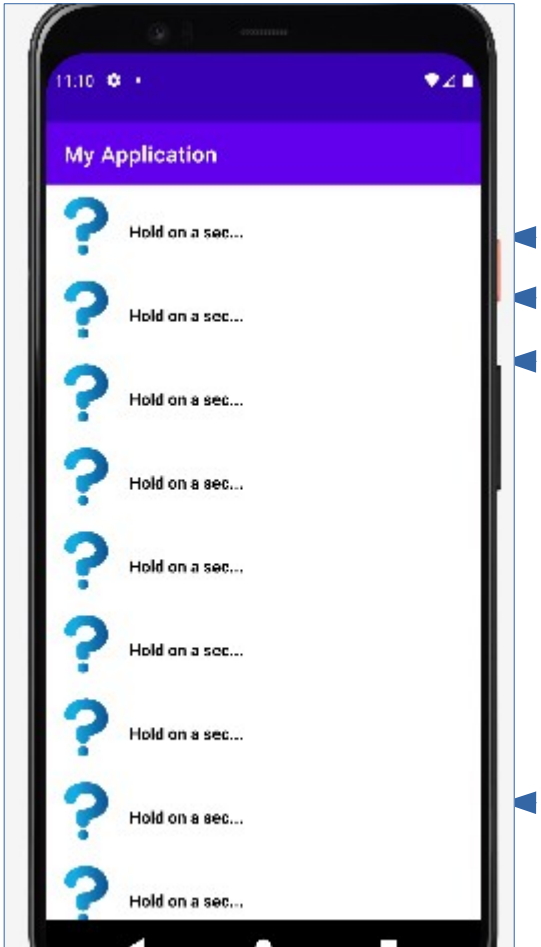

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

# Typically, some data is fast to get
# And some is not



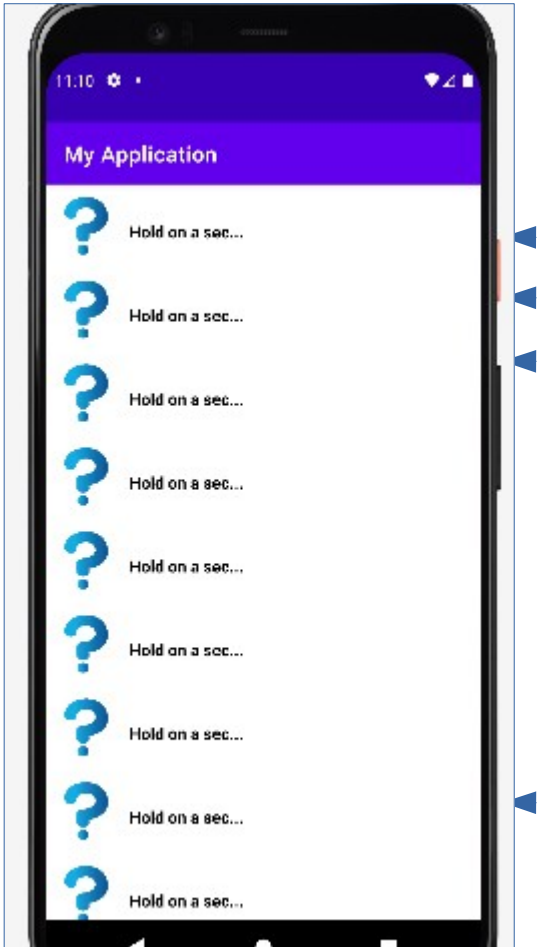

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

# So display the fast data
# And launch a thread for the slower data



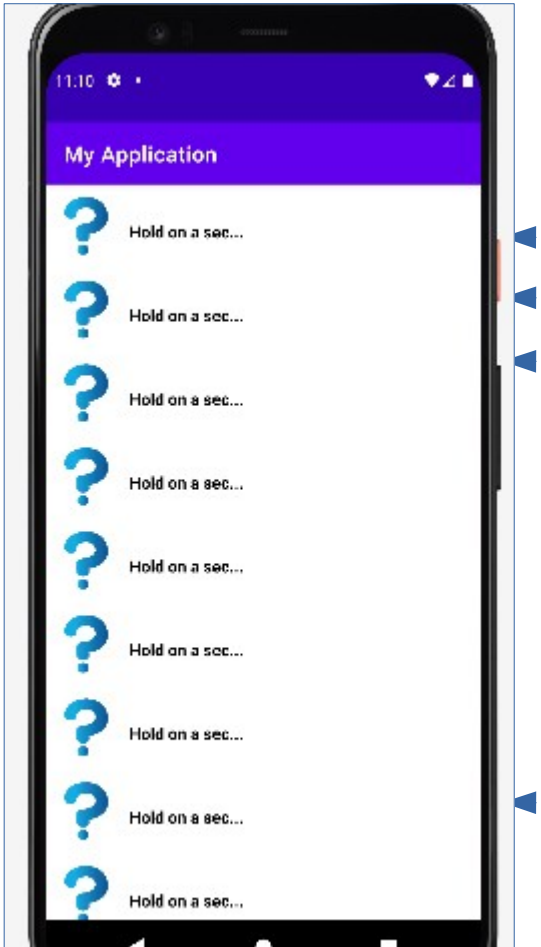

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

So display the fast data
And launch a thread for the slower data




```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

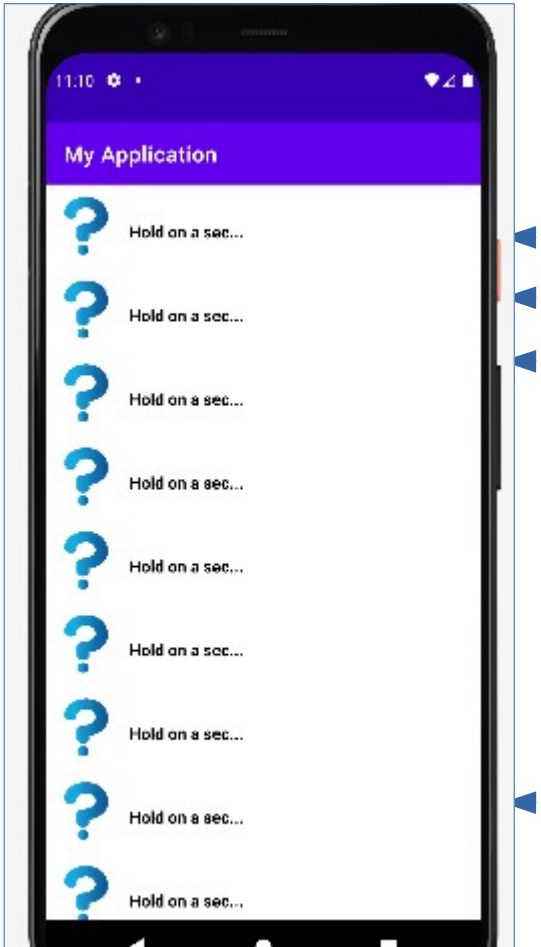
```
public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
        //passing in an existing instance, reuse the internal resources
        //pass our data to our ViewHolder
        RowViewHolder viewHolder = (RowViewHolder) holder;
        viewHolder.numb= holder.getAdapterPosition();

        //initialize the UI
        viewHolder.iv.setImageResource(R.drawable.unknown);
        viewHolder.tvInfo.setText("Hold on a sec...");
        viewHolder.tvResult.setText("");

        //launch a thread to 'retreive' the image (slow to get data)
        GetNumber myTask = new GetNumber(viewHolder,(MainActivity) ctx);
        myTask.start();

}
```

So display the fast data
And launch a thread for the slower data




```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
        //passing in an existing instance, reuse the internal resources
        //pass our data to our ViewHolder
        RowViewHolder viewHolder = (RowViewHolder) holder;
        viewHolder.numb= holder.getAdapterPosition();

        //initialize the UI
        viewHolder.iv.setImageResource(R.drawable.unknown);
        viewHolder.tvInfo.setText("Hold on a sec...");
        viewHolder.tvResult.setText("");

        //launch a thread to 'retreive' the image (slow to get data)
        GetNumber myTask = new GetNumber(viewHolder,(MainActivity) ctx);
        myTask.start();

}
```

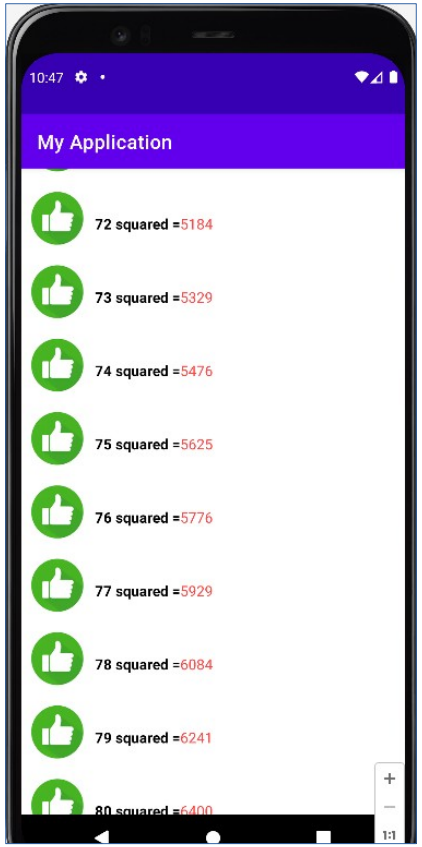
```
public GetNumber(RowViewHolder myVh, MainActivity act) {
    //hold on to a reference to this viewholder
    //note that its contents (specifically iv) may change
    //iff the viewholder is recycled
    this.myVh = myVh;
    //make a copy to compare later, once we have the image
    this.original_number = myVh.numb;

    //hold on to the activity
    this.act=act;
}
```

Thread constructor

Save the original Number we are multiplyng

# So display the fast data
# And launch a thread for the slower data




```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
        //passing in an existing instance, reuse the internal resources
        //pass our data to our ViewHolder
        RowViewHolder viewHolder = (RowViewHolder) holder;
        viewHolder.numb= holder.getAdapterPosition();

        //initialize the UI
        viewHolder.iv.setImageResource(R.drawable.unknown);
        viewHolder.tvInfo.setText("Hold on a sec...");
        viewHolder.tvResult.setText("");

        //launch a thread to 'retreive' the image (slow to get data)
        GetNumber myTask = new GetNumber(viewHolder,(MainActivity) ctx);
        myTask.start();

}
```

Thread constructor

```
public GetNumber(RowViewHolder myVh, MainActivity act) {
    //hold on to a reference to this viewholder
    //note that its contents (specifically iv) may change
    //iff the viewholder is recycled
    this.myVh = myVh;
    //make a copy to compare later, once we have the image
    this.original_number = myVh.numb;

    //hold on to the activity
    this.act=act;
}
```

Save the original Number we are multiplyng

Thread run

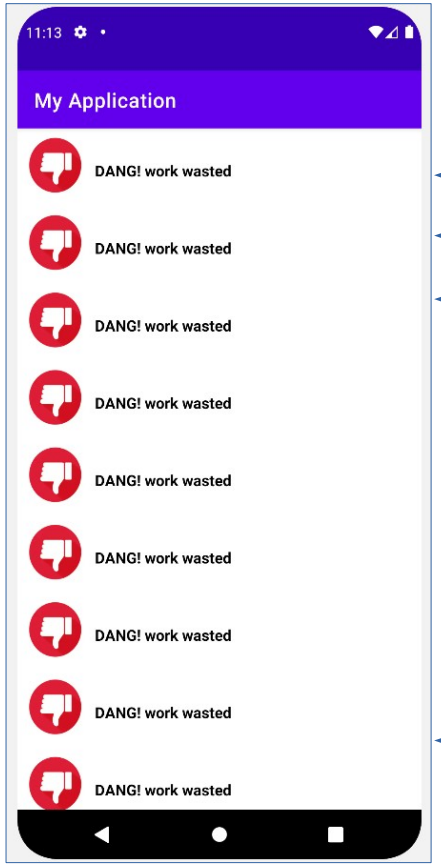
```
public void run() {
  super.run();

  //just sleep for a bit to simulate long running downloaded
  //but could just as easily make a network call
  try {
     Thread.sleep(2000); //sleep for 2 seconds
  } catch (InterruptedException e) {
     e.printStackTrace();
  }

  //create result (does this need protection? Not as written)
  result=original_number*original_number;
  act.runOnUiThread(new Runnable() {
     @Override
     public void run() {
        //got a result, if the following are NOT equal
        // then the view has been recycled and is being used by another
        // number DO NOT MODIFY
        if (myVh.numb == original_number){
           //still valid
           //set the result on the main thread
           myVh.iv.setImageResource(R.drawable.ok);
           myVh.tvInfo.setText(Integer.toString(myVh.numb) + " squared =");
           myVh.tvResult.setText(Integer.toString(result));
        }
        else{
           myVh.iv.setImageResource(R.drawable.notneeded);
           myVh.tvInfo.setText("DANG! work wasted");
           myVh.tvResult.setText("");
        }
     }
  });
}
```

If these numbers are the same, then the view has NOT been recycled

# So display the fast data
# And launch a thread for the slower data




```
class RowViewHolder extends RecyclerView.ViewHolder {
    int numb = UNINITIALIZED;
    ImageView iv;
    TextView tvInfo;
    TextView tvResult;
    //after construction, above member variables hold references
    //to widgets IN THIS PARTICULAR ROW!
    public RowViewHolder(@NonNull View itemView) {
        super(itemView);
        tvInfo = (TextView)itemView.findViewById(R.id.tvInfo );
        tvResult = (TextView)itemView.findViewById(R.id.tvResult );
        iv=(ImageView)itemView.findViewById(R.id.imageView1);
    }
}
```

```
public void onBindViewHolder(@NonNull RecyclerView.ViewHolder holder, int position) {
        //passing in an existing instance, reuse the internal resources
        //pass our data to our ViewHolder
        RowViewHolder viewHolder = (RowViewHolder) holder;
        viewHolder.numb= holder.getAdapterPosition();

        //initialize the UI
        viewHolder.iv.setImageResource(R.drawable.unknown);
        viewHolder.tvInfo.setText("Hold on a sec...");
        viewHolder.tvResult.setText("");

        //launch a thread to 'retreive' the image (slow to get data)
        GetNumber myTask = new GetNumber(viewHolder,(MainActivity) ctx);
        myTask.start();

}
```

```
public GetNumber(RowViewHolder myVh, MainActivity act) {
    //hold on to a reference to this viewholder
    //note that its contents (specifically iv) may change
    //iff the viewholder is recycled
    this.myVh = myVh;
    //make a copy to compare later, once we have the image
    this.original_number = myVh.numb;

    //hold on to the activity
    this.act=act;
}
```

Thread constructor

Save the original Number we are multiplyng

```
public void run() {
  super.run();

  //just sleep for a bit to simulate long running downloaded
  //but could just as easily make a network call
  try {
     Thread.sleep(2000); //sleep for 2 seconds
  } catch (InterruptedException e) {
     e.printStackTrace();
  }

  //create result (does this need protection? Not as written)
  result=original_number*original_number;
  act.runOnUiThread(new Runnable() {
     @Override
     public void run() {
        //got a result, if the following are NOT equal
        // then the view has been recycled and is being used by another
        // number DO NOT MODIFY
        if (myVh.numb == original_number){
           //still valid
           //set the result on the main thread
           myVh.iv.setImageResource(R.drawable.ok);
           myVh.tvInfo.setText(Integer.toString(myVh.numb) + " squared =");
           myVh.tvResult.setText(Integer.toString(result));
        }
        else{
           myVh.iv.setImageResource(R.drawable.notneeded);
           myVh.tvInfo.setText("DANG! work wasted");
           myVh.tvResult.setText("");
        }
     }
  });
}
```

Thread run

If these numbers are NOT the same, then the view has been recycled